«packt»

1 ИЗДАНИЕ

Справочник Исследователя Уязвимостей

Исчерпывающее руководство по нахождению, описанию и публикации уязвимостей безопасности

БЕНДЖАМИН СТРАУТ

Справочник Исследователя Уязвимостей

Исчерпывающее руководство по нахождению,

описанию и публикации уязвимостей безопасности

Бенджамин Страут

БИРМИНГЕМ - БОМБЕЙ

Перевод на русский язык **xss.is/handersen**, октябрь 2023 – март 2024.

Специально для **XSS.IS**.

Исправлены обнаруженные в английском оригинале опечатки и добавлены кое-какие ссылки по теме, отсутствующие в английской версии, а также другие полезные материалы.

Справочник Исследователя Уязвимостей

Ответственный за всё и всех: Мохаммед Рийан Хан

Ответственный за издание: Хушбу Самкария

Главный редактор: Рансил Ребелло

Технический редактор: Нитик Черувакодан

Литературный редактор: Safis Editing

Координатор проекта: Эшвин Харва

Корректор: Safis Editing

Составитель оглавления: Манжу Арасэн

Дизайнер: Алишон Мендонца

Координатор по маркетингу: Мэрилу ди Мелло

Первое издание: февраль 2023

Артикул: 1200123

Опубликовано Packt Publishing Ltd.

Ливери Плейс

Ливери-стрит 35

Бирмингем

Почтовый индекс: B3 2PB, Соединенное Королевство.

ISBN 978-1-80323-887-6

www.packtpub.com

Моему самому дорогому и старинному другу, Линде Мелхорн. Благодарю тебя за твою дружбу и поддержку на протяжении многих лет. Это много значило для меня.

– *Бенджамин Страут*

Над книгой работали:

Об авторе

Бенджамин Страут – ветеран технологической индустрии, увлеченный работой на стыке технологий. Его опыт работы в индустрии здравоохранения, биотехнологий, фармацевтики и финансов, привел его к роли ведущего пентестера одного из крупнейших конгломератов в области здравоохранения США. Основатель и связующее звено местной DEFCON групп (DC207), штата Мэн, он занимал важнейшее место в качестве приглашенного докладчика на множестве конференций. Он участвовал в работах, как технический обозреватель и опубликовал более 30 CVE для технологий распространенных по всей планете. Когда он никого не обучает или не возится с каким-нибудь техническим казусом – занят разучиванием аккордов блюграсса на своем банджо и игрой со своими кошками Дионисием и Льюисом Сэнксгивином (*странноватое имя для кота*).

В своей карьере, я безмерно признателен этим потрясающим людям, за помощь на моем пути: Дэвид Фридман, Тейлор Шейн, Скотт Аллен и выдающийся исследователь информационной безопасности Райан Бото. Куча благодарностей каждому из DC207 групп, кто сохранил традицию собираться каждый месяц, сохраняет живым хакерское сообщество в штате Мэн. В конце концов появление этой книги было бы невозможно без помощи постоянно развивающейся SANS ICS HyperEncabulatation research.

О рецензентах

Авинаш Синха эксперт по кибербезопасности с более, чем 12 летним опытом, работавший со многими мировыми бизнес-гигантами, такими как IBM, GE HealthCare, Schneider Electric, Airtel, Target, Aujas Networks, и UIDAI Aadhaar обеспечивая их безопасность от угроз следующего поколения.

Он возглавлял команды и управлял реализацией методик устанавливающих порядок действий по обеспечению безопасности OT/IT/клауд секьюрити проектов, архитектурой корпоративной безопасности, тестированием на проникновение, упреждающим обнаружением угроз, IR, работой в “красной команде“ и оценкой эффективности клауд секьюрити для O365, Azure, Citrix и AWS.

Он имеет ученую степень в компьютерных технологиях, специализируясь на искусственном интеллекте и является аспирантом в Интернэшнл Бизнес Симбиозис (IBS – Internetional Business from Symbiosis) в Индии. Владеет сертификатами GICSP, GCFA и SEC 541-Cloud Attacks, а также наблюдает за процедурами сертификации от SANS.

Я хотел бы поблагодарить моего отца, Раджендру Синха за то, что постоянно вдохновлял меня и мою маму за ее любовь и заботу. Так же благодарю, мою прекрасную жену Тану за то, что она заботилась обо мне и держала меня в тонусе, когда мое увлечение кибербезопасностью поглощало чрезмерное время.

Каждый из нас несет ответственность по защите мира от постоянно появляющихся угроз. Особая благодарность Джейкишану Нирмалу, Викраму Дхару и Анилу Кумару PV за их наставления.

Вайбхав Кушваха – ведущий инженер по безопасности, специализирующийся на архитектуре продуктов кибербезопасности и НИОКР, а также специалист по безопасности приложений, автоматизации тестирования на проникновение, сетевым и веб сканерам уязвимостей и многому другому.

Он начинал свою карьеру с пентеста, но позднее был поглощен интересом к исследованиям и разработке в области безопасности программного обеспечения, совершенствованию продуктов и индустрии кибербезопасности, а также автоматизации занудных и повторяющихся задач.

Кроме того, Вайбхав так же увлекается реверс-инжинирингом, работой над облачными сервисами и работал над коннекторами для нескольких облачных сервисов.

Отказ от ответственности

Информация в этой книге предназначена к использованию, только законным образом (например, если вы заключили договор о проведении пентеста и т. п.) Не применяйте никакой информации из этой книги, если у вас нет письменного разрешения от владельца оборудования. Если вы совершите противозаконные действия, то вероятно будете арестованы и привлечены к ответственности по всей строгости закона. Packt Publishing отказывается брать на себя любую ответственность, если вы злоупотребите любой информацией, содержащейся в этой книге. Информация отсюда, должна использоваться только для тестирования, при условии надлежащим образом оформленных разрешений от соответствующих ответственных лиц.

Предисловие

Приветствую, отважный исследователь!

Исследование уязвимостей, это действия по выявлению брешей в безопасности технических систем и последующая передача этой информации людям, которые могут эти бреши закрыть. Это завораживающий процесс, помогающий совершенствовать безопасность и конфиденциальность пользовательских данных по всему миру постоянно растущих технологий, в котором обитаем и мы сами. ***Справочник исследователя уязвимостей***, это книга нацеленная на всестороннее рассмотрение хода действий, которые кто-либо может совершить с новой, ранее неизвестной информацией об уязвимости безопасности, после ее обнаружения. Например вы можете работать в областях разработки ПО, его использования или информационной безопасности. В таком случае, вы возможно заинтересуетесь, каким образом уязвимость в публично выпущенном ПО становится известной и почему это в конечном счете направлено на всеобщее благо конечных пользователей? Ответ заключается в совместных усилиях, трениях между сторонами и упорстве исследователей безопасности в преодолении множества трудностей, входящих в этот сценарий.

В этой книге мы познакомим вас с миром исследования уязвимостей в публично выпущенном ПО, давая вам понимание основ тематики уязвимостей и того, почему они становятся опасными. После этого мы осветим методы начала выбора наиболее интересных объектов исследования и затем применим к этим целям методику основанную на испытанных, принесших результаты техниках. В итоге, вы научитесь составлять планы по поиску уязвимостей безопасности.

Далее, мы будем рассматривать большинство запутанных шагов, процесса ловли рыбки в мутной воде работы с вендорами, в то время, как вы делитесь своими исследованиями и открываете их миру. Мы займемся этим, изучая увлекательные примеры исследования уязвимостей и многочисленные препятствия, которые исследователям безопасности вроде вас, потребуется преодолеть. Наконец, когда вы дочитаете последние главы, вы будете вооружены знаниями, а также разными полезными шаблонами по организации и стандартизации ваших исследований и стратегиями построения лучших взаимоотношений вендор-исследователь с которых вы можете начать сотрудничество или работу на вендора.

Я написал эту книгу потому, что это было одним из моих желаний еще до того, как я начал работать с поставщиками ПО над раскрытием уязвимостей и убеждению их устранять проблемы безопасности в своих продуктах. Принятие в оборот технических систем и программного обеспечения для решения самых разных задач, распространяется (и распространилось) на все сферы человеческой деятельности. В целом индустрия аппаратного обеспечения и ПО, больше, чем когда либо нуждается в людях, которые знают как обнародовать результаты исследований с выгодой для пользователей, и должным образом передавать их ответственным лицам для устранения проблем, если они будут обнаружены.

В идеале, я ожидал бы от тебя, читатель обладания достаточными техническими навыками, которые ты сможешь применить к тем софт-скиллам, которые изучишь в этой книге. Мы не учим вас последним техникам исследования уязвимостей – какие-то книги, курсы и веб-контент на эту тему выходят каждый год. Мы ожидаем, что вы приступите с подготовленными записями или будете обращаться к таким источникам. Вместо этого, книга направлена помочь вам понять, как публично раскрытые уязвимости оказываются в системе CVE (или нет), как находить объекты для исследования и разрабатывать методы организации трудоемкого процесса раскрытия информации и публикации результатов исследований с ответственным подходом, который сделает мир вокруг нас лучше. Даже если вы не овладеете продвинутыми техническими навыками, с этой книгой вы найдете сам процесс в целом доступным, увлекательным и очень полезным – настолько, что станете пытаться повысить свои знания еще больше.

Для кого эта книга

Эта книга адресована аналитикам безопасности, исследователям, пентестерам, разработчикам ПО, инженерам технических отраслей и любым руководителям, которые хотят узнать, как уязвимости находятся и затем раскрываются широкой публике. Перед началом, мы предполагаем наличие у вас обобщенных знаний об операционных системах, программном обеспечении и взаимосвязанных системах. Наличие предшествующего опыта в нахождении и раскрытии уязвимостей – необязательно. Тем не менее, какое-никакое знакомство со сканерами уязвимостей и прочими инструментами информационной безопасности, помогло бы вам ускорить ваше продвижение в изучении объектов исследования, а также нахождению и раскрытию вашей первой уязвимости.

Какие темы охватывает эта книга

Глава 1, Знакомство с понятием уязвимости, включает ликбез по уязвимостям и в том числе то, как они классифицированы и упорядочены. Затем вы изучите, как аналитики безопасности находят известные уязвимости и какой софт они применяют.

Глава 2, Рассматриваем угрозы нулевого дня в реальности, подробно разбирает жизненный цикл уязвимости и знакомит с уязвимостями нулевого дня, их эксплуатацией и разнообразными опасностями исходящими от них. Немаловажно, что вы освоите нужную терминологию и то, как исследования безопасности прямо или косвенно создают эти риски и узнаете о дилеммах, как же раскрывать результаты таких исследований.

Глава 3, Исследование уязвимостей - начнем с проверенных стратегий, описывает исследование безопасности, применяемые стратегии, то как выбирать цели и использовать тестовые пакеты и инструменты, которые как правило применяются для исследования.

Глава 4, Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности, информирует вас об определениях типов раскрытия уязвимостей, которые вы можете использовать и стратегиях, которые вы можете применить для преодоления различных препятствий, которые встанут перед вами в процессе раскрытия.

Глава 5, Публикация уязвимостей – публикуем свою работу в базах данных, тщательно разъясняет процедуру публикации уязвимостей и используемые при этом методы, дает конкретные примеры установившихся практик, которые помогут вам уверенно поделиться вашим исследованием с миром.

Глава 6, Арбитраж уязвимости – что пошло не так и кто может помочь, дает определение кто такие арбитры, как их найти и воспользоваться наилучшим образом, когда во взаимодействии с вендором все пошло не так.

Глава 7, Независимая публикация уязвимостей, всесторонне описывает, что такое независимое раскрытие информации, каковы риски его применения и как избежать проблем с законом, с которыми исследователи обычно сталкиваются в лице враждебно настроенных вендоров.

Глава 8, Изучаем примеры из жизни – подробно разбираемся в удачных (и неудачных) отчетах об исследовании, раскрываем реальные истории от исследователей уязвимостей, показываем вам, как могут развиваться сценарии и как исследователи справлялись с этими трудностями.

Глава 9, Взаимодействие с исследователями в области безопасности – руководство для вендоров, сместим фокус и расскажем вендорам о читателях этой книги, намереваясь дать поставщикам ПО и технологическим лидерам инструменты и информацию, которую они могут использовать для построения хороших отношений при неизбежном контакте, который состоится с исследователями.

Глава 10, Шаблоны, ресурсы и заключительные наставления, делимся шаблонами тестов и методов их использования, шаблонами связи с вендорами, шаблонами раскрытия CVE, шаблонами организации рабочего процесса и наконец словами поощрения тем, кто применяет эту информацию для совершенствования мира технологий, исследуя и раскрывая уязвимости.

Получение максимальной отдачи от этой книги

Для получения максимальной отдачи от этой книги, мы ожидаем от начинающего, среднего уровня знаний о различных существующих техниках, разработанных и поддерживаемых пользователями и бизнесом. Однако, чтобы применить стратегии и провести исследование, которое будет представлено в виде отчета, мы ждем продвинутых и возможно специализированных знаний об используемых техниках раскрытия уязвимостей безопасности, определенных во фреймворках, рассмотренных в этой книге.

Загрузка цветных изображений

Мы также предоставляем PDF-файл, с цветными изображениями скриншотов и диаграмм использованных в этой книге. Вы можете скачать их здесь: https://packt.link/1iZ9q.

Используемые соглашения

Есть некоторое количество соглашений в тексте, используемых на протяжении этой книги.

`Код в тексте`: показывает в тексте, слова программного кода, наименования таблиц баз данных, наименования каталогов, имена файлов, расширения файлов, пути, подопытные URL, пользовательский ввод и приглашения в Twitter. Вот, произвольный пример: “Смонтируйте загруженный файл образа диска **`Web-Storm-10*.dmg`**, как диск в вашей системе.“

Жирный шрифт: показывает новые термины, какие-то ключевые слова или слова, на которые нужно обратить внимание на экране. К примеру слова в меню или диалоговых окнах, показаны **жирным шрифтом**. Пример: “Выберите **Система** в окне **Администрирование**.“

> *Подсказки или важные замечания.*
>
> *Выглядят, как это.*

Свяжитесь с нами

Отзывы наших читателей, всегда приветствуются.

Отзывы общего характера: Если у вас есть вопросы о каких-либо аспектах этой книги, пишите нам на customercare@packtpub.com и укажите название книги в тексте своего сообщения.

Опечатки: Хотя мы подошли со всей тщательностью, к обеспечению корректности нашего контента, ошибки могут случаться. Если вы нашли ошибки в этой книге, мы будем благодарны, если вы сообщите нам об этом. Пожалуйста, посетите www.packtpub.com/support/errata и занесите их в форму.

Пиратство: Ну, вы поняли...

Если вы заинтересованны в том, чтобы стать автором: Если есть тема, в которой вы являетесь экспертом и вы заинтересованы либо в написании, либо в ином вкладе в какую-нибудь книгу, пожалуйста посетите authors.packtpub.com.

Делитесь своими мыслями

Раз уж, вы читаете ***Справочник исследователя уязвимостей***, мы бы с удовольствием послушали ваши мысли! Пожалуйста, посетите страницу этой книги на Амазон и поделитесь своим отзывом. Ваш обзор, важен для нас и технического сообщества и поможет нам быть уверенными, что мы поставляем контент отменного качества.

Часть 1 – Основы исследования уязвимостей

В этой части вы узнаете об основах уязвимостей и процесса их исследования. Вы узнаете о разнообразных типах уязвимостей и различных эксплоитах. Вы также узнаете об инструментах и техниках, которые применяют исследователи безопасности, чтобы находить и анализировать уязвимости. Вооружившись этими знаниями, вы облегчите свой путь становления исследователя безопасности.

Эта часть, включает следующие главы:

- *Глава 1, Знакомство с понятием уязвимости*
- *Глава 2, Рассматриваем угрозы нулевого дня в реальности*
- *Глава 3, Исследование уязвимостей - начнем с проверенных стратегий*

Глава 1. Знакомство с понятием уязвимости

Уязвимость это характеристика чего-либо, что делает это что-то восприимчивым к повреждению или риску такового. Технологии однозначно, не защищены от уязвимостей. Например в 60-х годах в Соединенных Штатах, AT&T наблюдала один из первых широко распространившихся случаев эксплуатации уязвимостей в технологии. Люди выяснили , что они могли бы нарушить работу телефонных систем и обойти оплату за сервис, проигрывая в трубку телефона звук определенной тональности. Хакеры научились этим техникам потому, что они разобрались, как функционируют системы и могли ли в них оказаться изъяны, которые они смогли бы использовать непосредственно с выгодой для себя. Сегодня, такой же образ мышления и активность по испытанию технологий на прочность, используя их недокументированные возможности – цветут махровым цветом.

По всему миру, люди обнаруживают и эксплуатируют уязвимости, найденные в программном обеспечении. Софт управляет современным миром, едва ли не во всех областях нашей деятельности. Исследователи безопасности, могут найти программы и следовательно уязвимости, в простеньких приложениях, которыми вы пользуетесь в своем телефоне, бизнес-приложениях управляющих коммерцией, в устройствах применяемых в больницах для спасения жизней и промышленных контроллерах, которые обеспечивают общественные нужды. Эти уязвимости могут быть использованы для киберпреступлений, чтобы нарушить вашу приватность, повредить инфраструктуру и создать угрозы национальной безопасности. К несчастью, факты эксплуатации уязвимостей безопасности, могут быть сложны в обнаружении, особенно в случае нераскрытых уязвимостей.

Невзирая на растущее число отчетов об уязвимостях и информации в целом, люди продолжают становиться жертвами этих угроз, которые все дороже обходятся и в финансовом плане и в плане конфиденциальности. Итак, что же делается по этому поводу? В настоящей главе мы изучим, как эти угрозы реализуются, рассматривая ряд фундаментальных концепций.

Мы тщательно изучим следующее:

- Уязвимости ПО и CIA-триада
- Как уязвимости ПО классифицированы и упорядочены
- Сканеры уязвимостей, которые профессионалы часто используют для обнаружения и устранения дыр в ПО
- Ряд распространенных типов уязвимостей безопасности, как правило обнаруживаемых в ПО
- Жизненный цикл уязвимости и его воздействие на программное обеспечение

К концу этой главы, вы должны будете разобраться, что такое уязвимости, как они попадают в программы, как уязвимости упорядочены и ранжированы, как искать уязвимые компоненты ПО и что такое жизненный цикл уязвимости. Погнали!

Введение в уязвимости ПО

Преступникам нужны ваши данные. Одним из множества способов, которыми они могут добыть такие данные, является поиск и эксплуатация уязвимостей в программных продуктах, в которых ваши данные хранятся. С тех пор, как в нашу жизнь вошел интернет, произошло масса скандальных случаев, когда преступники эксплуатировали уязвимости, в результате чего была похищена банковская информация, медицинская информация, а так же корпоративные и государственные тайны.

Перед тем, как мы рассмотрим уязвимости, Неплохо бы разобраться, что же такое программное обеспечение. Лучше всего сформулировать это, как то, что программа - это набор инструкций, которые говорят компьютеру, что нужно делать. Программное обеспечение может включать разные инструкции и программироваться для каких-то устройств или оборудования. Эти инструкции делают возможным и позволяют пользователям, выполнять такие задачи, как обработка текста, интернет-серфинг,

использование социальных сетей типа Вконтактика и прочей хрени. ПО необязательно ограничивается компонентами, предназначенными исключительно пользователю в виде того, что мы называем приложениями. ПО разработано и для функционирования аппаратных компонентов и микроконтроллеров. Представьте современные автомобили и бытовую электронику – если вы покупали, что-либо по-видимому связанное с компьютерными технологиями в 21 веке, скорее всего программное обеспечение было написано и крутится в этих устройствах.

Уязвимости ПО, проще всего определить, как изъяны и недостатки софта. Представьте уязвимости, как логические ошибки или лазейки в инструкциях, которые мы передаем нашим компьютерам. Эти ошибки или лазейки могут представлять угрозу трем критическим областям, в отношении уязвимостей ПО. В индустрии информационной безопасности об этом обычно говорят, как о **CIA-триаде**. CIA это аббревиатура для **конфиденциальности, целостности** и **доступности (Confidentiality, Integrity, Availability)**, как проиллюстрировано на следующем рисунке:

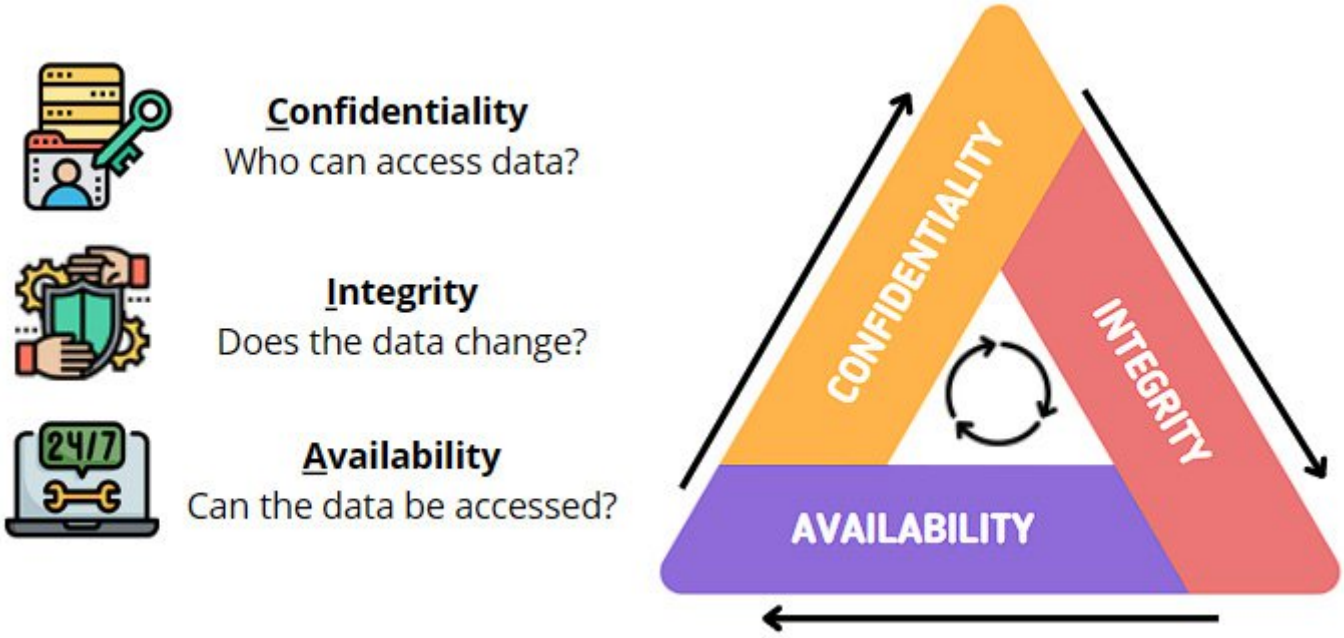


Рис. 1.1 - CIA-триада

Обсудим эти факторы, потому что они являются важнейшими для понимания, как уязвимости упорядочены по серьезности угроз.

CIA-триада. Конфиденциальность, целостность, доступность.

Конфиденциальность, это область, в которой данные сохранены в секрете и закрыты от широкого доступа. Уязвимости, которые могут атаковать это направление, часто проявляются в возможностях доступа неавторизованных пользователей к просмотру или экспорту информации, к которой они не должны иметь доступа. Эксплуатация этого вектора атаки, часто раскрывает данные, которые не были бы раскрыты другими способами.

Целостность проверяет, не изменены ли данные, по сравнению с теми, что были отправлены в ПО изначально. Уязвимости угрожающие целостности, часто имеют результатом изменение данных. Примите во внимание важность информации, относящейся к вашим банковским операциям или информации о вашем здоровье и насколько важна ее целостность. Эксплуатация с этой стороны, часто включает изменение значений по сравнению с оригинальными нацеливаясь на получение какой-либо выгоды для атакующего.

Доступность проверяет, есть ли доступ к данным. Уязвимости в отношении доступности, часто имеют результатом невозможность работы с приложением или отключение каких-то его возможностей. Вредное воздействие от эксплуатации таких уязвимостей нацелено на нарушение доступа или возможности использования данных и систем. В результате их эксплуатации, чаще всего, легальные пользователи не имеют возможности получить доступ к своим данным.

Когда мы размышляем об уязвимостях, важно понимать как они угрожают одному или более из этих факторов, в части причинения ущерба программному обеспечению. Итак, сейчас, когда мы разобрались, что представляют три ключевых понятия в процессе сравнения и измерения уязвимостей в ПО, как их упорядочить и расставить приоритеты?

Систематизация вероятных угроз

Чтобы помочь нам упорядочить и выстроить уязвимости по степени важности, исследователи безопасности разработали методы и системы для расчета уровня опасности уязвимостей. Наиболее распространенный метод расчета, с которым вы встретитесь это **Единая Система Оценки Уязвимостей** (**Common Vulnerability Scoring System - CVSS**). CVSS это открытая спецификация рекомендующая, что-то вроде языка описания степени серьезности уязвимостей, и их направленности на части CIA-триады. В дополнение, система оценки может быть использована, как разновидность калькулятора, который определяет по основным ключевым метрикам, насколько велик возможный ущерб от уязвимостей. Расчет, учитывает следующие факторы:

- Угроза конфиденциальности
- Угроза целостности
- Угроза доступности
- Как уязвимость эксплуатируется
- Сложность эксплуатации
- Какой доступ требуется для эксплуатации уязвимости
- Требуется ли для эксплуатации взаимодействие с пользователем

Вот набор данных, обработанный в CVSS-калькуляторе (см. рис. 1.2) и он обычно прикрепляется к уязвимостям в базе уязвимостей:

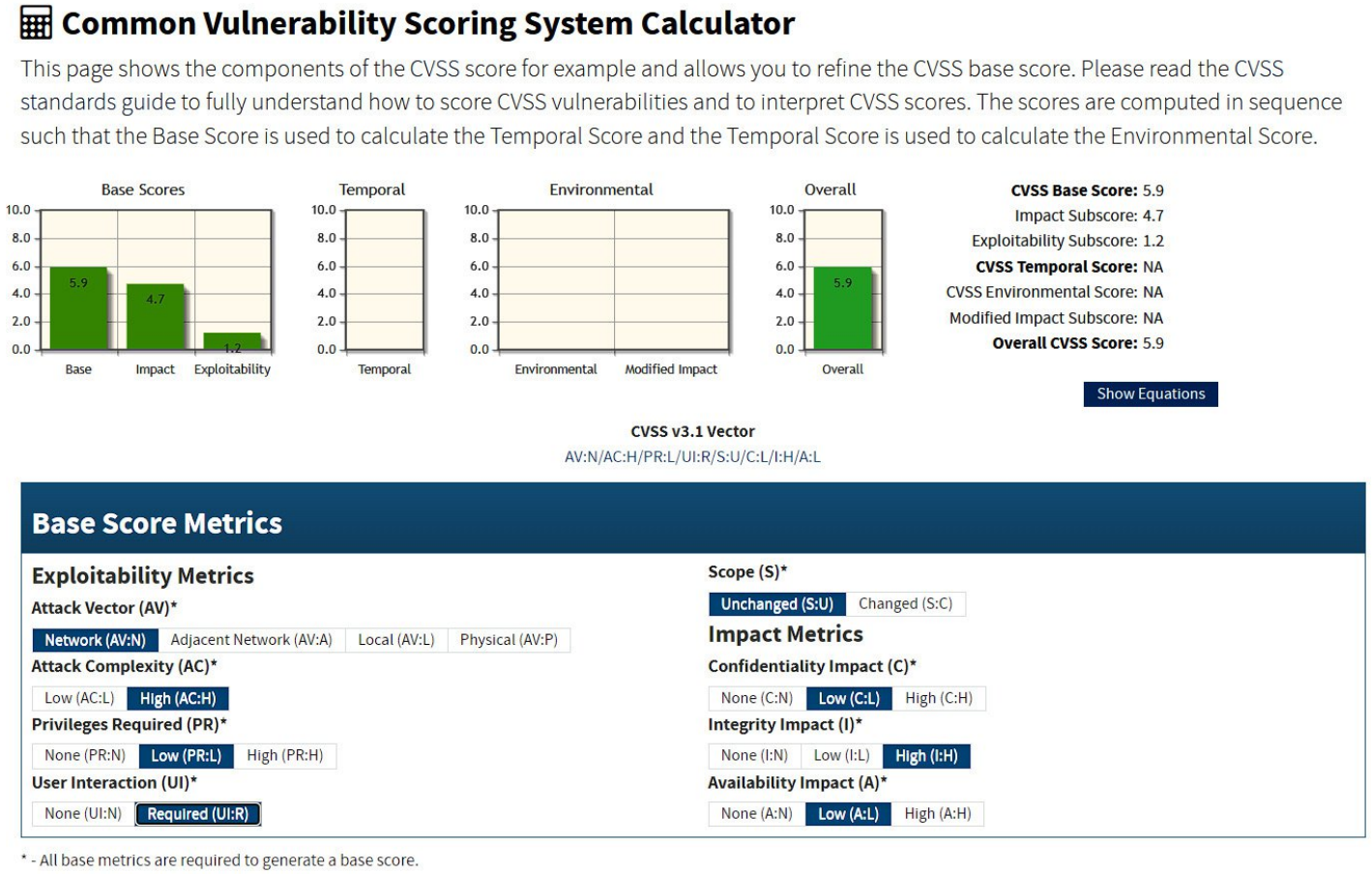
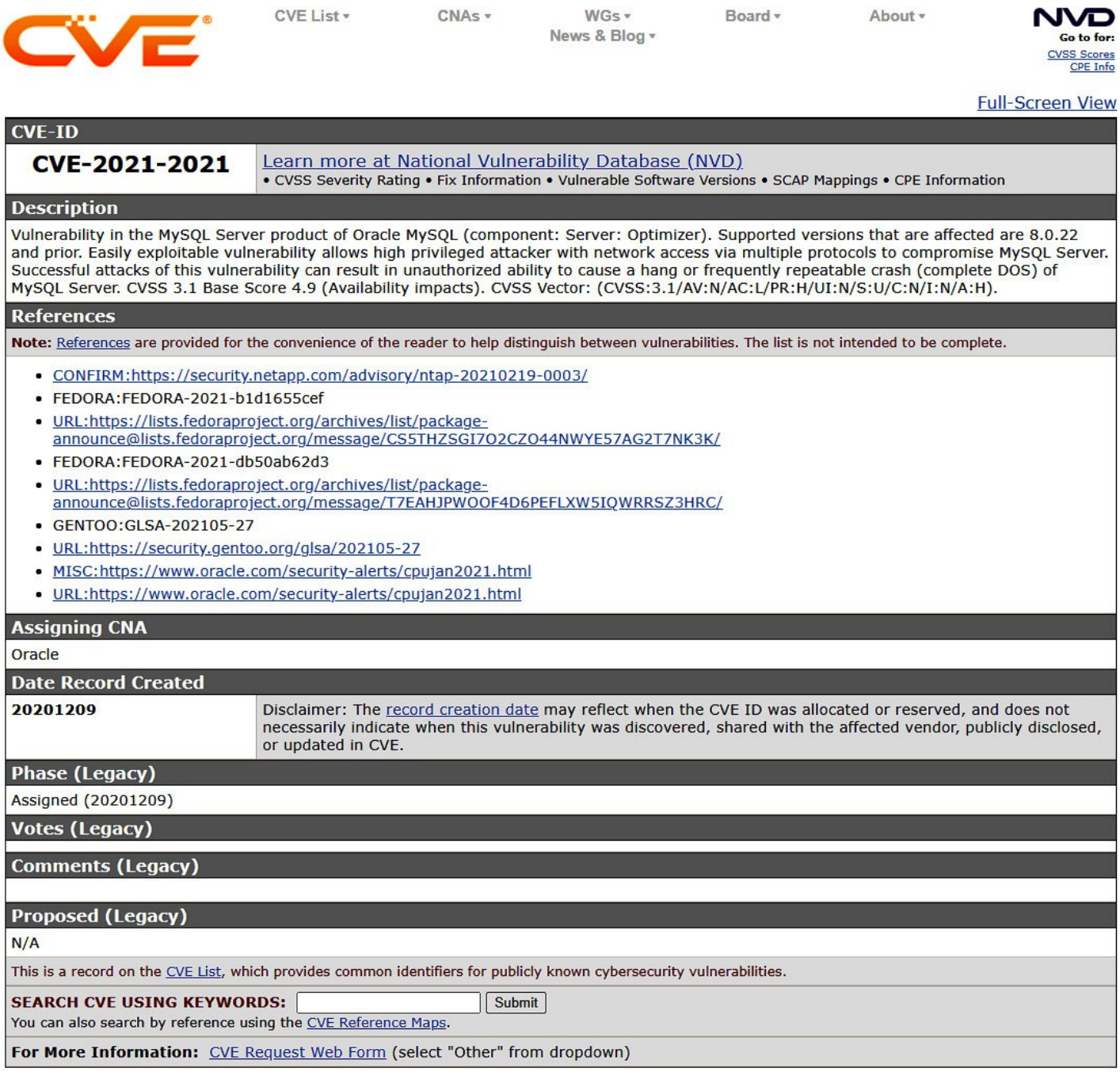


Рис. 1.2 - Пример CVSS-калькулятора

Несмотря на то, что мы тщательно изучим различные базы в дальнейших главах это важно, чтобы быть в курсе о наиболее распространенном формате описания уязвимостей в списке MITRE **Common Vulnerabilities and Exposures** (**CVE**) – **Известные Уязвимости и Дефекты**. Уязвимостям ПО, которые публично раскрыты, присваиваются сквозные номера в списке CVE. Списки CVE поступают в крупные базы уязвимостей безопасности, такие как **National Vulnerability Database** Соединенных Штатов **(NVD)** и многие другие. Каждая уязвимость в списке, называется просто CVE. Каждой уязвимости, присвоен номер CVE ID, оценка (баллы) CVSS и подробности относящиеся к уязвимости, такие как эксплуатация, исследование, краткое описание уязвимости и любая другая связанная информация. Типичная запись CVE, будет выглядеть как-то так:



CVE List ▾ CNAs ▾ WGs ▾ Board ▾ About ▾
News & Blog ▾

NVD
Go to for:
CVSS Scores
CPE Info

Full-Screen View

CVE-ID

CVE-2021-2021

Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

Vulnerability in the MySQL Server product of Oracle MySQL (component: Server: Optimizer). Supported versions that are affected are 8.0.22 and prior. Easily exploitable vulnerability allows high privileged attacker with network access via multiple protocols to compromise MySQL Server. Successful attacks of this vulnerability can result in unauthorized ability to cause a hang or frequently repeatable crash (complete DOS) of MySQL Server. CVSS 3.1 Base Score 4.9 (Availability impacts). CVSS Vector: (CVSS:3.1/AV:N/AC:L/PR:H/UI:N/S:U/C:N/I:N/A:H).

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- CONFIRM:https://security.netapp.com/advisory/ntap-20210219-0003/
- FEDORA:FEDORA-2021-b1d1655cef
- URL:https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/CS5THZSGI7O2CZO44NWYE57AG2T7NK3K/
- FEDORA:FEDORA-2021-db50ab62d3
- URL:https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/T7EAHJPWOOF4D6PEFLXW5IQWRRSZ3HRC/
- GENTOO:GLSA-202105-27
- URL:https://security.gentoo.org/glsa/202105-27
- MISC:https://www.oracle.com/security-alerts/cpujan2021.html
- URL:https://www.oracle.com/security-alerts/cpujan2021.html

Assigning CNA

Oracle

Date Record Created

20201209

Disclaimer: The record creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

Phase (Legacy)

Assigned (20201209)

Votes (Legacy)

Comments (Legacy)

Proposed (Legacy)

N/A

This is a record on the CVE List, which provides common identifiers for publicly known cybersecurity vulnerabilities.

SEARCH CVE USING KEYWORDS: Submit
You can also search by reference using the CVE Reference Maps.

For More Information: CVE Request Web Form (select "Other" from dropdown)

Рис. 1.3 – Пример записи CVE, CVE-2021-2021

Каждый год, тысячи CVE добавляются в список MITRE CVE, и затем отправляются в NVD. Эти уязвимости могут угрожать программам, сервисам, железу и не только.

Примечание

Списки CVE, записи CVE и производные наборы данных не отслеживают уязвимости для программных продуктов типа ***ПО как услуга*** *(****software as a service - SaaS****). Для уязвимостей из списков и наборов данных, упомянутых выше, ПО о котором идет речь, должно быть продуктом, контролируемым пользователем. Так, например Google Docs это офисный пакет, который главным образом функционирует, как веб-приложение, не содержащее элементов установленных под контролем пользователя. Следовательно Google Docs не подходит для присваивания CVE и регистрации в базах уязвимостей содержащих CVE.Аналогичным образом, офисные приложения вроде Microsoft Office, сейчас имеют схожие облачные сервисы, однако они также предоставляют установочные пакеты для “толстых“ клиентов, таких как Microsoft Word. В этом примере, версия Microsoft Word установленая на компьютер – подходит для публикации в списке CVE, за исключением облачной версии.*

Теперь мы знаем, как находить информацию об уязвимостях и как этот поиск организован. Итак, как же специалисты по безопасности, задействуют эти средства?

Осваиваем сканеры уязвимостей ПО

Множество уязвимостей, которые сегодня эксплуатируются – являются известными, зарегистрированы в списке CVE, имеют связанную оценку уязвимости (баллы CVSS), CVE-ID и описание прочих деталей. На момент написания этих строк, более 20 миллионов уязвимостей были зарегистрированы в этом списке. Несмотря на то, что вполне возможно собрать информацию об используемом ПО и затем, использовать перекрестные ссылки, пробивая цель по списку CVE – на деле, для регулярного применения, это малоэффективно и непрактично.

Чтобы решить эту проблему, разработчики программного обеспечения, создали решения для обнаружения этих уязвимостей. Приложения этого типа, обычно относят к сканерам уязвимостей. Несмотря на то, что вам могли рассказать о каком-нибудь полном энтузиазма производителе ПО и свежих фишках в их сканере, эти сканеры как правило пассивны, в том плане, что не обнаруживают неизвестных уязвимостей. Вместо этого, они только идентифицируют известные дефекты посредством скриптов, ищущих информацию о программном обеспечении. Сканеры сравнивают версии ПО в списках опубликованных уязвимостей, чтобы определить, есть ли известные уязвимости в установленом ПО.

Команды по разработке ПО, имеют возможность использовать эти инструменты для определения версий железа и софта, и даже уязвимого сетевого оборудования, такого как файерволы. Дополнительно, системы сканирования уязвимостей, дают системным администраторам знать, какое ПО нуждается в установке патчей безопасности во избежание эксплуатации уязвимостей. Современные системы, как правило представляют это в удобном виде, выделяют уязвимости с более высокой опасностью, предоставляют отчеты и дашборды, которые могут предназначаться различным аудиториям.

Сканеры уязвимостей, позволяют специалистам по кибербезопасности идентифицировать, классифицировать и располагать риски в соответствии с приоритетом. Есть много инструментов сканирования уязвимостей, помогающих их идентифицировать. Итак, какие же есть распространенные инструменты сканирования уязвимостей?

Распространенные инструменты сканирования на уязвимости

В этом разделе приводятся некоторые распространенные инструменты сканирования уязвимостей, применяемые практиками в сфере безопасности. Этот список не полон, тем не менее он должен дать вам серьезный вводный курс по основам этих инструментов и поможет вашему пониманию общих приемов, с которыми вы скорее всего столкнетесь, когда начнете исследовать и применять эти инструменты.

Nmap

Nmap или **Network Mapper** это свободный инструмент сканирования сети с открытым исходным кодом, созданный программистом Гордоном Лайоном. Взгляните на образец вывода Nmap, который он как правило выдает, когда пользователь запускает сканирование:

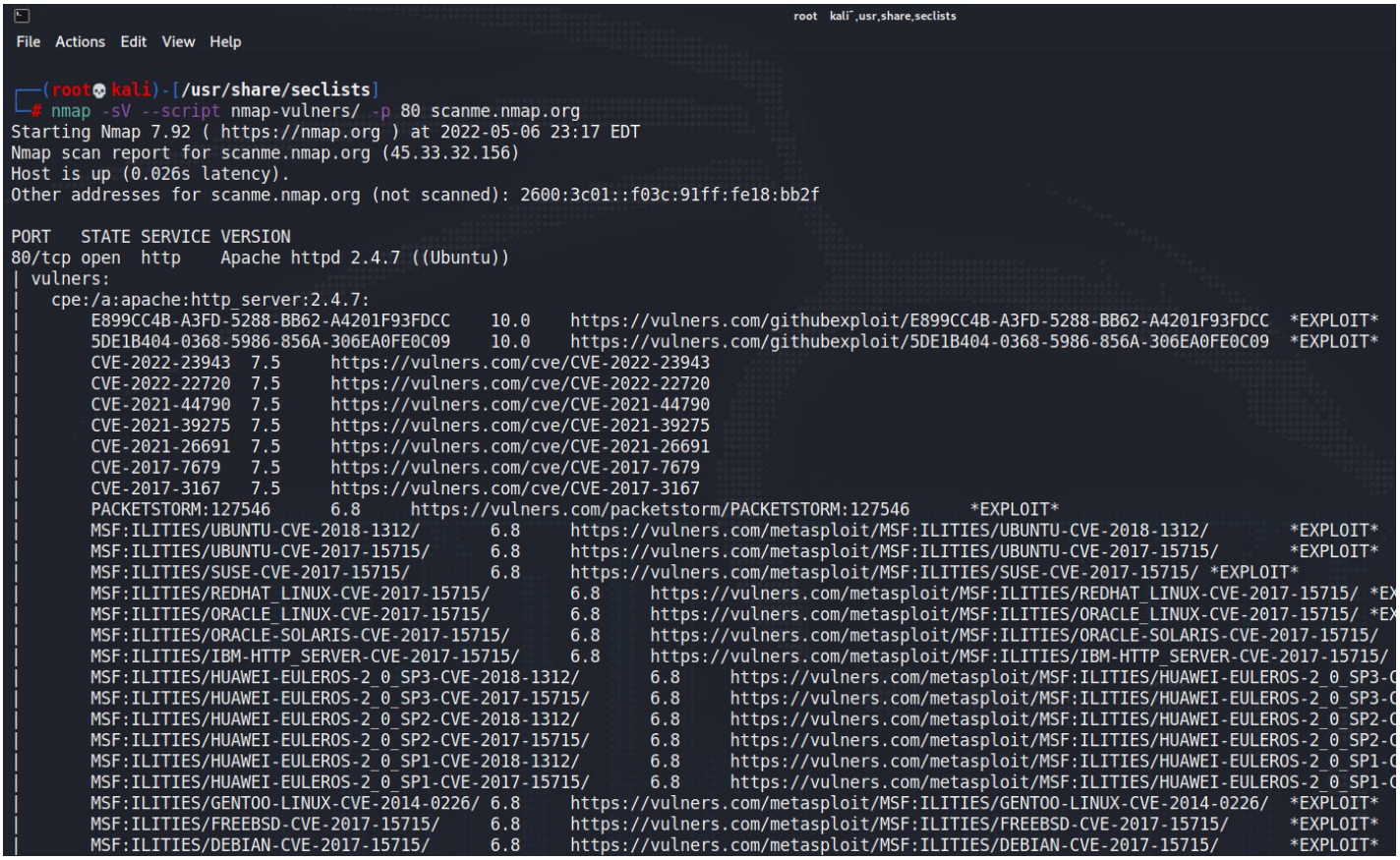


Рис. 1.4 – Скриншот вывода Nmap с использованием скрипта Nmap-Vulners

Nmap может сканировать диапазон различных сервисов на целевом хосте. Он может быть использован для сканирования одного или множества хостов в локальной сети или интернете. Сказать, что это средство сканирования уязвимостей, будет небольшим упрощением навороченного функционала Nmap, тем не менее он часто используется со специализированными скриптами, для проверки на уязвимости. Nmap получает данные ввода командной строки, а уже скрипты посылают запросы, чтобы выяснить, есть ли что-нибудь уязвимое.

OpenVAS

Open Vulnerability Assessment System, более известный как **OpenVAS** - свободный инструмент с открытым исходным кодом, отпочковавшийся от коммерческого продукта Nessus. Интерфейс OpenVAS, изображен на следующем рисунке с оформлением Greenbone Security Assistant:

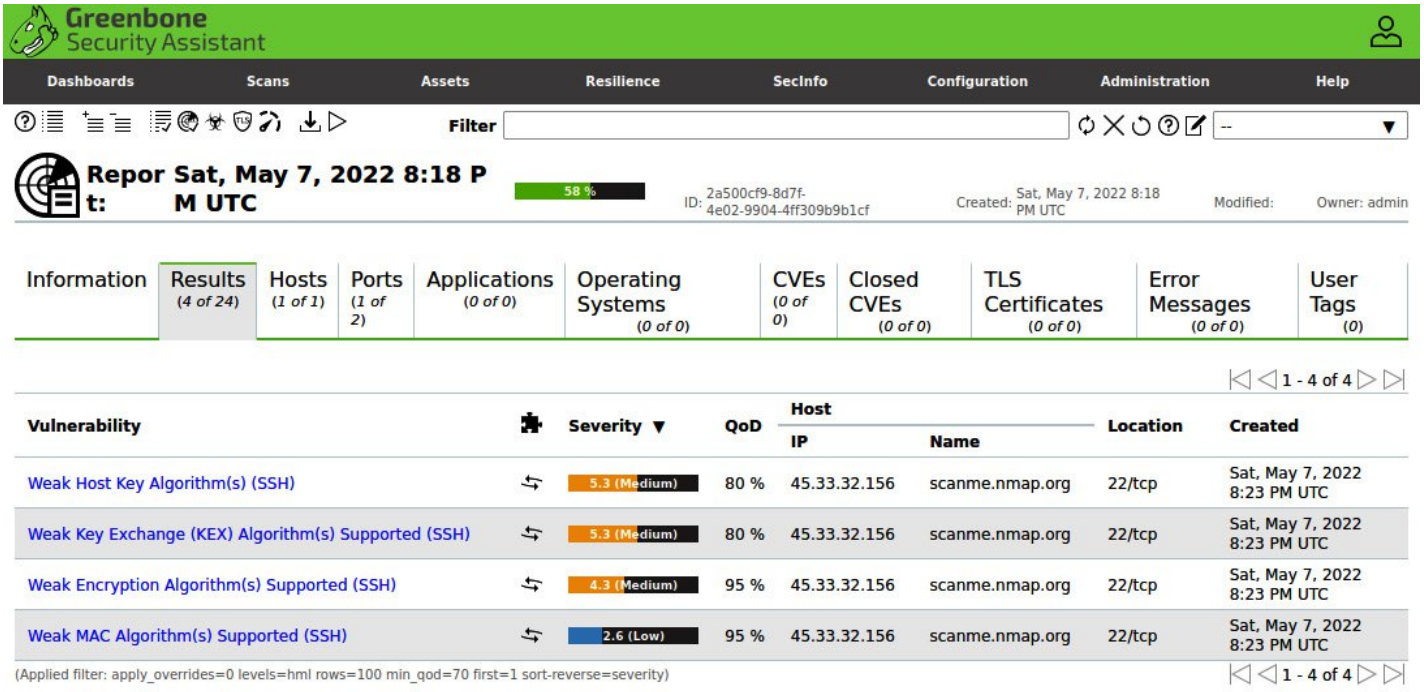


Рис. 1.5 – Дашборд OpenVAS Greenbone Security Assistant

OpenVAS помогает системным администраторам, сканировать на известные уязвимости различные платформы и операционные системы. Хотя это и опенсорс, это отлично поддерживающийся инструмент. Однако критики OpenVAS и прочие злопыхатели, часто указывают на то, что поддержка для корпоративного сектора скудна и извлечение результатов сканирования из системы может быть довольно сложным.

Nikto

Nikto, представляет собой инструмент с интерфейсом командной строки с открытым исходным кодом, который умеет сканировать веб-серверы на уязвимости. Его применение, ограничивается строго веб-серверами и будет требовать доступа к веб-сервисам для использования. Вывод результатов и интерфейс, схожи с Nmap, как вы видите на следующем рисунке:

```
└─# nikto -h https://nmap.org
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          45.33.49.119
+ Target Hostname:    nmap.org
+ Target Port:        443
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /CN=insecure.com
                   Ciphers:  ECDHE-RSA-AES128-GCM-SHA256
                   Issuer:   /C=US/O=Let's Encrypt/CN=R3
+ Start Time:         2022-05-06 23:09:57 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.6 (CentOS)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The site uses SSL and Expect-CT header is not present.
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a differe
```

Рис. 1.6 – Скриншот вывода Nikto в командную строку

Все таки, это замечательный инструмент, чтобы проверять веб-сайты и обрести понимание, почему они могли оказаться уязвимыми. Как и в Nmap, объем выдаваемых результатов ограничен, и недоброжелатели указывают на случаи с высоким уровнем ложных срабатываний.

Nessus

Nessus, одно из самых распространенных решений по оценке уязвимостей во всей индустрии информационной безопасности. Простой в использовании интерфейс и дашборд, выглядят так:

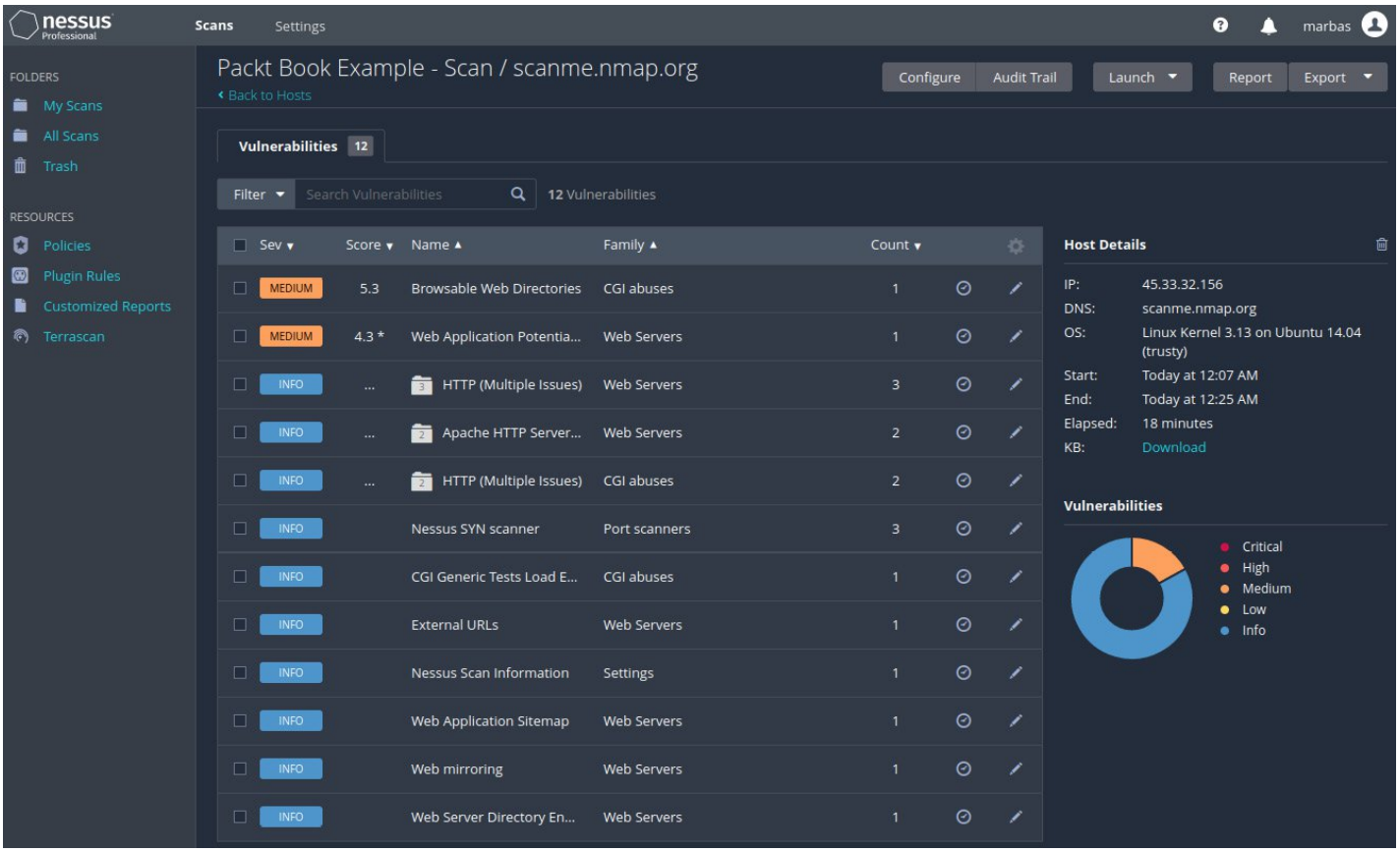


Рис. 1.7 – Пример сканирования Nessus

Nessus работает также, как и OpenVAS, позволяя оператору выявлять уязвимости и проводить аудит конфигурации, клиентских компьютеров, веб-серверов и т. п. Это коммерческий продукт, так же предлагающий, удобный экспорт отчетов с полученными данными и разнообразные плагины, расширяющие функциональность инструментов сканирования.

Rapid7 Nexpose

Rapid7 Nexpose, это сканер уязвимостей, который умеет сканировать различные цели на широкий спектр уязвимостей и ошибок конфигурации. Посмотрите на дашборд, чтобы увидеть отличия, по сравнению с Nessus на рисунке:

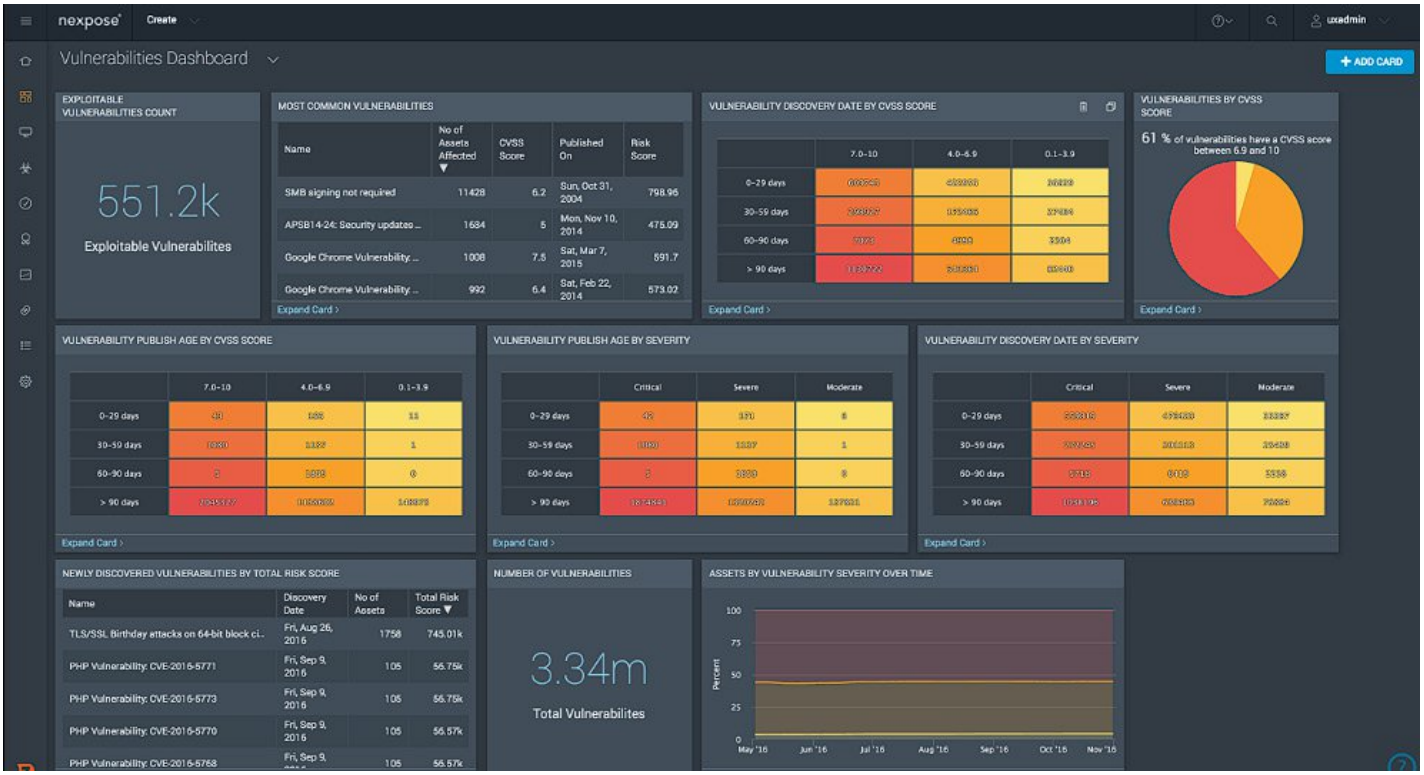


Рис. 1.8 – дашборд Rapid7 Nexpose

Он выдает операторам результаты на дашборде и ссылки на связанные инструменты Rapid7, используемые пентестерами.

Qualys Vulnerability Management

Qualys Vulnerability Management – коммерческий сканер уязвимостей. Аналитики безопасности используют этот инструмент так же, как и остальные сканеры, упомянутые ранее. Это помогает оценивать серьезность уязвимостей, располагать их в порядке приоритета и исправлять (установкой патчей или изменением конфигурации). Он работает так же, как OpenVAS, Nessus, и Rapid7 Nexpose.

> *Примечание*
>
> *Как бы ни были мощны сканеры уязвимостей, они строго ограничены в своих возможностях, поиском в собственных базах известных уязвимостей. Если там нет каких-нибудь скриптов, подтягивающих свежие базы, уязвимости могут быть не распознаны.*

Теперь, когда мы рассказали про сканеры уязвимостей, введем вас в курс дела, насчет распространенных видов уязвимостей ПО, которые вы увидите в их естественной среде обитания.

Рассматриваем распространенные типы уязвимостей в ПО

Несмотря на то, что исчерпывающий список уязвимостей ПО, выходит за рамки этой книги, мы покажем некоторые из наиболее распространенных уязвимостей, которые периодически находятся и эксплуатируются. Мы разобьем их на две отдельные группы – веб-приложения и клиент-серверные приложения.

Веб-приложения

Веб-приложения являются одной из самых распространенных точек входа, через которые организации оказываются взломаны. Одним из самых эпичных взломов всех времен, является утечка данных Equifax в 2017 году. Этот взлом случился потому, что у Equifax был уязвимый веб-сайт, который и подвергся эксплуатации. Эта эксплуатация уязвимости в программном обеспечении, была связана воедино с другими уязвимостями, и конечным результатом этих действий стала кража конфиденциальных данных. Организации обыкновенно имеют веб-сайты, которые предоставляют инструменты и приложения, помогающие им вести бизнес с их клиентами и другими организациями. Это, только несколько известных видов уязвимостей, найденных в этих популярных и повсеместно принятых технологиях:

Нарушения аутентификации и авторизации

Нарушения аутентификации и авторизации, одна из самых распространенных проблем, которые находят в современных веб-приложениях. В результате эксплуатации этой уязвимости, атакующие могут получить доступ к данным, доступ к которым не должен быть разрешен через действующий аккаунт веб-сайта или без аккаунта вообще. Хорошим примером нарушения аутентификации было бы, если условный злоумышленник смог обойти страницу логина веб-сайта для получения доступа и (или) манипуляции данными, которые ему не принадлежат. В случае нарушения авторизации, напротив, атакующий мог бы войти на сайт, используя учетные данные действующих аккаунтов, но после получения доступа к информации конкретного аккаунта, он смог бы манипулировать сессией приложения для доступа к данным аккаунтов других пользователей.

Кросс-сайт скриптинг

Кросс-сайт скриптинг (XSS) это техника, использующая внедрение злонамеренного содержимого в страницу веб-сайта, которое может быть использовано для запуска произвольного кода JavaScript в пользовательской сессии браузера. Атакующие делают это, выискивая в коде недостаточную проверку входных данных. Это может быть совершено, путем обнаружения на веб-сайтах мест, которые возвращают информацию обратно пользователям, или через постоянное хранилище или возвращенные значения, могущие быть представленными в URL. Есть три широко известных варианта этой уязвимости, которые мы вкратце рассмотрим:

- **Хранимый XSS** (**Persistent XSS**): Эта уязвимость эксплуатируется, когда пользователи загружают веб-сайты содержащие сохраненные объекты с вредоносным JavaScript. Для эксплуатации этой уязвимости, атакующие находят места в которых пользователь подтверждает некую информацию в веб-приложении путем сохранения значений, в виде разрешенного кода JavaScript, который затем, в свою очередь – отправляется обратно пользователю.
- **Отраженный XSS** (**Reflected XSS**): Эта разновидность уязвимости XSS, позволяет атакующим задействовать уязвимые входные данные через подстановку в браузер жертвы URL, содержащих вредоносный JavaScript. Обычно этот вид уязвимостей, распространен в поисковых функциях веб-сайтов.
- **XSS на основе DOM** (**DOM-based XSS**): Эта уязвимость появляется при недостаточной проверке при обработке входных данных в JavaScript, который использует элементы управления в окружении, называемом **Document Object Model** (**DOM**). DOM это способ отображения окон и элементов на веб-сайте, и управления потоками данных через эти объекты. Для эксплуатации этих уязвимостей, атакующим необходимо внедрить вредоносный JavaScript, который сможет злоупотребить механизмом обработки объектов.

SQL и NoSQL инъекция

SQL и NoSQL инъекция это тип уязвимости, использующая сформированные атакующим запросы к подключенной базе данных. Атакующие эксплуатируют этот вид уязвимостей, манипулируя способами, которыми данные передаются в формы и (или) в запросы к серверу. Результат успешной эксплуатации уязвимости, состоит в раскрытии данных, таких как пароли и персональная информация. В некоторых случаях, эти уязвимости могут эксплуатироваться способами, позволяющими злоумышленнику захватить основной сервер баз данных. Вариант этой уязвимости, описанный как NoSQL инъекция относится к похожим методам, используемым для эксплуатации не реляционных СУБД, подключенных к приложениям. Эти техники имеют схожие результаты, но выполняются совершенно разными способами.

Внедрение команд

Внедрение команд это тип уязвимости, очень похожий на SQL инъекцию. При внедрении команд однако, атакующие пытаются вставить значения в поля или процессы веб-приложения, которое в случае успешной эксплуатации, может запустить произвольные команды на сервере. Это дает атакующему возможность получить полный контроль над сервером и обеспечить себя точкой опоры в сетях, к которым в противном случае не было бы доступа.

Подделка межсайтовых запросов

Подделка межсайтовых запросов (**CSRF**) это уязвимость, которая будучи эксплуатируемой, может заставить пользователя, выполнить действия нужные злоумышленникам. Стандартные техники эксплуатации, включают отправку злоумышленниками пользователям вредоносных ссылок, которые выполнят поддельные запросы, если пользователь их откроет. Одним из ожидаемых последствий рассылки этих вредоносных ссылок потенциальным жертвам, являются действия по изменению пароля жертвы, адреса электронной почты или другой информации об аккаунте, чтобы приступить к его захвату.

Подделка запросов со стороны сервера

Подделка запросов со стороны сервера (**SSRF**) это тип атаки, которая эксплуатирует факт доверительных отношений, установленных между сервером и его клиентами. Она очень похожа на уязвимость CSRF, но фокусируется на сервере и его доверительных отношениях с клиентами и другими серверами, вместо конечных пользователей. Эксплуатация этого типа уязвимостей, может привести к аутентификации сервером запроса на повышение привилегий на нем самом или других доступных во внутренней сети ресурсах.

Изъяны бизнес-логики

Изъяны бизнес-логики это уязвимости, представленные недостаточным прогнозированием непредвиденного поведения или входных данных в алгоритме бизнес-процессов. Каких-то стандартных методов эксплуатации таких уязвимостей, разумеется нет. Эти уязвимости, не всегда следуют похожим шаблонам, потому-что разработчики приложений всегда проектируют логику, которая диктует как следует использовать, то или иное приложение, а любые атаки по шаблонам вытекающим из ошибок проектирования, должны быть досконально продуманы в каждом конкретном случае. Разработчики ПО, часто привносят такие уязвимости не обрабатывая непредвиденный ввод и ситуации, с которыми пользователи могут столкнуться.

Примечание

Превалирующие тенденции в безопасности веб-приложений, здорово изменились за последнее десятилетие. За самым свежим списком известных дыр в безопасности найденных в веб-приложениях, пожалуйста обратитесь к OWASP Top 10 list, который часто обновляется по следующей ссылке: https://www.owasp.org/www-project-top-ten.

Клиент-серверные приложения

В отличие от веб-приложений, которые демонстрируют восходящие и нисходящие тренды в дефектах ПО, традиционные клиент-серверные приложения, часто известные как толстые клиенты или полноценные приложения, значительно реже подвержены переменам. Такие приложения отличает, тот факт, что они преимущественно используются в какой-нибудь операционной системе и приложение как правило, должно быть установлено или сконфигурировано для запуска в этой ОС. Эти приложения, часто обеспечивают функциональность компонентов веб-приложений в серверных окружениях, однако они также дают возможность пользователям выполнять задачи локально, используя вычислительные ресурсы своих рабочих станций.

Раскрытие информации

Это происходит, когда приложениям требуется сохранять пароль, ключи API и персональные данные. Иногда такие подробности, оказываются сохраненными в виде обычного текста или зашифрованных файлов, к которым легко получить доступ путем декомпиляции приложений или расшифровке содержимого файлов.

Захват процесса (Process hijacking)

Когда приложения установлены в операционных системах, у них часто неправильно или небезопасно сконфигурированы права доступа к файлам приложения, сопутствующим сервисам, библиотекам и т. д. Атакующие могут злоупотребить такими ошибками конфигурации, захватив их компоненты, чаще всего проведя имперсонацию легального приложения или сессии пользователя.

Передача открытым текстом

Эти уязвимости, относятся исключительно к коммуникациям, которые приложение может устанавливать по сетевым протоколам. Как и в одной из предыдущих категорий раскрытия информации, приложения могут передавать информацию незашифрованной и содержащей секретные данные, которые могут оказаться похищены, если злоумышленник сумеет перехватить данные по сети между клиентом и его сетевым хостом.

Уязвимости веб-приложений

Иногда, традиционные приложения устанавливают веб-приложения, которые позволяют пользователям взаимодействовать с компонентами приложения, установленного локально. К этому разряду приложений, даже если приложение доступно локально, можно смело применять все пункты, ранее упомянутые в разделе “веб-приложения“.

Теперь, когда мы имеем хорошее представление какие уязвимости мы можем обнаружить в реальности, приступим к изучению типичного жизненного цикла уязвимости.

Изучаем жизненный цикл уязвимости ПО

Жизненный цикл уязвимостей не всегда линеен, тем не менее есть общие соображения, которые мы будем рассматривать в процессе детального изучения этапов этого цикла. Мы будем говорить о четырех стадиях, которые часто перекрываются и (или) начинают свое выполнение одновременно, как показано на следующем рисунке:

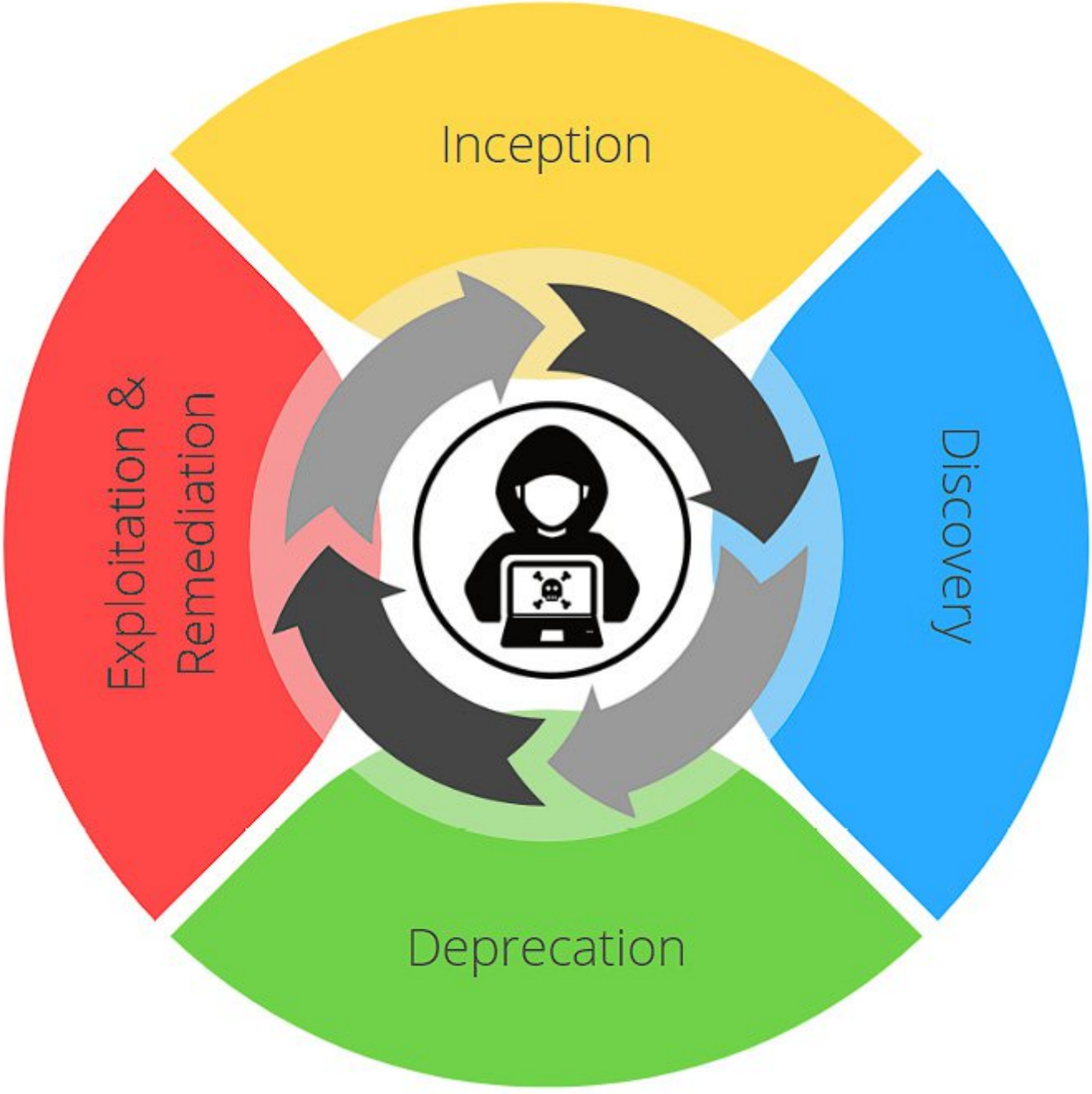


Рис. 1.9 – Стадии редко следуют в установленном порядке, но они согласованы на протяжении от возникновения до устранения уязвимостей

Итак, приступаем к детальному рассмотрению каждой стадии в нескольких следующих разделах.

Возникновение

Первый этап, который мы будем освещать это **возникновение**. В жизненном цикле в течение стадии возникновения, уязвимости заносятся в ПО в результате как злонамеренных, так и вполне легальных действий. Например, разработчики могли написать небезопасный код, системные администраторы, могли развернуть незащищенные конфигурации или злоумышленники могут целенаправленно встроить уязвимости и бэкдоры в ПО. Понимание времени появления уязвимости, является трудным из-за количества путей, которыми уязвимости могли быть внедрены.

Для того, чтобы ПО и системы применялись безопасным образом, инженеры должны знать как писать безопасный код и так же развертывать системы, или они должны получить рекомендации от специалистов по безопасности. Грамотность в сфере информационной безопасности, или точнее отсутствие таковой, могут быть причиной появления уязвимостей, но обычно это не единственный способствующий фактор. Разработка программного обеспечения и сопутствующие операции, довольно дороги и организации часто пытаются сэкономить деньги, посредством различных действий подразумевающих ускорение выпуска продукта. В результате, разработчиков ПО и инженеров, ответственных за эксплуатацию часто не хватает на объем всех проектов, а продукты болтает между добавлением новых возможностей для привлечения новых клиентов и удовлетворением старых. Этот напряг, сочетается с недостаточной культурой информационной безопасности и (или) образования в организации, часто приводит к тому, что какие-то шаги пропускаются и безопасность откладывается в долгий ящик, чтобы продукт гарантировано вышел на рынок в срок. Запомните, безопасность всегда связана с расходами и эти расходы часто урезают, когда бизнесу нужно побыстрее пропихнуть приложения или новые плюшки покупателям.

Еще одним фактором, который следует учитывать, является сложность программных систем и их развертывания или поддержки инфраструктуры. Если система является сложной или запутанной, есть большая вероятность появления слабых мест в конфигурации или коде.

В конечном счете, злоумышленники вносящие уязвимости в код, не такая уж и редкость, как кто-то мог бы подумать. Это может принимать несколько различных форм. Атакующим часто помогает результат инсайдерской угрозы, вроде недовольного сотрудника, который продаст им доступ. В других случаях, ювелирно проведенные атаки, финансируемые правительством, часто добивались создания точки опоры в ПО компаний, эксплуатации которых требовалось добиться. В 2020 году, такая кампания сделала мировые заголовки, когда атакующие нацелились на бизнес ПО Orion, компании SolarWinds, и завершилась в виде CVE-2020-10148. Эту уязвимость спалили прошаренные типы и протроянили Orion, который потом тысячи раз разошелся по пользователям. Пока эти типы атак и создаваемые ими угрозы, сложны для измерения многие верят, что полноценные результаты от подобного типа уязвимостей, часто остаются незамеченными и имеют потенциально катастрофические последствия, в случае эксплуатации злоумышленниками.

Обнаружение

В течение фазы **обнаружения**, уязвимости находятся при проведении исследования безопасности, поскольку мотивированные участники пытаются разобраться, является ли нечто уязвимым. Они делают это используя разные методы проверки ПО на типы уязвимостей, упомянутых в соответствующем разделе выше. Причины для обнаружения, могут отличаться. Одни люди, просто смотрят на обнаружение уязвимостей, как на улучшение безопасности ПО, в то время как другие ищут пути эксплуатации программного обеспечения, подразумевая противоположные цели.

В зависимости от мотивации действующих лиц, стадия обнаружения предоставляет исследователям сделать выбор. Во-первых, им необходимо решить, каким образом и будут ли они вообще передавать сведения о своих уязвимостях ответственным лицам. Однажды обозначившись,

исследователи могли бы уведомлять поставщиков ПО непосредственно, в идеальном случае. Один раз проинформированный поставщик, может начать работу по устранению проблем на этапе **исправления**. К сожалению, раскрытие информации об уязвимостях и определение следующих шагов по исправлению, редко оказывается таким же простым или позволяющим рассчитывать на благодарность.

Во многих случаях об уязвимостях, никогда не сообщается поставщикам программного обеспечения. Стоит упомянуть, что решения о раскрытии информации (если они вообще принимаются) иногда принимаются сомнительным с этической точки зрения путем. Например, известны случаи, когда исследователи решили продать исследованные уязвимости, фирмам по информационной безопасности. Некоторые из этих фирм, специализируются на скупке уязвимостей и действующих эксплоитов. Такие фирмы могут купить эксплоиты для включения в свои комплексы инструментов в сфере безопасности, затем они в свою очередь продают их другим вендорам, вендорам оригинального ПО (в котором найдены уязвимости) или даже государствам, которые могут включить эти эксплоиты в свой кибер-арсенал.

Эксплуатация и исправление

Стадии **эксплуатации и исправления**, могут наступить вскоре после обнаружения. Эта парочка связана в нашем жизненном цикле и выглядит в зависимости от того, как развивались события, после обнаружения уязвимости.

В одних случаях, уязвимость обнаруживается и затем нашедшие уязвимость исследователи, информируют вендоров о своих находках. Вендоры создадут планы по исправлению или даже патчи для программного обеспечения, которые будут выложены для пользователей, чтобы закрыть уязвимость в безопасности.

Допустим информация или программные патчи выпущены, но они прямо не указывают на то, как работает уязвимость. В таких случаях, исследователи иногда пытаются провести реверс-инжиниринг уязвимости, основываясь на документации или патчах, вплоть до тщательного изучения с целью создания действующих эксплоитов. Это обычное дело в работе исследователей безопасности или исследовании популярных программных продуктов вроде ОС Windows. Например вместе с раскрытой уязвимостью CVE-2019-0708, широко известной, как BlueKeep, Microsoft выпустила выпустила патч в мае 2019. Исследователи отреверсили патч, с целью создать действующий эксплоит в июле, при этом основные эксплоит-киты, получили рабочий эксплоит позже в этом году, в течение сентября месяца.

В других случаях, некоторые баг-хантеры могут сразу выложить информацию об их уязвимостях на всеобщее обозрение – когда-то бесплатно, когда-то за некоторую плату, как ранее обсуждалось на этапе обнаружения. Уязвимости, которые публикуются сразу, могут лидировать (и лидируют) в широко распространенных случаях эксплуатации. Современные модели атак, обычно показывают, что массовая эксплуатация не особо отстает по времени от обнаружения уязвимостей и публикации информации о них. Злоумышленники часто сканируют публичные ресурсы в интернете на уязвимые версии ПО, которые попадают под открытую уязвимость. Подготовленные варианты эксплуатации, разработанные и испытанные, часто пополняют инструментарий хакера, чтобы эксплуатировать уязвимости.

Наверное одной из самых неприятных характеристик, всего этого, является то, как люди узнают об этих уязвимостях. Даже в идеальном случае, когда исследователь обращается к компании и сообщает им информацию об уязвимости, не распространяя ее по общедоступным каналам, вендор в итоге может выпустить патчи устраняющие проблему, без какого бы то ни было уведомления пользователей об уязвимом ПО и рисках, которые влечет использование устаревших версий. К несчастью, такая практика “втихую“ распространена в индустрии.

Утрата актуальности

Как и множество других жизненных циклов, закончится и этот. С течением времени, программное обеспечение и его уязвимые версии, будут выведены из эксплуатации. Патчи или исправления временно добавлены в новейшие версии ПО. В конце концов, старые уязвимости будут выпадать из поля зрения хакеров, сканеров уязвимостей и людей ищущих уязвимости.

Держите в уме, что процесс окончательного ухода уязвимости с горизонта, может занять много времени и старые уязвимости всегда следует обдумывать и принимать во внимание, во время анализа технологий. Например, известный баг Shellshock или CVE-2014-7169 в большинстве случаев находится чуть ли не через 10 лет после опубликования. Уязвимость, присутствовала в программном обеспечении более 30 лет до того, как была обнаружена и это ПО было широко развернуто по всей мировой технической экосистеме.

Другие обвиняемые в длительных циклах устранения, это не поддерживаемые, однако необходимые сервисы и системы. Обсудим ситуацию в индустрии здравоохранения. Госпитали и организации здравоохранения, будут раз за разом инвестировать в оборудование для анализа крови, установки применяющие рентгеновские лучи и прочие технологии визуализации, вроде МРТ. Эти вложения могут составлять миллионы долларов. К сожалению, поставщики ПО для здравоохранения не могут поддерживать программное обеспечение этих устройств, так долго, как госпитали планируют использовать это оборудование, в рамках сделанных инвестиций. Такие дыры, могут привести ситуации в которой вложения не защищены от эксплуатации, или с ней приходится справляться с помощью других сдерживающих факторов. В частности, нередко можно увидеть софт 15 - 20 летней давности на многочисленном оборудовании крупных технологических отраслей, таких как системы промышленной автоматики поддерживающие критическую инфраструктуру, развернутую в коммунальном хозяйстве.

Будем надеяться, что это помогло вам разобраться в жизненном цикле уязвимостей и том, чего вы можете ожидать, когда уязвимость в итоге, появится в программном обеспечении.

Итоги главы

В этой главе вы узнали об уязвимостях, чем они являются, какое место они занимают среди ключевых понятий в сфере информационной безопасности. В первую очередь, мы рассмотрели стандартные инструменты профессионалов индустрии, применяемые для сканирования и поиска уязвимых компонентов. Затем, мы обсудили распространенные уязвимости, найденные в разных типах приложений. И наконец, мы изучили типичный жизненный цикл уязвимости.

Освоение вами этих общих положений – база, которая будет иметь решающее значение, если вы собираетесь исследовать и представлять отчеты об уязвимостях сами. Вообще говоря, практикующие в области информационной безопасности, должны понимать важнейшие фундаментальные концепции окружающие уязвимости, как они ранжированы и систематизированы, какие инструменты применять, чтобы сканировать на них и какие предпринимаются действия, когда исследователи безопасности их обнаружат.

Теперь, когда вы вооружены этими фундаментальными знаниями об уязвимостях, вы подготовлены к дальнейшему обучению исследованиям и тому, как приступить к самостоятельному нахождению уязвимости.

В следующей главе, мы рассмотрим классификацию уязвимостей, называемых **уязвимостями “нулевого дня“** (**zero-days**) и покажем конкретные примеры их угроз уязвимому ПО.

Дополнительные материалы

- Список MITRE CVE – https://cve.mitre.org/cve/
- Калькулятор Системы Оценки Уязвимостей (CVSS) – https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator
- Каталог известных эксплуатируемых уязвимостей – https://www.cisa.gov/known-exploited-vulnerabilities-catalog
- Nmap – https://nmap.org/
- Nikto – https://cirt.net/Nikto2
- Nessus – https://www.tenable.com/products/nessus
- Rapid7 Nexpose – https://www.rapid7.com/products/nexpose/
- Qualys Vulnerability Management – https://www.qualys.com/
- OWASP Топ 10 уязвимостей веб-приложений – https://owasp.org/www-project-top-ten/

Глава 2. Рассматриваем угрозы нулевого дня в реальности

Летом 2016 года, хак-группа **The Shadow Brokers** (**TSB**) заявила, что они выкачали и завладели кибер-оружием использованным органами разведки Соединенных Штатов, через социальную медиа-платформу Твиттер. Они планировали продать свою добычу на каком-нибудь аукционе в даркнете. Однако после нескольких безуспешных попыток продажи, они начали сливать украденное кибер-оружие в свободный доступ. В апреле 2017 они выложили несколько эксплоитов для уязвимостей нулевого дня, один из которых пробивал CVE-2017-0144, чаще известную как **EternalBlue**. К концу апреля, эксплоиты для EternalBlue, заразили компьютерные системы по всему миру, по заявлениям некоторых исследователей, было атаковано более полумиллиона компьютеров.

К тому времени как CVE-2017-0144 было выпущено, Microsoft были опубликованы патчи безопасности (MS17-010), двумя неделями ранее. Компании, имеющие возможность установки патчей, сделали исправление угрозы безопасности приоритетом, однако многие не сделали или не смогли. В предыдущей главе, мы узнали о жизненном цикле уязвимости. Исследователи часто сталкиваются с дилеммой, когда исследование брешей в безопасности, нацелено на эксплуатацию программного обеспечения. Они могли отправить отчет об изъянах в безопасности производителю так, чтобы их исправили, продать свои исследования или сохранить уязвимости в тайне для возможного использования и разработки боевого софта. В случае с эксплоитом для EternalBlue, правительство Соединенных Штатов, как утверждают, сохранило тайну уязвимости и разработало изощренное кибер-оружие, чтобы использовать его позднее в кампаниях технологического шпионажа.

По единственному признаку в жизненном цикле уязвимости EternalBlue, она была классифицирована, как уязвимость нулевого дня, но только в течение того времени, когда она применялась и эксплуатировалась при неосведомленности производителя ПО. Несомненно все уязвимости – являются zero-day в какой-то момент, но ровно до того, когда уязвимость нулевого дня будет классифицирована, как именно такая. Так каковы же причины перестать считать уязвимость – zero-day? Эта глава нацелена прояснить такие неопределенности, обсудить реальные уязвимости и рассмотреть этическую сторону, относительно уязвимостей нулевого дня. В этой главе, мы погрузимся в следующие темы:

- Мы познакомимся с уязвимостями нулевого дня
- Мы изучим ключевую терминологию, которая поможет нам понять, как говорить о zero-days
- Мы проанализируем несколько реальных эксплоитов для уязвимостей нулевого дня
- Мы обсудим этику и дилеммы в условиях раскрытия уязвимостей нулевого дня

Вопросы, которые будут рассмотрены на протяжении следующих основных тем этой главы:

- Zero-days – что они такое?
- Разбор конкретных примеров уязвимостей нулевого дня
- Обсуждение этической стороны уязвимостей нулевого дня

До конца этой главы, вы станете понимать, что такое уязвимости нулевого дня и их угрозу современному бизнесу, а также узнаете об этических вопросах, относительно подхода к zero-days. Действуя таким образом вы будете на верном пути к поиску, анализу и раскрытию уязвимостей собственноручно.

Zero-days – что они такое?

Zero-day, иногда обозначаемые, как **0-day** – это термин применяемый для наименования недавно обнаруженных уязвимостей, эксплуатируемых уязвимостей и кибератак. Термин zero-day, пришел из расчета, используемого аналитиками безопасности для определения степени серьезности

уязвимости. Когда **окно уязвимости**, равняется нулю, это означает наивысший потенциальный риск. Это также подходящим образом описывает демонстрацию ситуации и осведомленность об этом производителей ПО, при условии отсутствия исправления для такого прокола. В результате, уязвимости нулевого дня, являются чрезвычайно опасными, потому что против них не существует 100% способов защиты.

Вообще говоря, терминология zero-day, часто употребляется некорректно и путанно. Итак, давайте разберемся в терминах, которые вы можете услышать в контексте уязвимостей нулевого дня, и что они означают.

Уязвимости нулевого дня

Уязвимости нулевого дня, это одна из видов уязвимостей, обнаруживаемых исследователями безопасности. Этот термин, обычно относится к обнаруженным уязвимостям, которые не были раскрыты в других отношениях – вендорам или широкой публике. Такие уязвимости, как правило известны некоему кругу лиц, до того, как поставщик ПО узнает об уязвимости. Уязвимости нулевого дня, обычно находятся исследователями безопасности и передаются поставщикам программного обеспечения так, чтобы они могли создать патчи, для исправления этих проблем. В некоторых случаях, исследователь предпочитает раскрыть zero-day широкой публике так, чтобы пользователи узнали о проблеме и могли предпринять шаги по своей защите. Кроме того, исследователи могут найти выгодным поступить двумя обычными способами: сохранить уязвимость в тайне или продать zero-day перекупам в других случаях.

В примере с CVE-2017-0144/EternalBlue, открывавшем эту главу, правительство Соединенных Штатов, как утверждается, обнаружило уязвимость в ОС Windows. Они добились этого, целенаправленно изучая безопасность программного обеспечения, надеясь найти уязвимости нулевого дня. Однажды обнаружив их, они сохранили свои находки в тайне.

Поскольку уязвимости нулевого дня, обычно являются нераскрытыми, как правило нет способа предотвратить эксплуатацию уязвимости, даже в идеальных условиях. Zero-day уязвимости, могут быть обнаружены до начала атак, эксплуатирующих незащищенные места в ПО, одновременно с проводимыми атаками или после того, как атаки уже нанесли ущерб. Zero-day – это то, что мы бы классифицировали в жизненном цикле уязвимостей, как только что обнаруженную уязвимость .

Zero-day эксплоит

Zero-day эксплоит – это когда злонамеренные или не злонамеренные хакеры, разработали действующий эксплоит для уязвимости нулевого дня. Хотя так бывает не всегда, они часто доводятся до состояния боевых инструментов, которыми можно нанести удар по уязвимому софту. Следовательно, его также можно описать, как передаточный механизм, использованный для эксплуатации уязвимости.

Кроме того, в нашей исходной уязвимости, CVE-2017-0144, zero-day эксплоит был вероятно разработан из-за раскрытия информации в течение исследования уязвимости. В случае с TSB, они выпустили эксплоиты, состоявшие из EternalBlue, EternalRomance и EternalSynergy. Эти zero-day эксплоиты, появились в комплекте с инструментами и механизмами доставки, такими как **DOUBLEPULSAR**. Хотя DOUBLEPULSAR, и не использовался конкретно против CVE-2017-0144, он применялся как средство доставки и закрепления в системе, чтобы извлечь выгоду из уязвимости.

Когда zero-day используется как оружие, это часто происходит по характерным причинам, которые мотивированы некими целями. Эти цели, обычно включают следующее:

- **Шпионаж**: Эксфильтрация данных из целевой организации с целью добычи информации, которую сложно или невозможно получить иным путем.
- **Саботаж**: Нарушение работы целевой организации с целью получения некоей выгоды от этого факта или просто для причинения ущерба.
- **Нажива**: Использование zero-day для вымогательства у организаций или личностей, которые попали под удар.

Хотя zero-day эксплоиты могут быть использованы для множества разных целей, они также могут быть применены для защиты. В некоторых случаях, организации в сфере ИБ разрабатывают эксплоиты или даже покупают zero-day для тестирования своих или клиентских систем и узнавать, если они уязвимы к атакам. Поводом для компаний создавать собственные эксплоиты для внутреннего пользования при тестировании систем, является то, что это дает реалистичное понимание риска, куда сильно мотивированные действующие лица, нацеливающиеся на определенные отрасли, могли бы сделать такие же инвестиции.

В случае крайне успешной атаки вымогателя WannaCry, следующей за релизом EternalBlue, мотивированные злоумышленники, использовали эксплоит EternalBlue в связке с другой малварью. Преступники сделали это для разработки орудия, предназначенного для наживы на шифровании файлов, путем удержания их в “залоге“ и затем требования денег с пользователей угрожая, что в противном случае их файлы останутся зашифрованными навсегда.

Атаки нулевого дня

Атаки нулевого дня, включают в себя применение zero-day эксплоитов. Атаки часто составлены из вредоносного ПО, доставляющего эксплоит в целевую систему. В нашем примере с EternalBlue, эксплоиты были использованы до того, как они были слиты, однако он не мог дальше называться zero-day эксплоитом или атакой, когда уязвимость была пропатчена.

Эксперты по безопасности обеспокоены по поводу атак нулевого дня, так как они могут использоваться для выполнения задач эксплуатации на целевых системах, вообще без видимых признаков. Кроме того, новый zero-day эксплоит, когда-то будет использован, и для систем защиты может оказаться трудным сдерживать всевозрастающую интенсивность таких атак.

Наряду с тем, что атаки нулевого дня, стимулируют защиту от них же, в некоторых случаях организации нашли существенную пользу в стратегии **глубоко эшелонированной обороны** (**defense-in-depth**). Вообразите кибератаки, как цепочку событий. Атакующие редко эксплуатируют какой-то конкретный программный комплекс и продвигаются за ним всю его “жизнь“. Вместо этого, они группируют атаки вместе, чтобы достигнуть цели (см. рис. 2.1).

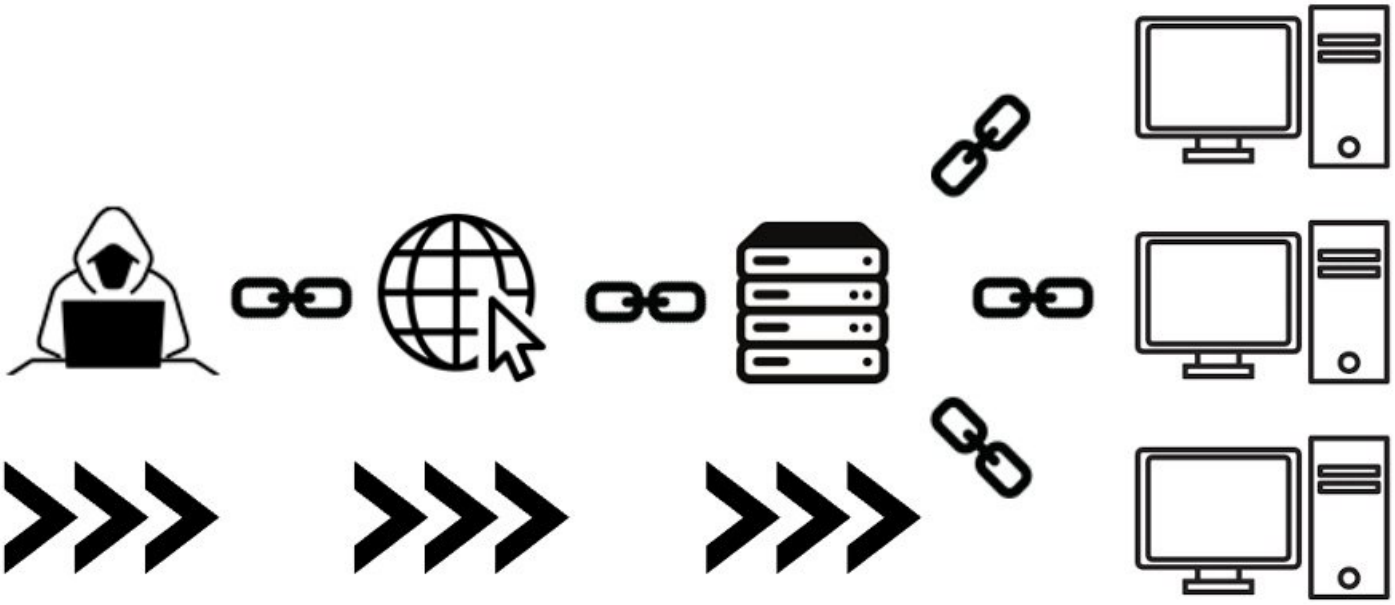


Рис. 2.1 – Отсутствие соответствующего контроля внутри цепочки, может позволить атакующему действовать более успешно

Глубоко эшелонированная оборона, помогает разорвать цепочку событий, вводя дополнительные элементы защиты, способствующие предотвращению процесса эксплуатации от начала или уже в ходе выполнения, позволяющего атакующему добиться своего. Еще раз рассмотрим наш пример с EternalBlue. В некоторых случаях, организации использовали средства управления безопасностью, которые предотвращали прямой доступ к определенным портам в Windows. Такие средства управления, действовали как защитный механизм:

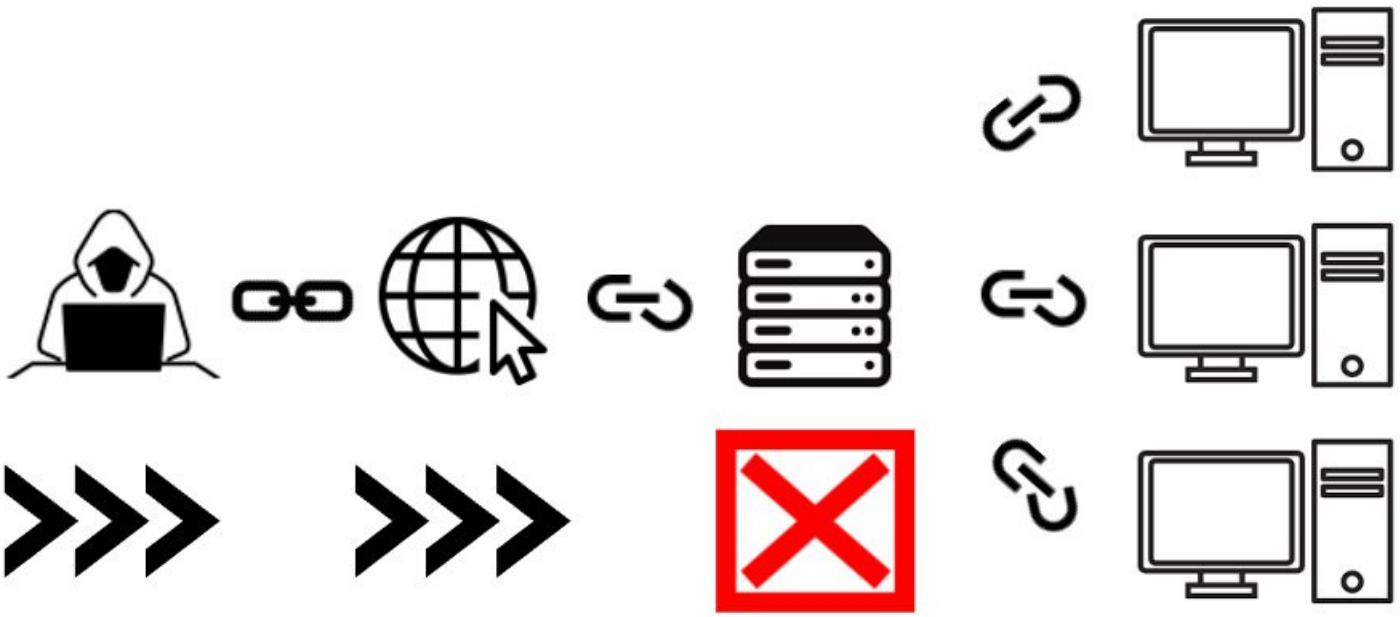


Рис. 2.2 – Глубоко эшелонированная оборона, разрывает цепочку событий, которые используются в атаках нулевого дня.

В результате, разрушение цепочки атак, предотвратило успешную эксплуатацию zero-day (см. рис. 2.2).

Некоторые аналогии в терминологии "нулевого дня"

Уязвимости нулевого дня часто используются как оружие и подобно большинству вооружения, они вызывают соответствующие аналогии, как такие термины могут быть использованы. Подумайте о другом оружии, таком как термоядерная бомба. Эти вооружения содержат опасное сырье, вроде урана. Хотя они опасны, они не обязательно являются чем-то таким, что может оказать негативное воздействие на общество в целом, если только они явно не предназначены для этого. Представьте, что сырье, помещенное в бомбу, как это – наши уязвимости нулевого дня. Чтобы использовать сырье (или уязвимости) для создания бомбы, мотивированным действующим лицам необходимо преобразовать их во что-то, что можно использовать как поражающие элементы, которые затем упаковывают в механизм доставки, типа ядерной бомбы. В нашей аналогии, zero-day эксплоит, как раз похож на взрывчатку, которая теперь имеет эффективное средство доставки и эксплуатации. Подобно большинству традиционных вооружений, они частно проходят испытания перед тем, как использоваться в наступательных операциях. В конечном счете, атака нулевого дня включает применение эксплоитов. Атаки часто принимают формы, которые причиняют ущерб, сразу после атаки и какое-то время спустя, после ее совершения (см. рис. 2.3):

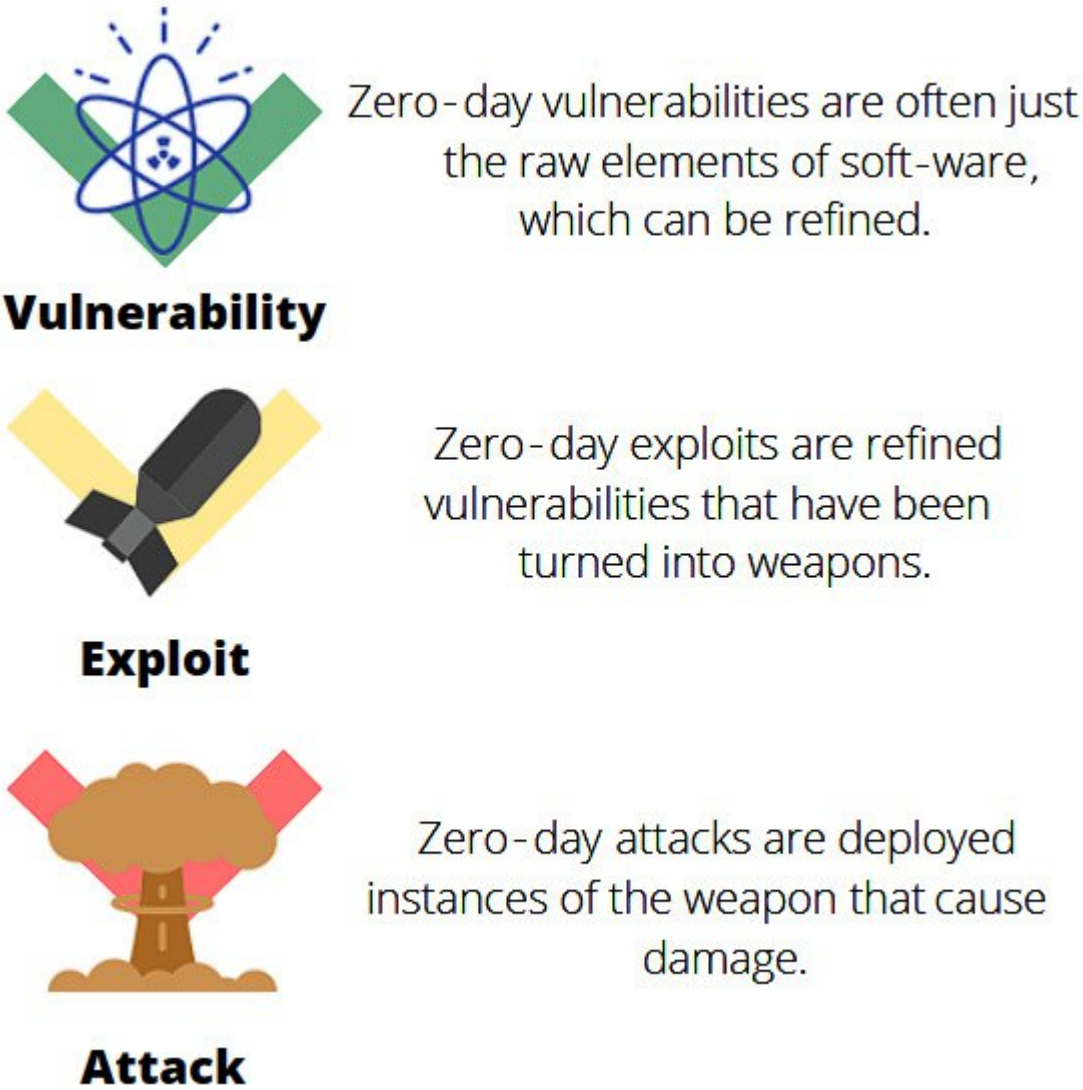


Рис. 2.3 – Zero-day начинаются как уязвимости, развиваются в эксплоиты и используются в атаках.

Разработчику программного обеспечения часто требуется время, чтобы узнать о наличии уязвимостей нулевого дня. Поэтому, разработка патча и развертывание его на системах пользователей, может растянуться на месяцы. Даже когда патчи разработаны, это не всегда

гарантирует, что пользователи будут обновлять софт. Опять же, в нашем примере с EternalBlue, по предварительным подсчетам, успешно взломали полмиллиона устройств, в первые 3 недели с момента появления доступного эксплоита – даже когда патч безопасности был срочно передан клиентам, как и информация о риске чреватом серьезными последствиями.

Теперь, приступим к изучению нескольких конкретных примеров широко эксплуатировавшихся уязвимостей, вошедших и вышедших из перечня уязвимостей нулевого дня, для лучшего понимания, как разворачиваются сценарии, подобные этим.

Разбор конкретных примеров уязвимостей нулевого дня

Отличный способ изучить уязвимости нулевого дня, это рассмотрение случаев для которых составлялись отчеты. Понимание того, как они были обнаружены, использованы и когда стали доступны патчи, может помочь нам понять ход событий и дать нам возможность подумать над потенциальными проблемами, которые могут вызвать раскрытия уязвимостей. Итак, рассмотрим четыре примера zero-day уязвимостей, найденных в естественной среде обитания, чтобы понять как они эксплуатировались. После того, как все эти уязвимости раскроют, мы запросим CVE ID, покажем занесение в MITRE, представим описание выпущенное уполномоченными организациями и кратко обсудим уязвимость.

Pulse – CVE-2019-11510

Начинаем наше обсуждение конкретных примеров с **CVE-2019-11510**, относящейся к семейству продуктов Pulse VPN. Мы будем обсуждать это, поскольку в течение 2019 и 2020 годов, она была одной из наиболее серьезных неисправленных уязвимостей, которую очень активно эксплуатировали злоумышленники. Давайте взглянем на описание уязвимости.

Описание уязвимости

В продукте Pulse Secure **Pulse Connect Secure** (**PCS**) 8.2, до версии 8.2R12.1, 8.3, до версии 8.3R7.1 и 9.0, до версии 9.0R3.4 неаутентифицированный, удаленный злоумышленник мог отправить специально подготовленный URI для эксплуатации уязвимости **чтения произвольных файлов** (см. рис. 2.4):

CVE-ID

CVE-2019-11510 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

In Pulse Secure Pulse Connect Secure (PCS) 8.2 before 8.2R12.1, 8.3 before 8.3R7.1, and 9.0 before 9.0R3.4, an unauthenticated remote attacker can send a specially crafted URI to perform an arbitrary file reading vulnerability .

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- BID:108073
- URL:http://www.securityfocus.com/bid/108073
- CERT-VN:VU#927237
- URL:https://www.kb.cert.org/vuls/id/927237
- CONFIRM:https://kb.pulsesecure.net/articles/Pulse_Security_Advisories/SA44101/
- CONFIRM:https://psirt.global.sonicwall.com/vuln-detail/SNWLID-2019-0010
- MISC:http://packetstormsecurity.com/files/154176/Pulse-Secure-SSL-VPN-8.1R15.1-8.2-8.3-9.0-Arbitrary-File-Disclosure.html
- MISC:http://packetstormsecurity.com/files/154231/Pulse-Secure-SSL-VPN-File-Disclosure-NSE.html
- MISC:https://badpackets.net/over-14500-pulse-secure-vpn-endpoints-vulnerable-to-cve-2019-11510/
- MISC:https://devco.re/blog/2019/09/02/attacking-ssl-vpn-part-3-the-golden-Pulse-Secure-ssl-vpn-rce-chain-with-Twitter-as-case-study/
- MISC:https://i.blackhat.com/USA-19/Wednesday/us-19-Tsai-Infiltrating-Corporate-Intranet-Like-NSA.pdf
- MISC:https://kb.pulsesecure.net/?atype=sa
- MLIST:[guacamole-user] 20190912 Re: [Guacamole hack attack?]
- URL:https://lists.apache.org/thread.html/ff5fa1837b6bd1b24d18a42faa75e165a4573dbe2d434910c15fd08a@%3Cuser.guacamole.apache.org%3E

Assigning CNA

MITRE Corporation

Date Record Created

20190424 Disclaimer: The record creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

Рис. 2.4 – Листинг MITRE CVE для CVE-2019-11510

Обсуждение уязвимости

CVE-2019-11510, является критической уязвимостью в PCS (ранее известным, как **Juniper SSL VPN**), которая позволяла чтение произвольных файлов. Атакующие могли сформировать специально подготовленный запрос к серверам, на которых крутятся веб-приложения, чтобы сделать возможным скачивание и извлечение имен пользователей и паролей – все это время оставаясь анонимным и не нуждаясь в какой-либо аутентификации. Если ее удачно связали с другими эксплоитами, жертвы могли ожидать результатом уязвимости опаснейшую компрометацию, потенциально позволяющую злоумышленникам внутри сети, беспрепятственный доступ в корпоративные сети. Эта уязвимость, поразила многие версии ПО, и согласно информации US-CERT и нескольких отчетов о пост-эксплуатации от компаний, которые пострадали от атак – этот дефект мог эксплуатироваться удаленно, без какого-либо взаимодействия с пользователем.

Цель Pulse Secure, подразумевалась в организации безопасного удаленного доступа со стороны пользователей, которые хотели войти в корпоративную сеть удаленно, из интернета. Слабые места точек входа в корпоративные сети, всегда живо интересуют исследователей безопасности, и Pulse Secure не оказалась исключением. Этот баг в безопасности, явился результатом изысканий, которые исследователи безопасности опубликовали непосредственно для вендора. Мотивированные исследователи начали наблюдать за Pulse, еще с тех пор, как он использовался в Google и Twitter, в числе других крупных организаций, таких как правительство Соединенных Штатов.

К счастью однажды, исследователи безопасности отправили отчет об уязвимости, вендор по быстрому отреагировал и исправил CVE-2019-11510 в пределах месяца с момента исходного отчета. Pulse выпустило для клиентов патч безопасности, а исследователи, изначально обнаружившие уязвимость, планировали не раскрывать результаты исследований, чтобы пользователи имели достаточно времени на ее устранение. Однако, как обсуждалось в разделе о жизненном цикле уязвимостей, потенциальная возможность для исследователей безопасности по реверс-инжинирингу патчей, стала реальной вскоре после того, как производитель ПО их выпустил. Эксплуатация **Proof-of-**

Concept (**PoC**) кода, начала всплывать по всему интернету, и такой код дорабатывался в автоматизированных сканерах, которые помогали автоматически обнаруживать и эксплуатировать программное обеспечение Pulse. Итак, мысленно возвращаясь к нашему обсуждению терминологии уязвимостей нулевого дня – только что раскрытый настоящий zero-day, сразу же становится уязвимостью. Последующие эксплуатации и атаки, которые производились на системы из-за этой уязвимости, технически не были атаками нулевого дня (см. рис. 2.5):

Рис. 2.5 – Единственным zero-day, который проявился в данном случае, была уязвимость

Движуха вокруг уязвимости

Критические уязвимости нулевого дня и их неизбежное использование, редко следуют по линейной траектории. Интересно, что после выпуска патча, другие исследователи безопасности, взялись рассматривать патч и начали его реверс-инжиниринг, чтобы выяснить потенциальную возможность эксплуатации. Прошло не более нескольких недель до тех пор, пока эксплоиты выложили в общий доступ и преступники начали их использовать в атаках на компании.

Это, кстати хорошая идея, чтобы проверить отклик клиентов на рекомендации по безопасности, опубликованные поставщиком ПО. Из-за широкого применения и слабого отклика на исправление, агентства безопасности в течение следующего года вели наблюдение за эксплуатацией уязвимости. Несмотря на то, что уязвимость нулевого дня, была найдена и устранена быстро, пользователи не применили патч. Неприменение патча, когда он опубликован – открыло двери перед эксплоитами, которые были массово использованы в атаках на крупные корпорации, правительственные учреждения, школы и организации здравоохранения.

Еще одним любопытным развитием этого, заслуживающего внимания случая, оказалось то что, исследователи, изначально нашедшие уязвимость и проинформировавшие об этом Pulse, позднее сами начали искать компании с уязвимыми системами, политиками ответственного раскрытия и баг-баунти, и эксплуатировать все уязвимые серверы Pulse, соответствующие их критериям. Примечательно, что на их наиболее интересных раскрытых ресурсах обсуждается поиск VPN используемой в Twitter, а затем использование этой уязвимости в цепочке слабых мест для получения удаленного доступа к компьютерным системам Twitter. Twitter заплатил исследователям более 20000$, за раскрытие этой секретной информации.

Размышления об уязвимости

Важно помнить, что все участвующие стороны верили, что поступают правильно. Тем не менее, размышляя об этой уязвимости – задайте себе следующие вопросы:

- Корректно ли вендор проинформировал клиентов о том, что их системы уязвимы? Множество пользователей были все еще уязвимы, после того как патчи были выпущены.
- Правы ли были исследователи безопасности, обнаружившие уязвимость – не раскрывая ее, после выпуска патча?

- Twitter заплатил более 20000$ исследователям безопасности, нашедшим уязвимых VPN клиентов – патчи безопасности при этом, были доступны. Как вы думаете, почему они не были пропатчены и мог Twitter ли сэкономить на выплате, если бы патчи безопасности, были бы применены?
- Если уж на то пошло, что бы вы сделали по-другому, в процессе раскрытия информации?

Confluence – CVE-2021-26084

Теперь, мы присмотримся к **CVE-2021-26084**, которая была найдена в чрезвычайно популярном программном обеспечении **Confluence** компании Atlassian. Эта уязвимость, использовалась весь 2021 год, помогая злоумышленникам без аутентификации добывать удаленный доступ к весьма секретным документам и доступ на уровне ОС. Рассмотрим описание уязвимости.

Описание уязвимости

В версиях Confluence Server и Data Center подверженных OGNL-инъекции, существовала уязвимость, позволявшая злоумышленнику выполнить произвольный код на инсталляциях Confluence Server или Data Center без какой-либо аутентификации. Уязвимы версии до 6.13.23, с 6.14.0 по 7.4.11, с 7.5.0 по 7.11.6 и с 7.12.0 по 7.12.5 (см. рис. 2.6):

CVE-ID

CVE-2021-26084	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

In affected versions of Confluence Server and Data Center, an OGNL injection vulnerability exists that would allow an unauthenticated attacker to execute arbitrary code on a Confluence Server or Data Center instance. The affected versions are before version 6.13.23, from version 6.14.0 before 7.4.11, from version 7.5.0 before 7.11.6, and from version 7.12.0 before 7.12.5.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- MISC:https://jira.atlassian.com/browse/CONFSERVER-67940
- URL:https://jira.atlassian.com/browse/CONFSERVER-67940

Assigning CNA

Atlassian

Date Record Created

20210125	Disclaimer: The record creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

Phase (Legacy)

Assigned (20210125)

Votes (Legacy)

Comments (Legacy)

Рис. 2.6 – Листинг MITRE CVE для CVE-2021-26084

Обсуждение уязвимости

Программное обеспечение компании Atlassian, Confluence – наверное один из самых используемых движков представления документации в стиле вики, применяемый в фирмах по разработке ПО. Плотная интеграция с Jira, Bitbucket и другими продуктами Atlassian, облегчает их продажу клиентам, уже инвестировавшим в эту экосистему. Если вы постоянно работали в организациях, сосредоточенных на разработке ПО или каких-то стартапах, вы скорее всего взаимодействовали, пользовались или работали непосредственно в программном обеспечении Atlassian. **CVE-2021-26084**, оказалась сюрпризом для Atlassian и была обнаружена исследователями, когда уже вовсю эксплуатировалась. Злоумышленники обнаружили возможность эксплуатации уязвимости**,** посредством отправки Confluence Server легко подготавливаемых HTTP-запросов с подмененным параметром. Atlassian, по-быстрому выпустила патч и рекомендации по безопасности.

Это не первый случай, когда Confluence оказался уязвимым к критической проблеме безопасности. В 2019 году, CVE-2019-3396 эксплуатировалась в “дикой природе“, что выражалось в удаленном выполнении кода. Достаточно интересно, что среди исследователей безопасности находились крайне осведомленные и заинтересованные в этой уязвимости, т. к. программное обеспечение Atlassian, среди прочего использовалось в Министерстве Обороны Соединенных Штатов, НАТО и нескольких прочих организациях высокого уровня секретности, в том числе правительственных.

В противоположность предыдущей уязвимости в Pulse VPN, этот zero-day, состоял из уязвимости, эксплоита и атаки:

Рис. 2.7 – Уязвимости, эксплоиты и атаки, в этом случае – полный комплект

В определенных случаях раскрытия zero-day, уязвимость обнаруживается уже после фактов эксплуатации, как в описанном случае (см. рис. 2.7)

Движуха вокруг уязвимости

Уязвимость нашлась, будучи активно эксплуатируемой в интернете, а т. к. недавно Atlassian имела предыдущий опыт с похожей проблемой, они подготовили улучшенный ответ на базе предыдущего. Можно было бы легко сказать, что Atlassian проявляет стремительную реакцию на инциденты безопасности, когда об уязвимости уже объявлено. Одним пользователям платформы Confluence, Atlassian могла рассылать уведомления и предупреждения о проблемах безопасности ПО, прямо внутри платформы. Эта функциональность позволяла администраторам сайтов, узнавать, что необходимы безотлагательные меры.

В частности, пользователи платформы Confluence заявляли, что их также агрессивно бомбардировали уведомлениями другие пользователи продукта по всем доступным каналам, за счет отношений с поставщиками. Однако, не каждый получал поддержку такого уровня. В других случаях, Atlassian предпочла не поддерживать пользователей, не обновивших их лицензию для получения патчей безопасности. В результате некоторым клиентам, требовалось оплатить дорогостоящие лицензионные отчисления, чтобы получить свои отчаянно необходимые, патчи безопасности. Кто-то мог бы утверждать, что это было неэтичным поведением со стороны Atlassian, а кто-то отнести к проблемам не желающих платить пользователей. Безотносительно обоих выводов, Atlassian хорошо отреагировала в этом случае, учитывая мгновенное уведомление и активную эксплуатацию в интернете.

Размышления об уязвимости

Обдумывая, в чем особенность этой уязвимости, найдите минутку подумать над некоторыми вопросами, которые мы могли бы себе задать в этом отдельном случае:

- Уязвимость была обнаружена после того, как злоумышленники применили в атаках zero-day эксплоит. Как вы думаете какие мероприятия, могли бы помочь снизить темп атакующих?
- Могли ли атаки быть предотвращены средствами управления? Почему, или почему нет?
- Что вы думаете насчет агрессивной политики реагирования на инциденты безопасности Atlassian? Думаете ли вы, что исправление уязвимостей, закончилось лучше чем у Pulse VPN?
- Должна ли была Atlassian, предоставить пользователям больше информации по выявленным атакам, которые могли быть проведены на установленные у них Confluence Server? Как это могло быть сделано?
- Как вы воспринимаете то, что Atlassian не предоставляло поддержки всем пользователям Confluence? Например они отказали в исправлениях безопасности, пользователям не имевшим актуальной лицензии на их продукты и не попадающих под поддержку?

Microsoft .NET CVE-2017-8759

В этом примере, мы критически рассмотрим **CVE-2017-8759**, найденное в платформе Microsoft .NET. Этот случай, занимателен тем, что обнаружился, после фактов эксплуатации. Для любого исследователя охотящегося за багами, это красная тряпка, сигнализирующая о вскрывшейся кампании активной эксплуатации уязвимости. Начнем с изучения описания, перед тем как погрузиться в дальнейшую дискуссию.

Описание уязвимости

Microsoft .NET Framework 2.0, 3.5, 3.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, и 4.7, позволяла злоумышленнику удаленно выполнять код посредством вредоносных документов или приложений, так же известная, как **Уязвимость Удаленного Выполнения Кода** (**RCE - Remote Code Execution**) в **.NET Framework**, (см. рис. 2.8):

CVE-ID

CVE-2017-8759 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

Microsoft .NET Framework 2.0, 3.5, 3.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2 and 4.7 allow an attacker to execute code remotely via a malicious document or application, aka ".NET Framework Remote Code Execution Vulnerability."

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- BID:100742
- URL:http://www.securityfocus.com/bid/100742
- CONFIRM:https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2017-8759
- EXPLOIT-DB:42711
- URL:https://www.exploit-db.com/exploits/42711/
- MISC:https://github.com/GitHubAssessments/CVE_Assessments_01_2020
- MISC:https://github.com/bhdresh/CVE-2017-8759
- MISC:https://github.com/nccgroup/CVE-2017-8759
- SECTRACK:1039324
- URL:http://www.securitytracker.com/id/1039324

Assigning CNA

Microsoft Corporation

Date Record Created

20170503 Disclaimer: The record creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

Phase (Legacy)

Assigned (20170503)

Votes (Legacy)

Рис. 2.8 – Листинг MITRE CVE для CVE-2017-8759

Обсуждение уязвимости

В 2017 году, Microsoft помогла опубликовать раскрытие проблемы безопасности в CVE-2017-8759. Это раскрытие относилось к неизвестной предыдущей уязвимости в Microsoft .NET Framework. CVE-2017-8759 основывалась на социальной инженерии для использования уязвимости. Эта конкретная уязвимость, проявляла себя через рассылку вредоносных документов. Создавая разнообразные экземпляры, злоумышленники нацеливались на офисный софт (Microsoft Word и PowerPoint), для впаривания нагрузки. Пользователей, требовалось первым делом, убедить открыть документ, как только документ открывался (в уязвимом ПО), малварь обычно загружалась на компьютер жертвы.

В наблюдаемых начальных этапах активной эксплуатации, злоумышленники разослали миллионы документов. По общему мнению исследователей безопасности, первоначальные образцы, которые они добыли (Project.doc на русском языке), указывали направленность на российских пользователей по геополитическим причинам. Однако наряду с тем, что дело обстояло именно так, как только первоначальный эксплоит был добыт и выложен в общий доступ, инженеры мгновенно начали разработку эксплоит-китов. Раз уж возможность эксплуатации стала доступной, преступные сообщества начали кампании по краже банковской информации у пользователей, путем рассылки миллионов электронных писем с зараженными документами во вложениях (см. рис. 2.9):

Рис. 2.9 – Уязвимости были обнаружены, доработаны до состояния эксплоитов и задействованы в атаках

Движуха вокруг уязвимости

Это замечательная уязвимость для изучения и анализа. Во-первых уязвимость, была во всех отношениях неизвестной – она была обнаружена позже, в качестве атаки нулевого дня, где жертвы возможно выбирались по национальному признаку и принадлежности к определенному государству. Исследователи безопасности тем временем, отрабатывали атаку вспять, разбираясь в механизме эксплуатации и сути уязвимости. Патч после этого был выпущен, но множество людей потеряли деньги в результате “кражи сетевой личности“, которую беспринципные хакеры, увидели и захватили с помощью уязвимости для своих целей.

Размышления об уязвимости

Microsoft быстро исправила положение, сразу же после обнаружения уязвимости и получения информации о ней. Однако, обдумывая эту уязвимость, задайте себе следующие вопросы о ее эксплуатации:

- Каким образом вы могли бы защитить свою организацию от атаки, подобной этой? Для эксплуатации требуется взаимодействие с пользователем.
- Что бы вы могли предпринять, если бы обнаружили в своем окружении следы такой атаки?

- Могли бы сторонние производители и продукты, такие как антивирусы – остановить загрузку и выполнение таких вредоносных документов?

Citrix – CVE-2019-19781

В нашем заключительном примере, мы обратим внимание на **CVE-2019-19781**, найденную в семействе продуктов Citrix. Продуктовая линейка Citrix, используется для предоставления безопасного доступа к приложениям, и в связи с этим, может быть заманчивой целью для исследований. Взглянем на описание уязвимости MITRE.

Описание уязвимости

Уязвимость была обнаружена в Citrix **Application Delivery Controller** (**ADC**) и Gateway 10.5, 11.1, 12.0, 12.1, и 13.0. Они допускают **обход каталога** (**Directory Traversal**), (см. рис. 2.10):

CVE-ID

CVE-2019-19781 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

An issue was discovered in Citrix Application Delivery Controller (ADC) and Gateway 10.5, 11.1, 12.0, 12.1, and 13.0. They allow Directory Traversal.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- CERT-VN:VU#619785
- URL:https://www.kb.cert.org/vuls/id/619785
- CONFIRM:https://support.citrix.com/article/CTX267027
- MISC:http://packetstormsecurity.com/files/155904/Citrix-Application-Delivery-Controller-Gateway-Remote-Code-Execution.html
- MISC:http://packetstormsecurity.com/files/155905/Citrix-Application-Delivery-Controller-Gateway-Remote-Code-Execution-Traversal.html
- MISC:http://packetstormsecurity.com/files/155930/Citrix-Application-Delivery-Controller-Gateway-10.5-Remote-Code-Execution.html
- MISC:http://packetstormsecurity.com/files/155947/Citrix-ADC-NetScaler-Directory-Traversal-Remote-Code-Execution.html
- MISC:http://packetstormsecurity.com/files/155972/Citrix-ADC-Gateway-Path-Traversal.html
- MISC:https://badpackets.net/over-25000-citrix-netscaler-endpoints-vulnerable-to-cve-2019-19781/
- MISC:https://forms.gle/eDf3DXZAv96oosfj6
- MISC:https://twitter.com/bad_packets/status/1215431625766424576

Assigning CNA

MITRE Corporation

Date Record Created

20191213 Disclaimer: The record creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

Phase (Legacy)

Assigned (20191213)

Votes (Legacy)

Рис. 2.10 – Листинг MITRE CVE для CVE-2019-19781

Обсуждение уязвимости

Citrix **CVE-2019-19781** была одной из самых часто эксплуатируемых уязвимостей между 2019 и 2022 годами. Дефект CVE-2019-19781 разрешал злонамеренным лицам, получить доступ и провести несанкционированное **удаленное выполнение кода** (**RCE**) на целевой системе. Это обеспечило атакующим возможность, выполнять вредоносный код и получать полный контроль над системами подверженными уязвимости.

Citrix ADC, ранее известный как **NetScaler ADC**, и Citrix Gateway, ранее известный как **NetScaler Gateway**, были подвержены этой уязвимости, метко обозванной **Shitrix-ом**, значительной частью ИБ сообщества. Производитель выпустил патч, через месяц после раскрытия уязвимости. Кроме того, Citrix опубликовал рекомендации по снижению риска, которые должны быть применены везде, где это возможно (см. рис. 2.11):

Рис. 2.11 – В данном случае, только уязвимость являлась zero-day, эксплуатация случилась после раскрытия

Выполнение произвольного кода (**ACE - Arbitrary Code Execution**) – опасная тактика атакующего, которую злоумышленники применяют для получения повышенных привилегий и удаленного доступа к целевому устройству. Как только дефект обнаружен, выполнение произвольного кода, может использоваться со злым умыслом. К сожалению, исправления безопасности не были доступны сразу, что предотвратило бы эксплуатацию этой уязвимости, вместо этого, чтобы защититься от этого типа атаки, Citrix опубликовал ряд шагов, которые специалисты по кибер-безопасности могли предпринять для нейтрализации угрозы.

Движуха вокруг уязвимости

Citrix выпустил руководство относительно этой проблемы, за несколько дней до раскрытия исследователями безопасности своих наработок. Citrix выпустил CVE совместно с исследователями и помогал консультировать клиентов, насчет выполнения шагов по снижению риска, вместо установки патча. Патчи выпустили через месяц по ходу распространения уязвимости.

Подробная публикация исследователей безопасности, помогла другим исследователям быстро соорудить действующий эксплоит. Позже, этот эксплоит использовался “ан масс“ против серверов Citrix, не установивших исправления или не применявших техник снижения риска этой угрозы. Исследователи, официально работавшие с Citrix, действовали совместно и они опубликовали шаги по снижению риска, но не патч.

После установки исправлений, некоторые экземпляры или инсталляции ADC, были повреждены так, что даже если установить патчи – они все еще оставались уязвимы к эксплуатации. Почти через год, после первоначальной публикации Citrix раскрыла, какие еще сервисы пострадали от этой уязвимости, консультировала клиентов, как это исправить, но не напрямую. Критики часто указывают, что группа исследовавшая безопасность, даже не дала Citrix времени на разработку патча. Кроме того, исследователи безопасности, восприняли как раскрытие, публикацию мероприятий по снижению риска, посчитав, что можно опубликовать свои исследования, через неделю или около того после CVE.

Размышления об уязвимости

В этом примере, некоторые похвалили подход Citrix к уязвимости, однако они оставили пользователей без исправлений на неопределенное время, а также не раскрыли явно уязвимые продукты. Подумайте, как бы вы могли поступить в этом сценарии, задав себе такие вопросы:

- Должна ли Citrix явным образом раскрыть все уязвимое ПО пользователям, вместе с сообщением об исправлении?

- Были ли мероприятия по снижению риска, опубликованные Citrix, достаточными для сдерживания угрозы?
- Дали ли исследователи безопасности раскрывшие уязвимость, время на ее исправление производителю?
- Публикация исследователей безопасности, помогла или навредила пользователям, т. к. доступного исправления еще не было?

Таким образом, мы рассмотрели четыре примера CVE, которые эксплуатировались, как zero-day на просторах интернета. Каждая из них обладает отличающимися характеристиками, но все они одинаково эксплуатировались до выхода исправлений. Понимание того, как они эксплуатировались, поможет нам быть лучше подготовленными в будущем, когда мы раскроем наши уязвимости нулевого дня. Обратим внимание, что исследователи должны учитывать общепринятые этические нормы при раскрытии уязвимостей нулевого дня, удостоверившись, что пользователи защищены от эксплуатации. Ответственность связанная с уязвимостями нулевого дня – тонкий и запутанный предмет обсуждения и в следующем разделе, мы изучим этику и ответственность участвующих сторон.

Обсуждение этической стороны уязвимостей нулевого дня

Этика раскрытия уязвимостей нулевого дня – это тема, о которой будут спорить годами. Однако эта дискуссия не о раскрытии самой уязвимости, а скорее о том, когда это делать и кому ее раскрывать. Не существует однозначного ответа, когда и как раскрывать zero-day уязвимости. Каждый сценарий, включает множество сторон (как правило, исследователь и вендор), которые потребуют рассмотрения. Главный решающий фактор, который нужно учесть во время раскрытия, это пользователи и угроза, которой они подвергнутся, если уязвимость нулевого дня окажется раскрытой. Вообще говоря, этические вопросы стали распространенными относительно ответственности этих сторон. Обсудим же их сейчас.

Ответственность исследователя

Некоторые исследователи полагают, что их обязанность найти и засветить уязвимости, чем быстрей, тем лучше. Это дает производителю время, исправить проблему, до того, как она станет широко известной, и дает время пользователям, чтобы обновить их системы. А что, если поставщики ПО, считают по-другому? Тем временем, крупные корпорации с продвинутыми структурами безопасности, могут иметь свои внутренние политики и публичную позицию и когда это касается исследователей, присылающих уязвимости ПО, исследователям приходится выполнять только то, что компании компенсируют, а это малая часть рынка ПО. Временами, вендоры принимают враждебную линию поведения по отношению к исследователям безопасности, рассматривая озвученные уязвимости, как допустимые издержки (которые не обязательно исправлять) или перерасход бюджета продукта, из-за которых производитель не заработает столько, сколько хотел. В результате, для исследователей безопасности не является редкостью, получить уведомление о запрете продолжения исследований от юристов, представляющих интересы производителей ПО. В этом случае, этично ли будет продать уязвимость перекупам или оказывать общественное давление на вендора, не дружественными, так скажем методами, в ответ на то, что он сам ведет себя враждебно?

Не существует правильного или неправильного ответа, ни в этом случае, ни вообще – в таких ситуациях, исследователь сам выбирает, как правильнее поступить. Итак, обсудим некоторые из обязанностей исследователя, когда это касается раскрытия уязвимостей.

Обязанности в рамках связи с производителями ПО

Исследователь должен, в первую очередь определить и опробовать способы взаимодействия с вендором. Это может принимать любую форму, но я советую завести и использовать какой-нибудь

адрес электронной почты для протоколирования в электронном виде, если когда-нибудь вам потребуется предъявить что-то в таком роде. Может быть полезно проверить, есть ли какие-нибудь политики раскрытия в области безопасности, в том числе на будущее. В зависимости от их политики или отсутствия таковой, могло бы быть правильнее проводить раскрытие исследования под псевдонимом, чтобы избежать идентификации.

Оценим многочисленные доводы за и против, о контакте с вендором в первую очередь и выкладывании информации об уязвимости на общедоступном форуме. Связь с производителями в первую очередь, позволяет им исправить проблему до того, как она станет общеизвестной, но пока они смогут отреагировать – пройдет время. С другой стороны, публикация на открытом форуме, делает проблему намного заметнее и подталкивает производителя, действовать значительно быстрее. Однако, также предполагается, что те многочисленные люди, кто оказался осведомлен об уязвимости до ее исправления, могут в свою очередь быстро привести к ее использованию. Итак, во-первых, крайне рекомендуется держать связь с вендором, стараться обмениваться с ним информацией и соглашаться на то, что лучше для пользователей. Если это не приносит успеха, рассмотрим другие способы донести эту информацию до общественности.

Мы обстоятельно поговорим об общении с производителем в *Главе 4*. Сейчас же, ключевой момент, который следует запомнить – убедитесь, что вы в разумных пределах дисциплинированны в своих исследованиях, ведь такое общение могло бы произойти. Ответственные стороны должны быть проинформированы об уязвимости и вашем явном намерении ее опубликовать.

Ответственность перед обществом

Исследования, обнаружение и познания в области уязвимостей безопасности, подобны обладанию сверх-способностями. Как говорится, с большими возможностями приходит большая ответственность. По мнению автора, такие исследователи безопасности, должны учитывать последствия для общества, если бы уязвимости были использованы. Наша работа как исследователей, должна применяться для усиления безопасности пользовательского опыта работы с ПО, а не ее ослабления. Следовательно, разделение информации о ваших исследованиях с широкой публикой, должно производиться только в направлении на снижение ущерба и последующих действий, которые защитят пользователей от участи жертв.

В нашем предыдущем разделе, про примеры уязвимостей, мы видели несколько примеров, в которых исследователи передали свои исследования в широкий доступ, безопасным, наиболее управляемым способом. Этот подход, привел к появлению патчей безопасности и позволил сознательным в плане информационной безопасности организациям, исправить дефекты до того, как они стали использоваться. Общий результат, в целом оказался положителен и первой это оценила общественность. Как исследователи, намеревающиеся улучшать безопасность программного обеспечения, мы должны стремиться к такому положительному результату.

Ответственность перед собой

Исследователи несут ответственность, также и перед собой. Вряд ли они продолжат исследования безопасности, если окажутся в тюрьме. Важно просчитать свое исследование досконально, убедиться в соблюдении вами действующего законодательства и том, что исследуемыми системами владеете, конкретно вы или имеете разрешение на тест. Никогда не тестируйте системы, не получив явного разрешения на проведение тестов.

Дополнительно к сказанному выше, обеспечьте так же защиту своих жизненно необходимых ресурсов. Старайтесь тестировать системы, создавая себе для работы, изолированное окружение для экспериментов. Это изолированное окружение, должно быть отделено от задач, выполняемых для любых других целей и не должно быть увязано, например с постоянной работой или другими

конкурирующими интересами. Со стороны представителей корпораций, не является редкостью, добиваться наказания вас или вашего работодателя, за исследование и раскрытие уязвимостей.

Мы ранее упомянули про раскрытие уязвимостей под каким-нибудь вымышленным именем или через третьих лиц, которые возьмут на себя риск, сопутствующий раскрытию. Эта практика, обычное явление для исследователей, которые желают избежать заморочек с юридическими отделами корпораций и не хотят ставить на кон свою работу и репутацию.

Ответственность производителя софта

За что отвечает производитель? Является ли его обязанностью, исправить zero-day как можно скорее? Или обязанность производителя, оценить риск и принять решение, когда его исправлять? Многовато вариантов.

Современные, ответственные компании, ориентированные на безопасность, должны принимать политики, которые позволяют исследователям безопасности работать совместно с ними, безопасно и конструктивно. Например, допустим условный исследователь безопасности, поддерживает связь с неким вендором. В этом случае, они заботятся о всеобщем благе пользователей и вероятно хотят сотрудничать, оправдывать доверие к своей непростой работе и улучшать программные продукты вендора неформальными способами. Это почти, как иметь бесплатного или (дешево обходящегося, если назначена награда) консультанта по безопасности. С другой стороны, устаревшие правила, сразу предусматривающие привлечение юристов защищающих организацию, в итоге без нужды создают слабые места в вашем ПО, т. к. исследователи, имеют возможность изучить его и не ставя вас в известность.

Принятие и ясное предложение гарантий безопасности и политика ответственного раскрытия информации – то, как поставщики ПО модернизируют свой подход к безопасности. Мы подробно обсудим в *Главе 9* то, каким может быть наилучший подход вендоров к исследователям безопасности.

Обязанности поставщика ПО перед пользователями

Если продукт, поддерживается производителем, пользователи рассчитывают на то, что вы безопасно храните их данные в своих продуктах. Поставщики ПО обязаны гарантировать пользователям, что любые дефекты программного обеспечения, которые могут быть использованы в целях нанесения ущерба будут исправлены, либо же будут предоставлены рекомендации по снижению риска и поддержка. У пользователя устаревшая версия вашего ПО, которая уже не поддерживается? Дайте им знать, о любых уязвимостях и предложите лояльным, давно работающим с вашим ПО пользователям, обновиться до версий для которых вы уже вложили средства в обновления и исправления безопасности.

В конечном счете, когда данные пользователей оказываются скомпрометированы, программным продуктом соответствующего производителя, они винят вендора и могут не обращать внимания на нюансы корпоративных или юридических правил, которые у него могут присутствовать. В результате клиент, может рассматривать других производителей при следующей закупке софта или, в случае корпоративного лицензионного соглашения, использовать это как средство давления для получения крупных скидок, урезая вам возможную рентабельность.

Ответственность вендора в качестве потребителя

Производители ПО часто строят свою работу и программное обеспечение, на основе других программных продуктов. Вендоры часто разрабатывают свое ПО с использованием библиотек с открытым исходным кодом или даже поверх операционных систем потребительского уровня. Производители ПО часто утверждают, что они отвечают исключительно за разработанное ими ПО, как

за цельный программный комплекс, а не за безопасность инструментальных средств на которых он базируется. Эта аномалия часто получается из-за муторных методов установления обязанностей потребителя относительно низлежащей платформы, наряду с тем, что сопровождающий ее вендор, не принимает доработки по безопасности, сделанные потребителем.

Этот вид политики безопасности, вреден для пользователей и лишает их возможности работать. Поговорим еще раз об уязвимости EternalBlue, которая дала старт атаке шифровальщика WannaCry – пользователи потеряли доступ к необходимым бизнес-функциям из-за того, что их не пропатченные и не поддерживаемые производителем устройства с Windows XP, неожиданно оказались зашифрованы.

**патч для Windows XP, вообще-то был выпущен – WindowsXP-KB4012598-x86*

Скажем об этом прямо. Вендоры, использовавшие и затем перепродавшие их, несут ответственность за исправление базового ПО, поверх которого ими был разработан, любой собственный код. Недостаточно, просто исправить ошибки в программном обеспечении и бросить пользователей барахтаться в море опасностей и угроз. Если производитель, продает своим клиентам аппаратно-программный комплекс, его компоненты требуют поддержки и периодических исправлений безопасности.

Итоги главы

В этой главе, вы изучили уязвимости нулевого дня, эксплоиты и атаки. В первую очередь, мы поговорили о каждом направлении и как они связаны друг с другом. Потом, мы рассмотрели четыре в прошлом значимых раскрытия уязвимостей, пока обсуждали уязвимости, эксплоиты и проведенные в некоторых из этих случаев атаки. И наконец, мы познакомились и этикой zero-day культуры и обрисовали кое-какую воображаемую схему, кто за что отвечает.

В следующей главе, мы обсудим, как мы можем начать исследование, которое поможет вам самостоятельно обнаруживать уязвимости, как правильно выбирать цели и какие стратегии вам потребуется усвоить для достижения успеха.

Дополнительные материалы

Для более подробного изучения тем, которые мы рассмотрели в этой главе, вы можете использовать следующие ресурсы самостоятельно:

- Список MITRE CVE: https://cve.mitre.org/cve/
- Список уязвимостей ФСТЭК: https://bdu.fstec.ru/vul?sort=datv
- Справка по EternalBlue: https://en.wikipedia.org/wiki/EternalBlue
- Справка по EternalBlue от ФСТЭК: https://bdu.fstec.ru/vul/2017-01099
- Wired Magazine – Не рассказанная история американского Zero-Day маркета: https://www.wired.com/story/untold-history-americas-zero-day-market/
- CVE-2019-11510: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-11510
- CVE-2021-26084: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-26084
- CVE-2017-8759: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-8759
- CVE-2019-19781: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-19781

Глава 3. Исследование уязвимостей – начнем с проверенных стратегий

Вы наверное знакомы со старой поговоркой: "Провал в планировании – это планирование провала". Обладание стратегией и следование ей, играет важную роль в дальнейших достижениях. Успешные бизнесмены, ученые и даже писатели, скажут вам, что искусство достижения целей – это больше, чем просто тяжелый труд. Оно требует тщательного планирования и действий в соответствии с выбранной стратегией. Без ясного плана действий, слишком просто зайти в тупик или запутаться в деталях. А с хорошо продуманной стратегией, вы остаетесь сосредоточенным на своей цели и увереннее продвигаетесь к ее достижению. Во многих случаях, разница между успехом и провалом, заключается только в наличии эффективной стратегии.

Как мы выяснили в предыдущих главах, исследование уязвимостей – это изучение слабых мест и изъянов технологии. Этот тип исследования, принимает множество форм – от распознавания новых уязвимостей, до оценки воздействия известных. Исследование уязвимостей, по сути служит защите от будущих, постоянно растущих угроз исходящих от кибер-преступности. Понимая, каким образом системы оказываются эксплуатированы, исследователи могут помочь разработать лучшие решения в сфере безопасности и делают технологии более защищенными для каждого.

Когда дело касается исследования уязвимостей – планирование и стратегии, должны быть продуманы. Провал в планировании проведения и оформления результатов исследований – это планирование провала. Немногие из большого числа успешных стратегий, специально разрабатывались с учетом конкретных требований исследователей и продуктов, которые они проверяют. Создание этих стратегий может вызвать трудности у новичка. Вместе с тем отметим, что существуют некоторые общие принципы, которые все исследователи уязвимостей должны помнить изначально и эти азы ведут к продвинутым и специализированным исследованиям.

В этой главе, мы собираемся учить вас, тем фундаментальным понятиям, которые помогут вам расти и начать исследование уязвимостей. Мы обсудим исследование уязвимостей, каким образом выбирать наиболее интересные объекты исследования, освоим методологию исследования, а также всерьез возьмемся за техники и инструменты, которые помогут вам достигнуть успеха в поиске уязвимостей.

Вот чему мы научимся в этой главе:

- Что такое исследование уязвимости и как его проводить
- Различные типы стратегий и доступные нам ресурсы
- Как выбирать лучшие объекты для наших исследований
- Тесты и тестовые пакеты, определение их как важнейшей составляющей успеха при исследовании уязвимостей
- Инструменты, которые помогут нам в исследованиях уязвимостей

Технические требования

Хотя это не обязательно для продолжения чтения, наличие этих принадлежностей и ресурсов под рукой, будет полезным:

- Компьютер с доступом к интернету
- Возможность скачивать программы и устанавливать их на компьютер

К концу этой главы, вы получите хорошее представление о том, что такое исследование уязвимостей, с чего начать, как выбрать ваш первый набор целей и какие есть распространенные инструменты применяемые исследователями при проведении и документировании исследования.

Что такое исследование уязвимостей?

В первой главе, мы познакомились с понятием жизненного цикла уязвимости. В начале этого жизненного цикла, уязвимости выявляются в процессе исследования. Исследование уязвимостей, является ключевой составляющей поддержания безопасности программного обеспечения и важно знать, что это такое и как оно работает.

Исследование уязвимостей – это процесс распознавания, анализа и составления отчетов об уязвимостях в программном обеспечении или системах. Это краеугольный камень информационной безопасности так, как с его помощью распознаются ранее неизвестные угрозы и гарантируется, что риски связанные с небезопасным программным обеспечением, будут тщательным образом изучены. Исследование уязвимости может быть сделано кем угодно, кто интересуется поддержанием безопасности программного обеспечения, включая ИБ специалистов, разработчиков и любителей. Общее мнение среди сообщества исследователей, что существует огромное число исследований в которых, если представится возможность уделить время, кто угодно мог бы выполнять задачи необходимые для нахождения уязвимостей и вносить вклад в исследование.

Хотя, задачи профессионалов, фокусирующихся на исследовании уязвимостей, могут разниться по организации, обычно исследователи уязвимостей, проводят время за выявлением изъянов безопасности в программных и аппаратных системах. Их цель – предоставить информацию, которая может быть использована разработчиками, хакерами или правительственными учреждениями для совершенствования мероприятий по информационной безопасности.

Существует множество способов проводить исследования уязвимостей. Одни исследователи применяют автоматизированные инструменты помогающие им в поиске уязвимостей, другие выполняют анализ вручную. Кто то, сосредоточивается на поиске новых уязвимостей, кто то на реверс-инжиниринге уязвимостей из уведомлений безопасности и патчей, тогда как остальные фокусируются на улучшении существующих методик поиска уязвимостей или создании продвинутых эксплоитов. Безотносительно направления, большинство исследователей уязвимостей, разделяют общую цель: делать программное обеспечения безопасней для каждого. Итак, что делает исследователь, разобрались – ну что, исследуем?

Проведение исследования

Для упрощения понимания ключевых этапов выполнения исследования, большинство исследователей обычно идут этим путем:

- Определение потенциального объекта исследования.
- Исследование объекта, чтобы выявить возможные слабые места.
- Попытаться эксплуатировать обнаруженные дефекты.
- В случае успеха, написать подробный отчет по результатам исследования, раскрывающий подробности уязвимости.
- Отправить отчет соответствующей стороне (напр. разработчику ПО, производителю оборудования и т. д.)

Некая аналогия, часто используемая для разъяснения сути исследования уязвимостей – взлом замков. Цель взломщика замков, открыть замок без ключа. Чтобы это сделать, он в первую очередь должен выбрать замок, который хочет взломать и произвести любые необходимые исследования, для

понимания, как этот замок работает. Для этого он будет изучать замок, на предмет выявления любых слабых мест, путем визуального осмотра, технического обследования, изучении всяческой документации от производителя и вероятно, информации о других замках, которые были взломаны, чтобы действовать по аналогии. Однажды найдя слабину, он будет пробовать различные техники, чтобы эксплуатировать ее. В случае успеха, он может быть документирует результаты и использует, чтобы помогать делать замки лучше.

Подобно нашему взломщику замков, исследователь уязвимостей проводит свое время, выискивая ПО, которое он хочет исследовать. Как только подходящий объект будет выбран, исследователи начнут пытаться разобраться, как работает целевой софт и есть ли в нем какие-либо признаки возможных слабых мест. Они занимаются этим, рассматривая поведение ПО в процессе тестирования, изучая исходный код, читая мануалы и проводя тщательные исследования, чтобы получить достаточный уровень понимания. Как только слабые места определены, применяются техники пытающиеся их эксплуатировать. В идеальных условиях, наш исследователь – проинформирует об этих уязвимостях, производителя программного обеспечения. Однако, как и большинство наших взломщиков замков, они могут и не поделиться этой информацией или использовать ее из соображений личной выгоды.

Было отмечено, что не все исследователи в точности следуют этому пути. Одни исследователи, могут выбрать только проведение изысканий и не пытаться эксплуатировать любые найденные ими уязвимости. Другие предпочтут, только эксплуатировать уязвимости и не составлять каких-либо отчетов. Важно запомнить тот факт, что не существует единственно верного варианта исследования уязвимостей и любые методы могут быть взяты и изменены, в соответствии с личными предпочтениями.

Приступим

Если вы заинтересованы в начале исследования уязвимостей, есть несколько вещей, которые вы должны сделать. Мысленно вернемся к нашей аналогии со взломщиком замков – взломщик обладал базовыми знаниями о замках, инструментами, которые применял для их вскрытия, умел находить больше информации о замках и иногда, имел опыт соответствующих манипуляций с ними. Подобно нашему взломщику, мы должны понимать основы функционирования систем, в чем они могут быть уязвимы, каков ассортимент применяемых в большинстве случаев инструментов и как такие инструменты использовать, чтобы эксплуатировать слабые места ПО.

Пространное руководство по исследованию уязвимостей и освоению инструментов, которое вам придется переварить для успешного тестирования на уязвимости, это лишь малая часть и некоторые вещи мы не сможем охватить в одной главе или даже в целой книге. Мы предполагаем, что вы подходите к этой главе с какими-то базовыми навыками, которые понадобились бы исследователю. Наша цель здесь, обеспечить вас основной методикой, которую вы в состоянии усвоить. Эта методика поможет вам систематизировать и упорядочить вашу работу, в отношении планирования и выполнения проектов исследования. Хотя, даже если некоторые ключевые понятия вам неизвестны, практические исследования и обсуждения методики должны помочь пополнить ваш объем знаний и улучшить понимание, невзирая на уровень навыков. Давайте уделим несколько минут быстрому обсуждению некоторых из лучших способов изучения этих тем, независимо от уровня квалификации.

Как могут помочь ранее раскрытые уязвимости

Ключевым источником информации, который будет полезен каждому целеустремленному исследователю, являются отчеты о ранее раскрытых уязвимостях и связанные ресурсы об эксплуатации. Вы можете учиться на них, анализируя огромный набор информации найденной в сети. Если вы хотите разобраться в ключевых техниках, которые раскрывают некоторые исследователи, поисковые системы приблизят вас к нахождению блогов или статей, соответствующих темам ваших

исследований. Известно, что в сообществе информационной безопасности, люди предпочитают делиться своими изысканиями для облегчения будущих исследований. Еще один прекрасный способ научиться – узнавать побольше об уязвимостях и их предыдущих раскрытиях. В предыдущей главе, мы тщательно исследовали несколько раскрытий уязвимостей нулевого дня. В каждом случае были засвечены слабые места или изъяны, способствовавшие обнаружению уязвимости.

Изучение ранее обнаруженных уязвимостей, может подкинуть вам идею на что обращать внимание во время поиска уязвимостей. Однако, собрать эти источники воедино, сложновато для начинающего, поэтому перечислим ключевые ресурсы, которые являются “маст-хэв“, когда вы начнете исследовать ранее раскрытые уязвимости:

- **National Vulnerability Database** (**NVD**): NVD, это репозиторий по известным уязвимостям безопасности. Это великолепный ресурс для понимания существующих типов уязвимостей, а также степени их серьезности (https://nvd.nist.gov/).
- База данных **Common Vulnerabilities and Exposures** (**CVE**): база данных CVE похожа на NVD, но часто включает больше исследований и источников информации для исследователей. Она может прояснить не только то, какие уязвимости существуют, но также каким образом они могли быть использованы и зафиксированы в базе (https://www.cve.org/).
- **Exploit-DB**: база данных эксплоитов, предоставленных исследователями безопасности. Это превосходный ресурс для рассмотрения того, как уязвимости были эксплуатированы в предыдущих сценариях. Исследователи безопасности, как правило разрабатывают их основываясь на CVE, но не всегда. Код содержащийся на Exploit-DB, обычно не проверен, поэтому будьте осторожны на счет того, что загружаете и точно понимайте, что делаете перед тем его запускать (https://www.exploit-db.com/).
- **Full Disclosure**: список рассылки, посвященный раскрытию уязвимостей безопасности. Подписавшись на эту рассылку, вы будете получать уведомления всякий раз, когда раскрывают новые уязвимости (https://seclists.org/fulldisclosure/).
- Последние доклады с **DEFCON**: DEFCON является старейшей хакерской конференцией в мире. Она проходит каждый год в Лас-Вегасе, штат Невада. Так как, не каждый может посетить конференцию, большинство последних докладов с DEFCON выкладываются в сеть для всеобщего обозрения. Исследования с DEFCON, обычно недоработаны. Они представлены реальными исследователями и не доведены до уровня коммерческих продуктов. Некоторые доклады настолько фундаментальны для конкретных областей исследований, что на них все еще ссылаются почти десять лет спустя после публикации (https://www.youtube.com/user/DEFCONConference).

По мере того как вы совершенствуете свои навыки и начинаете специализироваться, может оказаться полезным начать поиск ресурсов, относящихся к той специализации, которая вас интересует. Например, если вы ищете уязвимости в Windows, хороший источник для их поиска - **Microsoft Security Bulletin**. Если вы ищете уязвимости Linux, хороший ресурс – новостная лента **Ubuntu Security Notice**. Для поиска общих уязвимостей веб-приложений, подойдет **Open Web Application Security Project** (**OWASP**). Если уж не это, ваш любимый поисковик должен стать вашим лучшим другом в выявлении потенциальных уязвимостей. Ресурсы необходимые для дальнейшего пополнения ваших знаний уже ждут вас там.

В качестве дополнительных источников, вы могли бы рассмотреть печатные материалы по теме. Скажем, вам интересно разобраться, на фига вашему новенькому кухонному девайсу Wi-Fi для работы. Специальных книг по безопасности о вашем устройстве может не быть, но могут быть отдельные публикации по разработке встраиваемого оборудования, которые могут помочь вам узнать, как эти устройства создаются разработчиками. Изучение того, как устроено и запрограммировано оборудование, способствует значительной части понимания, необходимого чтобы узнать, какие уязвимости могли быть туда заложены. Однако не останавливайтесь на достигнутом, подробнее

изучите проблемы, с которыми другие могли столкнуться при использовании похожих устройств. Обращайтесь к раскрытиям CVE для похожих устройств, ищите руководства по техобслуживанию, которые рассекречивают скрытые функции оборудования или даже берите онлайн-уроки по разработке ПО или аппаратному взлому, чтобы разобраться как оборудование устроено и как оно функционирует. Какие бы ресурсы вы ни выбрали, убедитесь, что у вас хватает времени, чтобы хоть чуть-чуть разобраться в технологии, перед погружением непосредственно в поиск уязвимостей.

В заключение помните, что практика ведет к совершенству – так не бойтесь же испачкать руки и экспериментируйте самостоятельно! Вместе с этим базовым примером по поиску информации, вам нужно будет совершенствовать свои навыки исследования. Поговорим о том, как мы выбираем нужные объекты исследования.

Выбираем объекты исследования

Начало выбора объекта исследования – захватывающий процесс. На рынке представлен огромнейший выбор ПО. Вы сосредоточитесь на десктопных приложениях? Веб-приложениях? Как на счет плагинов, разработанных для программ? Возможно, устройство, которое вы купили и подключили у себя дома? Перед тем, как вы начнете суетиться или исследовать случайное ПО, вам стоит вспомнить, что *если мы проваливаем планирование, то мы планируем провал*. Чтобы преуспеть, нам потребуется стратегия для выбора объектов исследования. Перейдем к отличительным признакам и соберем их в стратегию по выбору объектов исследования:

Сохраняйте интерес – в любом исследовании выбирайте цель, в которой вы заинтересованы. Это будет поддерживать вашу мотивацию на протяжении всего исследования.

Выбирайте, потенциально уязвимые цели – выбирайте цель, которая скорее всего является уязвимой. Можно основываться на таких фактах, как сложное устройство цели, возраст кодовой базы и число пользователей.

Выбирайте что-нибудь в рамках своих возможностей – попытайтесь удостовериться, чтобы ваша цель, обладала чем-то, что вы можете тестировать без привлечения ресурсов, которых у вас нет. Например, вы могли бы реально заинтересоваться, исследованием уязвимостей безопасности автомобиля Тесла. Однако, если у вас нет возможности приобрести Теслу или нет возможности потерять деньги, из-за того, что исследование могло бы разрушить вещь, которую вы исследуете – это не самая подходящая цель.

Выбирайте что-нибудь такое, где у вас есть разрешение на тест – несмотря на то, что могло бы быть интересно, сосредоточиться на тестировании популярных веб-приложений, размещенных в каком-нибудь облачном окружении, вы можете не иметь разрешений на их тестирование или иметь крайне ограниченные из-за того, что вендор запустил тестирование по программе баг-баунти, только для авторизованных пентестеров. Вам следует избегать этого, особенно если вы собрались добывать CVE. Наоборот, фокусируйтесь на программном обеспечении, которое можете запускать на своем оборудовании или имеете доступ для тестирования.

Выбирайте цели с критическими данными и полезными качествами – вам следует выбирать цель, эксплуатировать которую выгодно. Это можно определить по таким признакам, как секретные данные, обрабатываемые на целевой системе и потенциальная подверженность эксплуатации. Например бесплатная программка, помогающая пользователям создавать поздравительные открытки, могла быть менее интересной хакерам, чем простенький инструмент управления взаимодействием с покупателями, который может принимать такую информацию, как данные кредитных карт, банковскую информацию и ценные персональные данные.

Соотносите исследование со своими силами – вам стоит учитывать свой опыт и уровень навыков, когда выбираете объект исследования. Выбирайте цель, в рамках своих компетенций или что-то, что мотивирует вас к развитию настолько, чтобы вы преуспели в своем исследовании. В противном случае, вы можете разочароваться.

Убедитесь, что вы имеете изолированное окружение – в заключение, нам необходимо убедиться, какую бы цель мы ни выбрали, мы развернули выделенное окружение ее для тестирования. Вы могли бы заинтересоваться тестированием мэйнфрейма использовавшегося для разработки ПО на КОБОЛе в 80-х. Возможно у вас есть легкодоступная копия софта, однако если у вас нет запасного мэйнфрейма или нет возможности его эмуляции, тогда это не самый подходящий выбор.

Несмотря на то, что эти общие рекомендации полезны, лучше выполнить практические примеры. Рассмотрим пример из этого руководства и применим его.

Находим интересующие вас цели

С тех пор, как программное обеспечение стало управлять современным миром, люди создали ПО для разных целей. Мы с тех же пор, стараемся оставаться сосредоточенными на занятии по выбору категорий ПО, представляющего интерес. Чисто ради этого упражнения предположим, что вы *действительно* интересуетесь лошадьми. Если вы по-быстрому заглянете в Google, вы выясните, что стоимость владения и содержания лошади в Соединенных Штатах, порядка 10000$ в год. Владельцы тратят много времени, сил и денежных средств, чтобы обеспечить своим лошадям достойный уход. Поскольку при владении лошадью, нужно многое учитывать (случки, выставки, верховая езда, обслуживание конюшни и услуги ветеринара), могло быть написано программное обеспечение, чтобы помочь в этом. Вы безусловно можете использовать для поиска ПО свой любимый поисковик, однако одним из лучших способов найти нишу программного обеспечения для исследования – это воспользоваться **SourceForge** (https://sourceforge.net/).

После набора **Horse Software** в поисковой строке в шапке SourceForge, появились около 75 вариантов для нашего изучения (см. рис. 3.1):

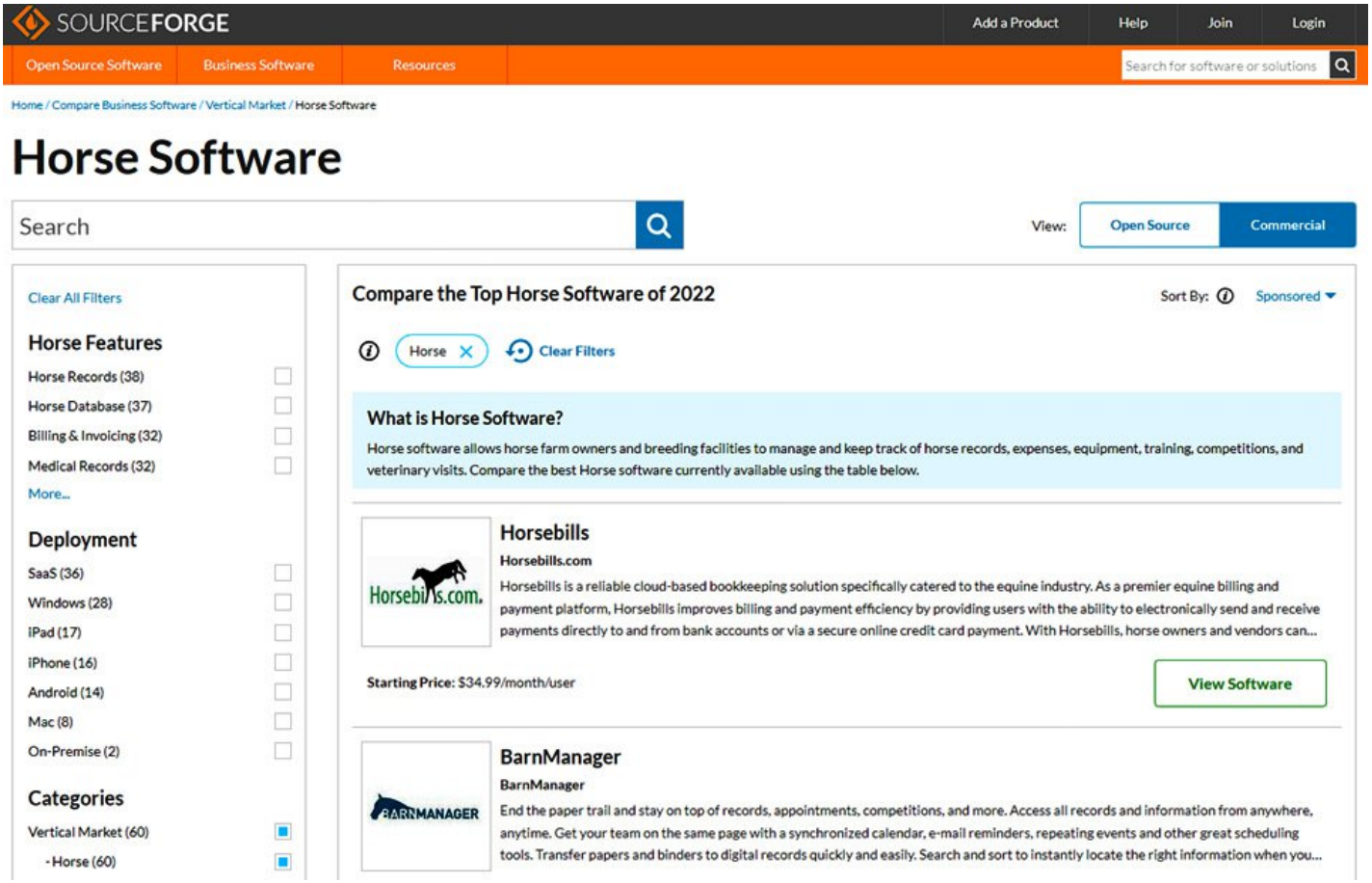


Рис. 3.1 – Используйте SourceForge, чтобы находить ПО, в тестировании которого вы заинтересованы, с помощью функции поиска

Мы управляем поиском ПО, которое интересно нам лично (гипотетически), и получаем массу вариантов из чего выбирать. Теперь, двинемся к следующей паре категорий, поиску потенциально уязвимого ПО, учитывая ограниченность в средствах..

Доступный к скачиванию софт, скорее всего содержащий уязвимости

Следующая вещь, которую нам потребуется сделать – найти ПО, которое скорее всего уязвимо и может быть исследовано с минимальными личными вложениями. Сегодня “вероятно уязвимое“ ПО может находиться под защитой с большой долей вероятности, однако в качестве “метода тыка“, я мог бы предложить следующий совет: **устаревший софт, с большой долей вероятности уязвим**. Еще одной зацепкой, могло бы стать то каким образом софт используется. Несмотря на то, что SaaS-решения часто оказываются уязвимыми, мы не всегда можем иметь доступ для их тестирования. SaaS-решения, всегда размещены на оборудовании, находящемся в собственности поставщика ПО. По сути, они немногим более закрыты, чем если бы нам разрешили установить приложение на устройства, вроде компьютеров или телефонов. В сущности, лучший вариант сделать это с минимумом усилий и максимальной отдачей – исследование приложений, которые находятся большей частью под вашим контролем.

Еще одно соображение, на которое следует обратить внимание - это связь программного обеспечения с техникой. Хотя это не всегда в порядке вещей, ПО используемое в нетехнических отраслях, часто обходится вниманием специалистов по безопасности. В нашем примере, где мы выбрали софт относящийся к коневодству, он является отличной целью имеющей по большей части нетехническую аудиторию. Вряд ли это крупный корпоративный рынок с высокими требованиями к безопасности или осознающей тематику безопасности основной массой пользователей.

Учитывая все это, давайте сосредоточимся на каком-нибудь старом приложении, которое можно установить в операционной системе Windows. SourceForge годами является отличным вариантом сузить область поиска и отфильтровать ПО по характеристикам. Это позволит нам сгруппировать приложения для Windows, как видно на рис. 3.2:

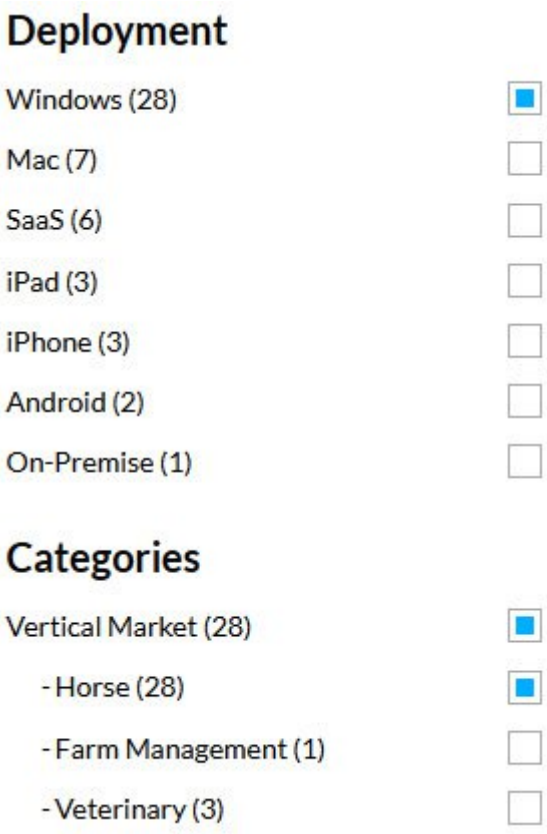


Рис. 3.2 – У SourceForge есть фильтры, которые полезные для сужения выбора конкретных объектов

Вместе с сужением области поиска по нашим критериям, существует еще несколько атрибутов, на которые мог бы ориентироваться исследователь. В нашем случае мы ищем софт, который мы можем скачать и попробовать, чем быстрей, тем лучше и мы рассчитываем сделать это без манипуляций с деньгами или личных вложений. После просмотра доступных вариантов, как показано на рис. 3.3, исследователь мог бы выбрать вариант, предлагающий бесплатную демо-версию доступную к скачиванию и установке:

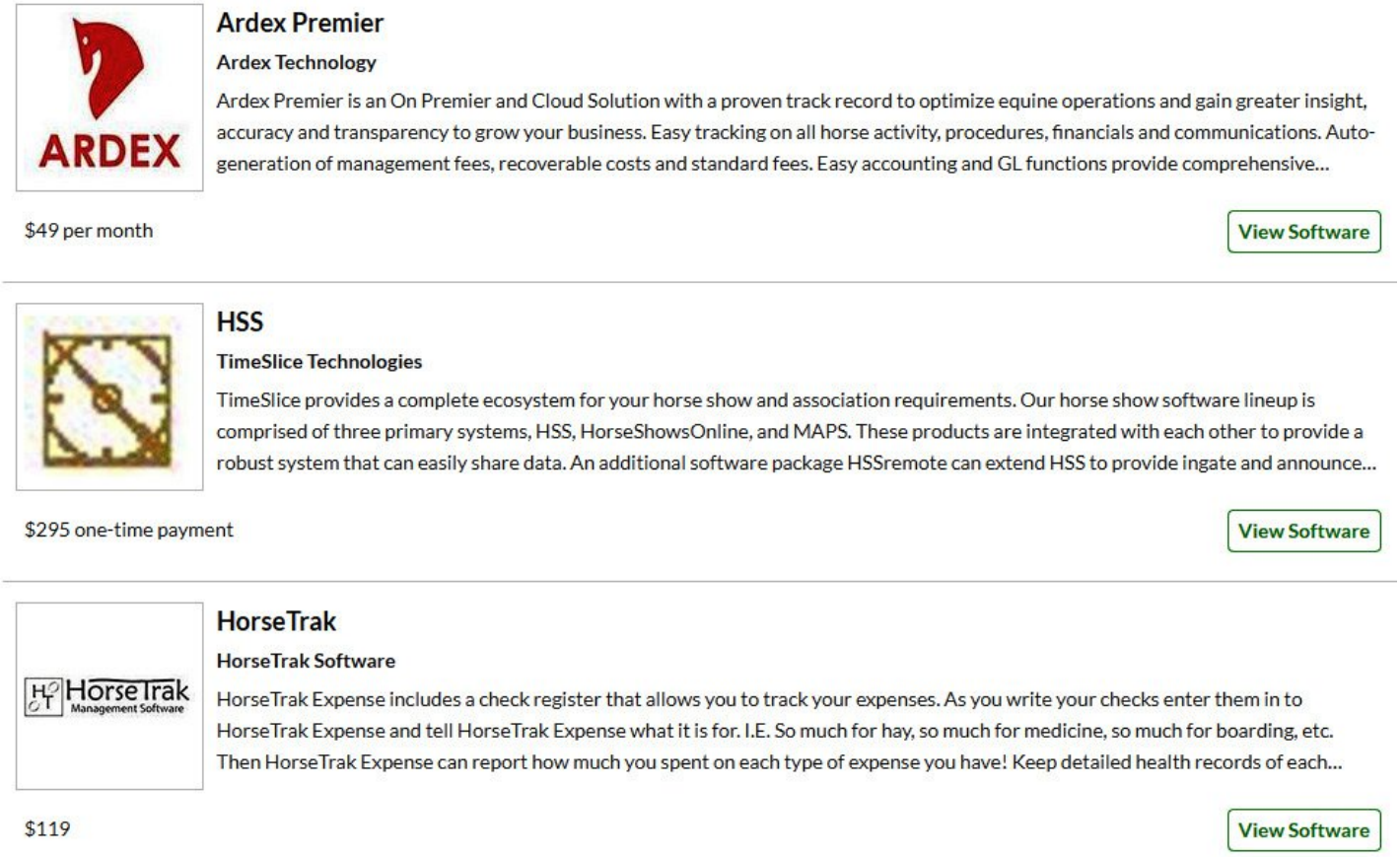


Рис. 3.3 – Какие-то из нескольких вариантов с доступными демо-версиями, могли бы представлять интерес

Выберите минутку, чтобы изучить любые маркетинговые материалы, которые могли бы найти, техническую документацию, которую возможно скачать и прочую информацию. Как исследователь, вы должны понимать особенности контекста в котором это ПО могло бы использоваться – так вы сможете разобраться, почему кто-то мог бы выбрать этот софт в качестве цели. Как показано на рис. 3.4, одна из вещей, которые мы ищем в материалах примера, это скачивание ПО для тестов:

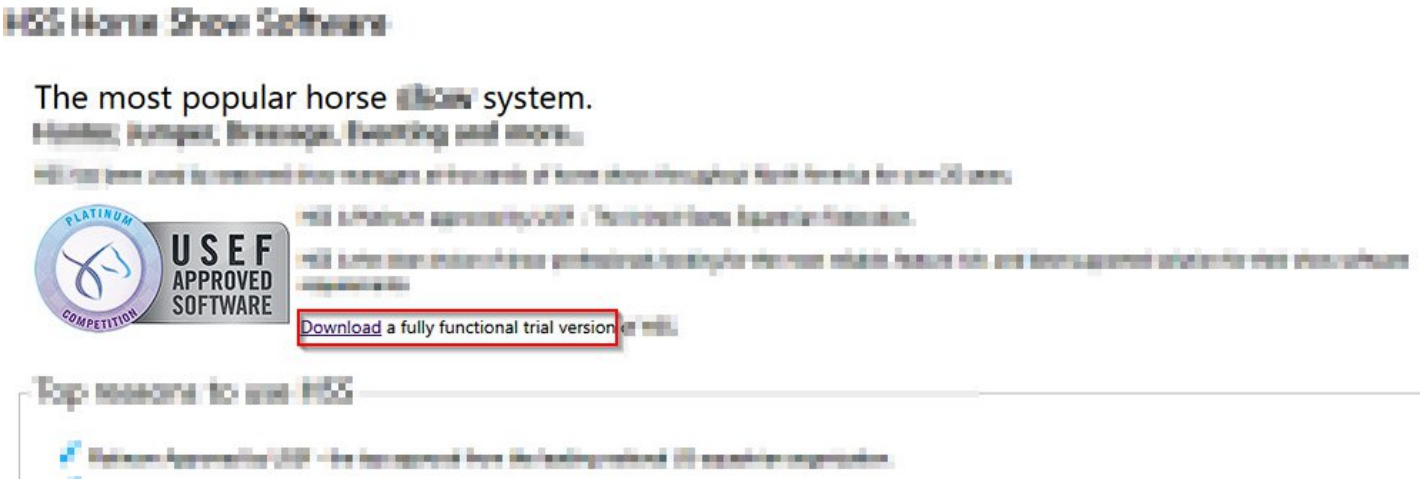


Рис. 3.4 – Материалы веб-сайта помогут раскрыть подробности, которые могут сделать объект, более привлекательным для исследования

После того, что мы изучили по материалам веб-сайта, мы приходим к пониманию, что исследуемое нами программное обеспечение, используется для управления регистрацией учетных данных, хранит личную идентификационную информацию, принимает информацию кредитных карт и

несколько других атрибутов. Дополнительно, ПО выглядящее похожее на это, было собрано несколько лет назад – на скриншотах из маркетинговых материалов, показана дата 2018 года (см. рис. 3.5). Остановка цикла разработки на несколько лет, как правило многократно повышает риск появления уязвимостей безопасности:

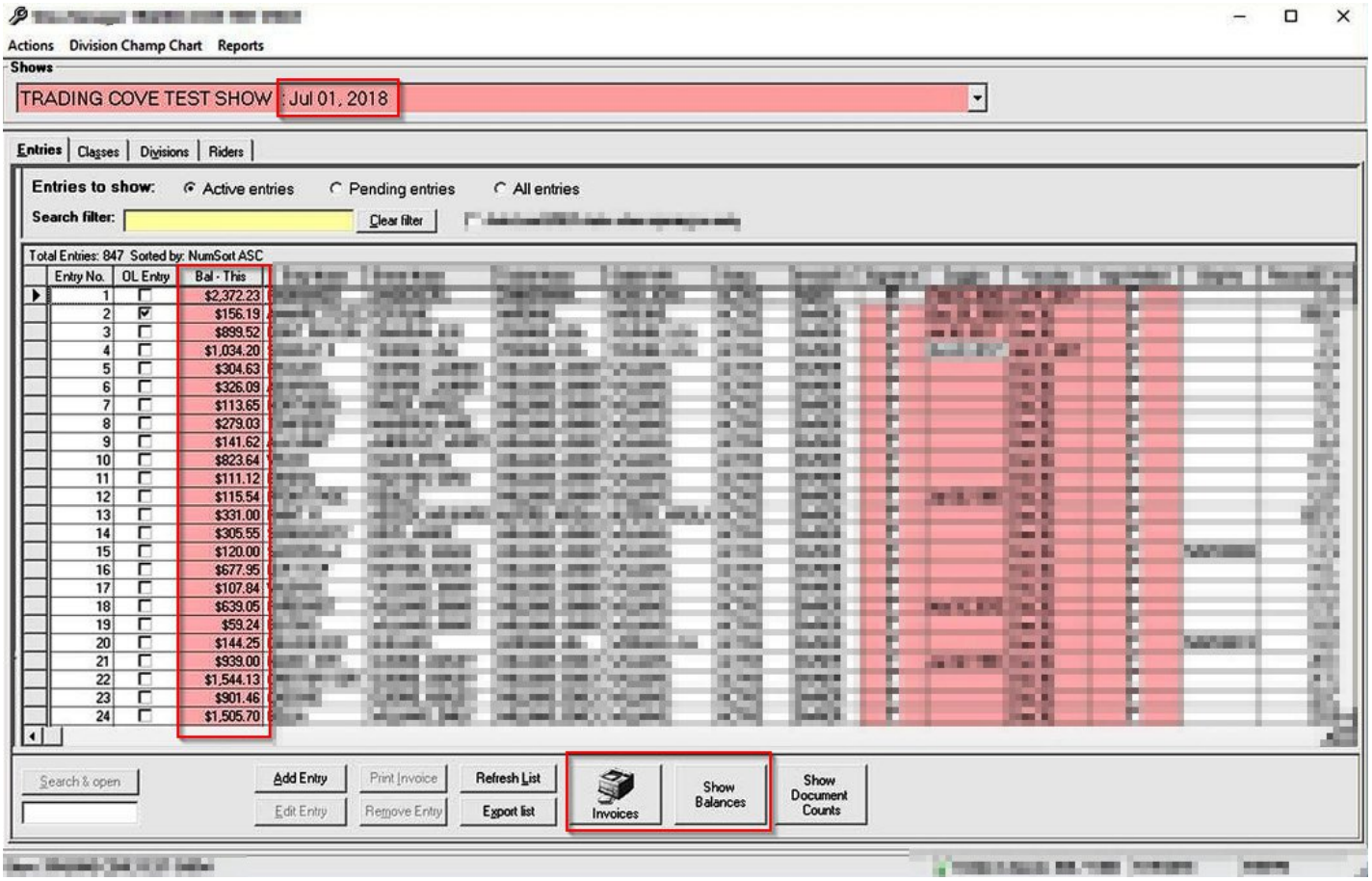


Рис. 3.5 – Скриншоты в маркетинговых материалах, могут приоткрыть функциональность до проведения тестирования, как эта доступная к просмотру накладная

Итак, в итоге – мы нашли интересующий нас софт. Это программное обеспечение, которое может иметь уязвимости в безопасности, доступное для скачивания и тестирования, а также содержит информацию о выставленных счетах и персональные данные. Согласно нашим условиям – допустим, что мы новички в исследовании. К счастью программы для Windows, являются легкой целью и многие уязвимости оказываются просты даже для начинающих их исследование (как мы выясним в нашем первом примере). Теперь нам потребуется подготовить изолированное окружение для проведения тестов.

Если у вас компьютер с Windows, вы вероятно сможете создать отдельные виртуальные машины с копиями ОС Windows, без особого труда. Майкрософт, предлагает эту функциональность в качестве встроенной в большинство операционных систем, как инструмент под названием Hyper-V. Есть также другие известные коммерческие и некоммерческие варианты на которые вы можете положиться для создания изолированных установок Windows. Мы хотим сконцентрироваться на событиях, которые происходят на компьютере, где вы проводите свои тесты. Поэтому, лучше установить тестируемое программное обеспечение в каком-то изолированном окружении, чтобы быть уверенным в отсутствии возможных взаимных помех между приложениями. Если вы незнакомы с гипервизорами или виртуальными машинами, мы предложили некоторые рекомендации в разделе *Знакомимся с распространенными исследовательскими инструментами* этой главы, где мы рассматриваем возможные варианты создания и запуска изолированных окружений.

Итак, в конце концов, мы заполучили подходящую цель, возможность и ресурсы для проведения тестирования. Все наши виртуалки проверены. Теперь приступим к обсуждению того, каким образом мы будем тестировать программное обеспечение на уязвимости, используя тестовые пакеты.

Обнаружение уязвимостей с помощью тестов

Раз у нас есть подходящая цель (или цели), пора начать исследование. Как и в большинстве предыдущих разделов, планирование и стратегия должны занимать ведущее место в нашем подходе. Одним из способов сделать это, является создание и (или) использование тестовых пакетов. Тестовый пакет, это набор инструкций, который можно применить для выявления возможных слабых мест в программном обеспечении. Многие исследователи безопасности, предпочитают выполнять большую часть первоначального исследования и тестирования, вручную, однако когда вы станете больше знакомы с потенциальными способами “ронять“ системы, автоматизация могла бы быть полезной для изучения. Сейчас же, на протяжении следующего раздела мы будем определять, что такое тестовый пакет и как их разрабатывать на практике.

Тесты – ликбез

В мире разработки программного обеспечения, тестовые пакеты – один из важнейших инструментов обеспечения качества инженерной работы, для проверки ПО на отсутствие багов. По аналогии, профессиональные охотники за багами на баг-баунти и исследователи уязвимостей, точно также получают существенную выгоду, когда осваивают методики тестирования. Простейшее определение тестового пакета – это набор условий или переменных, которые будут определять, работают ли функции ПО так, как должны (по замыслу разработчиков). Высококачественные тестовые пакеты, содержат три элемента:

Краткое изложение целей – здесь мы разместим сводку причин тестирования и целей, связанных со слабыми местами ПО, которые мы наметили во время тестирования. В этом разделе, мы будем очерчивать, где и когда тест принес результат и кратко определим область его применимости.

Инструкции тестирования – хороший набор тестов, будет содержать конкретные примеры и инструкции по выполнению проверки на наличие дефектов заданного типа. Это будет включать ссылки на инструменты и техники, которые могли бы быть использованы. Инструкции тестирования, могут также содержать дополнительные справочные материалы, там где это имеет смысл.

Исправления – поскольку вы обнаруживаете уязвимости, будет полезным иметь шаблоны уведомлений, чтобы дать объяснение сути уязвимости и процесса ее исправления, которые часто применяют для снижения риска или уменьшения последствий. Когда вы начинаете сообщать об уязвимостях, людям создавшим программное обеспечения – вещь, которую они хотят узнать в первую очередь, это как исправить проблему.

В то время, как при разработке программного обеспечения, назначением тестирования – является проверка работоспособности заявленных функций ПО, цель преследуемая нами, будет другой. Мы сосредоточимся на определении, являются ли компоненты ПО восприимчивыми к распространенным типам уязвимостей. Хорошо написанный тестовый пакет, должен быть прозрачным, лаконичным и простым для понимания, он также должен быть повторяемым, в том смысле, что сможет использоваться для проверки, что функциональные возможности ПО работают как задумано, даже после изменения кода соответствующих компонентов ПО в будущем.

Создание эффективных тестовых пакетов, может оказаться задачей требующей значительного напряжения сил. Важно помнить, что цель – найти баги и (или) информацию, которая приведет к нахождению багов, а не проверить, что система работает должным образом. Следовательно, тестовый

пакет должен быть спроектирован таким образом, чтобы иметь высокие шансы выявления дефектов, в то же время будучи лаконичным и простым для понимания. В некоторых случаях, тестовый пакет может также включать дополнительную информацию, такую как скриншоты или лог-файлы, которые могут пригодиться при ближайшем изучении дефекта в будущем.

Итак, с какого места мы начнем создание тестового пакета? Что ж, хорошая новость в том, что в большинстве случаев, их лучше всего скопировать и использовать с ресурсов в свободном доступе.

Избегайте изобретения велосипеда

Написание и сборка эффективных тестовых пакетов, может быть довольно затруднительной, особенно для начинающих исследователей безопасности. Каким образом вы могли бы придумать тестовый пакет проверки на некое слабое место, когда вы понятия не имеете, как его для начала выявить? К счастью большая часть кропотливой работы, связанной с созданием тестов, уже выполнена исследователями безопасностями. Поговорим же о таких ресурсах, сейчас же.

OWASP, является одним из лучших, если вообще не лучшим источником, если вы хотите приступить к обзору шаблонов тестовых пакетов. OWASP специализируется конкретно на веб-приложениях, однако краткий и упорядоченный метод тестирования ресурсов, опубликованный в качестве примера, заслуживает внимания каждого интересующегося темой. Применительно к веб-приложениям, он представляет отличное руководство по тестированию, включающее всесторонние примеры тестирования веб-приложений. **Web Security Testing Guide** (**WSTG**), на момент выпуска этого материала, имело версию 4.2 и состояло из 400 страниц, хорошо оформленных справочных материалов по тестированию, по полочкам раскладывающих задачи тестовых пакетов, демонстрирующих как производится тестирование на уязвимость, а также оформление возможных исправлений (см. рис. 3.6):

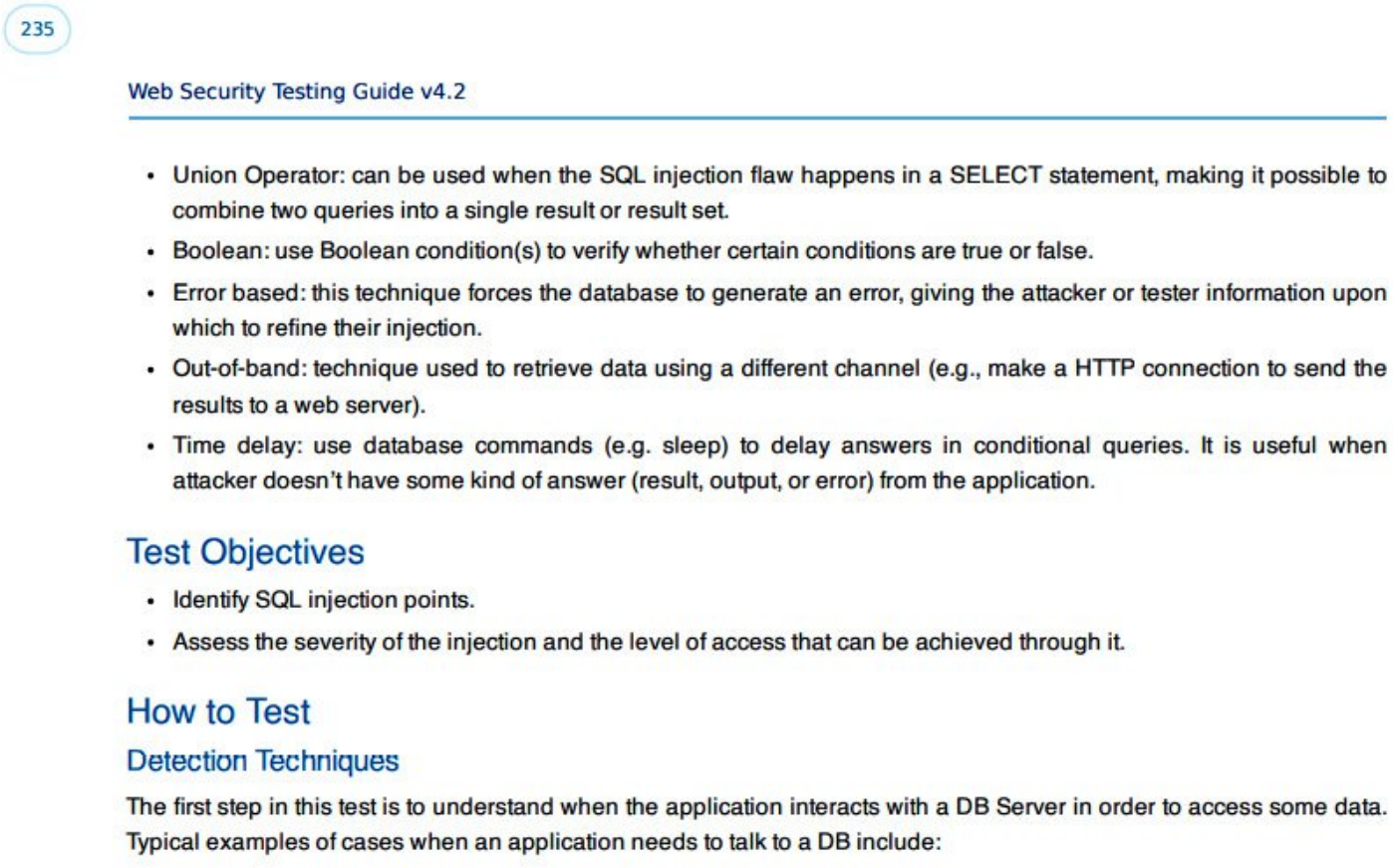


Рис. 3.6 – Примеры тестов в OWASP WSTG

В дополнение к превосходному WSTG, есть еще много ресурсов от OWASP, на которые исследователи могут полагаться. Команда OWASP, публикует ресурсы связанные с API и тестированием мобильных приложений, оба из которых быстро обнаруживаются в прекрасных разработках, которые объединены в структуры большей частью аналогичные WSTG.

Для тестов, связанных с инсталлируемым ПО, к примеру для ОС Windows, UNIX-like систем, мобильного тестирования и тестирования встраиваемых устройств, значительно меньше ресурсов объединенных этими тематиками. Иногда это подталкивает исследователя собирать эффективные тестовые пакеты, к счастью мы изучим эти моменты подробнее, чуть позже в этой главе.

Тестовые сценарии, особенно ценны для начинающих свои исследования и могут пригодиться в качестве образца, для подающих надежды исследователей, которые пока не обладают значительным опытом в поиске багов. Предостерегаем, однако не принимать тестовые сценарии, за истину в последней инстанции при изучении и тестировании приложений на дефекты. Думайте о тестовых пакетах, как о поваренной книге. Когда вы читаете рецепт, там для вас в правильном порядке, перечисленны все ингридиенты. Если вы следуете инструкциям, вы сможете приготовить довольно вкусное блюдо. Однако иногда, вам хочется положить свой ингридиент из любопытства, как новые вкусы, проявят себя вместе. Может быть у вас нет всех компонентов рецепта, указанных в нем или на месте рецепта в книге пятно и вы не можете прочесть его полностью, поэтому вам придется импровизировать. То же самое, справедливо и для отдельных тестов и тестовых пакетов. Следуйте инструкциям, но не бойтесь творческого подхода, когда вы хотите поэкспериментировать или ваши объекты тестирования не соответствуют точной спецификации в ваших комплексных тестах.

Получив хорошее представление о доступных нам ресурсах, давайте поговорим о том, что такое тестовые пакеты и как они могли бы быть использованы в эффективной стратегии исследования.

Создание эффективных тестовых пакетов

Тестовый пакет, это сборник отдельных тестов, которые могут применяться тестирующими. Считайте WSTG, который мы обсуждали ранее – тестовым пакетом. Во время исследования конкретных наборов ПО, может возникать искушение протестировать абсолютно все, однако иногда такая стратегия дает не лучший результат ваших изысканий. Если вы к примеру решили, что хотите сосредоточиться на одном из конкретных типов программного обеспечения, а в таком ПО нацелились на какой-то вид уязвимости, тогда – ваш кратчайший путь из пункта А в пункт Б, совершить вполне определенный набор действий над тем, что вы хотите протестировать. Это позволит вам тестировать софт быстро и эффективно, сосредоточиваясь именно на своих интересах. Если вы оставляете исследование без четких критериев и принимаете подход “тестировать абсолютно все“, при котором вы выполняете все тесты какие сумеете, в конце концов вы может, что-нибудь и найдете. Однако, лучше организованный исследователь, за то же время выполнит тестирование множества целей, и вероятно найдет больше уязвимостей за единицу времени.

Как мы уже выяснили, тесты – это здорово, но где все сходится воедино, так это в вашем тестовом пакете. Тестовые пакеты, несколько отличаются, при рассмотрении вашего тестирования с той и другой стороны. С одной стороны, это способ сортировки тестов и распределения их по категориям. Однако с другой стороны, пакеты также являются объединением всего необходимого вам, для полноценных комплексных тестов (конкретного назначения).

Тестовые пакеты очень помогают в сценариях, когда вам требуется объединить вместе, ключевые тесты предназначенные конкретной технологии, которую вы сейчас исследуете. Обладая тестовым пакетом, который фокусируется на определенной области, вы можете снизить временные затраты на исследование и посвятить больше времени тестированию.

Лучшим способом собрать тестовый пакет, является использование существующих источников, которые могут проинформировать, как бы вы могли объединить тестовые примеры. В сети доступно множество отличных ресурсов, по тестированию безопасности. Часто, их можно приспособить для ваших конкретных нужд. Мы рассмотрели OWASP, в качестве прекрасной стартовой площадки, когда искали широко распространенные источники по тестированию безопасности. Другие ресурсы, такие как NVD и MITRE, помогут получить ретроспективные данные по проверке на уязвимости, обнаруженные ранее. Еще один замечательный ресурс, который смог бы помочь вам собрать, какой-нибудь эффективный тестовый пакет, это **MITRE ATT&CK** framework. ATT&CK, определяет типы атак, которые будут использовать хакеры, а защитники применяют это для планирования оборонительных тактик. Это может оказаться весьма полезным, в качестве справки по распространенным типам вторжений и атак. Освоение распространенных моделей атак и отбор их в тематические пакеты для тестирования, отличное начало по конфигурированию вашего исследования. Каждый из этих векторов атаки, тщательно документирован в ATT&CK framework и стоит времени, потраченного на изучение.

В заключение, важно не забывать, что тестовые пакеты это “живые“ документы, а следовательно их нужно регулярно обновлять. Как только обнаружена новая уязвимость или выпущены новые версии программного обеспечения, ваш тестовый пакет должен быть обновлен с отражением этих изменений. Поддержание вашего тестового пакета в актуальном состоянии, будет гарантировать, что он остается полезным и соответствует текущим реалиям.

Написание ваших собственных тестов

Написание своих собственных тестов, это способ набраться опыта в понимании уязвимостей. В жизни каждого исследователя, они потребуются для тестирования приложений, о которых очень мало собранной информации, по тестированию на определенные типы уязвимостей. Что мы сделаем сейчас – это пройдем создание собственного теста, который будет направлен на тестирование известного изъяна и уязвимости в ПО для Windows, часто называемой **DLL injection**.

Нашим первым шагом в создании эффективного документирования, это использовать что-то, что поможет нам упорядочить нашу документацию. Мы будем использовать один из множества популярных кросс-платформенных инструментов с открытым исходным кодом, применяемых пентестерами: **CherryTree**. Продвигаясь по этому разделу, вы можете скачать и установить CherryTree для операционных систем Windows, MacOS или Linux с ее веб-сайта (https://www.giuspen.net/cherrytree/). Как-нибудь скачайте, установите и запустите его. Запустив, CherryTree вы должны увидеть, нечто выглядящее как текстовый редактор (см. рис. 3.7):

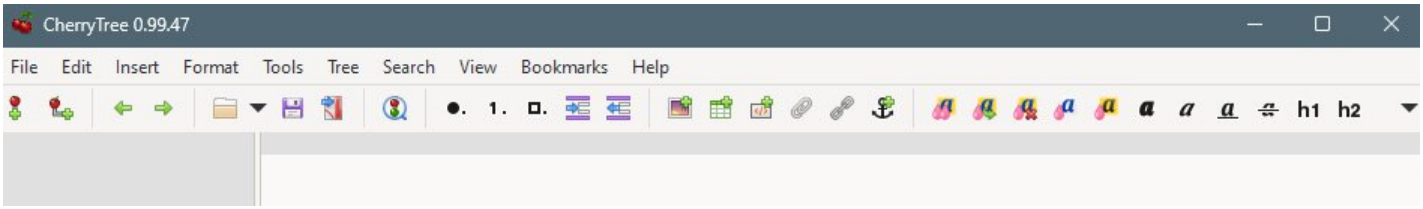


Рис. 3.7 – Первый запуск CherryTree

Лучшее качество CherryTree, это то, что она обладает великолепной организационной функциональностью, так что давайте выберем время, чтобы упорядочить структуру наших тестов. CherryTree использует соглашение об именовании *узлов (nodes)*, для описания иерархических отношений, которые мы можем создавать между документами. Так как область тестирования, которое мы выполняем, фокусируется на Windows, давайте создадим первый узел, который упорядочит наше тестирование в тестовом пакете (см. рис. 3.8):

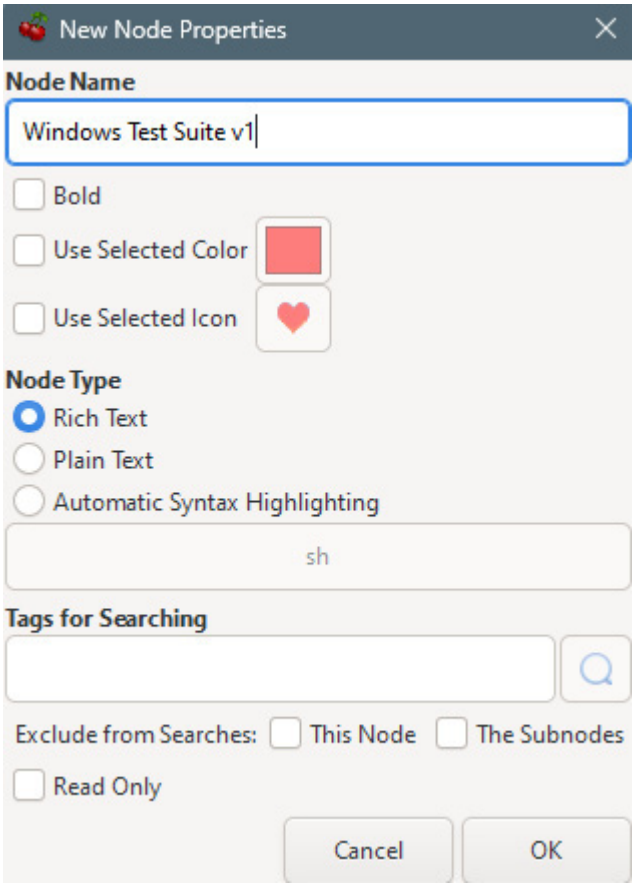


Рис. 3.8 – Создание нашего первого узла в CherryTree

Мы продолжим этот процесс и хотя окончательно это вам решать – сейчас создадим рекурсивные связи между заметками, для этого примера. Первый узел, должен описывать отдельный тест, аналогичный тому, что мы рассмотрели. В этом случае, мы создадим тест для **DLL Hijacking**. Продолжая упорядочивать структуру нашего примера, мы должны включить дочерний узел **TESTING NOTES**, где мы сможем сохранять заметки об исследовании, затем еще один рядом - **TEST CASE**, для хранения описания нашего теста (см. рис. 3.9):

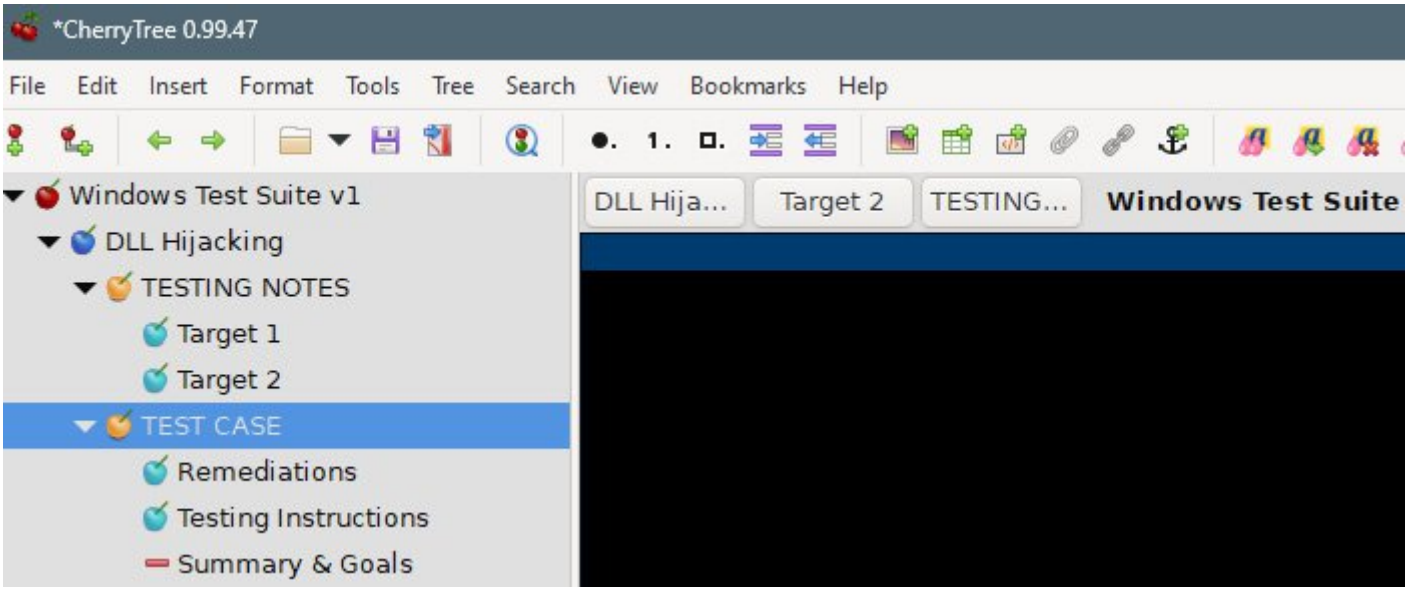


Рис. 3.9 – Конфигурирование взаимосвязей узлов в CherryTree, помогает упорядочить содержимое

Предложенная нами структура, удобна для ведения записей о тестировании наших целевых объектов, создании документации по краткому изложению целей, инструкциям тестирования и

исправлениям. Давайте погрузимся непосредственно в построение наших тестов, взяв за основу наши начальные критерии. Приступим к заполнению раздела теста – **Краткое изложение целей**.

Краткое изложение целей

Чтобы заполнить этот раздел нашего теста, нам необходимо четко понимать, что мы тестируем, краткое описание уязвимости и какие результаты хотим получить от тестирования. Прежде всего, если вы не слишком много знаете об этой уязвимости, проведите какое-то первоначальное исследование. Вы можете использовать свою любимую поисковую систему, для выполнения большей части первоначального поиска, однако мы советуем продолжить усердно работать с ресурсами, которые мы упоминали ранее, такими как MITRE. Например, если вы на начальном этапе посвящали время сайту MITRE, вы могли бы найти их страницу с ATT&CK, которая является всеобъемлющей информационной структурой, описывающей большинство уязвимостей в контексте их использования в составе примеров. Для нашего примера с DLL injection, это не исключение и послужило отличным источником информации (см. рис. 3.10):

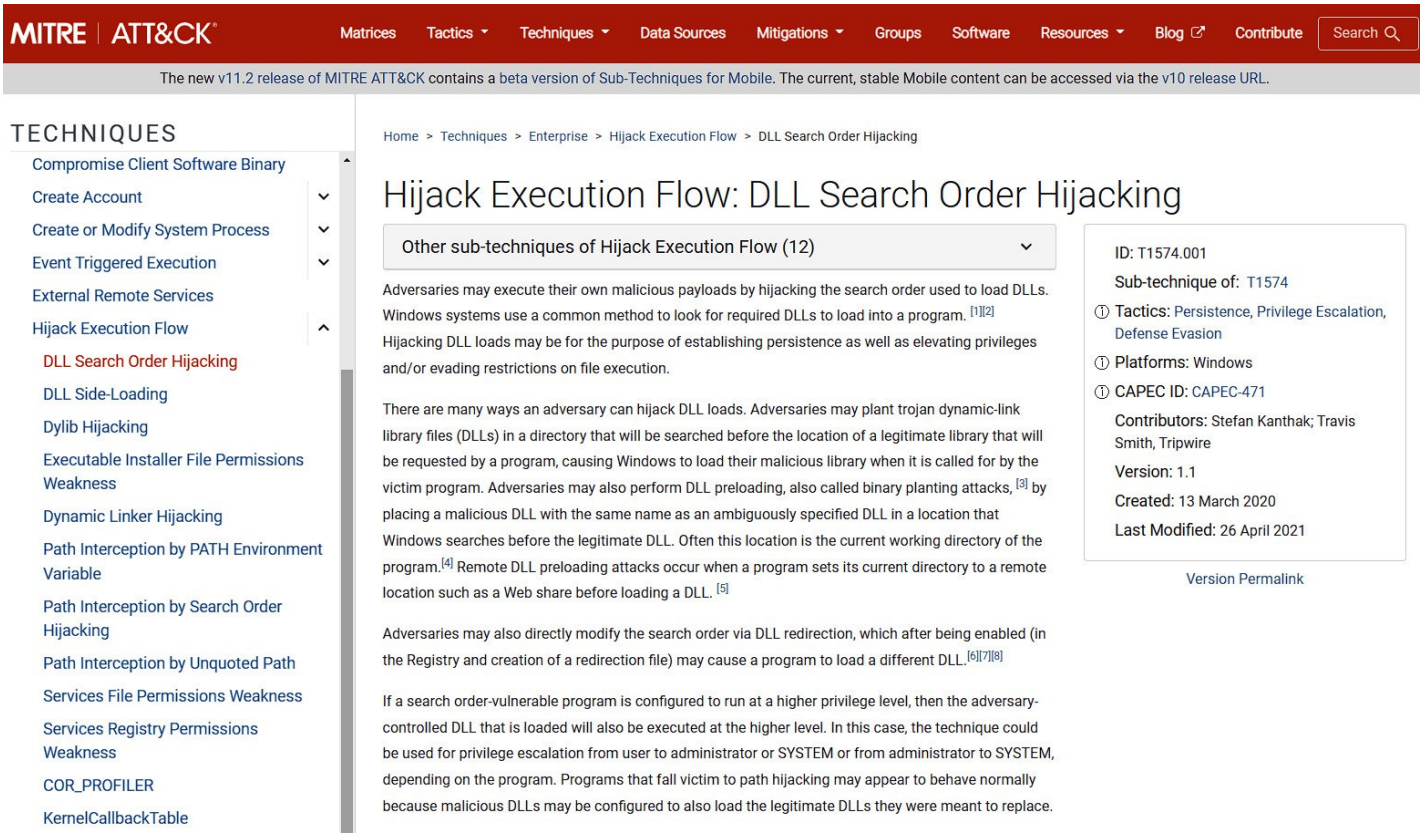


Рис. 3.10 – MITRE ATT&CK, это бесценный исследовательский ресурс для изучения хакерских техник

Теперь у нас есть примеры CVE с DLL hijacking и информация о техниках hijacking-а. Если вы впервые рассматриваете этот класс уязвимостей, потратьте достаточно времени на их изучение. Как только вы разберетесь, начинайте создавать свой шаблон краткого описания. Хотя вы могли бы его просто скопипастить, возможно полезно потратить время на создание собственного описания, основываясь на том, как *вы* понимаете уязвимость. Основываясь на полученных результатах, мы могли бы предложить следующее краткое описание, основанное на наших источниках:

DLL hijacking, это вид уязвимости, которая использует метод загрузки DLL ОС Windows, для выполнения вредоносного кода. Это может быть совершено как путем размещения вредоносной DLL в каталоге, который является частью системного пути поиска, так и через внесение изменений в реестр, для указания на вредоносную DLL. Когда приложение пытается загрузить DLL, Windows будет искать ее в предопределенном порядке. Если DLL не найдена на в одном из

каталогов, Windows будет искать ее в текущем каталоге. Это может быть использовано атакующим, который поместил вредоносную DLL в тот же каталог, откуда легальное приложение загружает динамические библиотеки. Легальное приложение загрузит и выполнит код из вредоносной DLL и атакующий может использовать это для получения контроля над приложением или системой.

DLL hijacking, является серьезной уязвимостью безопасности, так как может использоваться для выполнения произвольного кода с правами приложения или процесса, загрузившего DLL. Это может привести к повышению привилегий и позволить злоумышленнику захватить полный контроль над системой. DLL hijacking часто используется в связке с другими атаками, такими как социальная инженерия или распространение малвари.

Основываясь на этой информации, мы могли бы подытожить наши цели, следующим образом:

Цель данного теста – найти слабые места в способе загрузки DLL ОС Windows, через неверные конфигурации путей в программном обеспечении или разрешений, которые могли быть неправильно сконфигурированы, таким образом, который допускает повышение привилегий или выполнения произвольного кода.

Наши заполненные заметки в секции **Краткое изложение целей**, должны выглядеть примерно так, когда мы завершим создание нашего описания (см. рис. 3.11):

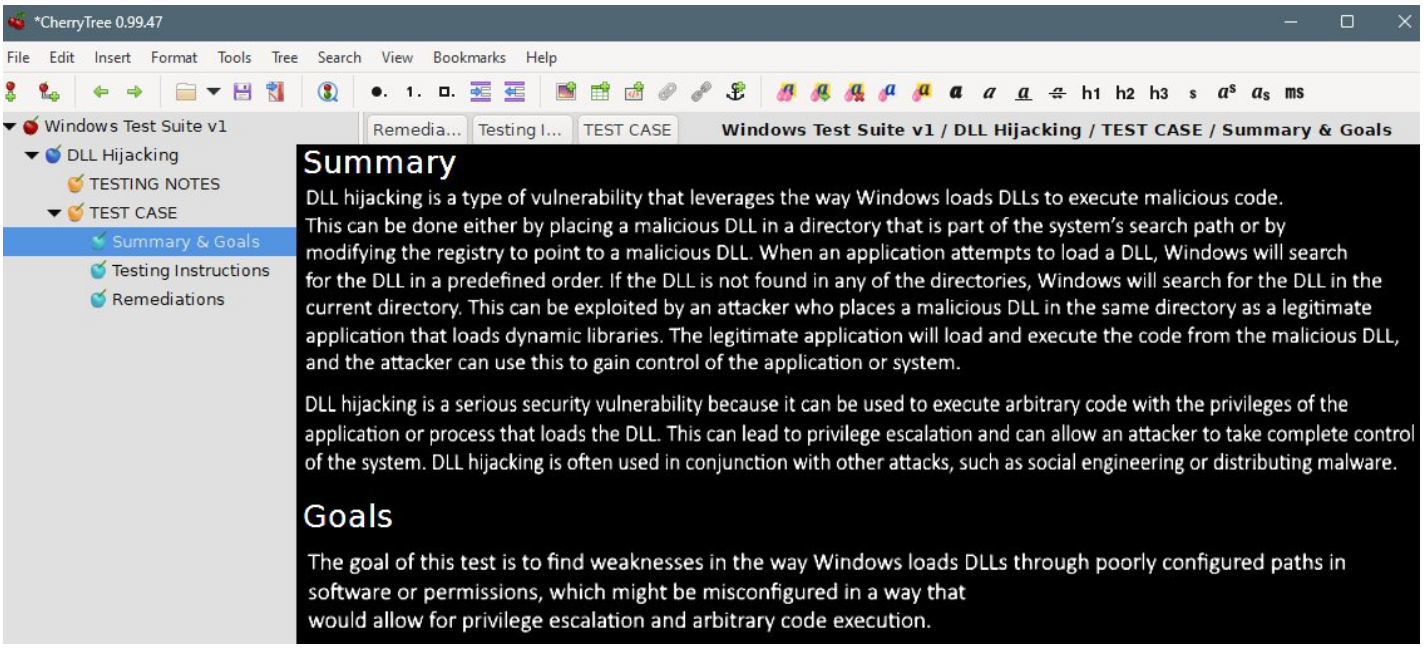


Рис. 3.11 – Наша секция краткого изложения целей, должна быть заполнена исследованиями в том виде, в каком мы их понимаем

Сейчас, когда у нас есть первоначальный шаблон, в котором определено наше краткое изложение целей, давайте перейдем к тому, каким образом найти лучший способ составить тест для проверки на этот дефект.

Инструкции тестирования

Инструкции тестирования, будут конечно во многом отличаться от случая к случаю, однако в большинстве случаев, существует обилие информации уже нам доступной. Есть несколько разных способов, которыми вы можете проверять на уязвимости к DLL hijacking. Если вы не знаете даже этого, инструкции тестирования вряд ли придут вам на ум. Даже если это для вас в новинку, начните изучение с использования вашей любимой поисковой системы. **How to test for DLL hijacking** или **DLL hijacking**

Tutorial, могли бы быть отличными стартовыми запросами. Во время выполнения такого поиска, одной из найденных нами вещей оказалось исчерпывающее руководство с сайта **hacktricks** (https://book.hacktricks.xyz/), где были описаны шесть подходов к тестированию. Это хорошо сочетается с мыслью не изобретать велосипед, в данном случае мы просто скопируем определения и краткое описание тестов, в наш тестовый пакет. Убедитесь, что вы ссылаетесь на источник ваших материалов, в своих записях (см. рис. 3.12). Пока, мы по сути может и не разрабатываем сейчас свои собственные заметки к тестам, мы можем добавлять и убирать элементы, по мере освоения разработки собственной методологии:

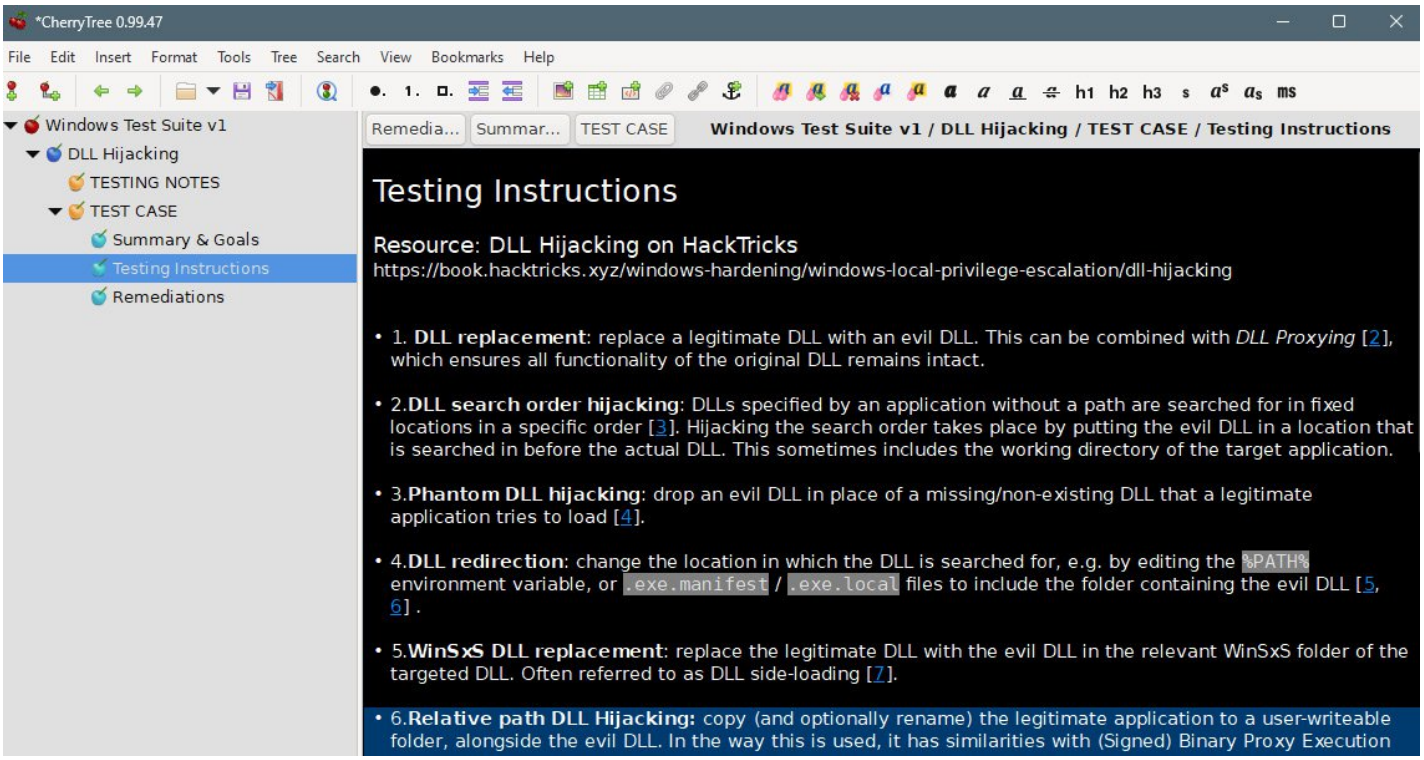


Рис. 3.12 – Избегайте изобретения велосипеда, собирайте заметки для тестов с ранее исследованных внешних ресурсов, когда это возможно

Давайте продолжим с нашей следующей и заключительной секцией по исправлениям, которая будет завершающей в нашем первом тесте.

Исправления

Как и в большинстве предыдущих разделов, мы снова будем учиться по доступным источникам. В секции краткого описания, мы обращали внимание на отличные ресурсы имеющие отношение к исправлению. Однако одним из лучших источников, которые дают официальное руководство по такого рода вещам – это инструкции производителя или ресурсы разработчика. Что касается нашего примера, мы ранее выяснили, что DLL hijacking – это вид недостатков, затрагивающий только системы Windows. Держа это в уме, уделим время изучению руководства Microsoft по предотвращению такого рода проблем программного обеспечения. Быстрый поиск ресурсов имеющих отношение к этому - идентифицировал уязвимость и выдал лучшие инструкции по снижению риска связанного с этой проблемой. Мы добавим их в наш тестовый пример, на тот случай, если нам понадобится уведомить поставщика ПО о найденной уязвимости (см. рис. 3.13):

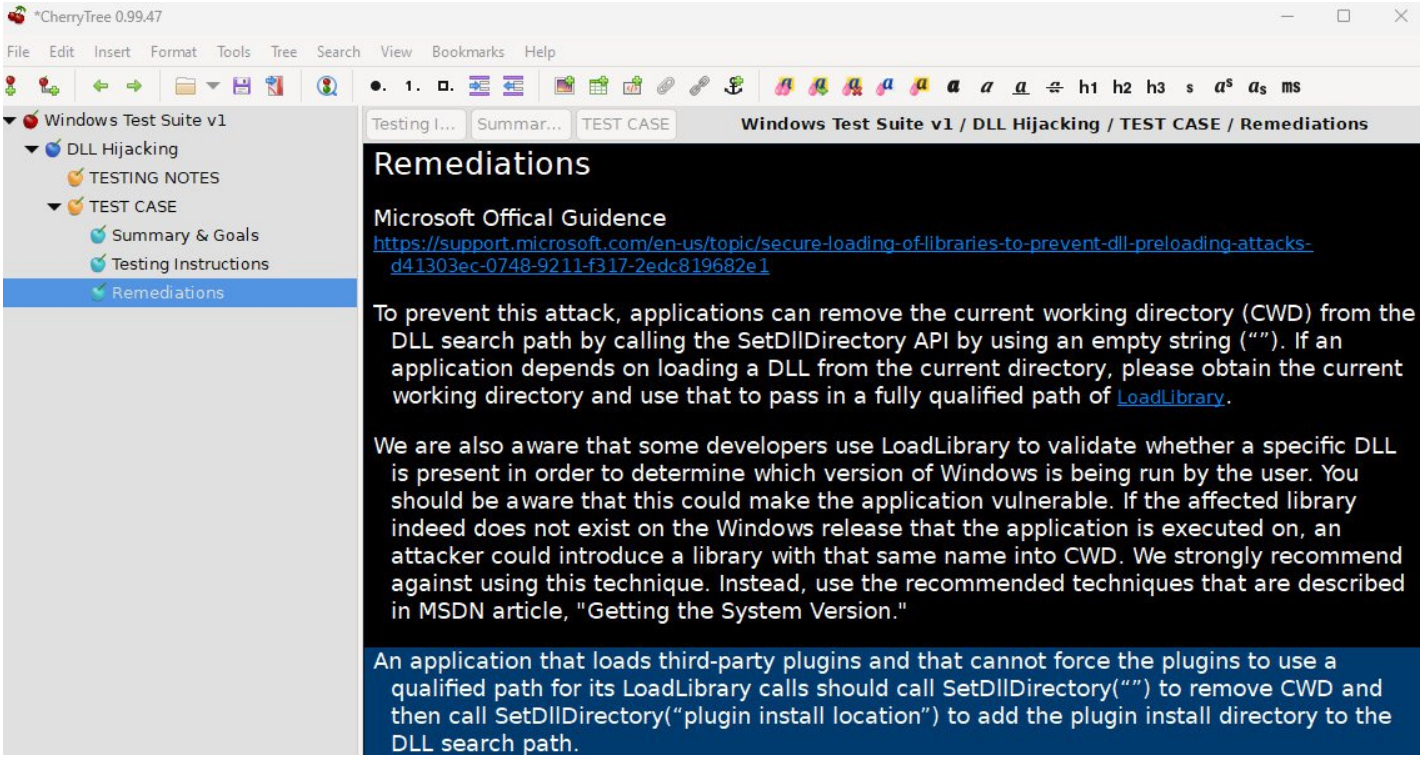


Рис. 3.13 – Основа ваших исправлений – выдержки из официальных руководств, когда это возможно

Отлично, теперь у нас есть наш первый тест. Если вы совсем недавно приступили к исследованию уязвимостей, это мог бы быть первый и единственный тест, с которого вы захотите начать. Хотя это отличный вариант для начала, мы призываем вас продолжать искать уязвимости, интересные вам, которые связаны с вашими увлечениями в этой области. Скажем в нашем случае, мы добавили несколько дополнительных тестов, связанных с тестированием приложений Windows. Вот, как три дополнительных теста, могли бы быть структурированы в CherryTree (см. рис. 3.14):

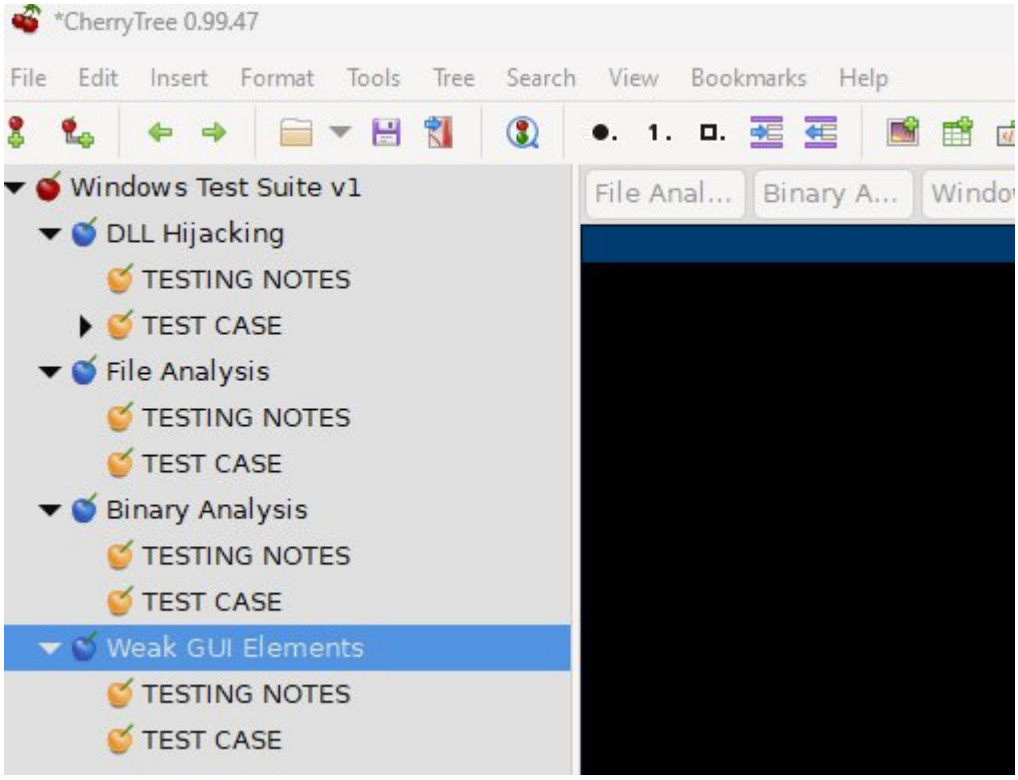


Рис. 3.14 – Не останавливайтесь на единственном тесте, составляйте другие, которые отражают тематику пакета тестов, по которому вы хотите проводить исследования

Как только ваши отдельные тесты и пакеты, сконфигурированы по вашему вкусу, вам может захотеться, сохранить их теперь в виде шаблона. CherryTree позволяет использовать ваш проект в качестве шаблона, таким образом что, когда вы приступаете к исследованию новой цели, вы можете открыть новую копию своего проекта и секция **TESTING NOTES**, будет выделена под заметки, которые вы можете заполнять данными конкретного объекта исследования. Чтобы это сделать, кликните **File** и **Export to CherryTree Document** (см. рис. 3.15):

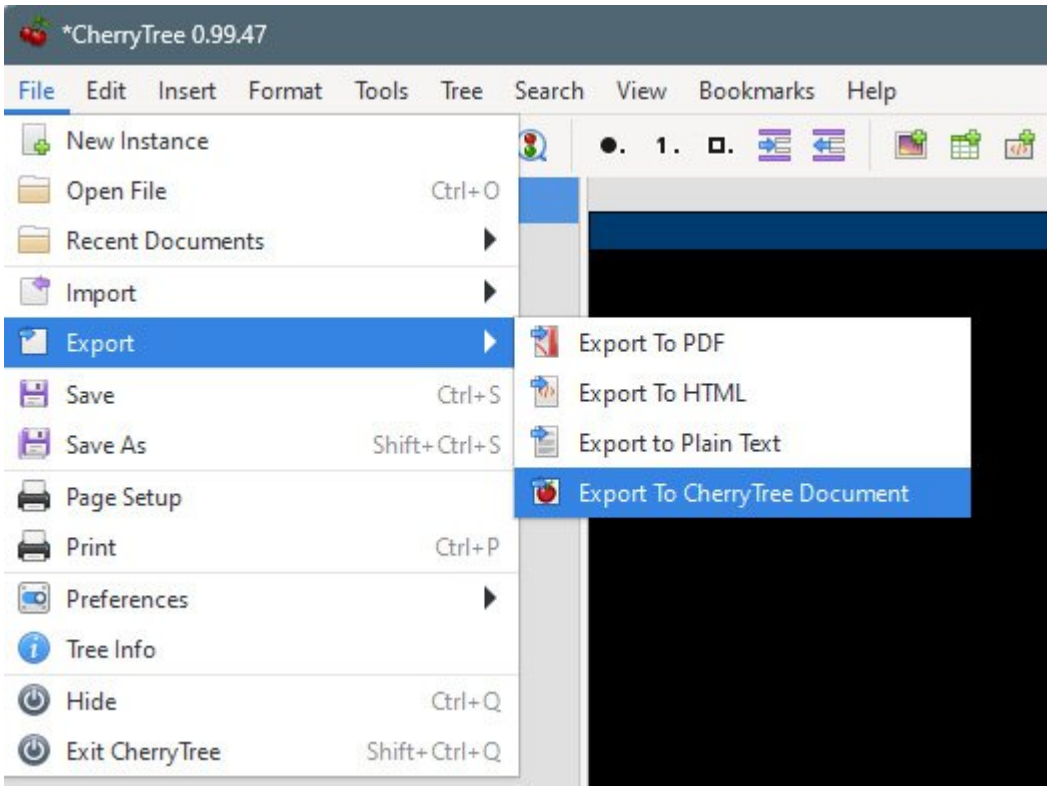


Рис. 3.15 – Экспорт вашей работы в качестве шаблона, для экономии времени в будущих проектах исследований

В следующий раз открывая CherryTree и желая запустить новые тесты, вы сможете импортировать проект CherryTree как свой шаблон, использовав функцию импорта.

При создании тестов собственной разработки важно учитывать, чего вы хотите добиться своим тестированием. Ищете ли вы конкретный тип уязвимости? Или вы просто пытаетесь получить представление о безопасности какого-то приложения? Четко понимая свои цели, вы гарантируете, что ваши тесты точно подходят для конкретной задачи. Эффективные тестовые пакеты, наверное лучший инструмент, который нужно иметь в своем арсенале для выявления уязвимостей. В следующем разделе мы охватим некоторые из других наиболее подходящих исследовательских инструментов, которые вам нужно будет научиться применять для проведения определенного рода исследований.

Знакомимся с распространенными исследовательскими инструментами

Давайте уделим немного времени, разговору о некоторых инструментах, которые понадобятся для исследования уязвимостей. Есть множество доступных в сети источников, которые помогут вам начать, однако мы потратим некоторое время, чтобы обсудить распространенные инструменты, на которые полагаются исследователи.

Ведение заметок, создание скриншотов и средства записи экрана

Как мы ранее выяснили, документация достаточно важна. А значит исследователю нужны подходящие инструменты, чтобы использовать их для упорядочивания своих исследований, синхронизации между разными окружениями, комментирования, создания скриншотов и видео-презентаций, когда это необходимо. Давайте поговорим о самых популярных инструментах, которые помогают в выполнении таких задач.

Joplin

Joplin, это приложение для заметок, напоминаний и планирования, с помощью которого вы можете синхронизировать свои заметки и списки дел между всеми своими устройствами. Оно доступно для Windows, macOS, Linux, Android, и iOS. Вы можете использовать Joplin в своем браузере, посетив страницу расширения на сайте Web Clipper и добавив его в свой браузер.

С помощью Joplin вы можете создавать заметки и списки дел, которые могут быть объединены в “блокноты“. Вы также можете пометить свои записи для удобства структуры и упрощения поиска по ним:

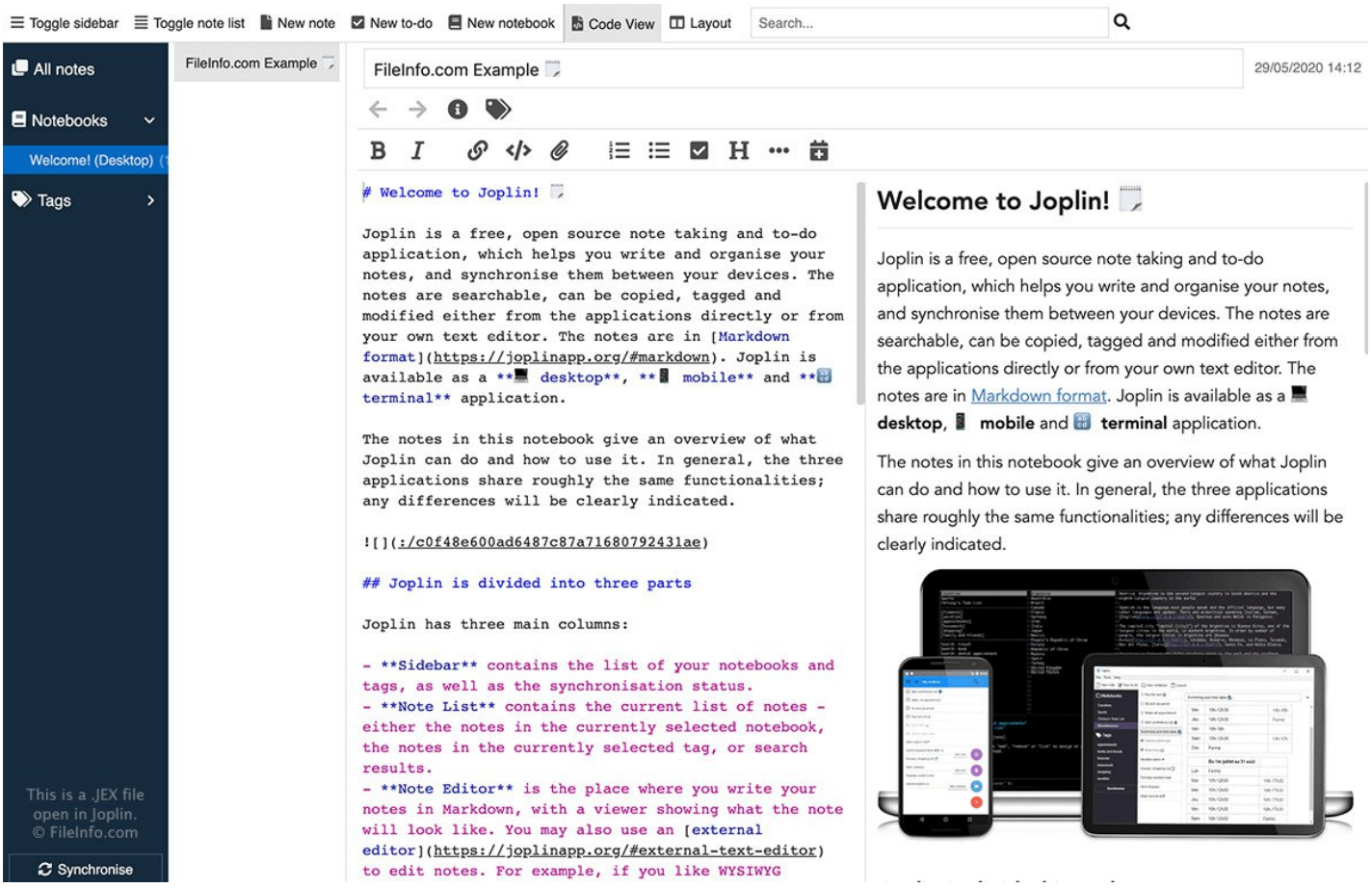


Рис. 3.16 – Пример использования в Joplin разметки Markdown

Ключевой момент, который полюбили исследователи, это то, как все замечательно синхронизируется между устройствами и как отформатированы заметки на языке разметки, называющемся **Markdown** (см. рис. 3.16).

CherryTree

CherryTree, мы уже упоминали это приложение с иерархической организацией записей, с возможностью форматирования текста и подсветки синтаксиса, сохранении данных в едином файле **.xml** или **.sqlite**. Их возможно экспортировать в HTML, PDF и другие форматы. CherryTree доступно для Windows, Linux, и macOS.

С помощью CherryTree, вы можете создавать заметки, объединенные в узлы. Вы также можете присваивать метки своим записям для удобства поиска. Ваши заметки, сохраняются в едином файле, поэтому легко сделать их резервное копирование или перенести на другие устройства:

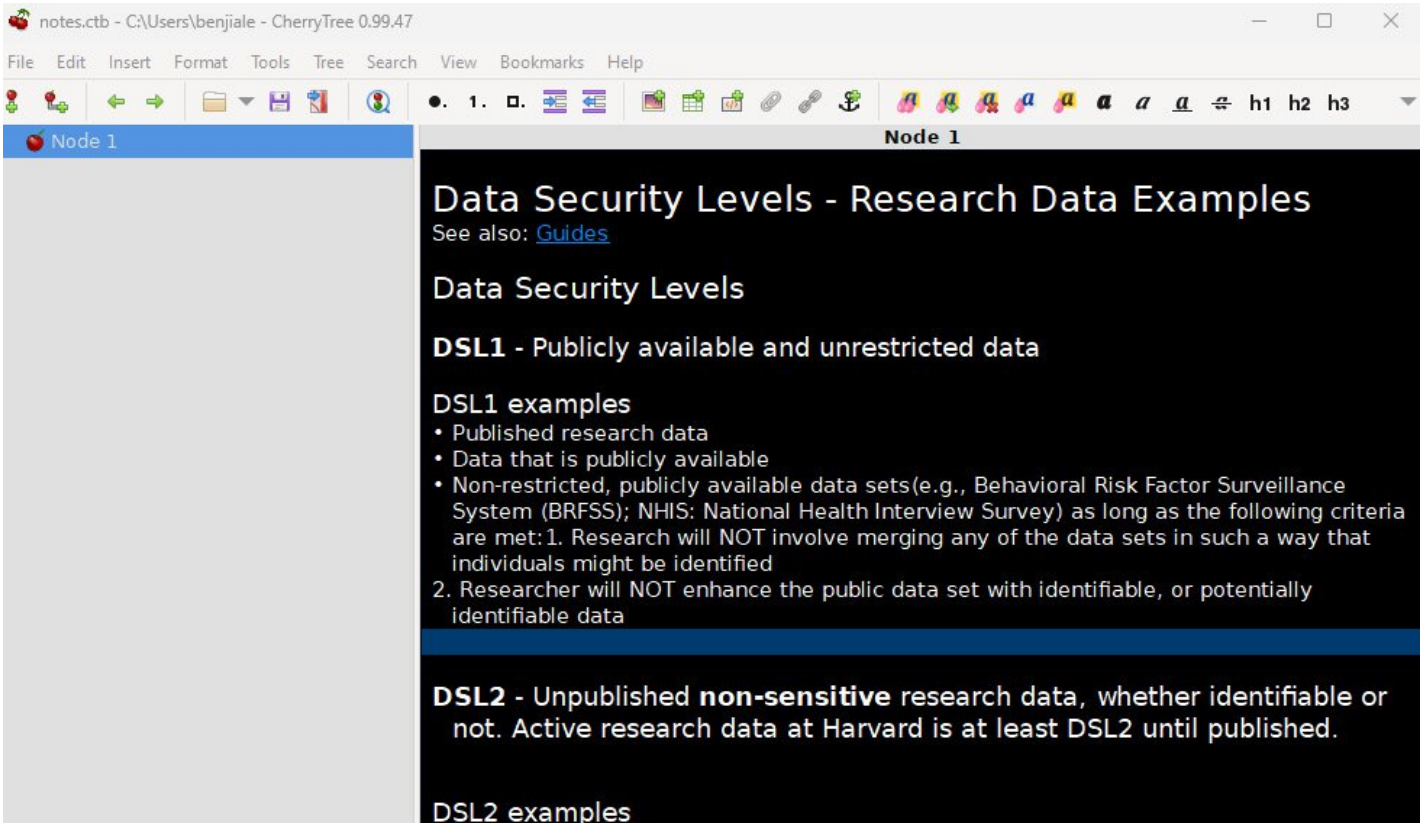


Рис. 3.17 – Пример CherryTree

Как мы к тому же рассказали в нашем предыдущем разделе по написанию тестов, это отличный инструмент, который можно использовать для составления шаблонов, чтобы можно было их наследовать и применять, в случае необходимости протестировать определенное ПО (см. рис. 3.17).

Greenshot

Вы можете применять простейшие инструменты для создания примеров своих исследований в картинках или можете поставить турбонаддув на свою рутину по иллюстрированию, с помощью чего-то вроде **Greenshot**. Greenshot это бесплатный инструмент создания скриншотов с открытым исходным кодом, поддерживающий Windows и macOS. Он умеет делать скриншоты всего вашего экрана, выбранной области или определенного окна:

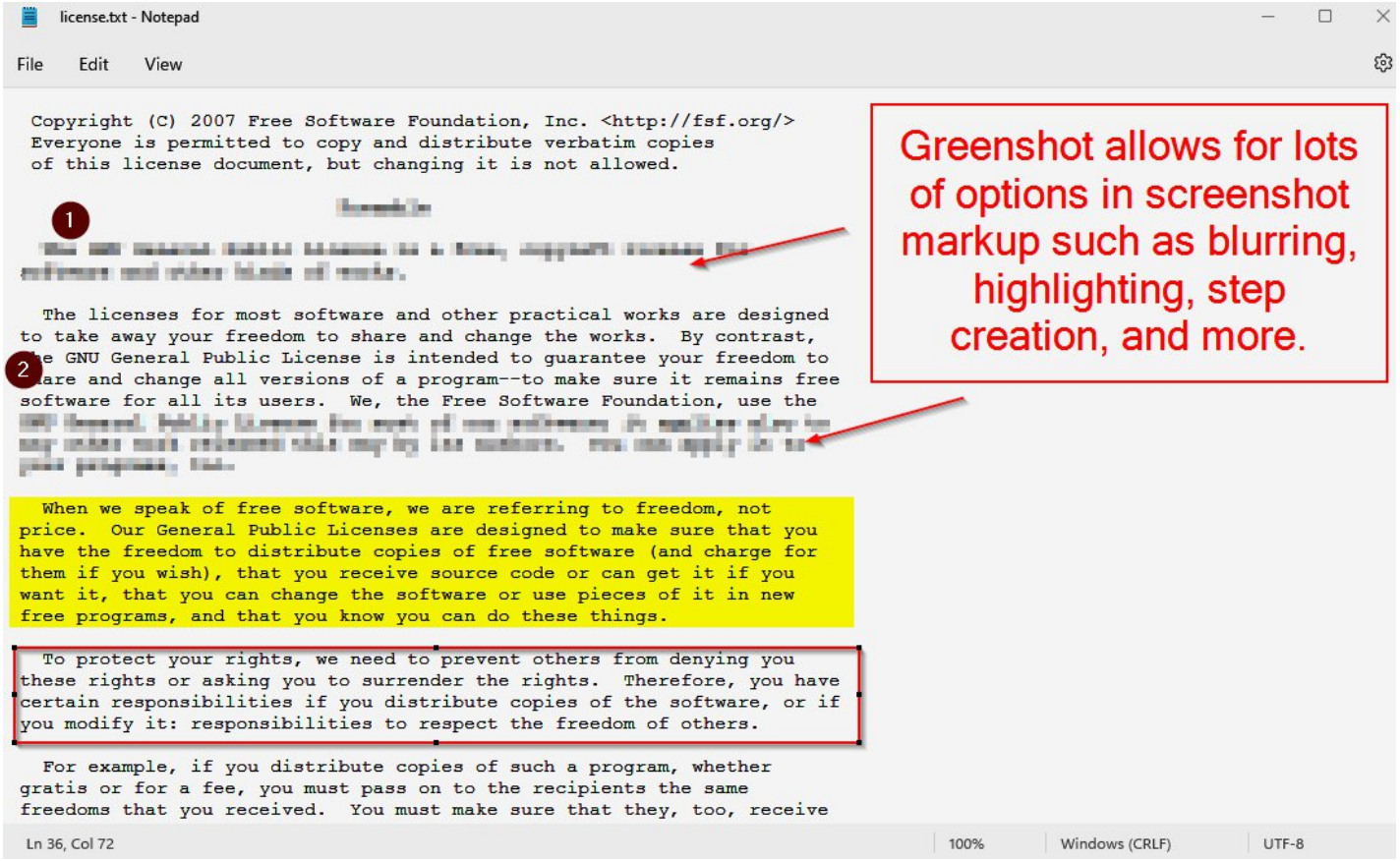


Рис. 3.18 – Некоторые из множества возможностей по редактированию скриншотов, которые вы можете применить с Greenshot

Лучше всего, Greenshot проявляет себя богатыми возможностями аннотаций и редактирования, которые позволяют писать заметки, а также размывать и подсвечивать элементы вашего скриншота (см. рис. 3.18).

Open Broadcaster Software (OBS)

С помощью **Open Broadcaster Software** (**OBS**), вы можете записывать ваш экран целиком или выбранную область. Вы также можете транслировать свой экран в популярные стриминговые сервисы, типа Twitch или Youtube. OBS это отличный выбор, для создания видео по своим исследованиям, чтобы доставлять их пользователям, которые могли бы воспользоваться вашими исследованиями. Видео это очень эффективный способ коммуникации и демонстрации всем желающим, как можно воспроизвести уязвимость (см. рис. 3.19):

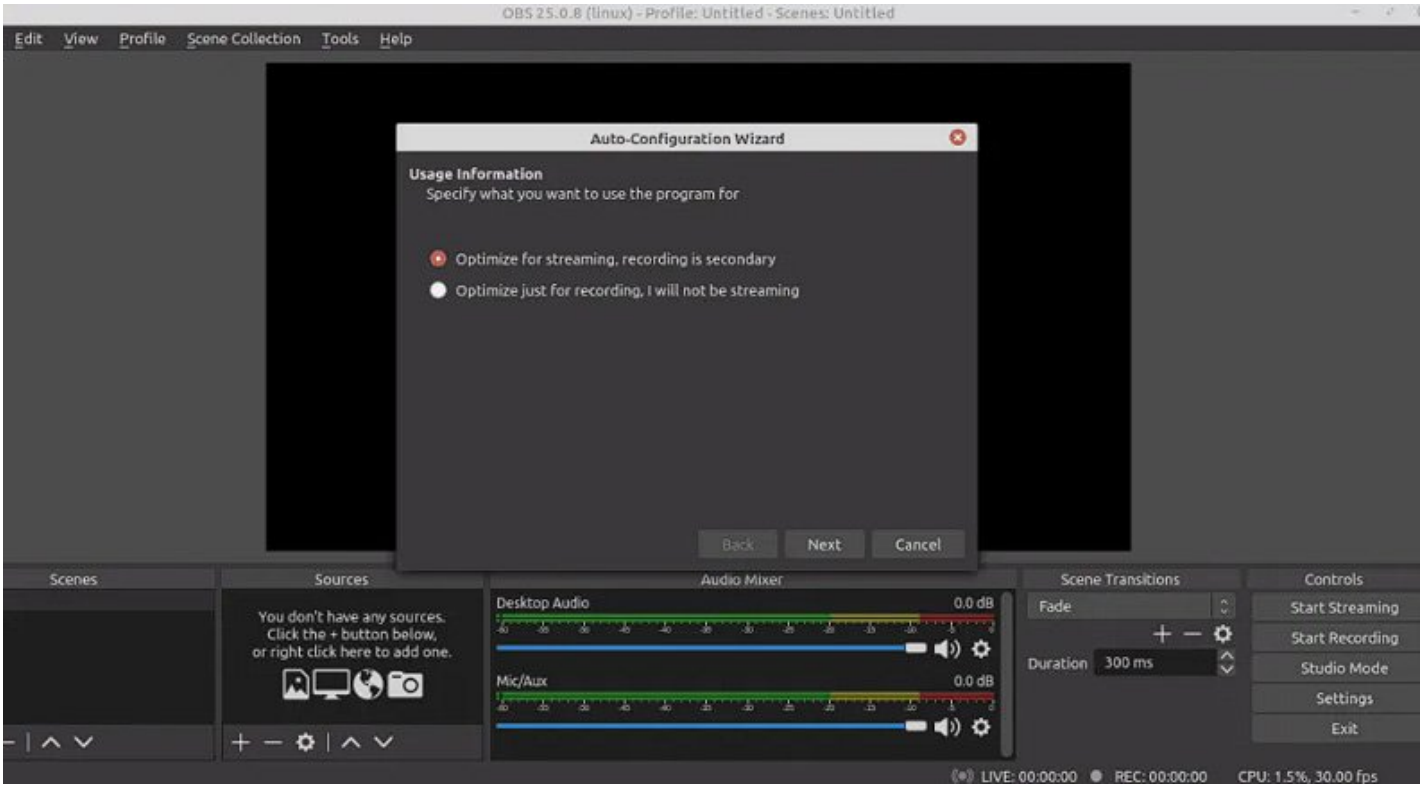


Рис. 3.19 – Пример конфигурирования OBS

Средства записи, связанные с нашим тестированием, довольно важны. Это должны быть подходящие приложения, в базовой функциональности которых, вы найдете то, что поможет вам внятно документировать результаты вашего тестирования. Теперь, давайте выберем минутку и обсудим технологию, которую мы применим для изоляции нашего тестового окружения.

Гипервизоры и виртуальные машины

Гипервизор это класс программного обеспечения, который позволяет вам создавать и запускать виртуальные машины на физическом компьютере. Гипервизоры используются и в серверной и в десктопной виртуализации. Использование гипервизора позволит вам заниматься исследованием уязвимостей на множестве виртуальных машин одновременно, не беспокоясь о влиянии на стабильность своей физической машины. Дополнительно некоторые гипервизоры, поставляются с функционалом, который позволяет легко захватывать и анализировать сетевой трафик или производить другие типы анализа. В отношении исследования, лучше всего, чтобы тестирование программного обеспечения было сделано в изолированной операционной системе или “песочнице“ для того, чтобы минимизировать риск случайного повреждения своей основной системы. Это может быть сделано с применением **виртуальной машины** (**VM**), запущенной поверх гипервизора. Давайте немного поговорим о том, какие гипервизоры обычно используют исследователи.

VMware Workstation Pro

VMware Workstation Pro это коммерческое ПО виртуализации, поддерживающее Windows и Linux. С ее помощью можно создавать виртуальные машины на вашем компьютере, позволяющие вам запускать множество операционных систем одновременно. VMware Workstation Pro также предлагает многочисленные функции для разработчиков и исследователей, такие как возможность сохранять снапшоты вашей виртуальной машины (см. рис. 3.20):

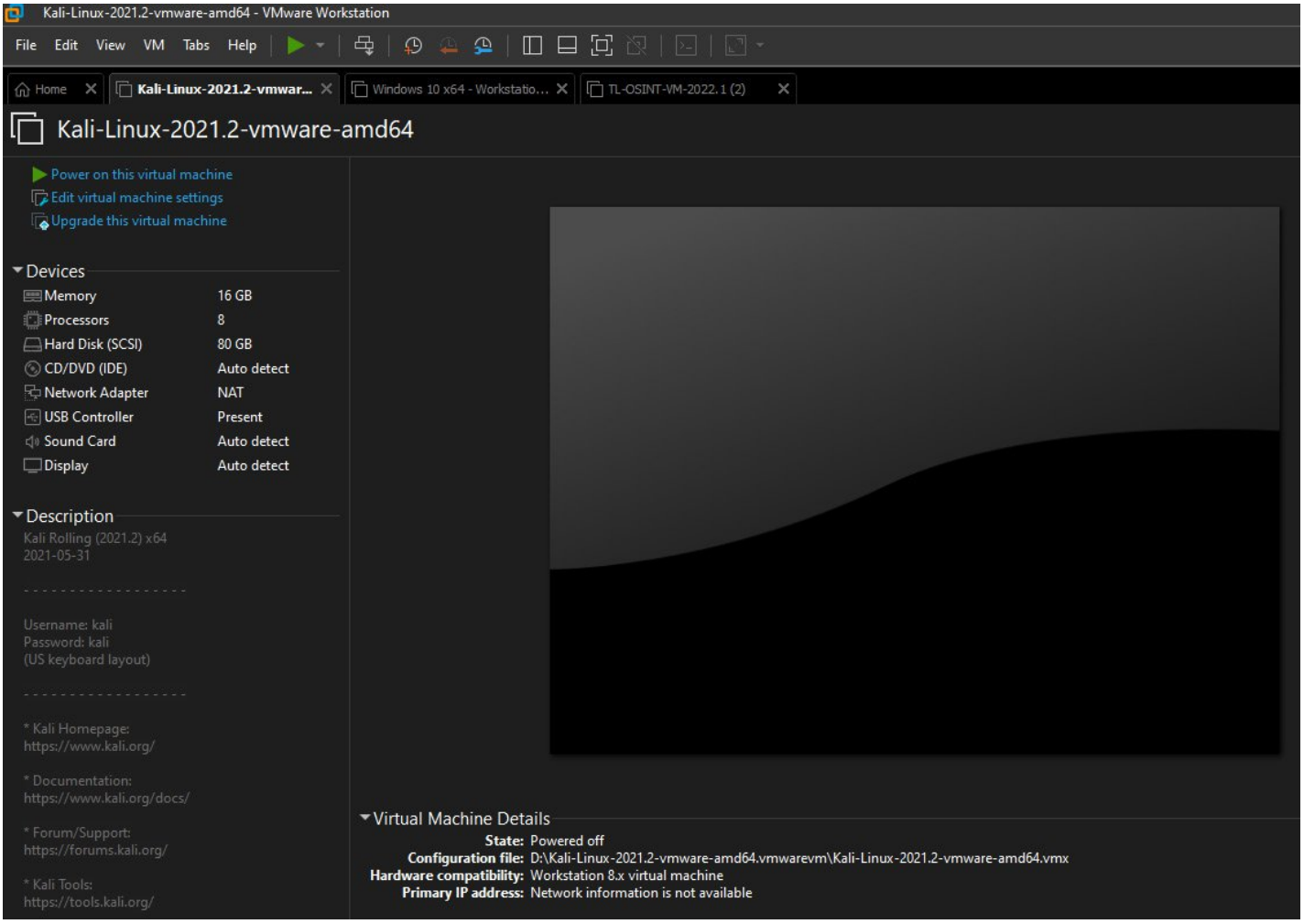


Рис. 3.20 – Пример виртуальных машин в VMware Workstation

Схожая функциональность, доступна пользователям macOS, в продукте под названием *VMware Fusion*.

VirtualBox

VirtualBox это бесплатный инструмент виртуализации с открытым исходным кодом, поддерживающий Windows, macOS, Linux и Solaris. Он позволяет запускать несколько операционных систем на одном компьютере одновременно. VirtualBox также предоставляет многие другие функции, вроде создания снапшотов и клонирования (см. рис. 3.21):

Рис. 3.21 – Пример виртуальных машин в VirtualBox

По набору функций, он похож на VMware Workstation, однако он менее эффективно управляет виртуализацией и чуть более склонен к сбоям, по сравнению с VMware Workstation.

Hyper-V

Hyper-V это инструмент виртуализации для Windows, который поддерживает одновременную работу нескольких операционных систем. Он так же может использоваться для создания виртуальных машин. Hyper-V также предоставляет массу функций, таких как создание снапшотов и клонирование:

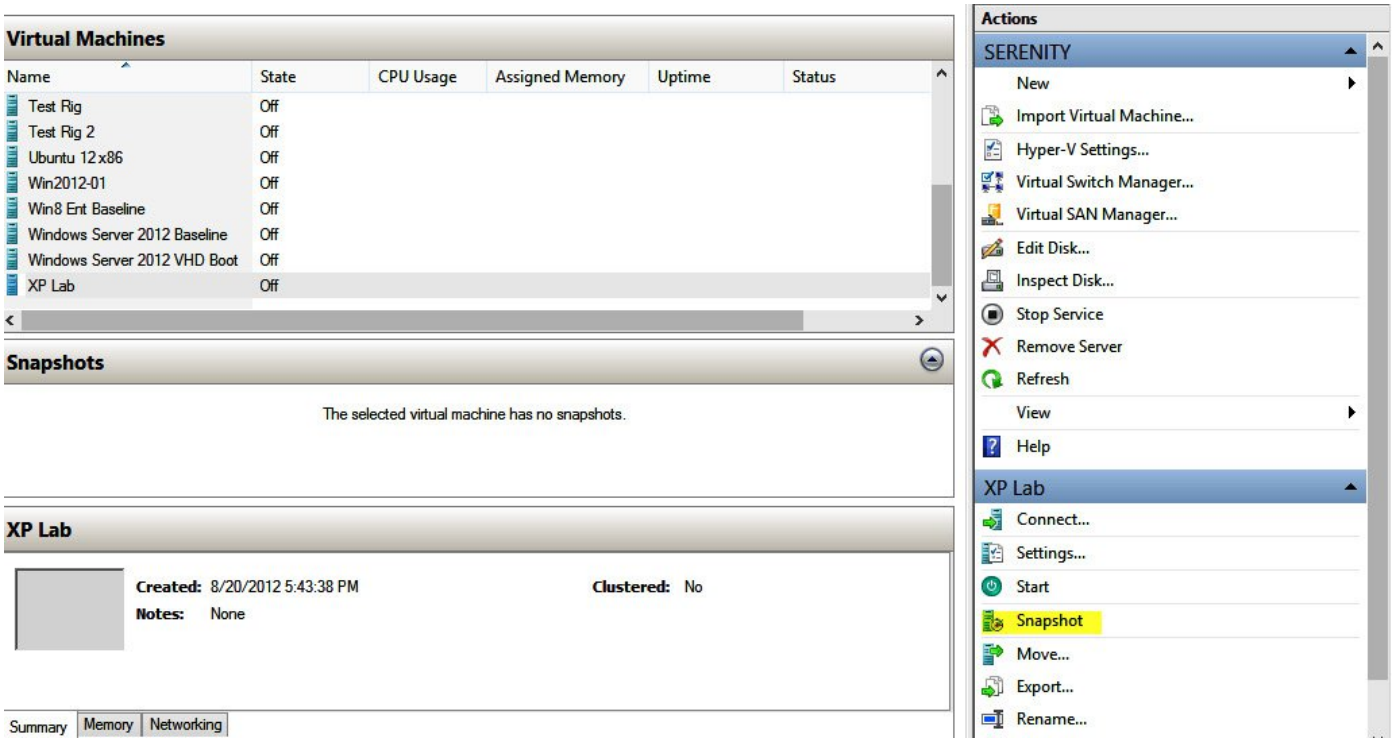


Рис. 3.22 – Пример Hyper-V менеджера в Windows

Хотя Hyper-V включает ограниченный набор функций, по сравнению с другими двумя гипервизорами, он имеет некоторое преимущество в том, что будучи частью ОС Windows, может быть установлен и настроен через Панель Управления в большинстве версий Windows (см. рис. 3.22).

Готовые виртуальные машины

Обзаведясь гипервизором, вы можете создавать с нуля свои собственные виртуальные машины или взять готовые и загрузить их в выбранный вами гипервизор. Это потрясающий способ создавать изолированные окружения для тестирования и оставлять больше времени на работу. Вот несколько предпочтительных ресурсов с машинами различных видов.

Windows

Майкрософт поддерживает и распространяет множество серверного и десктопного программного обеспечения, в виде готовых виртуальных машин, которые вы можете бесплатно использовать скачав их с https://developer.microsoft.com/en-us/windows/downloads/virtual-machines/.

Linux

Годный ресурс со множеством готовых виртуалок с Linux – это OSBoxes https://www.osboxes.org.

Операционные системы, ориентированные на безопасность

Большинство исследователей, используют специально сконфигурированные для работы в области безопасности виртуальные машины, которые помогают исследователям начать с уже подготовленных инструментов. Хотя это не обязательно, что-то из этого полезно для экономии вашего времени на собирание комплекта общих инструментов:

- Kali Linux (Сосредоточена на пентесте и исследованиях безопасности) https://www.kali.org/get-kali/
- Moblexer (Фокусируется на проверке и исследовании безопасности мобильных устройств) https://mobexler.com/
- REMnux (Заточена под анализ малвари и реверс-инжиниринг) https://remnux.org/
- Trace Labs OSINT VM (Подвид Kali Linux, концентрирующийся на исследованиях в области сетевой разведки) https://www.tracelabs.org/initiatives/osint-vm

Кроме подходящего гипервизора и виртуальных машин, чтобы проводить наши исследования, посвятим немного времени обсуждению прокси-серверов веб-приложений.

Прокси-серверы веб-приложений

Веб-прокси это программа, которая действует как посредник между вашим компьютером и интернетом. Когда используется веб-прокси, весь ваш интернет трафик, будет пропускаться через прокси-сервер перед отправкой по адресу назначения. Это может пригодиться по многим причинам, таким как обход файерволов, фильтрации веб-контента и то есть, выявления уязвимостей веб-приложений.

Например, если вы тестируете страницу входа и хотите видеть, какие данные пересылаются, когда вы подтверждаете вход, вы должны воспользоваться веб-прокси для перехвата такого трафика. В таком случае вы сможете просмотреть трафик и увидеть, если ваши учетные данные передавались открытым текстом (что сигнализировало бы о серьезной уязвимости безопасности). Давайте прямо сейчас, взглянем на несколько примеров прокси-серверов веб-приложений.

Burpsuite

Burpsuite это перехватывающий прокси-сервер, поддерживающий Windows, macOS и Linux. Он применяется для перехвата и модификации трафика. Burpsuite также предлагает множество функций, таких как **фаззинг** и **брутфорс**:

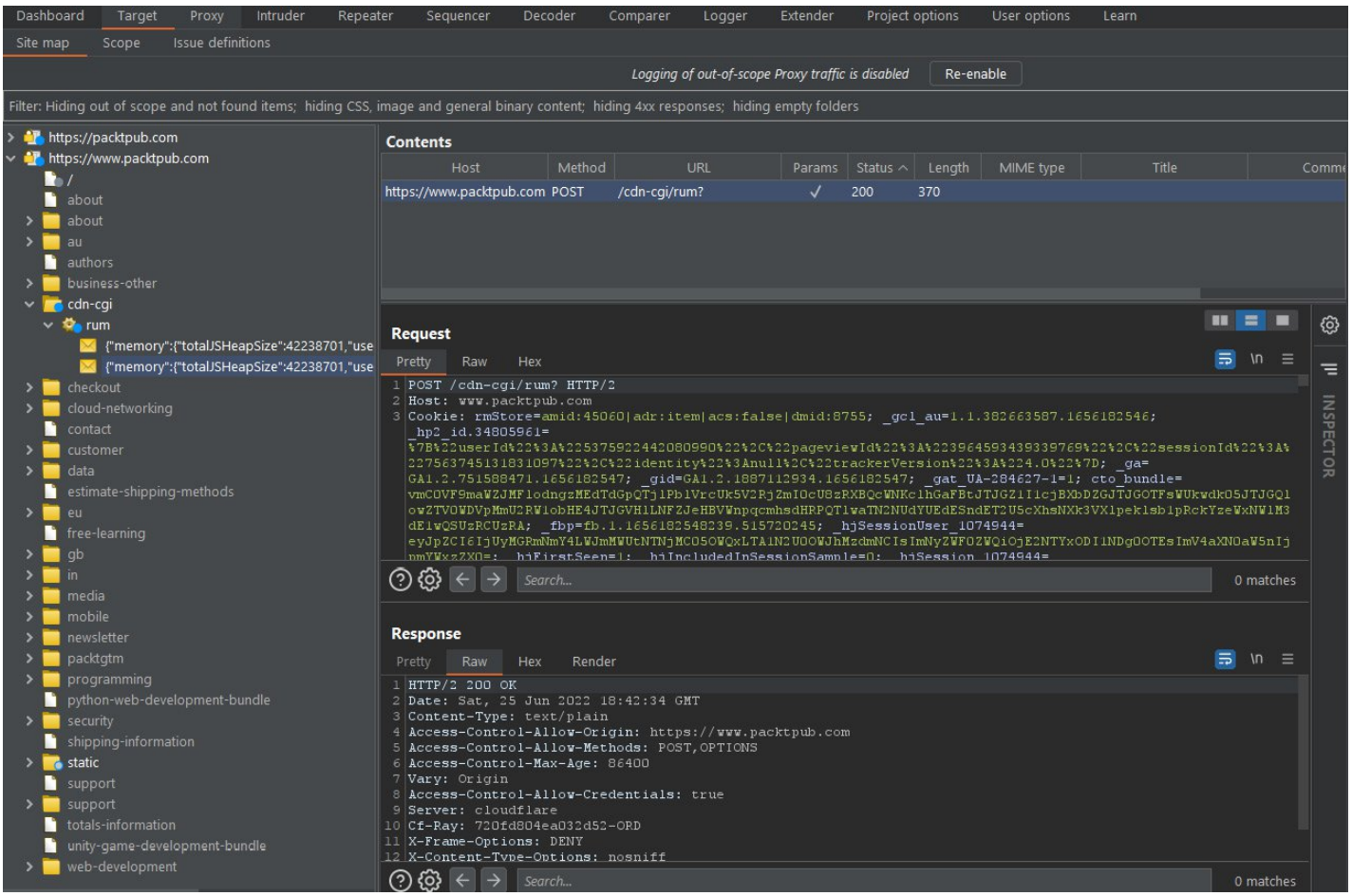


Рис. 3.23 – Пример использования Burpsuite для захвата и просмотра трафика веб-сайтов

Burpsuite один из наиболее распространенных “швейцарских ножей“, которые пентестеры имеют в своем инструментарии, при тестировании веб-приложений. Учебные ресурсы и поддержка сообществом этого инструмента, просто превосходны.

ZAP

ZAP это бесплатный инструмент тестирования безопасности веб-приложений, с открытым исходным кодом, поддерживающий Windows, macOS и Linux. Он применяется для перехвата и модификации трафика.

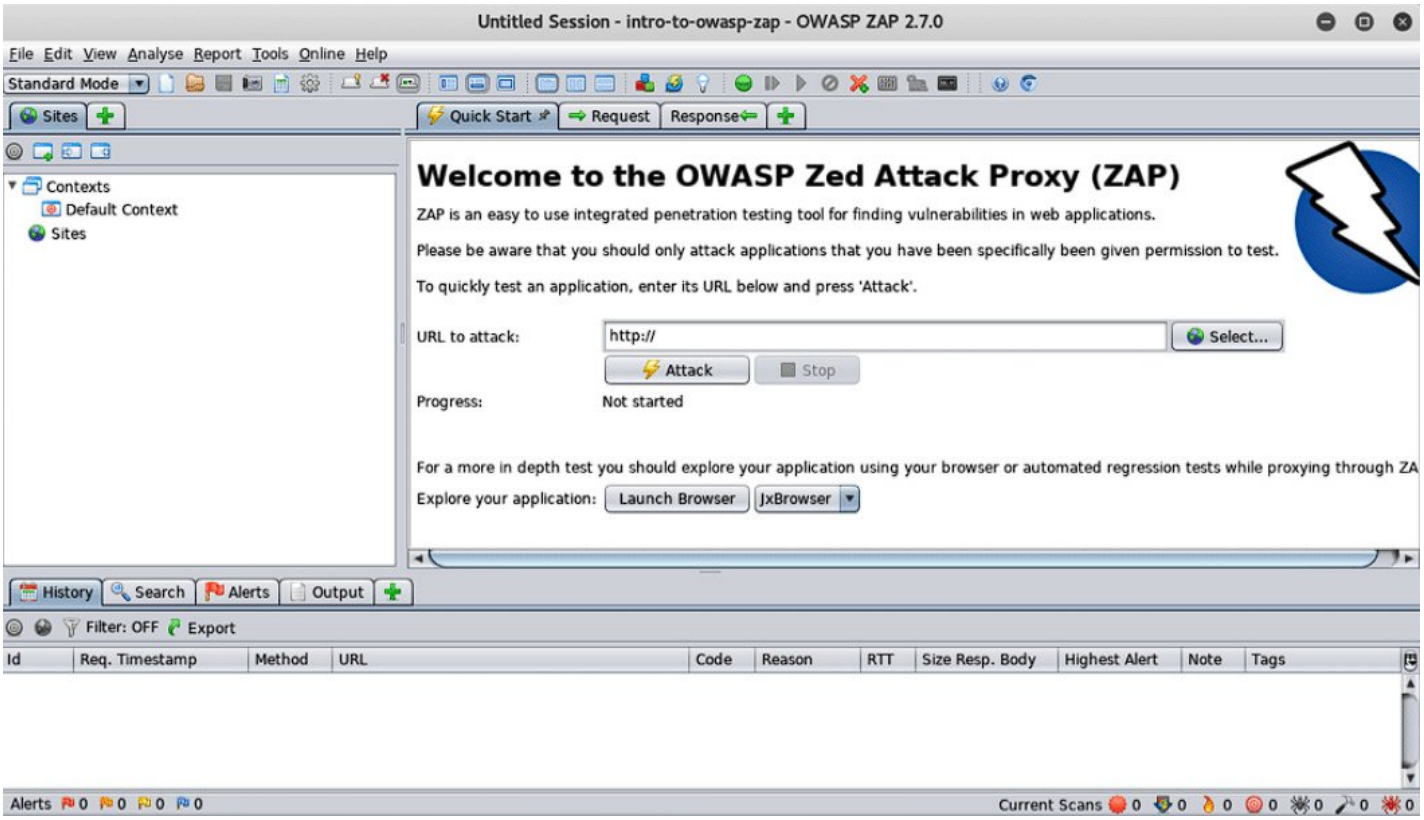


Рис. 3.24 – Пример ZAP в конфигурации по умолчанию.

ZAP также предлагает множество функций, таких как фаззинг и брутфорс, которые могут оказаться полезны для поиска уязвимостей в веб-приложениях (см. рис. 3.24).

Fiddler Classic

Fiddler Classic является бесплатным отладочным веб-прокси, поддерживающий ОС Windows. Он применяется для перехвата и модификации трафика:

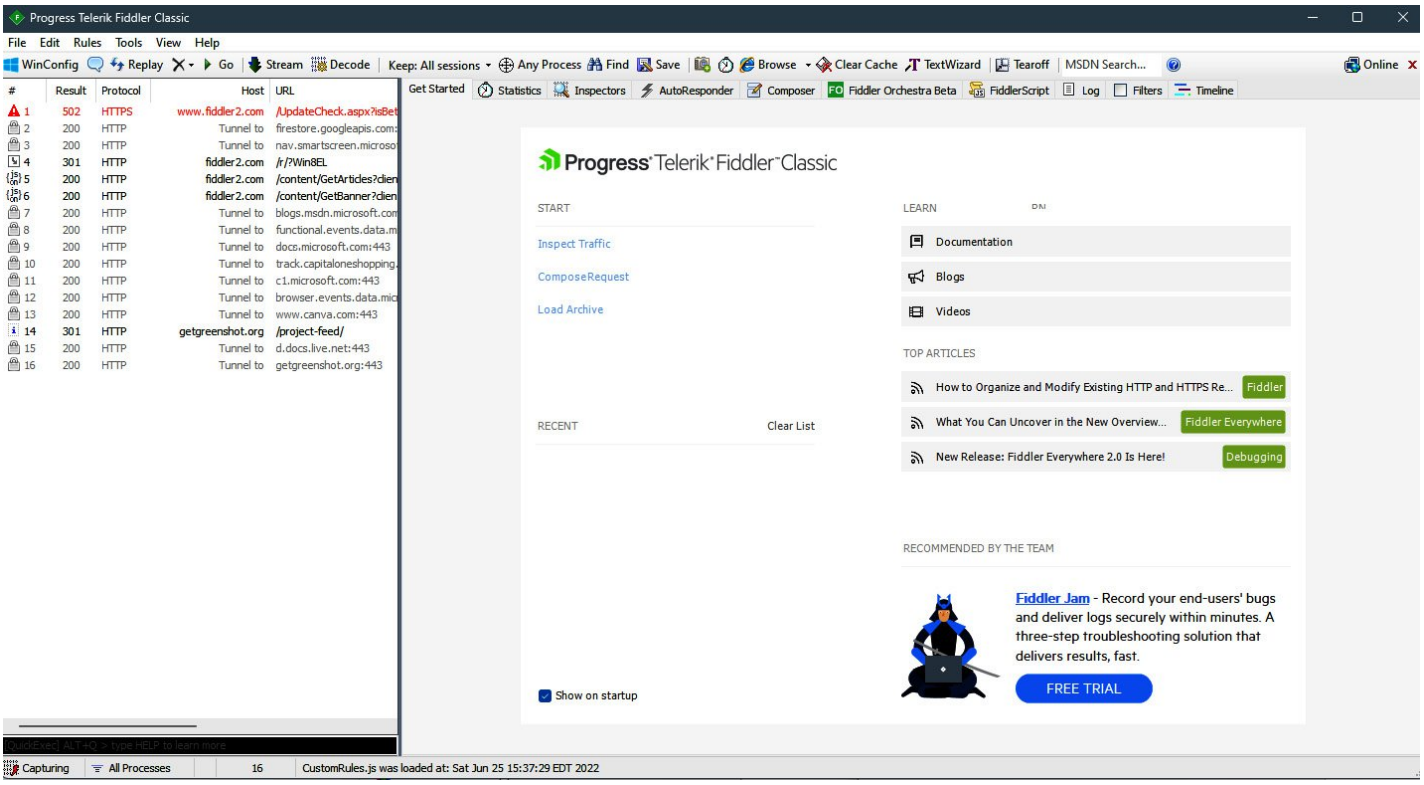


Рис. 3.25 – Пример использования Fiddler для проксирования и изучения веб-трафика

Fiddler также предлагает множество функций, таких как нагрузочное тестирование и дешифрование HTTPS (см. рис. 3.25).

Отладчики и декомпиляторы

Отладчик программного обеспечения это инструмент, который помогает вам искать ошибки в коде. По сути, он применяется для анализа работы программ изнутри. Это делается путем предоставления вам возможности пройти по коду строка за строкой, видя все что происходит и где могли бы обнаружиться уязвимости безопасности. Это чрезвычайно полезно, когда вы пытаетесь найти и исправить уязвимости безопасности.

Отладчики часто совмещены с декомпиляторами, которые позволяют вам провести реверс-инжиниринг скомпилированного кода. В идеале, это позволит вам взять исполняемый файл и преобразовать его обратно в исходный код. Это полезно при исследовании уязвимостей, потому что помогает вам разобраться, как работает программа и идентифицировать возможные уязвимости безопасности.

Если вы новичок в отладке, есть масса доступных ресурсов, чтобы помочь вам начать. Существует множество различных книг и онлайн пособий, которые научат вас основам. Как только вы разберетесь с основами, вы сможете начать экспериментировать с различными отладчиками и найти тот, которые лучше вам подходит. Независимо от уровня вашего опыта, отладчик является ценнейшим инструментом для исследования безопасности. Так что, если вы его все еще не используете, настоятельно рекомендуем начать. Давайте поговорим о нескольких из множества популярных инструментов исследователя, применимых на сегодняшний день.

OllyDbg

OllyDbg это бесплатный отладчик с открытым исходным кодом, поддерживающий Windows. Он применяется для отладки программ. Это помогает в поиске багов и уязвимостей в ПО, с помощью различных методов тестирования:

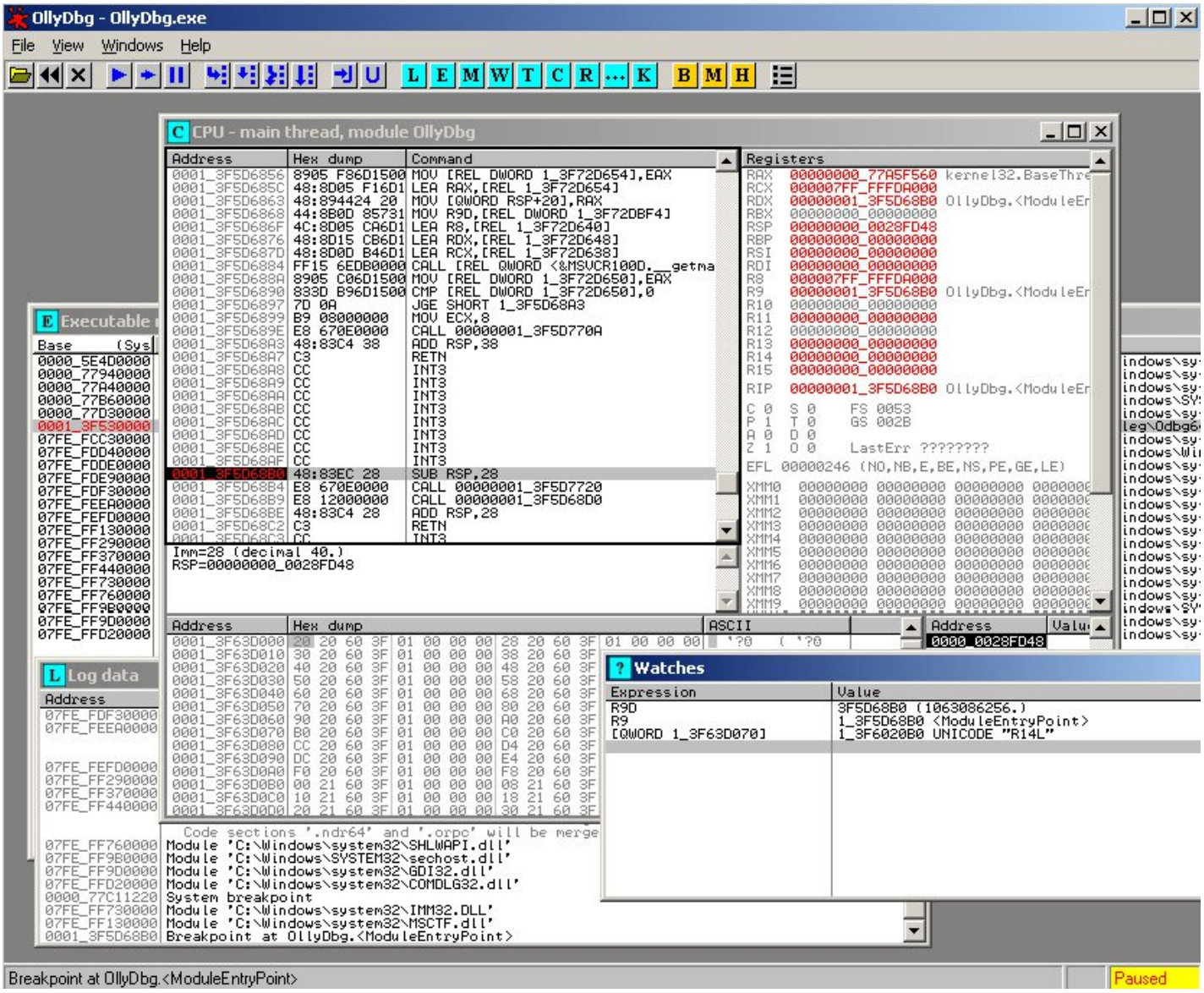


Рис. 3.26 – Пример отладки приложения Windows в OllyDbg

OllyDbg также предоставляет многочисленные возможности, такие как анализ кода и дизассемблирование, которые помогают в задачах реверс-инжиниринга (см. рис. 3.26).

Immunity Debugger

Immunity Debugger это бесплатный отладчик с открытым исходным кодом, поддерживающий Windows. Он применяется для отладки программ. Immunity Debugger также предоставляет многочисленные возможности, такие как анализ кода и дизассемблирование, что делает его похожим на выбор OllyDbg (*на котором он вообще-то и основан*), (см. рис. 3.27):

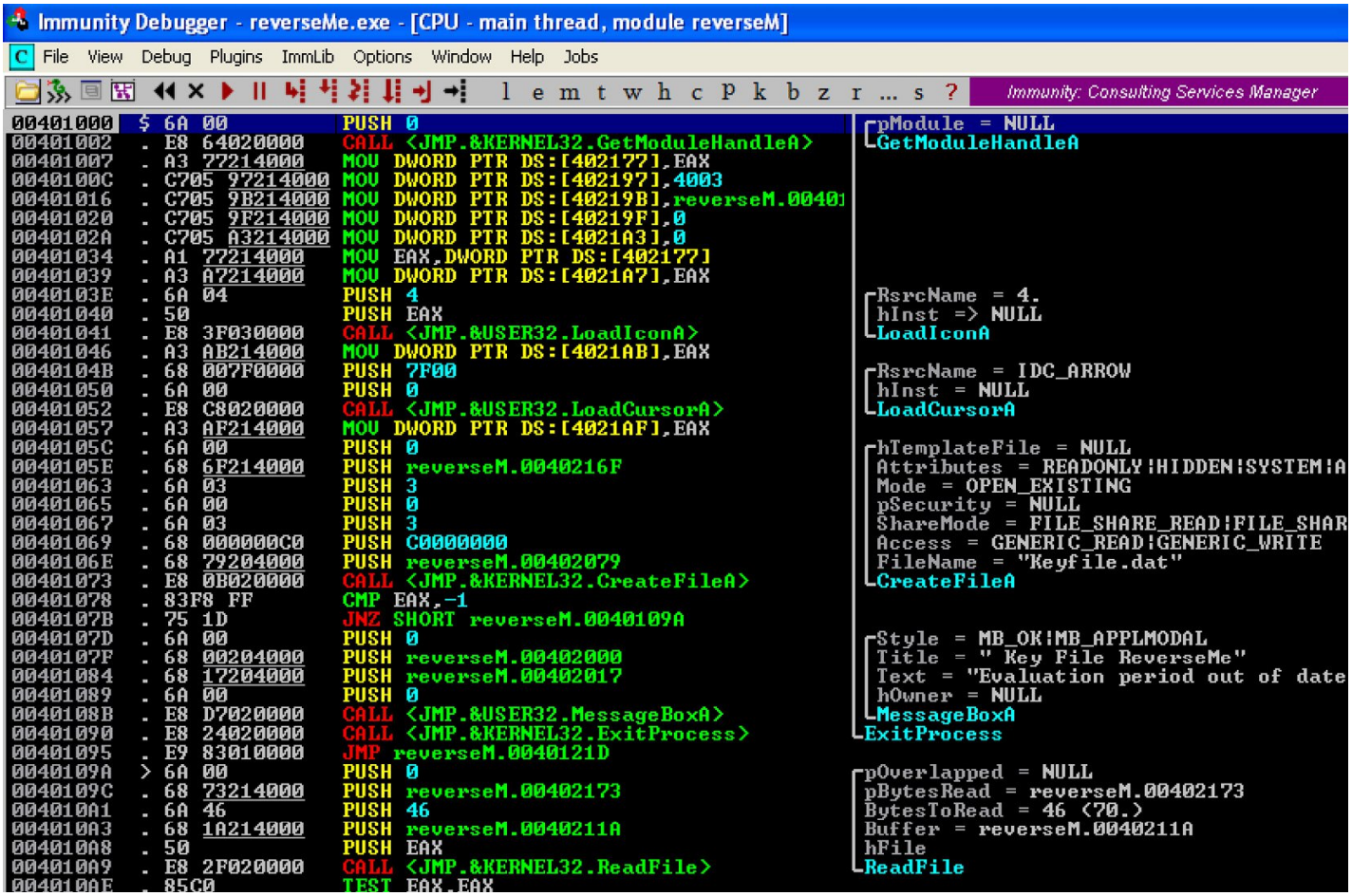


Рис. 3.27 – Пример отладки приложения Windows в Immunity Debugger

В зависимости от исследователя, некоторые предпочитают цветной вывод в Immunity, выводу OllyDbg, из-за возможности выделять текст и сосредоточиваться на наиболее важных элементах.

Windbg

Windbg является бесплатным отладчиком, созданным Майкрософт, для поддержки в Windows разработки ПО. Он применяется для отладки программ:

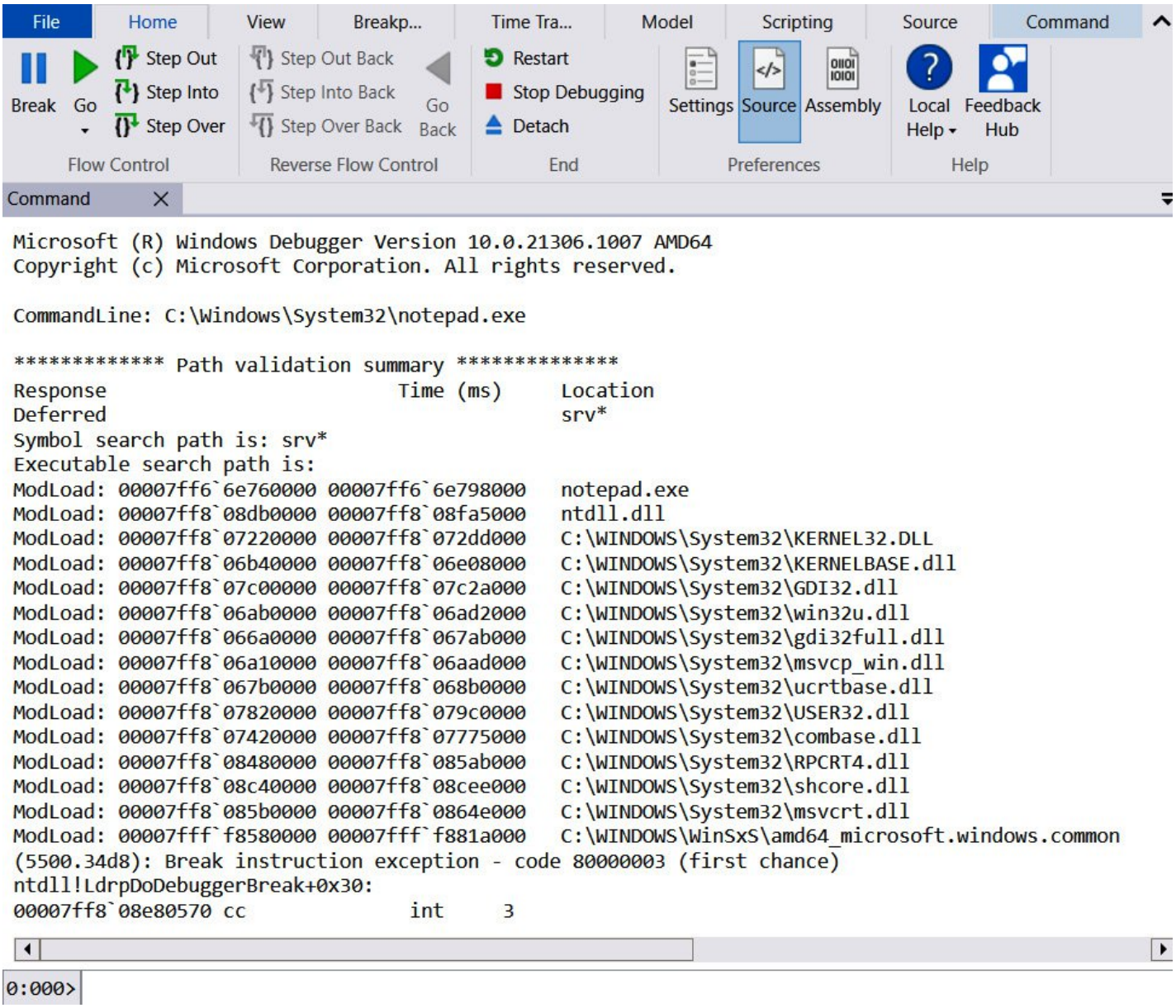


Рис. 3.28 – Отладка приложения Windows в Windbg

Windbg наверное, самый актуальный (обновляющийся) вариант из этого списка, однако злопыхатели, могут ссылаться на то, что Windbg больше применяется в командах разработчиков, чем в инфосеке (см. рис. 3.28).

UWP версия Windbg – Windbg для людей

UWP версия **Windbg**, имеет гораздо более приятный интерфейс и удобнее в использовании. Доступна в Microsoft Store, кроме этого ее можно установить оффлайн, скачав https://windbg.download.prss.microsoft.com/dbazure/prod/1-2308-2002-0/windbg.msixbundle и установив ее командой:

```
Add-AppxPackage -Path .\windbg.msixbundle
```

Radare2

Radare2 – бесплатный комплекс программ и вспомогательных утилит, с открытым исходным кодом, предназначенных для реверс-инжиниринга, поддерживающий Windows, Linux и macOS. Применяется соответственно, для реверс-инжиниринга. Radare2 разумеется, предоставляет множество функций, таких как анализ кода и дизассемблирование:

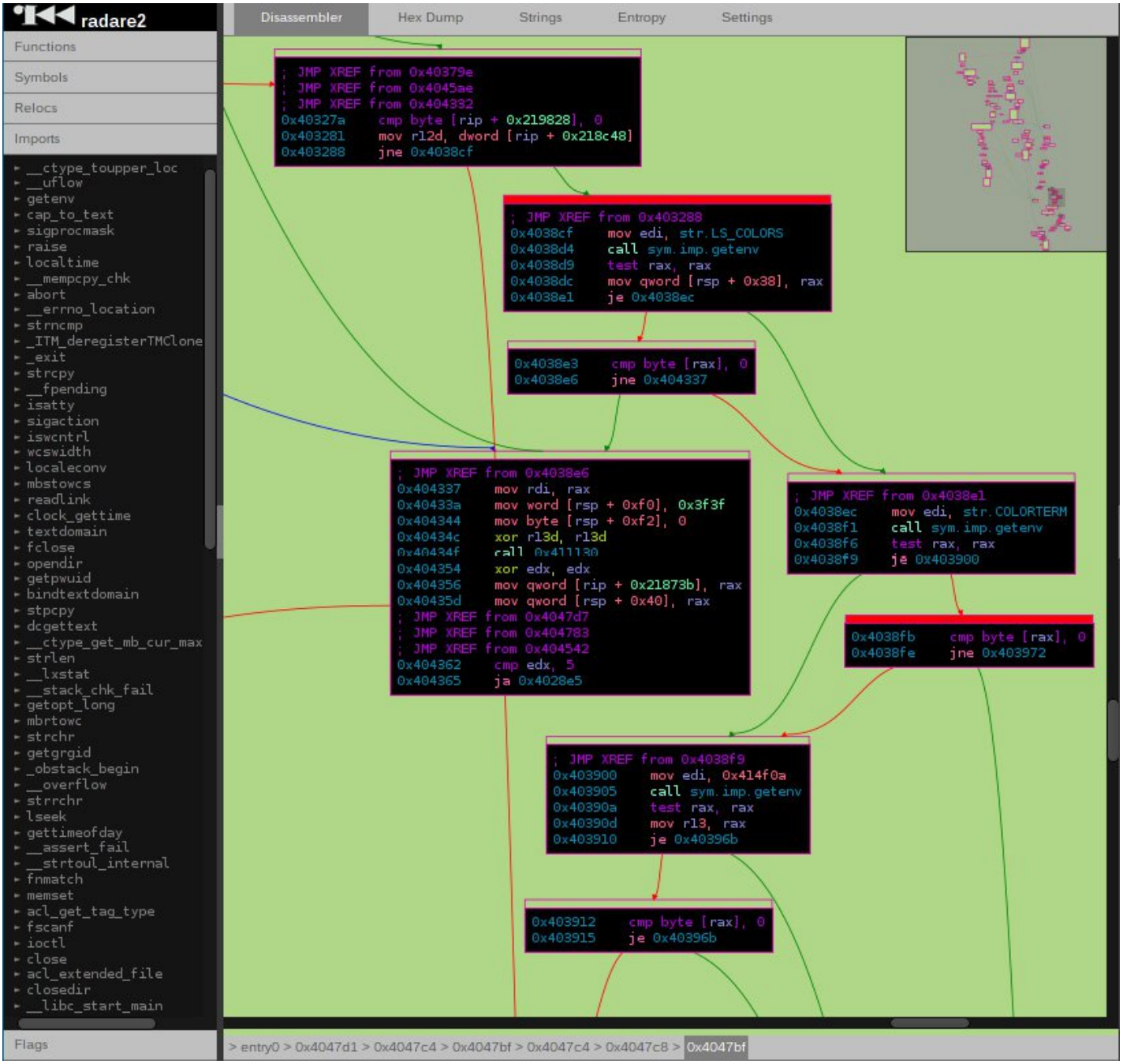


Рис. 3.29 – Дизассемблирование двоичного файла в Radare2

С помощью Radare2, вы можете проводить реверс-инжиниринг программ. Это способствует пониманию того, как работают программы и нахождению в них уязвимостей. Radare2 также предлагает множество функций, таких как анализ кода и дисассемблирование, которые могут быть полезны для создания патчей или модификации программ (см. рис. 3.29).

Ghidra

Ghidra это бесплатный (и с открытым исходным кодом) инструмент реверс-инжиниринга, который поддерживает Windows, macOS и Linux. Он был разработан **Агентством Национальной Безопасности** (**АНБ**), ведомством правительства Соединенных Штатов, чтобы помочь ликвидировать пробел в инструментарии, по их мнению, имевший место быть. В конечном счете, этот инструмент был широко разрекламирован, как отличное средство, которое поможет исследователям осуществлять реверс-инжиниринг двоичных файлов. С помощью Ghidra, вы сможете реверсить программы. Это помогает понять, как работают программы и искать в них уязвимости:

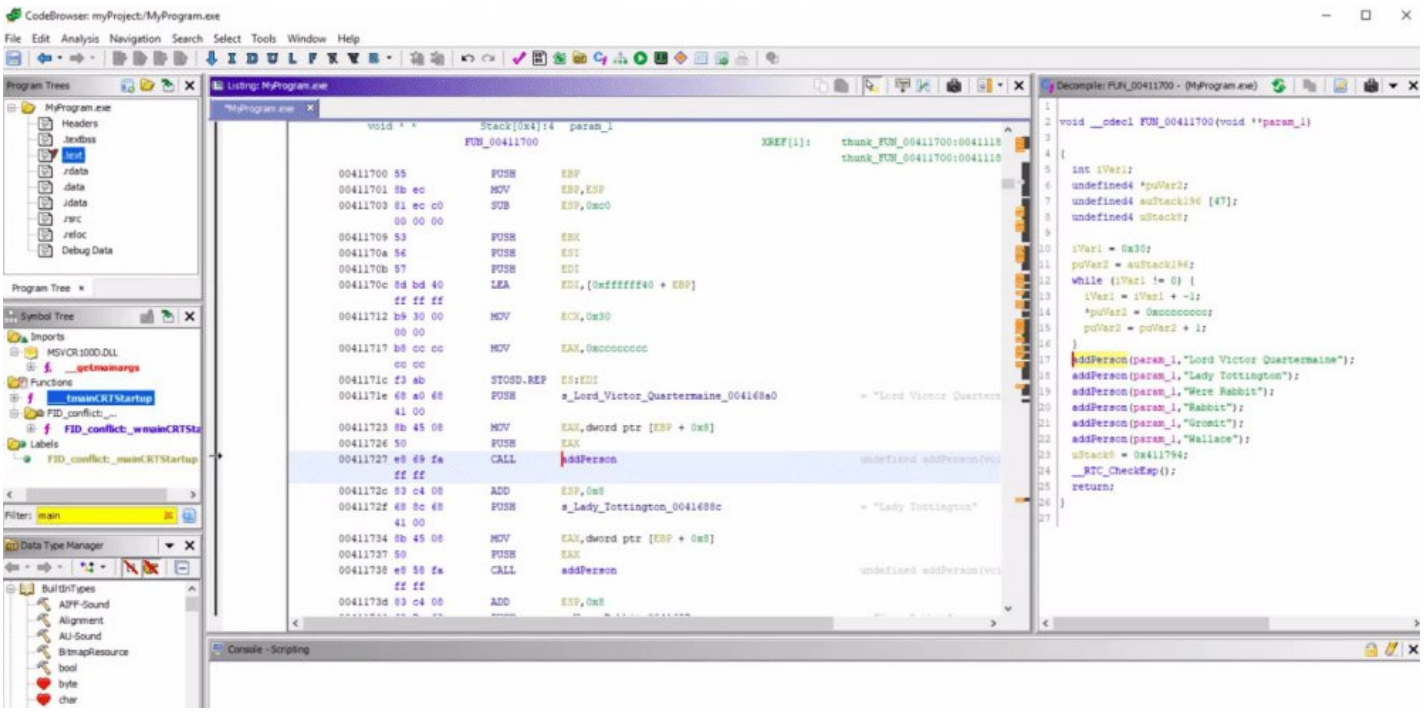


Рис. 3.30 – Дизассемблирование двоичного файла в Ghidra

Ghidra также предлагает множество функций, таких как анализ кода и **декомпиляция** (по другому известный, как процесс обратный компиляции), которые могут быть полезны для создания патчей и эксплоитов или модификации программ (см. рис. 3.30).

IDA Pro

IDA Pro это коммерческий дизассемблер, поддерживающий Windows, macOS и Linux. С помощью IDA Pro, вы сможете заниматься реверс-инжинирингом программного обеспечения (см. рис. 3.31):

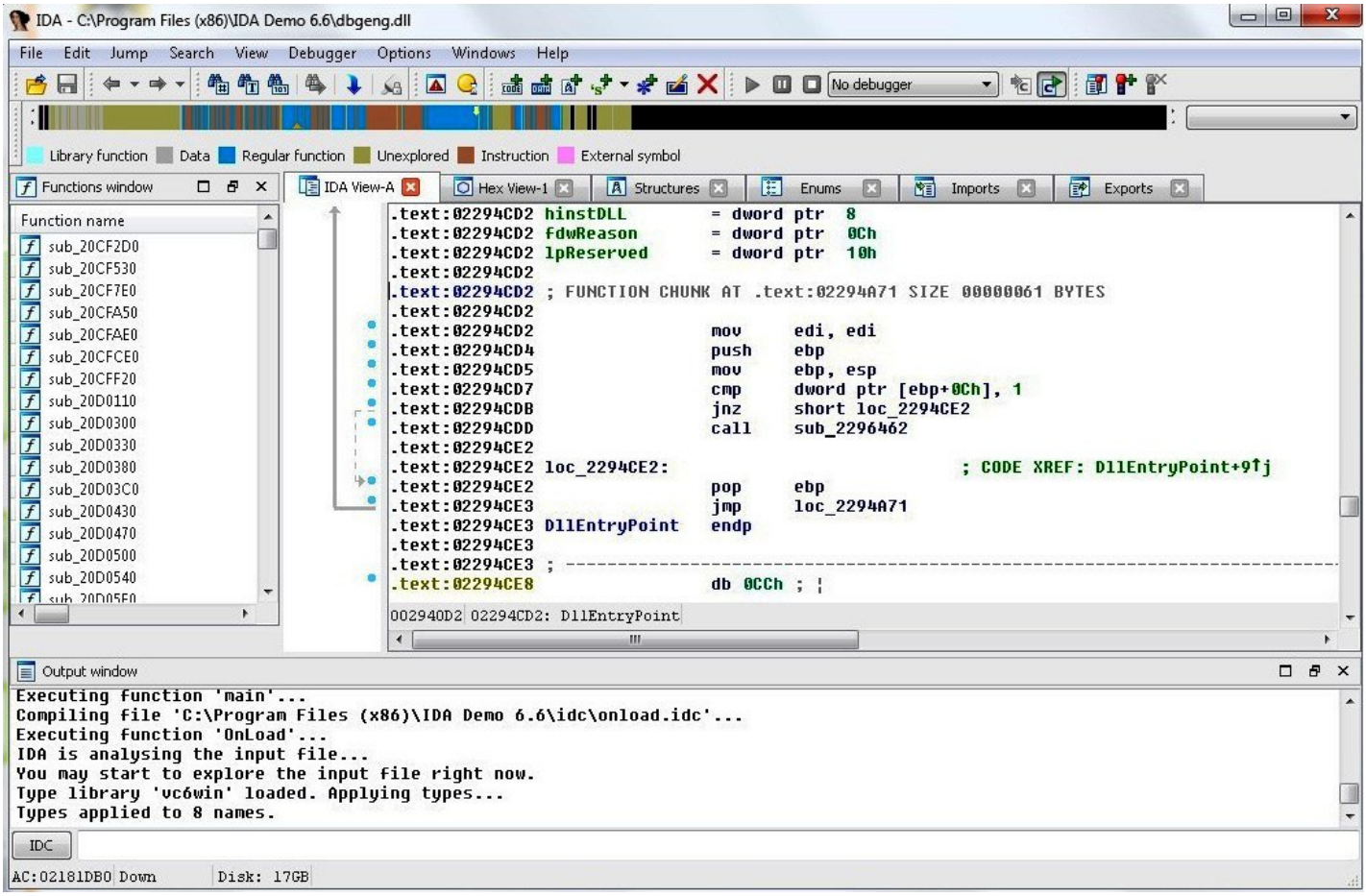


Рис. 3.31 – Дизассемблирование двоичного файла в IDA Pro

IDA Pro несомненно, лучший из существующих инструментов для дизасемблирования – однако лицензия на это ПО для одного пользователя, включающая все опции (все декомпиляторы), что-то около 30000$, на момент публикации этого материала.

Sysinternals Suite

Мы были бы полными разгильдяями, не включив этот феноменальный набор инструментов в наш список. Хотя технически, это не декомпилятор или отладчик, это наиважнейший набор утилит и приложений, которые могут помочь и очень сильно, при их использовании для тестирования приложений Windows. **Sysinternals Suite** содержит более 70 бесплатных инструментов командной строки и GUI-утилиты, которые помогут в просмотре дампов памяти, конфигураций Active Directory, запущенных процессов, исследовании процессов на уровне потоков и много чего еще. Если вы рассматриваете исследование приложений Windows, скачайте ее, как только получится – это золотая жила!

В завершение этой главы, подытожим – эти инструменты представляют арсенал превосходных средств, которые вы можете использовать для построения тестовых окружений, эффективного заполнения заметок об исследованиях и подтверждения уязвимостей, изучения сетевых соединений, отладки и дизассемблирования приложений. Несомненно, исследуя и узнавая больше об уязвимостях, вы будете добавлять в свой арсенал, инструменты наиболее приглянувшиеся лично вам.

Итоги главы

В этой главе мы узнали, что такое исследование уязвимостей и как его проводить. Мы также рассмотрели некоторые из наиболее распространенных техник, применявшихся при исследовании уязвимостей. Мы изучили различные типы стратегий и ресурсов, которые используются для обнаружения новых уязвимостей. Мы также узнали о кое-каких трудностях, встающих перед исследователями уязвимостей. Мы выучили, как выбирать наилучшие цели для исследований. Мы тщательно исследовали примеры тестов и то, насколько они важны в качестве метода, который поможет стать вам более результативными в поиске уязвимостей. Мы также выяснили, с какими инструментами нам стоит хорошенько познакомиться, чтобы начать деятельность в качестве исследователя уязвимостей.

Вся эта информация, очень пригодится нам в будущем, при проведении наших собственных исследований уязвимостей и вам следует овладевать ей на протяжении своего пути начинающего исследователя. Хотя, во время ваших исследований, совершенствование в искусстве обнаружения уязвимостей неизбежно, что же вам нужно делать дальше? В следующей главе, мы поговорим о том, как разумно поступать исследователям и с чего начинать продвижение отчетов об уязвимостях.

Дополнительные материалы

Чтобы подробнее узнать о программном обеспечении и ресурсах, рассмотренных нами в этой главе – просмотрите эти источники:

- NVD – https://nvd.nist.gov
- База данных CVE – https://www.cve.org
- Список уязвимостей ФСТЭК: https://bdu.fstec.ru/vul?sort=datv
- Exploit-DB – https://www.exploit-db.com
- Full Disclosure – https://seclists.org/fulldisclosure/
- Канал DEFCON на Youtube – https://www.youtube.com/user/DEFCONConference
- SourceForge – https://sourceforge.net/
- CherryTree – https://www.giuspen.net/cherrytree/
- MITRE ATT&CK – https://attack.mitre.org/
- HackTricks – https://book.hacktricks.xyz/
- Joplin – https://joplinapp.org/
- Greenshot – https://getGreenshot.org/
- OBS – https://obsproject.com/
- VMWare Workstation Pro – https://www.vmware.com/products/workstation-pro.html
- VirtualBox – https://www.virtualbox.org/
- Hyper-V – https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/about/
- Burpsuite – https://portswigger.net/burp
- ZAP – https://owasp.org/www-project-zap/
- Fiddler Classic – https://www.telerik.com/download/fiddler
- OllyDbg – https://www.ollydbg.de/
- Immunity Debugger – https://www.immunityinc.com/products/debugger
- WinDbg – https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/debugger-download-tools
- Radare2 – https://rada.re/n/
- Rizin – форк Radare2, развивающийся параллельно – https://rizin.re/
- Ghidra – https://ghidra-sre.org/
- IDA Pro – https://hex-rays.com/ida-pro/
- Sysinternals Suite – https://docs.microsoft.com/en-us/sysinternals/downloads/

Часть 2 – Раскрытие информации об уязвимостях, ее публикация и составление отчетов

Раскрытие, публикация и составление отчетов – являются критичными стадиями в исследовании уязвимостей. В этой части вас ждет обучение правильным методам раскрытия уязвимостей поставщикам ПО и передаче ваших пожеланий по распространению исправлений среди их клиентов. Затем вы изучите, как опубликовать свою работу в распространенных базах данных по безопасности, к кому вам обращаться, если в течение процесса раскрытия, что-то пойдет не так, и наконец – как поделиться этой информацией самостоятельно.

Эта часть, содержит следующие главы:

- *Глава 4, Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности*
- *Глава 5, Публикация уязвимостей – публикуем свою работу в базах данных*
- *Глава 6, Арбитраж уязвимости – что пошло не так и кто может помочь*
- *Глава 7, Независимая публикация уязвимостей*

Глава 4. Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности

В 2008 году Дэн Камински, обнаружил серьезную уязвимость в протоколе **Domain Name System** (**DNS**) - **Системы Доменных Имен**. Эта уязвимость была чрезвычайно опасна, так как она могла быть использована для перенаправления трафика с целых доменов (со всеми поддоменами), таких как .com или .org. Это позволяло бы атаки MITM (man-in-the-middle – человек посередине), перенаправления сайтов и еще некоторые сценарии. Это была существенная уязвимость с широчайшей поверхностью атак.

Камински понял угрозу и немедленно предупредил Майкрософт, которые с ним работали и многих других поставщиков ПО тайно, до устранения уязвимости. Как только патч был готов, Камински созвал пресс-конференцию, на которой он и производители ПО анонсировали патчи, защищавшие технологию. На пресс-конференции, небольшие подробности были разглашены, из-за необходимости исправления уязвимых систем. Однако Камински, разболтал вообще все, что будет раскрыто, в докладе на конференции BlackHat в Лас-Вегасе, штат Невада, через несколько недель после пресс-конференции.

Исследователи уязвимостей, заметили это и начали строить предположения об уязвимости Камински. Некоторые исследователи, начали делиться своими мыслями насчет этого бага в своих блогах – были озвучены как ценные, так и бесполезные подробности связанные с его эксплуатацией. В то же время, исследователь случайно опубликовал некую инсайдерскую информацию об усилиях прилагаемых к устранению бага, до доклада Камински информирующего о возможности эксплуатации. Пользователи уязвимых систем, имели около 2 недель на исправление уязвимости перед ее случайным раскрытием. Камински поделился с прессой, так как рассчитывал, что производители ПО и общественность имели больше времени, по крайней мере несколько недель на исправления, до того как уязвимость была использована. Хотя это далеко от идеала, вендоры получили этот короткий промежуток времени, для исправления уязвимых систем, потому-что Камински сообщил о результатах своего анализа безопасности, нужным людям в нужное время, в ходе того, что называют **раскрытием информации об уязвимости**.

Когда мы неизбежно найдем угрозы безопасности, нам необходимо держать о них в курсе, причастных производителей ПО. К сожалению, иногда это заставляет нас работать с вендорами, в режиме секретности, разглашая информацию, только если все участники согласятся с раскрытием. Хотя не все раскрытия информации следуют этому пути, мы можем сделать несколько вещей, чтобы убедиться, что используем лучший из возможных подходов к сообщениям о результатах анализа безопасности. В этой главе, мы погрузимся в эту тему знакомясь с принципами в области эффективных стратегий сообщения о результатах анализа безопасности.

Вот, что мы изучим в этой главе:

- Мы изучим, что такое раскрытие информации об уязвимости и почему это важно.
- Мы обсудим различные виды раскрытия информации об уязвимостях, и когда их применять.
- Мы познакомим вас со стратегиями, которых вы сможете придерживаться в раскрытии уязвимостей, затронутым производителям ПО.
- Мы узнаем о некоторых распространенных препятствиях, с которыми вы столкнетесь и каким образом их преодолевать.

Эти результаты будут представлены на протяжении следующих основных тем этой главы:

- Раскрытие информации об уязвимостях – зачем и почему.
- Различные типы раскрытия информации.
- Введение в процедуру раскрытия информации.

- Обратим внимание на распространенные трудности.

До конца этой главы вы разберетесь, что такое раскрытия информации об уязвимостях и почему они важны. Вы также получите хорошее представление о различных типах раскрытия информации, применяемых исследователями безопасности. И наконец, вы узнаете как поддерживать связь с поставщиками программного обеспечения и осознаете известные трудности сопутствующие процедуре раскрытия информации.

Раскрытие информации об уязвимостях – зачем и почему.

Давайте начнем эту главу с определения, что такое раскрытие информации об уязвимостях и почему оно является важным элементом исследований в сфере безопасности, уязвимостей программного обеспечения и его безопасности.

Что такое раскрытие информации об уязвимости?

Раскрытие информации об уязвимости – это процедура рассекречивания уязвимостей перед обществом. Для примера, раскрытие уязвимости, может быть сделано непосредственно всей публике, производителю создавшему программное обеспечение, консультантам по безопасности или сочетанию этих сторон. Идея состоит в предположении, что информация об уязвимости, переданная разным сторонам – “продавит“ создание исправлений для уязвимости и вендор будет, вынужден предпринять действия по обеспечению безопасности своего продукта и пользователей. Во многих случаях, исследователи безопасности совершают эти раскрытия, чтобы помочь сделать технологии и их использование безопаснее.

Мы начали эту главу с истории про раскрытие уязвимости Дэном Камински Майкрософт и другим производителям ПО, насчет изъянов найденных в DNS. Сообщество информационной безопасности представило в ярких красках, что могло бы произойти и почему это было не лучшим способом раскрытия информации об уязвимом программном обеспечении. В конце концов, это подтолкнуло исследователей безопасности к определению лучшего порядка действий по раскрытию уязвимости и в этом варианте, вендоры получали возможность разработать набор исправлений для своего продукта, а пользователи этих продуктов, могли исправить уязвимость до того, как случится массовое использование уязвимости. Значит, это лучший подход, верно?

На каждую историю о совместном раскрытии уязвимости, приходится куча раскрытий крайне узкому кругу лиц. Вот, например – когда исследователь безопасности Ли Энн Гэллоуэй, обратилась в Myspace в 2017 году, по поводу нахождения бага в Myspace, позволявшего доступ к профилям пользователей без аутентификации, она не получила ответа от Myspace и спустя месяцы, после неоднократных обращений. По ее воспоминаниям, она подумала, что лучшим способом сохранить безопасность пользователей, была публикация ее исследований в своем блоге (ссылку на оригинальное раскрытие, найдете в разделе *Дополнительные материалы*, в конце этой главы). Myspace исправили проблему, в пределах нескольких часов, после их уведомления о публикации в ее блоге. В данном случае, хотя связь с поставщиком программного обеспечения, в какой-то степени произошла, действия с его стороны, были предприняты только после открытой публикации уязвимости.

Суть в том, что раскрытие уязвимостей может выглядеть совершенно по разному, если конечной целью является защита пользователей, но исследователи уязвимостей могут подходить к раскрытию, по своему для достижения конечной цели. Но почему это имеет значение?

Почему раскрытие информации об уязвимостях важно?

В зависимости от того, кто вы такой и кто вас спрашивает, вы можете давать разные ответы на вопрос, почему раскрытие информации об уязвимостях является важным. С точки зрения исследователя, раскрытие уязвимостей важно, потому что помогает им определить угрозы для пользователей программных продуктов, обеспечивая общественное давление на вендоров, которые в ином случае, могли бы не захотеть исправлять проблемы. Общественное давление, в конечном счете подталкивает производителей ПО к созданию лучших продуктов. Еще одним важным моментом для исследователей, является то, что это способствует достижению ими профессиональных целей. А также для исследователей, которые может быть не занимаются этим полный рабочий день, но хотят вкатиться в индустрию. Публикация исследований, это отличный метод, помогающий наполнению вашего резюме и представления сообществу своего опыта проработки вопросов безопасности. С точки зрения поставщиков ПО, раскрытие информации об их продуктах и практических аспектах безопасности, могло считаться источником проблем.

В Соединенных Штатах, существует правительственное учреждение по обеспечению безопасности продуктов питания, называющееся **Управление по контролю за продуктами и лекарствами** (**FDA**). FDA было сформировано в 1906 году, после возмущения общественности из-за антисанитарных условий, описанных Эптоном Синклером в его новелле *Джунгли*. Синклер основывался в своем рассказе, на личном опыте работы тайного агента на скотном дворе в Чикаго. Синклер написал о компаниях, которые расфасовывали и продавали тухлое мясо, испортившуюся продукцию бодяжили химикатами и даже отрезанные части тел работников – отправлялись в колбасу. В то же самое время, рабочие больные туберкулезом кашляли, сплевывая кровь в продукцию, которая затем фасовалась и продавалась покупателям. В результате Синклер и его вымышленное разоблачение, были облиты грязью мясоперерабатывающей промышленностью. Хотя в Соединенных Штатах, есть разногласия вокруг правительственных учреждений и их эффективности, правила установленные в конце 19 века, относительно безопасности продуктов питания, чрезвычайно улучшились на протяжении их применения и контроля. Компании обязаны удовлетворять специальным требованиям к упаковыванию и продаже пищевых продуктов и дать согласие, чтобы продукция продавалась в торговых точках, соответствующим стандартам безопасности.

Хотя это и не применимо к каждому производителю программного обеспечения, изрядная часть контор по разработке, отражает положение в мясоперерабатывающей промышленности, описанное в Чикаго в 1906 году. Современные программные продукты, изготавливаются и продаются в сомнительных условиях, лидируя по небезопасным для покупателей продуктам. Можно с уверенностью говорить о недостаточном качестве программной гигиены. Перегруженные работой недоучки, собирают продукты на коленке и софт выбрасывается на рынок, без достаточного внимания к его безопасности.

В случаях, когда компании осведомлены об уязвимостях (или на их взгляд, неприятных издержках) в своих программных продуктах, они могут предпринять действия для их исправления. Хотя это может быть затратно, производители программного обеспечения, часто предпринимают шаги по предотвращению будущих расходов, культивируя внедрение мероприятий по безопасности на ранней стадии жизненного цикла процесса разработки ПО.

Прогрессивные в отношении безопасности компании, думающие о защите своих продуктов, могут приветствовать раскрытие информации, но так бывает не всегда. Иногда вы будете общаться с вендорами, которые отказываются от конструктивного диалога по решению проблем, найденных в их продуктах и еще меньше, хотят публиковать информацию о них. Попросту говоря, раскрытие информации об уязвимостях, это больная тема для некоторых производителей, однако им необходимо понять, что раскрытие уязвимостей – неотъемлемый элемент, в конце концов повышающий качество продукта.

Мы коротко обсудили пару разных сценариев, включающих отличающиеся виды раскрытия информации. Теперь, давайте обсудим их развернуто.

Различные типы раскрытия информации

Раскрытия информации, проходят в нескольких различных форматах. Вы, как исследователь безопасности, могли предполагать, что раскрытия не будут одинаковыми. Однако, исследователи безопасности достигли общего соглашения по разновидностям раскрытия информации, преследующих отличающиеся цели. Мы разъясним, каждый из этих общепринятых типов раскрытий и затем обсудим, как выбрать из них наиболее подходящий к вашему сценарию и поставленным целям.

Согласованное раскрытие информации

Согласованное раскрытие, часто известное как **Ответственное раскрытие информации** – это, когда вся информация об уязвимости публикуется, но только после того, как компания получает возможность исправить ее, в процессе совместной работы с исследователем безопасности. В сценарии с DNS, начинавшим эту главу, так в точности и произошло. Это дало компаниям, время на исправление проблемы, до того как злоумышленники ее используют. Ответственное разглашение информации, требует присутствия всех сторон для оценки угроз, выработке и исполнению плана по исправлению ситуации.

Представители производителей программного обеспечений, готовые работать по ответственному раскрытию информации о безопасности, часто размещают *политику ответственного раскрытия информации*, на своих веб-сайтах. Такие политики, могут иногда подпадать под политику конфиденциальности *Safe Harbor*, разработанную правительством США для защиты личных данных граждан и резидентов стран-членов Европейского Союза, обрабатываемых на территории США. Прекрасный пример подобной политики, опубликован **агентством безопасности и защиты инфраструктуры** – ***Cyber Security and Infrastructure Security Agency** (**CISA**), по этому образцу, компании могут разрабатывать собственные политики раскрытия, подходящие под их нужды:

(в России это **Федеральная служба по техническому и экспортному контролю (**ФСТЭК**), **Национальный координационный центр по компьютерным инцидентам** (**НКЦКИ**) и **Центр мониторинга и реагирования на компьютерные атаки** (**ГосСОПКА**))*

What we would like to see from you

In order to help us triage and prioritize submissions, we recommend that your reports:

- Describe the location the vulnerability was discovered and the potential impact of exploitation.
- Offer a detailed description of the steps needed to reproduce the vulnerability (proof of concept scripts or screenshots are helpful).
- Be in English, if possible.

What you can expect from us

When you choose to share your contact information with us, we commit to coordinating with you as openly and as quickly as possible.

- Within 3 business days, we will acknowledge that your report has been received.
- To the best of our ability, we will confirm the existence of the vulnerability to you and be as transparent as possible about what steps we are taking during the remediation process, including on issues or challenges that may delay resolution.
- We will maintain an open dialogue to discuss issues.

Рис. 4.1 – Шаблон CISA с информацией ожидаемой от обеих сторон, чтобы ни у кого не оставалось неопределенности

За ссылкой на этот шаблон, загляните в раздел *Дополнительные материалы*, этой главы.

Качественные политики, должны ясно устанавливать, что могут делать вендоры и что могут делать исследователи в процессе совместной работы. Когда производитель ПО имеет такую политику, он вероятно более продвинут в отношении безопасности и продумывает конструктивное взаимодействие с исследователями.

Когда исследователи безопасности, должны применять согласованное раскрытие?

Вообще говоря, мы рекомендуем его, в качестве первоочередного метода раскрытия, который следует проводить всем, кто стремится в итоге защитить пользователей. Мы коротко рассказали про политику конфиденциальности Safe Harbor и ответственное раскрытие информации. Как правило, компании вместе с этими политиками имеют какие-то методы или процедуры связанные с согласованным раскрытием, которые могут дать вам хорошие ориентиры, в плане предполагаемых результатов от работы такой компанией.

Допустим что, у компании нет процедуры или политики согласованного раскрытия. В таком случае, нет причин, по которым исследователь не может попытаться совместно поработать с производителем программного обеспечения, в формате без вышеупомянутой политики. Конечно, добросовестные вендоры, захотят с вами работать, однако в таких ситуациях вы, как исследователь безопасности, должны руководить порядком действий для обеспечения лучшего из возможных исходов. В следующем разделе посвященном взаимодействию с компаниями, мы обсудим, что влечет за собой такое руководство.

Хотя это могло бы быть наиболее распространенным и предпочтительным способом заинтересовать производителей ПО, некоторые другие методы, могут и применяются для достижения похожих результатов. Следующим, мы обсудим конфиденциальное раскрытие.

Конфиденциальное раскрытие

В модели **конфиденциального раскрытия информации**, исследователи раскрывают баг, непосредственно вендору, большей частью как и в согласованном раскрытии. Затем производитель программного обеспечения, в закрытом порядке полностью решает, публиковать ли и исправлять ли уязвимость. Как правило, конфиденциальное раскрытие принято использовать, когда исследователь состоящий в штате, находит уязвимости в продуктах своей компании или используемых ей продуктах, которые разработаны или поддерживаются сторонним производителем. Например, это часто происходит, в случае заключения договора на проведение пентеста определенных сервисов, когда оплачиваемый эксперт тестирует продукты и отчитывается только перед клиентом, в рамках оплаченного контракта. Процесс конфиденциального раскрытия, может быть остановлен после начала обмена информацией или выпуска отчета. Сторона, получающая раскрытую информацию об уязвимости, возможно не захочет публиковать о ней подробности, и может исправить, а может и не исправлять уязвимости.

Почему это важно? Действительно, для компаний использующих ПО какого-то производителя, иногда, сохранение угроз в тайне, является допустимым риском для всех участвующих сторон. Иногда, угроза имеет низкую или среднюю значимость для той и другой стороны или ни одна из сторон не заинтересована, в том чтобы широкая публика знала об этой уязвимости.

Когда же исследователи безопасности, должны применять конфиденциальное раскрытие информации?

Если вы нанялись для работы на компанию непосредственно, заключив с ней договор, требование конфиденциального раскрытия информации, могло быть включено в ваш контракт. Это

редкость для соглашений содержащих пункты о публичном или согласованном раскрытии, но все же убедитесь, что понимаете юридические ограничения своих действий в рамках любого контракта. В ходе конфиденциального раскрытии информации, любопытно то, что некоторые компании, будучи осведомленными об угрозах любых видов публичного разглашения информации, а также юристы и ведущие специалисты по безопасности таких производителей ПО, часто включают в свои продукты, условия или формулировки ограничений применения, помогающие защитить компании и устанавливающие юридические методы возмещения ущерба, если кто-то соглашается использовать их ПО, а потом ставит целью сделать широко известной любую найденную уязвимость.

С такими вводными, перед тем, как заняться любым раскрытием, мы считаем, что условия использования ПО вами прочитаны и любые обязательства, соглашения или условия договора учтены. Не каждый производитель программного обеспечения, склонен к исследованиям в области безопасности, однако достаточно будет указывать, где это может привести к проблемам, при вашем обращении к вендорам. Теперь, давайте поговорим о более резонансных раскрытиях: полноценные раскрытия информации.

Полноценное раскрытие информации

Полноценное раскрытие информации – это, когда исследователь публично раскрывает всю информацию об уязвимости. Такое раскрытие, может включать информацию о том, каким образом использовать уязвимость и любых доступных методах снижения или предотвращения ущерба. Некоторые могли утверждать, что как только пользователи узнают о проблеме, они смогут предпринять меры по своей защите. Скажем так: полноценное раскрытие информации, не всегда гарантирует наличие техник снижения или предотвращения ущерба. Существует много доводов за и против при оценке полноценного раскрытия, вам следует уделить время и учесть их, перед выкладыванием информации о своем исследовании на всеобщее обозрение.

Предположим, что злоумышленники смогут использовать уязвимость, до того, как она будет исправлена. В таком случае, вы можете оказаться втянутыми в конфликт связанный с любым ущербом, который может придти на ум пользователям, не имеющим защиты против обнаруженных вами уязвимостей. Тем не менее, как правило, бремя последствий ложится на поставщика программного обеспечения, в случае их неадекватной реакции на уязвимости безопасности, в случае когда они честно уведомлены о них.

Когда исследователям безопасности, следует использовать полноценное раскрытие информации?

В большинстве случаев, полноценное раскрытие информации, должно использоваться только в качестве крайней меры. Обычно, это когда вендор не идет на согласованное раскрытие и исследователь предполагает, что пользователи подвергаются угрозе атак. Например ранее в этой главе, мы рассказали о раскрытии уязвимости в Myspace. В этом случае, исследовательница безопасности предпочла сделать полноценное раскрытие информации, написав об этом в своем блоге, после того, как производитель ПО игнорировал любые ее сообщения, на протяжении нескольких месяцев. Как только это было сделано, вендор подсуетился, закрыв уязвимость в течение нескольких часов.

Если вы решите добиваться полноценного раскрытия информации, убедитесь, что у вас есть документированное подтверждение того, что вы предоставляли компании квалифицированное уведомление об уязвимости, и либо не получили ответа, либо компания не пожелала сотрудничать. Уберите любую дискредитирующую или секретную информацию из вашего раскрытия и, смотря по величине угрозы, подготовьтесь к повышению внимания к своей персоне. Компания может быть завалена запросами СМИ и заставлена исправить уязвимость. Вдобавок, вы можете увидеть отзывы

и цветистые комментарии сообщества информационной безопасности по поводу того, почему бы они поступили или не поступили, так как это сделали вы.

Определенной альтернативой, подобной полному раскрытию, но несущей меньший риск разоблачения существенной информации об эксплуатации, является частичное раскрытие информации. Давайте, поскорее рассмотрим это.

Частичное раскрытие информации

Частичное раскрытие информации, это когда избранная информация об уязвимости, открыто опубликована, непосредственно самим исследователем. Подобно полноценному раскрытию, частичное может содержать краткие доказательства эксплуатации уязвимости, в доступной подаче СМИ. Однако там, обычно нет конкретной информации о том, как могла бы быть выполнена эксплуатация или конкретные детали уязвимости. Обычно частичные раскрытия, случаются в социальных сетях вроде Twitter или личных блогах, чтобы способствовать привлечению внимания к производителю ПО, иначе нежелающему сотрудничать или игнорирующему обращения.

В каких случаях, исследователи безопасности, должны применять частичное раскрытие информации?

По большей части, как и *полное раскрытие*, *частичное* должно быть тщательно проверено, до того, как оно будет опубликовано. Тем не менее, если целью исследователя, является привлечение внимания к полученным результатам, когда другие способы окончились неудачей – это прекрасный метод подталкивать поставщика программного обеспечения в нужном направлении. Частичные раскрытия информации, часто использовались для создания давления (на производителей), подобно полным раскрытиям, помня однако, об интересах пользователей. Раскрытие не всех деталей целиком, помогает обезопасить пользователей от угрозы немедленной эксплуатации. Добавим к сказанному выше, что раскрытие информации об уязвимостях привлекает внимание других исследователей, которые хотят провести реверс-инжиниринг или повторное исследование сами, что может привести к полному раскрытию информации, другой стороной.

Теперь, когда мы имеем хорошее представление о наиболее применяемых типах раскрытия информации, давайте повернем к обсуждению согласованного раскрытия информации по программам баг-баунти. Это распространенная и развивающаяся практика среди компаний, чтобы отдать раскрытие информации об уязвимостях на сторону.

Баг-баунти и согласованное раскрытие информации

С нашей стороны было бы безответственно, не выбрать немного времени для обсуждения **баг-баунти** в этой главе. Хотя баг-баунти не обязательно являются процедурами раскрытия информации, это программы, которые компании, прогрессивные в сфере безопасности используют, чтобы способствовать согласованным раскрытиям. По-сути, программы баг-баунти – выступают посредниками, которые принимают результаты от исследователей безопасности, подтверждают их и передают дальше в компании, которые их наняли. Обычно эти посредники, хранят финансовый депозит, который может и будет расходоваться как премия, если компания в конце концов, признает значимость результатов исследования. На момент написания этой книги, две наиболее крупных платформы баг-баунти, с открытым исходным кодом и доступные всем желающим – это HackerOne и BugCrowd. Сильным мотивом для исследователей безопасности, является то, что раскрывая значимые уязвимости, они получают хорошее вознаграждение за свои раскрытия. Производители ПО, регулярно объявляют вознаграждения в размере от 10$ до миллионов долларов. Как правило, программы баг-баунти, сосредоточиваются на приложениях, основывающихся на SaaS (*ПО как услуга (software as a service - SaaS)*). В предыдущей главе, мы предостерегали против поиска и исследования случайных

SaaS-решений потому что, незаконно тестировать приложения работающие на серверах, которые вам не принадлежат, без соответствующих разрешений. Платформы, подобные этим, помогают избавиться от таких юридических затруднений и помогают исследователям тестировать баги в безопасности, до некоторой степени определяя, что допускается тестировать и обеспечивая вознаграждение мотивированным исследователям.

В дополнение к уже сказанному, эти компании обычно определяют точные границы того, что может быть протестировано. Любые из найденных уязвимостей, могут быть отвергнуты по различным причинам – некоторые из них, приводят исследователей в раздражение. В частности, бывало немало инцидентов, когда исследователям не удавалось получить выплаты или результаты их исследований были изначально отклонены, а потом использованы для исправления уязвимостей, без оплаты за сделанное раскрытие.

Важное замечание

Не забывайте, что эти компании, работают брокерами и финансово не заинтересованы в гарантиях выплат исследователям, и если прижмет - они будут на стороне производителя ПО. Отдавайте себе в этом отчет.

Несмотря на все эти опасения, мы предлагаем вам ознакомиться с одним из нескольких доступных вариантов. Используем, в качестве примера HackerOne и войдем на сайт платформы, предлагающей исследователям программы баг-баунти, в которых они могли бы принять участие (см. рис. 4.2):

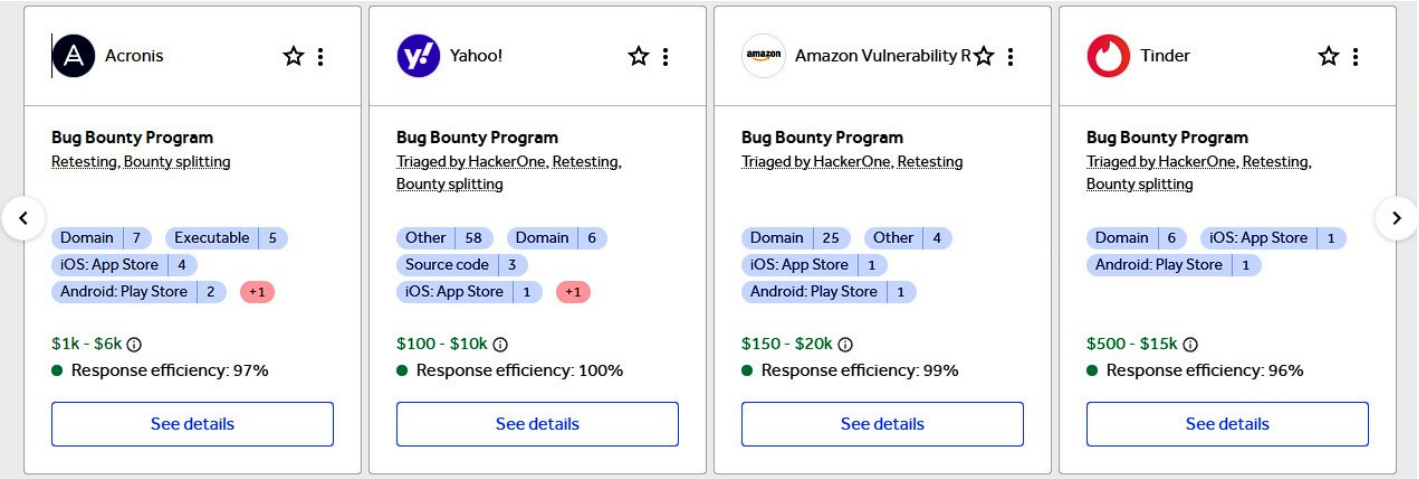


Рис. 4.2 – HackerOne это отличный ресурс для исследований в рамках баг-баунти, насчитывающий тысячи действующих программ

Обычно, каждая программа приводится с ясными ограничениями и правилами, связанными с программой и тем, что ожидается от исследователей, а также условия премирования по результатам. *Чрезвычайно важно не забывать перечитывать все эти правила и политики (см. рис. 4.3).* Их нарушение, может лишить вас права на назначенную награду и участие других программах, а также может подвергнуть вас юридическим последствиям, включая уголовные и гражданские иски против вас:

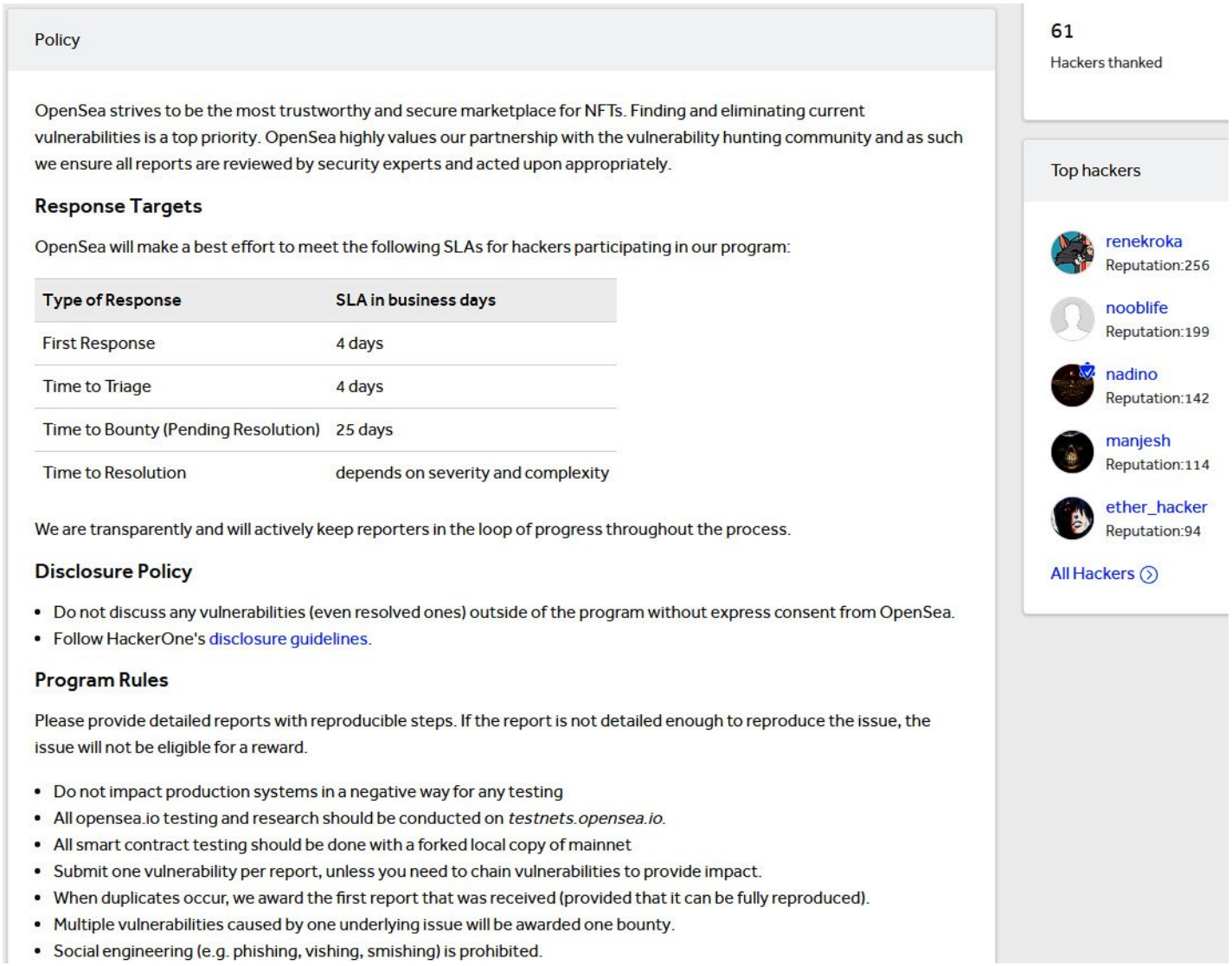


Рис. 4.3 – Убедитесь, что вы поняли правила, заложенные в каждой программе баг-баунти, перед началом своих исследований

Баг-баунти, несмотря на свои страшилки, достойны изучения. Ознакомившись с предыдущей главой, вы можете предпочесть этот тип исследования, большинству исследований с расплывчатыми критериями. Решающим моментом, который любят баг-хантеры, является то, как происходит взаимодействие с производителями ПО. Программы баг-баунти помогают успешно наладить пути взаимодействия с поставщиками и разработчиками программного обеспечения с минимальными усилиями. Такие раскрытия информации и даже логи переписки, часто публикуются на платформе. Это позволит вам быстро понять, какие деньги выплачивают исследователям, как проходит взаимодействие исследователей в процессе программы баг-баунти и, что более важно – какие опубликованы подробности уязвимости, которые помогут информировать остальных, о возможных техниках эксплуатации (см. рис. 4.4):

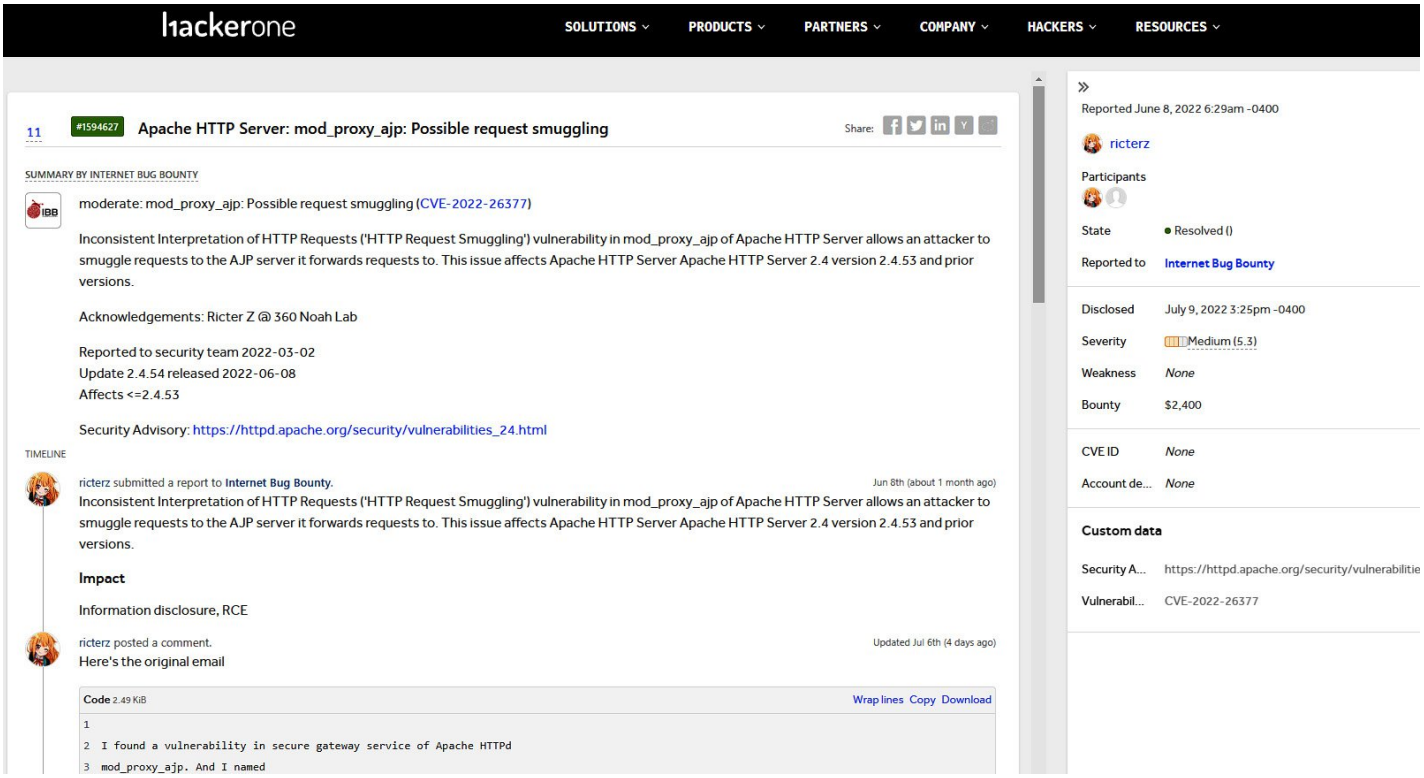


Рис. 4.4 – Типичный журнал раскрытия информации по уязвимости, в котором показаны выплаты и логи переписки

Программы баг-баунти, несмотря на свои недостатки, вносят вклад в улучшение процесса раскрытия уязвимостей производителям ПО, с помощью точных руководств и механизмов, материально поощряя исследователей.

А что, если вы добьетесь большего, свободно исследуя программы, которые не объявляли баг-баунти? Давайте поговорим о том, как следует начинать переговоры насчет уязвимостей и немного об установившейся практике налаживания взаимодействия.

Введение в процедуру раскрытия информации

Встречают по одежке, и в области раскрытия информации о безопасности, это решает судьбу успешного раскрытия, на стадии подготовки вами совместной работы с производителем ПО. Этот раздел поможет вам собраться и научит, как лучше выйти на связь с нужными людьми и произвести прекрасное первое впечатление. Он также даст вам полезные советы, чего *не следует* делать.

Когда вы нашли уязвимость в некоем программном продукте – время подумать о том, что делать дальше. Выработалось общее мнение, что если вы надеетесь защитить пользователей от уязвимостей в программном обеспечении, лучшее что вы можете сделать, это начинать связываться с производителем ПО и попытаться договорится о согласованном раскрытии информации.

Еще одним пунктом памятки, которая поможет вашим планам, является определение, что вы собственно хотите получить от процесса раскрытия информации. Мы сможем создать из этого стратегию взаимодействия, когда четко поймем – что мы хотим получить в результате. Задайте себе следующие вопросы, чтобы разобраться чего вы хотите:

- Вы хотите, чтобы производитель ПО, просто знал об угрозе?
- Вы хотите, чтобы производитель ПО, устранил уязвимость за установленный период времени?
- Вы хотите опубликовать это исследование в СМИ или личном блоге?

- Вы хотите представить свое исследование в базах уязвимостей, так чтобы, другие специалисты по информационной безопасности или широкая публика, узнали об уязвимости и смогли действовать?
- Вы хотите получить награду за свое исследование?

Понимание своих намерений в процедуре раскрытия информации, помогает определиться, как вам действовать. В идеале, наши общие рекомендации – это информировать производителей ПО, делиться результатами исследований, публиковать любую подходящую информацию в блогах и базах уязвимостей и получать доход за свою работу. Однако, раскрытие информации о безопасности, может требовать значительных усилий и вам следовало бы подготовиться к выполнению задач, достижения которых вы планируете.

Определившись со своими намерениями, вам желательно начать собирать предварительную информацию о производителе ПО и своем исследовании, до отправки какого-либо письма в компанию. В дальнейшем было бы полезно, если бы вы скомпоновали свои намерения, **proof of concepts** (**POCs**) и факты, и поместили их в таблицу для составления качественного раскрытия информации. Вот что, вам нужно сделать, чтобы грамотно подготовиться:

- Соберите всю значимую информацию о проблеме, включая код proof of concepts или скриншоты связанные с вашим исследованием. Содержите все это в порядке и храните в надежном месте для будущего использования, на случай если вам понадобится добавить факты или поднять определенные фрагменты информации.
- Узнайте, есть ли у производителя ПО, политика раскрытия проблем безопасности, программа баг-баунти или хотя бы политики безопасности и конфиденциальности. Вам это необходимо, чтобы связаться с наиболее подходящим человеком, который сможет ответить вам по начальной стадии раскрытия информации.
- Проверьте, не является ли информация о проблеме, уже раскрытой производителю программного обеспечения. Например иногда, если вы исследуете какую-нибудь старую программу, уязвимость может оказаться проблемой, которая уже раскрыта и опубликована в их материалах, вместе с техниками предотвращения или снижения последствий ущерба.
- Соберите убедительный комплект сведений, который описывает уязвимость и разработайте четкие инструкции по воссозданию условий, которые продемонстрируют наличие уязвимости. Если есть возможность, создайте пошаговое руководство, включающее стандартные инструменты, которые могут быть скачаны и использованы технически продвинутыми пользователями, которые возможно не слишком разбираются или не имеют доступа к специальным инструментам по исследованию безопасности.
- Если вы хотите связаться с компанией, вам нужно решить, свяжетесь ли вы анонимно, используете псевдоним или ваше настоящее имя. Действуете ли вы в рамках закона или даже политики безопасности компании, обычно следует с осторожностью использовать свое имя. Однако в дополнение к сказанному выше, следует отметить, что компания может вас выследить и натравить на вас юристов для вручения приказа о запрете продолжения исследования. Некоторые исследователи, начинают налаживать контакты анонимно или под вымышленным именем на начальном этапе взаимодействия, для определения, будете ли вы работать с компанией сообща.
- Как только вы соберете всю относящуюся к делу информацию, можете начинать выстраивать первые связи с компанией. Если вы находили контактные данные, кого-либо связанного с безопасностью, на этапе первоначального сбора информации или в политике раскрытия данных о безопасности, вы можете связаться по таким каналам. В ином случае, контакт из общей информации о компании или использование электронной почты их клиентского сервиса – обычно срабатывает.

Ваше письмо, должно содержать следующую информацию:

- Объясните, что вы исследователь безопасности, владеющий информацией об уязвимости безопасности и вам необходимо сообщить о ней представителю компании.
- Если вы связались, не с официальным представителем службы ИБ, попросите о разговоре с ним, чтобы вы смогли озвучить подробности уязвимости.
- Если вы говорите с представителем службы ИБ, включите всю существенную информацию о проблеме, включая пошаговую инструкцию по ее воссозданию, разработанную заранее.
- Сообщите им, что готовы сотрудничать и помогать в исправлении уязвимости.
- Дайте им знать о своем плане, заострив внимание на следующих 30 днях, если у них возникнут вопросы, а вы не получили от них никакого ответа.
- В заключение, ознакомьте их со своими намерениями. Поделитесь с ними, если вы планируете зарегистрировать **CVE**, написать в блоге или возможно заработать, помогая устранить угрозу безопасности, когда уязвимость будет раскрыта пользователям. С другой стороны, это важное предварительное замечание о намерении раскрыть уязвимость и обозначить свое ожидание, что на определенном этапе, компания разгласит информацию.

Ваше письмо, ни в коем случае **не должно**, включать следующие пункты:

- Угрозы насчет раскрытия уязвимости, если они не ответят на ваши письма.
- Угрозы эксплуатировать уязвимость на системах клиентов или угрозы выпуска эксплоита.
- Требование денежных выплат за свое исследование.
- Устрашающие или малопонятные речевые обороты, описывающие уязвимость. Такие письма, обычно пугают людей, которые никогда не получали их в прошлом. Спокойный тон и детальное изложение, выглядят как письмо от дружелюбно настроенного человека, а не сверх-злодея.
- Бесформенные описания или заметки по воспроизведению, в дальнейшем не помогут прояснить уязвимость, при этом достаточно сильно подвергнут риску, налаживание связи.
- Оскорбления, неэтичные высказывания и излишне резкие замечания об уязвимости.

Запомните, первое письмо, должно быть наполнено духом сотрудничества. Пытайтесь подходить к компании как кто-то, кто искренне желает помочь и обеспечить этим сообщением благо пользователей. Итак, когда вы отправили свое первое письмо, что происходит дальше? Давайте поговорим о том, чего вы можете ожидать.

Что происходит после раскрытия?

Это разнится от случая к случаю. Во-первых, компании обычно будут изучать проблему и пытаться определить адекватны ли результаты исследования. Затем они могут связаться с вами, чтобы обсудить ваш первый разговор или сохранить все в кругу своей команды. В большинстве случаев, это побуждает к неформальным переговорам и доработке программы с целью устранения проблемы, которые могут происходить без каких-либо уведомлений.

В своем первоначальном обращении, вы должны были сообщить вендору, что подождете его ответа в течение 30 дней. Если вы так и не получили от него ответа, попытайтесь снова. Спросите, как вы можете помочь и поделитесь новым планом отслеживания прогресса на 30 дней. Установившейся практикой, является предоставление компании до 90 дней для ответа на ваш запрос. Если вы все еще не получили от них ответа, вы должны решить, что делать. Сейчас вам необходимо принять решение, продолжите ли вы связываться с ними или собираетесь раскрыть информацию об уязвимости, применяя полное или частичное раскрытие. Если вы собираетесь открыть CVE, опубликовать это в блоге или разместить в другой базе уязвимостей, дайте им знать. Избегайте угрожающих выражений и изложите, в чем заключается ваш план продвижения вперед.

Теперь, давайте уделим время рассмотрению шаблона, который вы можете подготовить для первого контакта с производителем программного обеспечения. Придется здорово повозиться с тем,

чтобы ваше письмо для налаживания связи, выглядело как надо, если вы хотите произвести прекрасное первое впечатление.

Пример шаблона документа по раскрытию информации

Это шаблон электронного письма, в котором мы в фигурных скобках, укажем переменные для вашей контактной информации, информации об уязвимости и о производителе ПО. Вы должны стремиться к общению в доброжелательном русле и с готовностью к сотрудничеству, которое включает ваши намерения и всю существенную и полезную информацию о вашем исследовании:

Здравствуйте {наименование компании вендора},

Меня зовут {имя исследователя} и я являюсь независимым исследователем безопасности. Я недавно заинтересовался вашим программным обеспечением {название ПО} и потратил определенное время на изучение безопасности этой платформы.

Это письмо, для того, чтобы информировать вас об уязвимостях безопасности, которые я обнаружил в течение моего исследования. Я делюсь этим с вами, потому что я тоже хочу помочь в повышении безопасности вашего замечательного продукта и снизить шанс того, что ваши клиенты пострадают от угрозы безопасности, если уязвимость используют преступники.

Существует {перечисление уязвимостей} в {наименование ПО}, конкретно в версии {номер версии}. Я опишу эти уязвимости:

Уязвимость 1

*{Наименование и ID, указанное в **CWE(Common Weakness Enumeration – Список Распространенных Уязвимостей**)}*

Эта уязвимость найдена в {где была найдена уязвимость} части приложения. Результатом этого дефекта, может оказаться {описание угрозы из CWE}, который может повредить вашим клиентам в {сценарий, как эта уязвимость могла бы повредить пользователям}.

Для воссоздания этой уязвимости, вы можете проделать следующие шаги, которые я сопроводил скриншотами, чтобы помочь проиллюстрировать proof-of-concept.

{Действия по воссозданию условий для уязвимости}

Уязвимость 2

{Дополнительные уязвимости}

В качестве части моего исследования, я планирую {написать в блоге / опубликовать CVE}, поэтому я бы с удовольствием принял любую помощь от участников, когда вы исправите и информируете об уязвимости своих клиентов. Кроме того, я с радостью предлагаю любое содействие, которым я могу помочь исправить уязвимость угрожающую {наименование ПО}.

Надеюсь на ваш ответ в ближайшее время. Я знаю, это длительный процесс, так что, я напомню об этом через 30 дней, если не получу ответа от вашей команды по нашим дальнейшим действиям. Если вам требуется любое содействие или помощь, пожалуйста держите со мной связь по электронной почте {email}.

С уважением,

{имя исследователя}

Это отличное начало ясно излагающее, чего мы хотим, когда мы о себе напомним, какие уязвимости были найдены и каким образом воссоздать proof-of-concept. Кроме того мы в обязательном порядке осветили угрозу, определили наименование и версию ПО, и повторили адрес электронной почты исследователя в теле письма, если обратный адрес окажется удаленным при пересылке его другим участникам.

Несмотря на то, что это отличный старт, и скорее всего он обеспечит вам позитивный опыт, мы обсудим, с какими трудностями вы можете столкнуться и как справиться с ними наилучшим образом.

Обратим внимание на распространенные трудности

Раскрытие информации об уязвимости, редко бывает линейным или очевидным процессом. Даже с прекрасным вступительным письмом и подходом к ситуации в духе сотрудничества, дела могут быстро пойти наперекосяк. А сейчас, давайте потратим некоторое время на обсуждение, таких трудностей.

Двойная работа

Хотя это могло быть одной из довольно редких проблем, это стоит обсуждения. Поставщики ПО, периодически могут (справедливо) утверждать, что другое исследование безопасности, уже привлекло внимание к уязвимости, о которой вы сообщили, но вам не стоит об этом переживать. Если вы вложили в исследование силы и время, как вы поступите дальше?

Углубленное исследование

Компании, могли получать частые запросы от исследователей, в зависимости от популярности их продукта, предлагаемого на рынке. В первую очередь, вам следует предложить поделиться своими записями по proof-of-concept уязвимости, если вы этого еще не сделали. Исследователи, не всегда приходят к одинаковым выводам и иногда уязвимость с которой они работают, выглядит довольно похоже на некоторые другие уязвимости, которыми они занимались и исследователи строят предположения, без подтверждения.

Обсуждение плана

Если вы в состоянии проверить, что ваш вендор действительно работает над этим или осведомлен об уязвимости, которой вы с ним поделились, поинтересуйтесь каковы его планы. Производители могли знать о чем-то, но убедитесь, что они собираются это исправлять. Если у них нет планов, самое время подтолкнуть их к тому, чтобы их создать. Если уязвимость подходит под оформление CVE, спросите, не открыли ли ее уже? Если они этого не сделали, предложите им, содействие в открытии уязвимости от их лица.

Случай из практики автора

Как автор, я поделюсь одним случаем из личного опыта работы совместно с поставщиком ПО, по закрытию уязвимости. После прихода к вендору, они заявили, что в курсе об уязвимости и сказали, что работали над ее устранением. Когда мы говорили об их планах по устранению проблемы, оказалось, что их исправлением было обновить клиентов и со временем отказаться от этого программного обеспечения. Я напомнил им, что платные клиенты, работающие непрерывно – могли не принять это, как исправление и должен быть выпущен патч, соответствующий хранящимся на платформе данным, оказавшихся под угрозой. По сути я участвовал в открытии CVE от их лица. Стороны, подверженные уязвимости, получили патчи для своего окружения 8 месяцев спустя. Иногда стоит уточнить, каковы планы, чтобы убедиться, что они есть с самого начала.

Притормозите

Если поставщик ПО прав и вы предоставили дубликат, хорошим планом было бы прекратить. Иногда, приостановка работы и перепроверка, когда исправление будет выпущено, является разумным вариантом действий, если уже есть исследователь, потративший на это время и силы. Если время и силы ликвидацию этой уязвимости, потратили в том числе и вы, всегда можно предложить помочь в тестировании патча, когда от выйдет или вы можете попросить уведомить вас, когда будет выпущено исправление безопасности.

В то время как, задвоение исследования, могло быть одной из тех вещей, которые вы увидите единственный раз за все время, более распространенной проблемой, наблюдаемой в индустрии – являются не отвечающие вендоры. Давайте поговорим об этом сейчас.

Не отвечающие вендоры

Хотя мы кратко обсуждали это, в разделе про планирование взаимодействия, одна из распространенных трудностей, это то, что компании могут не реагировать на ваши сигналы. Это может расстраивать и приводить в уныние, однако есть несколько вещей, которые увеличат ваши шансы на получение ответа.

Проверьте ваши контакты

Убедитесь, что вы связываетесь с нужным человеком или командой – используйте творческий подход к налаживанию связей, если у вас есть сложности с первыми попытками. Например установка связи и общение с вероятно ответственными сторонами/сотрудниками в LinkedIn (офицеры безопасности, служащие работающие с финансами, практики в сфере безопасности и т. д.) – это отличный способ начать разговор. Сообщение в духе *“Привет, я увидел, что вы работаете на компанию X. Вы получили информацию об уязвимости, которую я отправил компании? Мне оказалось довольно затруднительно достучаться до кого-нибудь и я думаю, что вы могли бы мне помочь.“* может дойти очень далеко.

Будьте настойчивы

Старые люди говорят: *под лежачий камень вода не течет* – мотайте на ус! Продолжайте подталкивать ход обновлений и вежливо предлагайте помощь. Чем чаще вас будут слышать, тем более вероятно, что это сработает, если поставщик ПО не отзывается по другому. Продолжая преследовать цель, убедитесь, что ваш тон и подход к делу, является спокойным и профессиональным.

Будьте терпеливы

Убедитесь, что даете разумное время на ответ. На поставщиков ПО влияют их финансовые трудности, правила компании и техническая безграмотность, которые могут замедлить развитие событий. Без связи могут пройти месяцы – но это не обязательно означает, что вендор не работает над исправлением. Они могут скрывать информацию о разработке патча, не доверять вам или не считать, что вас стоит допускать к информации об их действиях. Это вполне справедливо и вам следует набраться терпения в таких обстоятельствах.

Еще одна распространенная трудность, в том, что поставщик ПО, может получить ваш отчет об уязвимости, но решить не исправлять ее по различным причинам. Давайте обсудим такой вариант.

Вендоры, не желающие сотрудничать

Вендоры могут либо изначально, либо со временем стать не желающими сотрудничать, по нескольким причинам. Например, поставщики ПО могут считать, что уязвимость безопасности, это не то, что следует исправлять, из-за представления риска незначительным или такой продукт активно не продвигается и не продается, а следовательно, не стоит тратить средства на принятие мер. Еще одним участвующим фактором, могло бы быть то, что вендор, просто не имеет сотрудников, чтобы правильно понять информацию, раскрытую вами в обращении к нему. И наконец, вендоры иногда занимают позицию, при которой они не доверяют вам, как ни крути. Если вы столкнулись с этой проблемой, попробуйте следующие техники.

Нарисуйте ясную картину риска

Потратьте время на составление формулировки причин, из-за которых уязвимость безопасности обязательно должна быть устранена. Попытайтесь составить этот текст в ключе, который привлечет внимание их клиентов ясностью формулировок. Постарайтесь тщательно изучить, угрозы для пользователей и используйте их в качестве побудительных стимулов. Однако будьте осторожны в ваших высказываниях, избегая запугивания. Например, когда вы пишете *“Это важно, я мог бы взломать это прямо сейчас и похитить все данные ваших пользователей“*, может это и правда, но звучит враждебно. Откорректируйте это сообщение, чтобы оно более походило на *“Это важно. Как показано в моем proof-of-concept, данные ваших клиентов подвергаются риску, в случае использования преступниками этой уязвимости, это может стать причиной крупных финансовых проблем у вас и ваших клиентов. Как я могу вам помочь, в ликвидации этой угрозы?“*, это будет намного более эффективно.

Предлагайте помощь более активно

Они могут нуждаться в большей помощи по разъяснению этой проблемы и донесению ее до их подразделений. Или могло быть так, что ответственные лица не видят в этом проблемы. Проясните проблему с которой вы столкнулись и ответьте с посылом “Я помогу!“, если вы вложили время и силы в решение. Поинтересуйтесь, нужна ли им помощь в доведении информации об уязвимости и связанных угрозах, до разработчиков. Спросите, нужна ли им помощь со средствами снижения последствий эксплуатации. Узнайте, не нужна ли помощь в разработке заметок по раскрытию информации. Если вы возьмете на себя, определенный объем работы, чтобы сделать раскрытие информации проще, они могли бы быть более отзывчивыми насчет исправления уязвимости. Кроме того, это не что-то совсем уж неслыханное для исследователей – получить предложение о работе от компаний, чтобы принять участие в улучшении их продуктов.

Напоминайте им о своих намерениях

Решительно подтверждайте свои намерения. Допустим вы собираетесь открыть CVE, а поставщик ПО не хочет заниматься этой уязвимостью, и ясно подтвердил это. Было бы разумно обеспечить, понимание им ваших дальнейших действий. Напомните им, что как исследователь, вы собираетесь исполнить свой план по раскрытию информации об уязвимости и без их участия, если потребуется. Напомните им, что CVE это международная база данных уязвимостей безопасности, и программный продукт с выпускающей его компанией – свяжут с этим раскрытием информации. Сообщите им, что вам хотелось бы опубликовать эту информацию, *с их участием, а не без него*.

Необычным затруднением в процессе раскрытия информации об уязвимостях, является случай, когда вендор больше не в индустрии. Давайте обсудим наши ограниченные возможности, в таких случаях.

Вендоры, забросившие свой продукт или ушедшие из индустрии

Возможно, создание небезопасного программного обеспечения, не было достаточно прибыльным для производителя, чтобы продолжать зарабатывать, функционируя как коммерческая организация. Такое возможно, и случается, что программное обеспечение могло быть выпущено и применялось по всему миру, но уже не поддерживается производителем, потому что он больше не существует. Если это так, вот пара подходящих вариантов действий.

Обращение к сообществу

Может быть существует сообщество пользователей, которые все еще применяют это программное обеспечение. Попробуйте поискать объединения пользователей или веб-сайты, которые возможно упоминают уязвимый продукт. Связь с этими ресурсами и ознакомление их с потенциальной уязвимостью безопасности, обычно приветствуется и в результате приводит к сотрудничеству.

Неофициальная поддержка

Предположим, что разработчик или может быть кризисное подразделение компании, взяли на себя поддержку устаревшего ПО, в качестве консультанта. Обычно эта информация могла быть опубликована на ресурсах, подобных LinkedIn или даже непосредственно на сайте компании, если он до сих пор в сети. Стоит попробовать, по крайней мере поверхностно взглянуть, не видно ли признаков чего-то такого.

Предстоящее полное раскрытие информации

Полноценное раскрытие информации, как правило лучший вариант, в случае неподдерживаемых устаревших продуктов, сделанных людьми ушедшими из этой сферы деятельности. Однако, так как это неподдерживаемое ПО, оно вероятно не получит исправления. Продумайте это, при своем раскрытии информации. Наверное не надо предоставлять действующий эксплоит или выкладывать в широкий доступ proof-of-concept, если вы не можете обеспечить средства снижения или предотвращения ущерба.

В заключение скажем, что поставщики ПО могут, не отнестись благожелательно к вашим сообщениям, а наоборот, могли бы отреагировать на ваше раскрытие информации – враждебно. Давайте проанализируем эти возможности.

Враждебно настроенные вендоры

В случае если вы начинаете сталкиваться с враждебно настроенными вендорами, вам следует подготовиться к неприятному общению. Поставщик ПО, может преследовать вас в судебном порядке или применить атаки методами социальной инженерии, с целью дискредитировать вас. Если вы при других обстоятельствах, не были информированы о тактике, применяемой враждебными вендорами – они выглядели бы безнравственно и немного нереально.

Например в 2019 году, исследователь безопасности подвергся нападкам высшего руководства (*COO - Chief Operating Officer*) компании, называвшейся Atrient. Компания продавала широко распространенные терминалы для казино Лас-Вегаса, которые клиенты применяли для выплат выигрышей. Исследователь пытался связаться с компанией, чтобы предупредить их об использовании в терминалах, персональных данных без применения шифрования. Вендор проигнорировал, раскрытую ему информацию. После нескольких попыток связаться с вендором, исследователь частично раскрыл информацию об этом факте в Twitter. Твит был замечен, сотрудниками служб безопасности казино Лас-Вегаса и впоследствии **Федеральным Бюро Расследований** (**ФБР**).

После привлечения внимания руководством служб безопасности казино, ФБР впоследствии помогло организовать и проконтролировать переговоры между поставщиком оборудования и исследователем, подтвердить исправление бага через несколько месяцев, а Atrient сообщило исследователям о готовности выплаты вознаграждения, если они подпишут согласие о неразглашении информации. Исследователям ни разу не заплатили. Высшее руководство Atrient, встретило исследователя на конференции, вскоре после объявления компанией о скором выпуске новых продуктов, они были возбуждены и насильно схватили и сорвали с исследователя его бейдж. На этом Atrient не остановились. Они продолжили запускать кампании по дезинформации против исследователя, утверждая, что он угрожал вымогательством и последующим взломом.

К счастью, исследователь был в этой ситуации прав. Там была видеозапись факта нападения, а подробности переговоров относительно сотрудничества исследователя с вендором и ФБР - были зафиксированы и сохранены.

Хотя это исключительно редкий пример раскрытия информации, которое пошло наперекосяк по вине компании, мы искренне надеемся, что вы не запустите сценарий, когда в службе безопасности почувствуют, что вас пора реально отпинать. Давайте поговорим о распространенных тактиках, с которыми вы могли бы столкнуться и мерами, которые мы можете предпринять для самозащиты.

Тактики дискредитации

Одной их обычных тактик, применяемых враждебно настроенными вендорами, является попытка дискредитировать исследователя, нашедшего уязвимость. Это опорочит их работу, и снизит вероятность того, что другие люди воспримут их серьезно в будущем.

Есть несколько разных способов, которыми это может быть совершено. Один из типичных методов, это изобразить исследователя, как кого-то не являющегося компетентным или заслуживающим доверия. Это может быть сделано путем распространения лживой информации об исследователе или его работе, или путем найма кого-то, кто выдаст себя за исследователя и наделает ложных утверждений о его настоящей работе.

Еще один метод, это попробовать разрушить репутацию исследователя, например выставив его, действовавшим якобы неэтично. Это можно сделать, разглашая личную информацию об исследователе или размещая лживую информацию, которая выставляет исследователя, ну хотя бы алкашом.

Лучшей вещью, которую вы можете сделать, это сохранять копии своего исследования и сообщений – публикуйте ваш материал или работайте с кем-то, кто получит ваш рассказ. Выпуски СМИ по информационной безопасности и журналы, могли бы быть заинтересованы услышать, вашу версию изложения событий.

Судебное преследование

Некоторые компании могут выбрать судебное преследование исследователей, раскрывающих информацию об уязвимостях. Это может включать начало судебного процесса или уголовного расследования. В некоторых случаях, эти действия могут быть несерьезными, с единственным намерением заставить исследователей замолчать. В других случаях, это может быть законными попытками наказать исследователей, которые нарушили условия использования сервиса или другие соглашения.

Исследователям, следует отдавать себе отчет о возможности судебного преследования и консультироваться с юристами, перед раскрытием информации о чем бы то ни было или общением с враждебно настроенным вендором, относительно уязвимости.

Завершая главу на этой теме, уже не удивительно, что исследователи безопасности иногда предпочитают использовать вымышленные имена, когда связываются с поставщиками программного обеспечения. Тем не менее, было бы лучше избегать, непреднамеренно нервировать вендора излишне агрессивным подходом к раскрытию информации о безопасности.

Итоги главы

В этой главе мы изучили раскрытие информации об уязвимостях. Во-первых, мы обсудили, что они из себя представляют и почему они так важны. Затем мы разобрали различные типы раскрытий информации, и в каких случаях их использовать. Далее мы обрисовали основы налаживания связи в ходе раскрытия информации о безопасности, на примере нескольких общепринятых приемов. В заключение, мы рассмотрели трудности, связанные с раскрытием информации и полезные рекомендации для достижения поставленных целей сообщения об уязвимостях. Теперь мы имеем, отличное представление о раскрытии информации, что же происходит дальше? Нам необходимо опубликовать уязвимость в базе данных, такой как MITRE CVE или NVD в большинстве случаев, если программное обеспечение, соответствует определенным критериям. В следующей главе, мы рассмотрим все вам необходимое, чтобы опубликовать ваше исследование в таких базах данных.

Дополнительные материалы

Чтобы подробнее узнать о программном обеспечении и ресурсах, охваченных нами в этой главе – просмотрите эти источники:

- Иллюстрированное руководство к уязвимости DNS, обнаруженной Камински: http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html
- Это не Yourspace, это - Myspace: https://leigh-annegalloway.com/myspace/
- Образец политики раскрытия информации об уязвимости: https://www.cisa.gov/vulnerability-disclosure-policy-template
- HackerOne: https://www.hackerone.com/
- BugCrowd: https://www.bugcrowd.com/
- Казино Раздолбаный рояль: история “этичного хакинга“, закончившаяся черт знает чем: https://arstechnica.com/information-technology/2019/03/50-shades-of-greyhat-a-study-in-how-not-to-handle-security-disclosures/

Глава 5. Публикация уязвимостей – публикуем свою работу в базах данных

В сентябре 2017 года, исследователи безопасности работающие в Armis Labs, опубликовали исследование по слабым местам в протоколе Bluetooth. Они назвали эти слабые места и связанные уязвимости – **BlueBorne**. Уязвимости BlueBorne, позволяли злоумышленникам получить доступ к устройству без уведомления пользователя или взаимодействия с ним. Следовательно злоумышленник, может полностью контролировать устройство жертвы. Исследователи обнаружили, что там было восемь уязвимостей в протоколе Bluetooth, которые могли быть использованы для выполнения этой разновидности атаки. Этим уязвимостям подвергались все устройства, которые использовали Bluetooth, включая смартфоны, ноутбуки и устройства интернета вещей (**IoT - Internet of Things**). Их исследование было опубликовано в огромном докладе на 41 страницу, который обнародовал всю подноготную протокола, работающего примерно на **8,2 миллиардах** устройств по всему миру. Этим докладом об исследовании, Armis заложил основу для сегодняшнего исследования Bluetooth и помог индустрии пересмотреть отношение к безопасности повсеместно принятых технологий. В каждом приведенном примере исследования в докладе BlueBorne, команда ссылалась на подробности об уязвимости в **Common Vulnerability and Exposure** (**CVE**). Перед выпуском этого доклада и обсуждения этих CVE – их необходимо было раскрыть и опубликовать.

К этому месту нашего повествования, вы должны отлично ухватить суть уязвимостей, zero-day, техник исследования уязвимостей и раскрытия информации о ваших результатах анализа безопасности. Как ранее обсуждалось в первых двух главах, CVE – это уникальные идентификаторы присвоенные уязвимостям, когда они раскрыты организацией, уполномоченной резервировать и публиковать эти данные. Но как им удалась, публикация в этой базе? Единственный ли это вариант публикации уязвимостей? Всегда ли резонно публиковать уязвимость в базах данных?

В этой главе, вы изучим такие вопросы, как опубликование вашей работы для широкой аудитории с пользой для всех остальных. Это особенно важно для CVE, которые обычно используют, чтобы распространять и точно описывать данные по уязвимостям. Вот, что мы узнаем из этой главы:

- Что собственно публиковать и подробности связанные с получением возможности опубликовать уязвимость.
- Доступные варианты и какой из способов публикации мы можем выбрать.
- Примеры опубликования уязвимостей вместе с **CVE Numbering Authorities** (**CNA**) и **CVE Numbering Authorities of Last Resort** (**CNA-LR**).

Эти результаты будут представлены на протяжении следующих основных тем этой главы:

- Разъяснения о публикации уязвимостей.
- Выбираем подходящий способ публикации уязвимости.
- Практические примеры публикации уязвимостей.
- Установившиеся практики публикации уязвимостей

Следуя советам из этой главы, вы будете на верном пути к получению возможности публиковать уязвимости в базах данных. Давайте приступим к разъяснению процедуры публикации уязвимостей.

Разъяснения о публикации уязвимостей

Давайте начнем эту главу с определения, что такое публикация уязвимостей, важности публикации в жизненном цикле уязвимости и последствий публикации в целом, для сообщества информационной безопасности, пользователей и вас.

Публикация уязвимости, является процессом формирования информации об уязвимости безопасности, повышающим информированность и позволяющим остальным, предпринять действия по собственной защите. Как правило, публикация может принимать различные формы. В открывающем главу примере BlueBorne, был опубликован монументальный докладище, но был ли он технически проработанной публикацией уязвимости? Действительно, в данном случае публикация уязвимости, произошла до ее раскрытия в виде заявки о публикации CVE. В первую очередь, исследователи обращаются за идентификаторами CVE в организацию MITRE, процедуру мы будем рассматривать далее в этой главе, в разделе *Практические примеры публикации уязвимостей*. Это обычно происходит сразу за этапом раскрытия информации, который мы обсуждали в прошлой главе *Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности*. Как только информация об уязвимости оказалась раскрытой, уязвимость могут устранить или обнародовать средства предотвращения ущерба, или снижения его последствий. Однако, если раскрытие информации об уязвимости не вызвало реакции вендора, публикация все еще возможна, в виде процесса, происходящего с участием вендора или без такового.

Хорошим примером отсутствия реакции поставщика ПО на раскрытие информации об уязвимости, могло бы быть исследование *GPS трекера MiCODUS MV720*, опубликованное в июле 2022 года. *MiCODUS* это производитель оборудования и программного обеспечения, который разрабатывал устройства отслеживания по GPS, применяемые в системах управления на флоте. **Агентство безопасности и защиты инфраструктуры** – **Cyber Security and Infrastructure Security Agency** (**CISA**), выпустило информационное сообщение номер ICSA-22-200-01, в котором они описали пять раскрытий информации в CVE, связанных с этим устройством. В исследовании безопасности описаны такие вещи, как небезопасные конфигурации и пароли, все относительно простые вещи для обнаружения грамотным исследователем. Несмотря на это, угроза была серьезной. Эти трекеры, допускали удаленное конфигурирование и управление устройством в виде удаленного отключения функций, геолокацией, отключением аварийных сигналов и маршрутизацией данных. Если бы эти уязвимости были использованы, их последствиями могли стать человеческие жертвы. Исследователи, изначально пытались работать с поставщиком оборудования и начали налаживать с ними связь в самом начале сентября 2021 года. В ходе своего первого общения с поставщиком они не сообщили никаких конкретных деталей об уязвимости. Они получили письмо из отдела продаж вендора, в котором заявлялось, что работы по уязвимостям – ведутся. Компания отказалась говорить с исследователями об уязвимости и не отвечала на любые дальнейшие вопросы о проблеме. В январе 2022 года, исследователи обратились в CISA с просьбой помочь, в CISA просьбу удовлетворили. CISA начало пытаться воздействовать на вендора, однако они столкнулись с похожими трудностями, во время общения с производителем и потерпели неудачу в привлечении вендора к объединению или сотрудничеству по улучшению безопасности GPS трекера. В июле 2022 года, исследование было совместно выпущено исследователями и CISA, в отсутствие рекомендаций по снижению ущерба, или устраненных уязвимостей со стороны вендора на время публикации (https://www.cisa.gov/uscert/ics/advisories/icsa-22-200-01).

Что касается CVE, после того, как подробности уязвимости были опубликованы на общедоступных ресурсах, таких как, CISA, раскрытие информации посредниками, раскрытие компанией-поставщиком ПО или даже независимое раскрытие информации, CVE могут быть *опубликованы* исследователем безопасности или вендором. Какими бы увлекательными ни были публикации по исследованию уязвимостей, зачем исследователям вообще утруждать себя доведением этой информации до сведения общественности?

Почему публикуется информация об уязвимостях?

Когда мы рассматривали жизненный цикл уязвимости, представленный в *Главе 1, Введение в уязвимости ПО*, мы упоминали, что уязвимости не всегда доходят до публикации. Почему же некоторые могли, тем не менее рассчитывать на публикацию информации об уязвимостях? В

действительности есть множество причин, по которым исследователи безопасности предпочитают опубликовать информацию об уязвимостях. В некоторых случаях, они могут захотеть повысить осведомленность о проблеме, так чтобы остальные смогли предпринять действия по собственной защите. В случае с MiCODUS, исследователи опубликовали результаты своего анализа, чтобы помочь информировать общественность о риске использования этих устройств и угрозе, которую они представляли, национальной безопасности Соединенных Штатов. Не все исследователи движимы этими мотивами, тем не менее некоторые исследователи, могли захотеть улучшить содержания своих резюме и надежно подтвердить результаты своего труда и имеющийся опыт, чтобы получить предложение о работе. В других случаях, они могли захотеть оказать давление на вендора, с целью закрыть уязвимость как можно быстрее. А в отдельных случаях, они могли просто захотеть поделиться результатами своих исследований с сообществом информационной безопасности, так чтобы остальные смогли учиться по их работе.

Какой бы ни была причина, публикация это наверное самая простая (хотя и самая долгая) из всех стадий жизненного цикла уязвимости и этот процесс, возможно принесет исследователям прибыль и общественное признание. Это определенно существенные причины для публикации, однако давайте уделим немного времени, рассмотрению этого под разными углами.

В уязвимости BlueBorne, во введении этой главы, как только исследование было обнародовано, это привело к обнаружению новых уязвимостей высокого и критического уровня опасности. Вскоре после доклада 2017, множество практиков в области безопасности опешили от того, насколько много уязвимостей в Bluetooth, вскрылось после первых результатов исследования. Некоторые предположили, что их вероятно было бы значительно больше, будь проведены дополнительные исследования.

Они не ошиблись в своих подозрениях. **Исследование притягивает исследование**, и спустя год после публикации этого отчета, исследователи безопасности со всего мира, опубликовали новые уязвимости в Bluetooth, более серьезные, чем в прошлом году и в намного большем количестве. Публикация важна тем, что помогает доводить исследования безопасности до сообщества, которое в свою очередь помогает досконально изучать технологии используя все богатство своих навыков и разума. Это в конечном счете улучшает качество технологии, повышая ее безопасность.

Когда вы публикуете результаты исследования, вы привлекаете внимание к уязвимостям и способствуете дальнейшим исследованиям. Публикация информации об уязвимостях, отличный способ для исследователей безопасности, поделиться своими результатами с сообществом, так чтобы остальные могли учиться по их работам. Исследователи не созданы одинаковыми, то что видит один человек, может отличаться для другого. Привлечение внимания, в конце концов приводит к большему познанию которое, в свою очередь, заставляет технологии совершенствоваться, чтобы повысить безопасность, иначе существует риск их устаревания.

Еще одним соображением, как мы думаем, насчет важности публикаций – это то, что пользователи обычно от этого выигрывают. Когда уязвимости найдены и опубликованы, это повышает осознание проблемы и позволяет остальным, предпринять действия по собственной защите. Руководители начинают задавать вопросы о рисках связанных с выбором технологий на своих предприятиях, а пользователи становятся более информированными и образованными в вопросах безопасности. В большинстве случаев это, в общем создает давление на производителя ПО, чтобы он улучшал свои продукты. Если вы просто остановитесь на этапе раскрытия информации об уязвимости, вы можете упустить ключевые выгоды от публикации. Если же давления на вендора не возникнет, это может послужить причиной того, что производитель ПО не будет торопиться исправлять дефекты найденные в программном обеспечении. А может вообще просто увильнет от исправления первоначальной проблемы.

Как и в предыдущей главе, где мы рассмотрели первоначальное раскрытие информации об уязвимости, всегда существует риск во время открытия этой информации для общего доступа. Предлагаем рассмотреть некоторые из таких моментов.

Связана ли публикация уязвимостей с какими-либо рисками?

Жизнь такова, что опасности есть в любом занятии, и публикация уязвимостей не исключение. На момент печати этого текста, произошли огромные улучшения за последние 10 лет в подходе к публикации уязвимостей. Крупным и(или) современным производителям, в целом удается делать программное обеспечение более безопасным, а также получать информацию об уязвимостях и публиковать их, без значительных усилий со стороны исследователей.

Сопутствующие этому риски, обычно идентичны рискам, сопровождающим раскрытие информации об уязвимостях. Исследователям безопасности, следовало бы выяснить до опубликования с кем придется работать и попытаться определить, могла ли публикация привести к враждебным действиям производителя ПО. Если вендор был настроен враждебно в ходе раскрытия информации об уязвимости, ждите от него еще большей неприязни в процессе публикации.

В конечном счете, всякий раз, когда вы публикуете информацию об уязвимости, следует предполагать, что ей скорее всего могут злоупотребить преступники. В докладе ***Совершенствование Исправления Уязвимостей Путем Лучшего Прогнозирования Эксплуатации***, исследователи выяснили, что из уязвимостей опубликованных между 2009 и 2018 годами, приблизительно 5%, подвергались постоянной эксплуатации. Из этих 5% уязвимостей, только половина имела опубликованные эксплоиты. Так что, советовали бы, если вы собираетесь раскрывать подробности уязвимости, отказаться от публикации proof-of-concept экплоита, это может снизить шансы использования уязвимости.

В примере, который мы ранее привели относительно производителя программного и аппаратного обеспечения MiCODUS, исследователи вероятно, долго взвешивали, следует ли им раскрывать данные по уязвимости. В этом примере, уязвимости не были исправлены, однако при рассмотрении с участием CISA в качестве арбитра, исследователи и команда арбитража решили, что отказ от раскрытия информации, был бы значительно опаснее, чем ее публикация. В конечном счете эта публикация, привела к тому, что некоторые розничные продавцы убрали продукт из своих магазинов. И эта информация помогла пользователям, принимать более осознанные решения о том, использовать ли им технологию, которая так уязвима и обслуживается компанией, которой плевать на уязвимости, реакцию на запросы исследователей и правительственных учреждений. Мы поговорим более подробно о командах арбитража, таких как CISA, которые могут вам помочь в принятии этих решений, в следующей главе, *Арбитраж уязвимости – что пошло не так и кто может помочь*.

В других случаях, уязвимости могут быть исправлены, значительно медленнее, если они не сделаны достоянием общественности. Иногда просто достаточно, дать вендору знать, о своих планах опубликовать результаты исследований и о том, что вы начнете раскрытие информации, с тем чтобы зарегистрировать ее в базе уязвимостей, для того чтобы побудить его к действиям. Эти опасности не должны отпугивать исследователей от публикации уязвимостей, но следует о них помнить и предпринимать меры по снижению их возможных последствий. Теперь, когда мы закончили этот раздел о разъяснении хода публикации уязвимостей, давайте погрузимся в изучение наиболее популярных вариантов публикации.

Выбираем подходящий способ публикации уязвимости

Если вы впервые начали поиск в базах уязвимостей, вы могли оказаться легко поражены количеством доступных вариантов. Иногда базы данных имеют избыток публикуемых элементов или

они могут помогать при этом, коммерческим организациям, а это часто вносит сумятицу в алгоритмы определяющие, чью информацию, в каком случае выдавать. Чтобы помочь упростить все это, давайте познакомимся с ключевыми издателями исследований, относительно традиционных приложений и решений **ПО как услуга** (**SaaS**), а затем поговорим о паре отклонений, о которых вам следует знать, как о том, что может пригодиться для достижения целей вашей публикации.

CVE

Один из наиболее распространенных способов публикации информации об уязвимостях – сделать это через систему CVE. Как ранее обсуждалось в предыдущих главах, CVE это общедоступная база данных, которая используется множеством различных организаций, включая поставщиков ПО, исследователей безопасности и программы баг-баунти.

Для добавления уязвимости в список CVE, программное обеспечение должно быть в определенном объеме, обслуживаемым конечным пользователем самостоятельно – в том плане, что у пользователя есть возможность, отключить небезопасные компоненты. Это обычно исключает продукты SaaS, кроме аппаратной части решения SaaS, владельцем которой является конечный пользователь. Для понимания этого отправного пункта, давайте пройдемся чуть дальше, по нескольким примерам того, удовлетворяет ли приложение критериям CVE или нет:

- **Приложение для Windows, установленное на вашем компьютере**
- **Подходит для CVE:** Приложения установлены локально, на устройстве управляемом пользователем, следовательно они подходят для публикации CVE.
- **Приложение для Windows, установленное на сервере, обращаться к которому вы можете только через удаленное подключение.**
- **Не подходит для CVE:** Распространенное новшество, предлагаемое SaaS, это предоставление ограниченного удаленного доступа к традиционным приложениям Windows через веб-сервисы. Поскольку эти ресурсы полностью контролируются поставщиком ПО, они как правило, не подходят для CVE.
- **Библиотеки с открытым исходным кодом**
- **Подходит для CVE:** ПО с открытым исходным кодом, это один из наиболее распространенных элементов, который стабильно приносит раскрытие информации в CVE, и в общем случае для CVE подходит.
- **Библиотеки с закрытым исходным кодом**
- **Подходит для CVE:** Даже если у некоторых библиотек закрыт исходный код, если вы имеете к ним доступ и можете непосредственно взаимодействовать с ними, подключая их в разрабатываемом вами ПО, или видите их в программном обеспечении, которым владеете – они подходят для CVE.
- **Приложения, которые можно загрузить и установить на компьютер, но данные при этом, хранятся в облаке поставщика ПО**
- **Частично подходит для CVE:** Даже если компоненты ПО, являются SaaS-подобными, большая часть вычислительной мощности, расположена локально, следовательно программное обеспечение и любые связанные с SaaS компоненты взаимодействующие с приложением, подходят для раскрытия информации в CVE.
- **Программное обеспечение, целиком расположенное в облаке, но требующее загрузить и использовать расширения браузера Chrome**
- **Частично подходит для CVE:** В зависимости от области применения расширений браузера, элементы связанные с этими расширениями могут подходить для CVE.
- **Устройство IoT, взаимодействие с которым возможно исключительно в облаке и которое должно быть сконфигурировано на SaaS платформе вендора**
- **Частично подходит для CVE:** В примере, где мы обсудили оборудование MiCODUS, облачное API поставщика ПО, подошло для раскрытия уязвимости, из-за того факта, что оно подвергало

угрозе локальные устройства пользователей. Если отдельные элементы, находились бы за пределом контролируемым пользователем, например блог на сайте MiCODUS у которого был XSS, они не являлись бы подходящими для CVE, потому что не зависят от потребителя.

- **Облачные приложения SaaS**
- **Не подходит для CVE:** В основном облачные приложения SaaS, не являются подходящими для CVE, т. к. они не содержат элементов, контролируемых пользователем.

Еще одной составляющей обсуждения, относительно CVE, является роль CNA (организаций регистрирующих CVE) в публикации. CNA это организации, имеющие налаженные взаимоотношения с MITRE и допущенные к участию в программе присвоения номеров CVE (вы можете почитать о них более подробно на https://www.cve.org/ProgramOrganization/CNAs). CNA, обладают полномочиями присваивать уязвимостям CVE ID и публиковать информацию об уязвимостях. CNA обычно состоят из крупных корпораций, таких как Майкрософт, Али-баба, Гугл или Мета. В ходе запроса CVE, (кое-что мы покажем далее в этой главе, в разделе *Практические примеры публикации уязвимостей*) вам потребуется определить, принадлежит ли программное обеспечение, которое вы тестировали, какой-либо уже зарегистрированной CNA. Если это так, то существуют процедуры, которым вы должны будете следовать, чтобы запросить CVE от такой CNA. Если ПО, которое вы тестировали не принадлежит CNA, то вам потребуется запросить CVE через какую-то CNA-LR. Существует очень мало CNA-LR, на момент выхода в печать этого материала, всего две – одна в составе MITRE, а другая в составе CISA для систем промышленной автоматики. Их взаимосвязь показана на рис. 5.1:

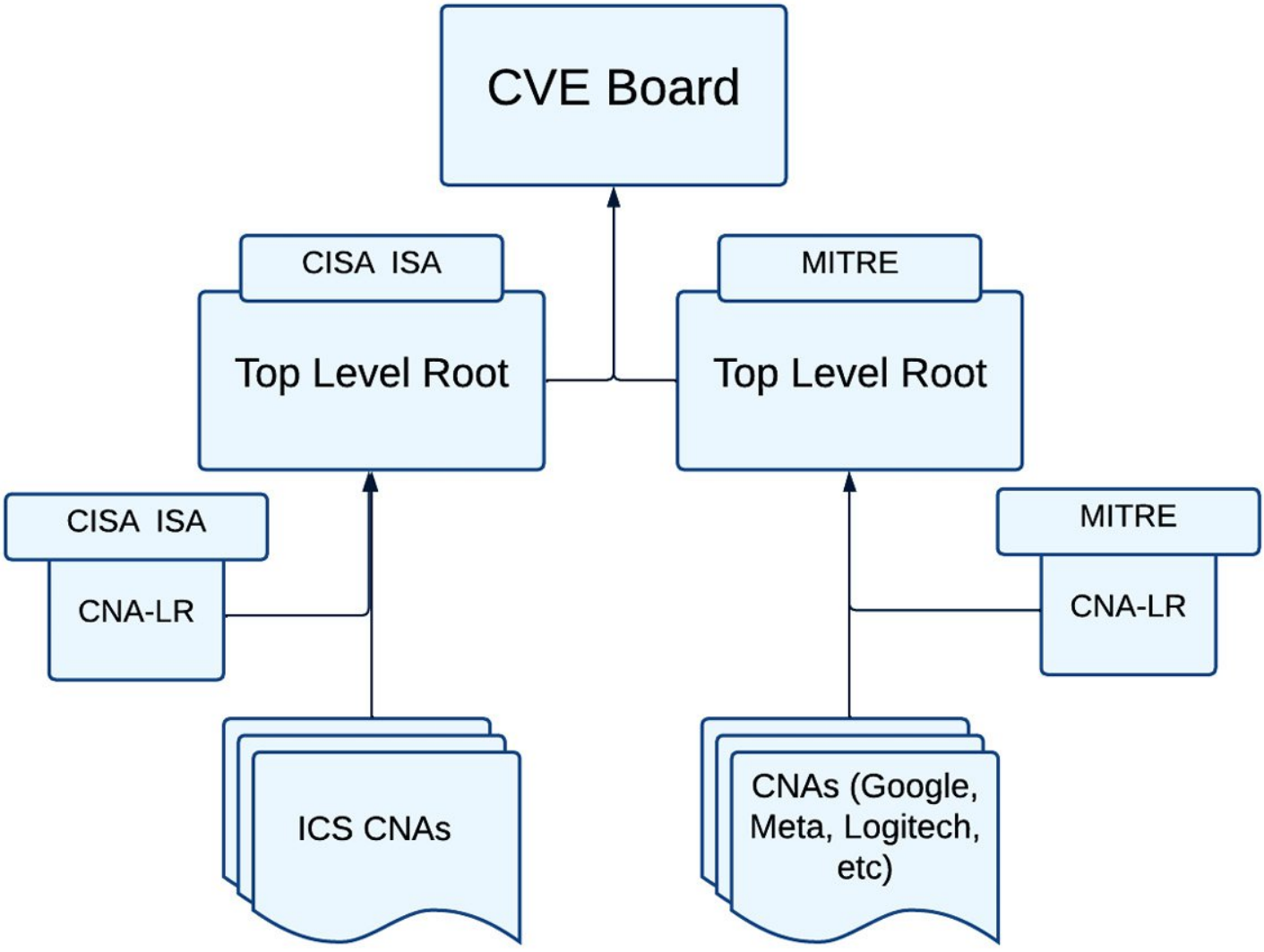


Рис. 5.1 – Пример иерархии информационных потоков CVE

Согласно организационной схемы раскрытия информации в CVE, как только CNA или CNA-LR принимают запрос, информация передается на один из *верхних уровней*, которые применяют собственные методы обработки уязвимости.

Дополнительной порцией информации, жизненно важной для понимания процедуры получения своих CVE опубликованными – является трех-стороннее одобрение информации об уязвимостях перед опубликованием. Давайте обсудим, по отдельности каждую из этих трех составляющих:

Запрос CVE: Первая часть получения CVE, это подача заявки. В зависимости от того, используете ли вы CNA или CNA-LR, эта процедура может выглядеть по разному. Хотя в обоих случаях, должна быть подана заявка с необходимой информацией, которая обычно включает информацию об уязвимости, предоставляемую исследователем на рассмотрение уполномоченному лицу. Для перехода на следующий этап, информация должна быть полной и соответствовать техническим условиям CNA или CNA-LR. Если первоначальная заявка, не соответствует техническим условиям, она будет возвращена для уточнения информации.

Присвоение CVE: Как только заявка на CVE, будет одобрена проверявшим ее уполномоченным лицом, уязвимости присвоят CVE ID. Этот CVE ID будет установлен как **RESERVED** и затем, на заключительном шаге, он будет опубликован. Как правило, присвоение CVE ID, происходит сразу после раскрытия исследователем информации об уязвимости и намерении опубликовать CVE:

CVE-ID

CVE-2022-37001 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

** RESERVED ** This candidate has been reserved by an organization or individual that will use it when announcing a new security problem. When the candidate has been publicized, the details for this candidate will be provided.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

Assigning CNA

N/A

Date Record Created

20220728 Disclaimer: The record creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

Phase (Legacy)

Assigned (20220728)

Votes (Legacy)

Comments (Legacy)

Proposed (Legacy)

N/A

This is a record on the CVE List, which provides common identifiers for publicly known cybersecurity vulnerabilities.

SEARCH CVE USING KEYWORDS: Submit
You can also search by reference using the CVE Reference Maps.

For More Information: CVE Request Web Form (select "Other" from dropdown)

Рис. 5.2 – Пример CVE в статусе RESERVED

Публикация CVE: Если подробности уязвимости сделаются общедоступными производителем ПО, исследователем безопасности или обоими сразу, то в дальнейшем может быть опубликовано CVE. Чтобы сделать это, исследователь или производитель ПО представляет на рассмотрение подробности уязвимости и официальное заявление. Как только детали будут рассмотрены и утверждены, CVE станет активной и статус **RESERVED** будет снят раскрывая оригинальный запрос о CVE.

Теперь, когда вы имеете хорошее представление об основах работы CVE, давайте поговорим о вспомогательной роли CNA в присвоении этих идентификаторов, при публикации уязвимостей в компонентах традиционных приложений.

Вспомогательная роль CNA (организации регистрирующие CVE) в присвоении CVE ID

Публикация это тяжелый труд, и некоторые компании и организации облегчают публикацию деталей уязвимостей с помощью своих сервисов. Несмотря на это, помните, что большинство таких организаций – коммерческие компании, которые используют полученные вами результаты для продажи своих продуктов. Поговорим о нескольких таких компаниях:

VulnDB

VulnDB, является продуктом принадлежащим и управляемым швейцарской компанией рухур .inc. Его цель – находить уязвимости в глобальной базе и затем показывать представляемую ими опасность:

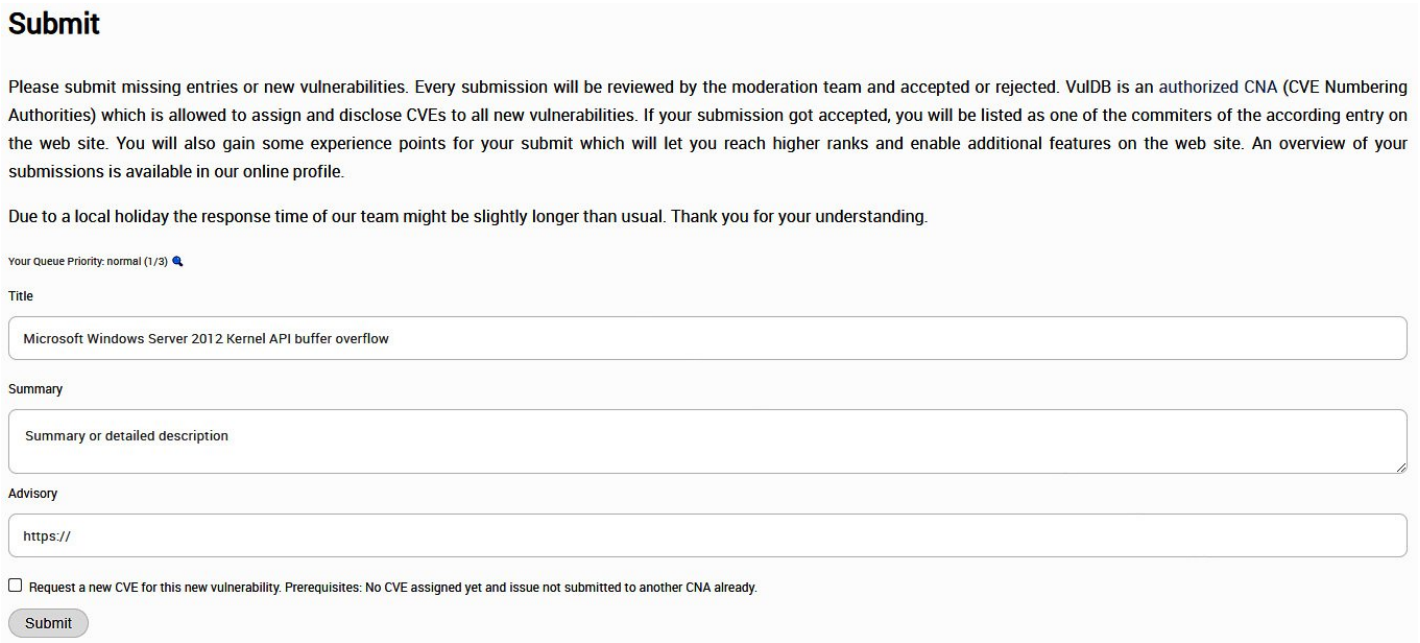


Рис. 5.3 – Форма заявления об уязвимости на сайте VulnDB

Этот продукт, который представлен как бесплатный сервис, с дополнительными платными возможностями. Они работают как CNA, где если вы предлагаете им на рассмотрение уязвимости, для которых они могут открыть CVE с вашей помощью, то оценив ваше исследование, они откроют дополнительные возможности сервиса, в зависимости от того, насколько часто вы предлагаете платформе исследования уязвимостей.

Snyk

Snyk, это коммерческая организация, специализирующаяся на обнаружении, каталогизации и раскрытии информации об уязвимостях в библиотеках **ПО с открытым исходным кодом** (**open source software – OSS**). Большинство технологий, используемых в продуктах SaaS, построено с большим количеством библиотек с открытым исходным кодом. Печально известная утечка данных бюро кредитных историй Equifax, была вызвана непропатченными OSS библиотеками. Поэтому компании инвестируют в технологии, которые будут проверять OSS на наличие уязвимых элементов:

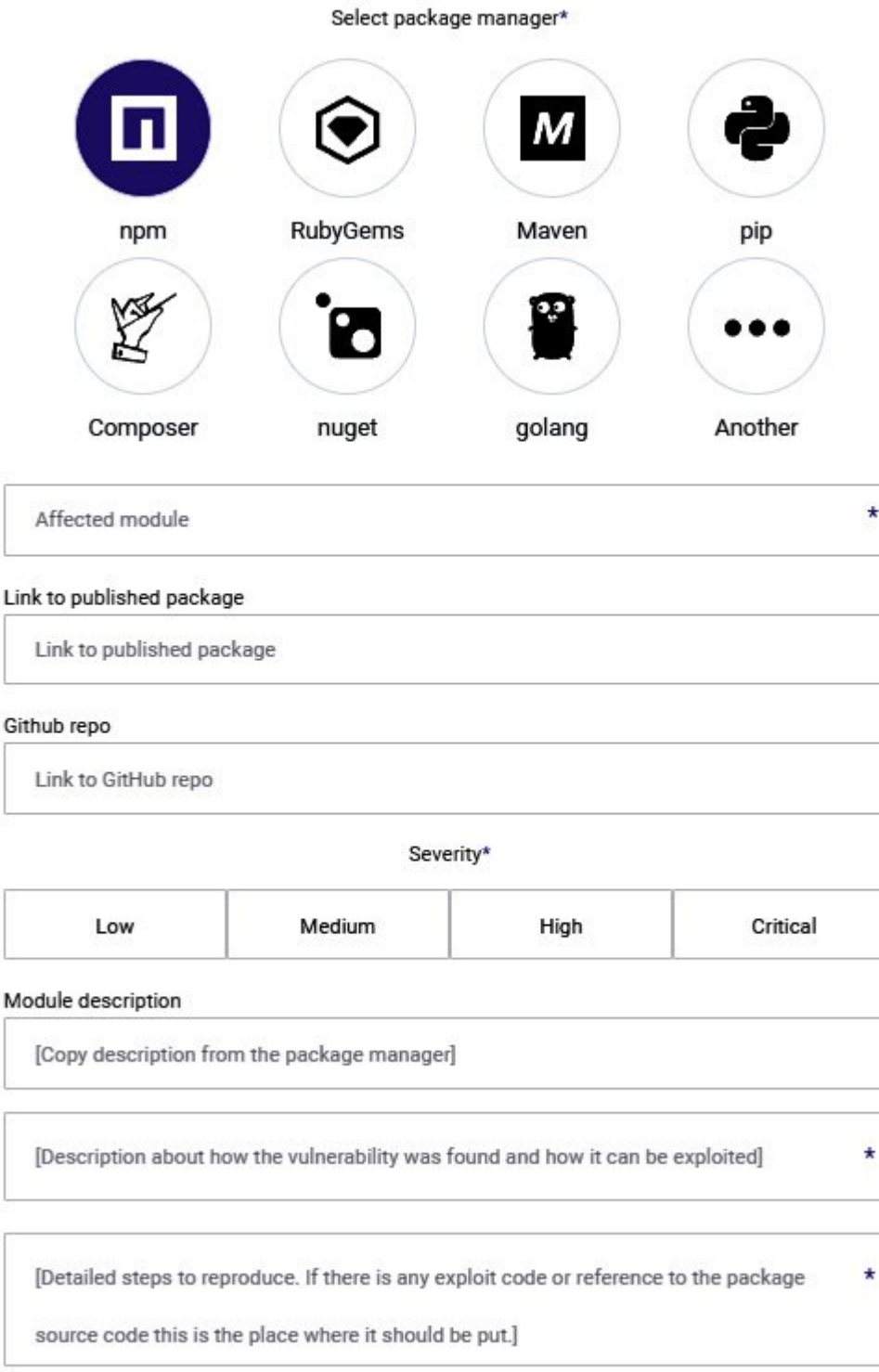


Рис. 5.4 – Форма заявления об уязвимости на сайте Snyk

Став в стороне от конкурентной борьбы, Snyk тесно сотрудничает с исследователями, чтобы дать клиентам информацию об уязвимостях, настолько быстро, насколько это возможно, обеспечивая возможности проверки и поиска уязвимостей, предоставленных исследователями. Если вы исследователь безопасности, интересующийся компонентами ПО с открытым исходным кодом, Snyk является легким способом получить опубликованные подробности вашей уязвимости. Snyk является посредником CNA в регистрации CVE, соответственно они могут подтверждать и публиковать уязвимости в базе CVE, с вашей помощью.

huntr.dev

huntr.dev – это команда хакеров, которые имеют общие ресурсы, для создания программ поощрения исследователей безопасности и майнтейнеров OSS, размещенных на GitHub:

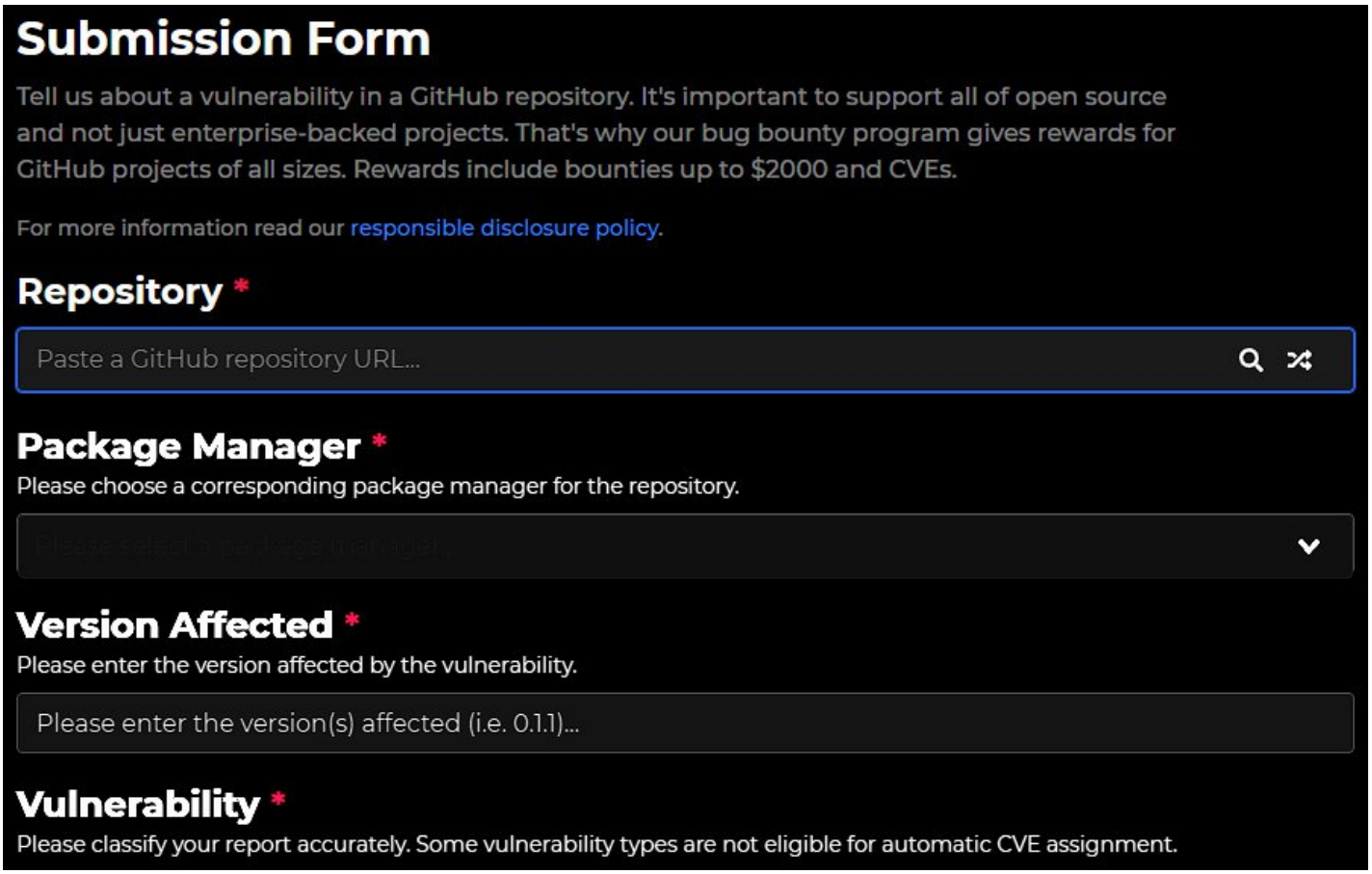


Рис. 5.5 – Форма заявления об уязвимости на сайте huntr.dev

На этой платформе исследователи раскрывают информацию об уязвимостях, платформа начинает процедуру раскрытия информации майнтейнерам и тема отслеживается до тех пор, пока уязвимость не исправят. Это открытая платформа как для исследователей безопасности, так и для майнтейнеров проектов, но только на GItHub. Очень похожая на Snyk, команда huntr.dev, выступает в роли CNA и будет автоматически открывать CVE и публиковать их, представляя в качестве результатов работы платформы.

Zero Day Initiative

Большей частью как Snyk и huntr.dev, **Zero Day Initiative** (**ZDI**) может открывать CVE при вашем участии, но только делают это так, чтобы предоставленные исследователем выводы были проверены, признаны и вознаграждены. ZDI инициирует этот процесс, только после того, как исследователь подтвердит уязвимость, а ZDI предложит ее купить. Критерии спроса и предложения, целиком основаны на том, широко ли распространен исследуемый продукт и насколько вероятна эксплуатация этого изъяна безопасности. Подробности – тема довольно мутная и даже приведенные критерии оценки, не выложены где ни попадя. Вознаграждение предлагается в долларах США и в токенах, названных *ZDI rewards points*.

То, что ZDI называют формой заявления об уязвимости, выглядит как:

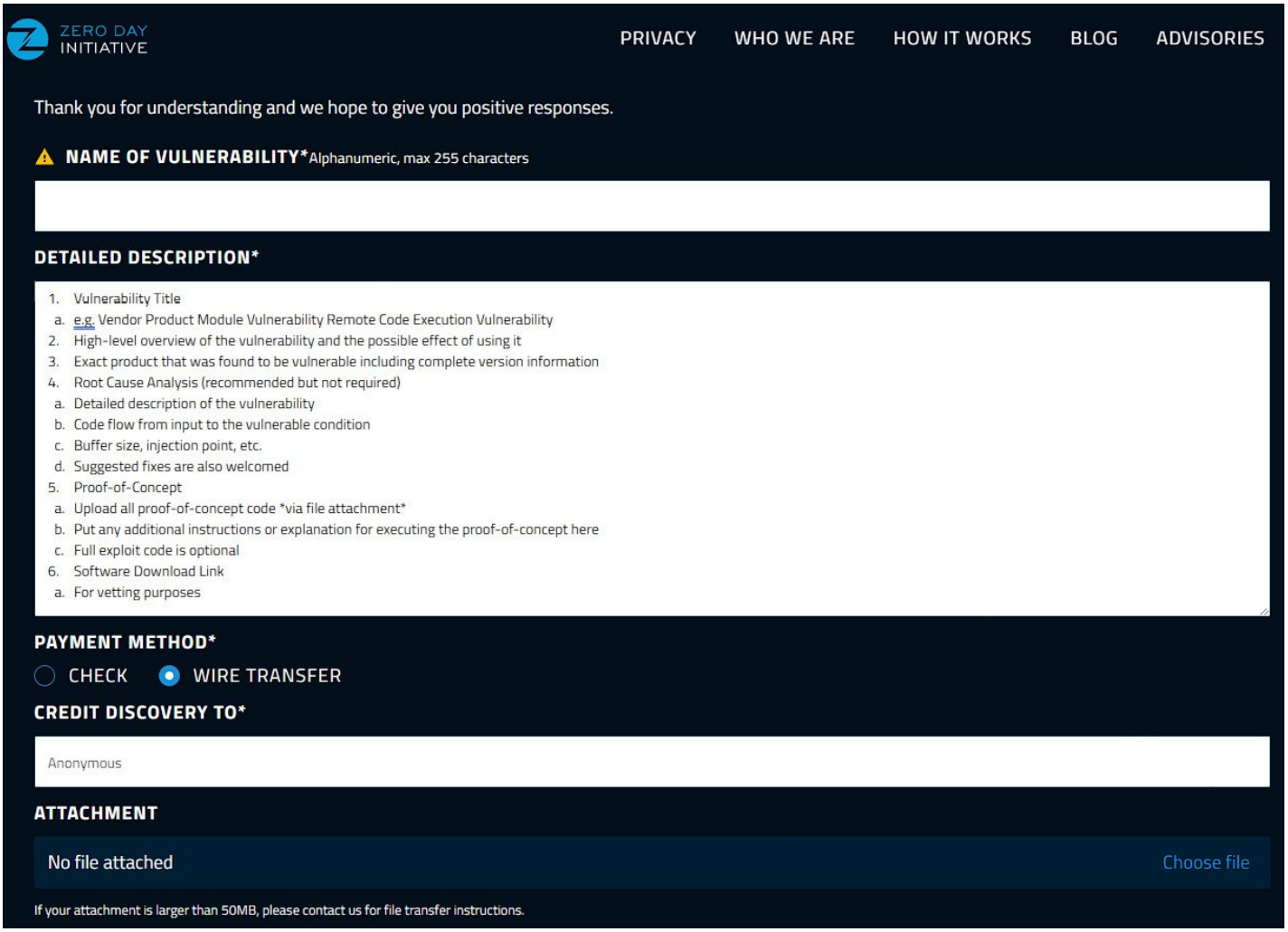


Рис. 5.6 – Форма заявления об уязвимости на сайте ZDI

Потратив время на обзор всех этих вариантов публикации CVE, теперь давайте поговорим о приложениях, не соответствующих в настоящее время, критериям публикации CVE, т. к. это определение вмещает множество прикладных сервисов, находящихся в эксплуатации у пользователей на сегодняшний день.

Способы публикации для приложений не соответствующих критериям CVE

В большинстве случаев, не существует баз данных или общедоступных руководств по раскрытию информации об уязвимостях или слабых местах, найденных в приложениях SaaS, так как они не соответствуют критериям CVE. В этом сценарии, раскрытие информации об уязвимости поставщику программного обеспечения, могло бы быть рекомендованным выбором и местом, на котором работа исследователя могла бы закончиться. Как обсуждалось в *Главе 3, Исследование уязвимостей - начнем с проверенных стратегий*, в большинстве случаев, когда провайдеры SaaS информированы об уязвимостях, обычно не бывает дополнительных публикаций, на сторонних ресурсах, о том, что обнаружили вендор или исследователь.

В дополнение к сказанному выше, некоторые компании публикуют *Release Notes* о том, что касается изменений интерфейса пользователя, в своих программных продуктах, а во многих продвинутых в плане информационной безопасности компаниях, такие заметки (о выпуске), включают слова благодарности исследователям.

В некоторых программах баг-баунти, организаторы могут публиковать результаты исследований и делиться ими с сообществом. Если подробности найденной вами уязвимости, сделаны

общедоступными, то вы можете ссылаться на них или даже написать о них в блоге. Перед тем, как делать любую из этих вещей, хотя бы убедитесь, что вы ссылаетесь на политику раскрытия информации или руководство написанное организатором баг-баунти, чтобы избежать нарушения условий участия в программе.

В целом, все-таки не существует единой базы данных, в которой вы можете публиковать информацию об уязвимостях в приложениях SaaS. В главе 7, мы немного более подробно рассмотрим варианты самостоятельной публикации, если говорить об уязвимостях, однако общедоступная информация о приложениях SaaS, является в высшей степени недостаточной.

Публикация информации об уязвимостях, подходит не каждому – иногда исследователи мотивированы выпускать эксплоиты в продолжение своего исследования. Давайте поговорим о таких мотивах и куда вы могли бы, представить на рассмотрение подробности эксплуатации, если бы вы вдруг захотели углубиться во что-то подобное в будущем.

Базы данных эксплоитов

Одним из любопытных способов, которым уязвимое ПО, оказывается *опубликованным*, однако может быть и *не опубликованным* в базах уязвимостей, является публикация через базы данных эксплоитов. Это такие веб-сайты, где люди могут выкладывать и делиться эксплоитами для различных уязвимостей.

Важно знать, что не для каждой CVE, существует эксплоит и наоборот, не для каждого эксплоита существует CVE. Некоторые базы эксплоитов не всегда различают или вдаются в тонкости, что к чему принадлежит. Там, где правила CVE, могли бы сказать, что SaaS-приложения выходят за рамки, в пределах условной базы эксплоитов - таких правил могло и не быть.

Есть несколько причин, почему некоторые могли захотеть выложить свои эксплоиты в какую-нибудь базу эксплоитов охотнее, чем отправить в базу уязвимостей. Во-первых, они могут быть заинтересованны не в том, чтобы программное обеспечение получило исправления, а как раз наоборот – в возможности применения эксплоита в своих целях или продажи его кому-нибудь. Во-вторых, они могут не считать, что информация об уязвимости, это достаточное условие, чтобы гарантировать ее исправление. И в заключение, они могут не желать преодолевать сложности одобрения заявки, при обращении в “официальные“ базы.

В зависимости от используемой вами базы данных, вы вероятно сможете ускорить процесс исправления уязвимости. Например 0day.today, публикует для вас уязвимости в виде CVE, уведомляет поставщиков ПО и что наиболее важно, позволяет вам сохранить инкогнито. Давайте рассмотрим, несколько таких баз эксплоитов.

Предупреждение

Базы эксплоитов, являются очень интересным местом по нахождению кода и эксплоитов для различного программного обеспечения. В дополнение к сказанному, если вы заинтересовались загрузкой или запуском любого кода, пожалуйста отдавайте себе отчет, что загрузка и выполнение кода без соблюдения мер предосторожности – представляет опасность. Будьте осторожны и попытайтесь разобраться в любом коде, перед его запуском, а если вы его запускаете – пожалуйста, выполняйте это в изолированном окружении. Вы никогда не знаете, что могло бы скрываться во фрагментах шеллкода, опубликованных злонамеренным исследователем безопасности.

0day.today

0day.today одна из наиболее впечатляющих, среди всех баз эксплоитов. Представьте ее, чем-то вроде рынка эксплоитов, где какие-то бесплатные, какие-то платные. Они выражают интересы сообщества исследователей, обеспечивая анонимность, проверенными zero-day эксплоитами и, что намного важнее – площадкой, где открыто делятся уязвимостями, что может стать важным шагом для получения CVE, при некоторых условиях:

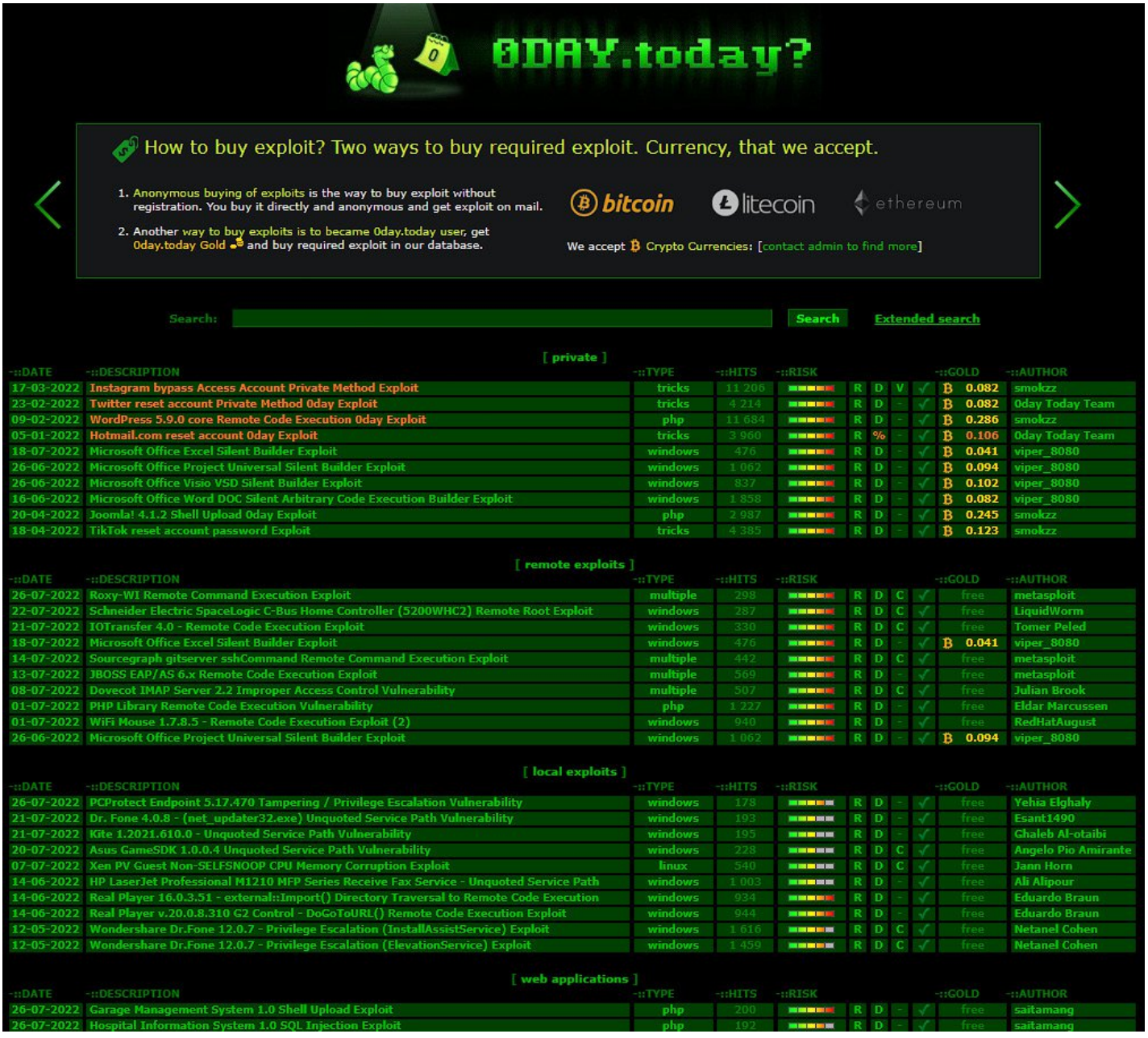


Рис. 5.7 – Домашняя страница базы эксплоитов 0day.today, с перечислением бесплатных и платных эксплоитов

Проверенные исследователи, которые пожертвовали (или даже продали) эксплоиты, получают доступ к еще большему количеству информации об эксплоитах и особый доступ к средствам сообщения с другими исследователями, для обсуждения уязвимостей. 0day.today содержит около 37000 эксплоитов, на момент написания этих строк.

ExploitDB

ExploitDB, является вероятно наиболее известной базой эксплоитов. Она была создана в 1998 году, HD Moore (создатель Metasploit framework) и в настоящее время поддерживается Offensive Security:

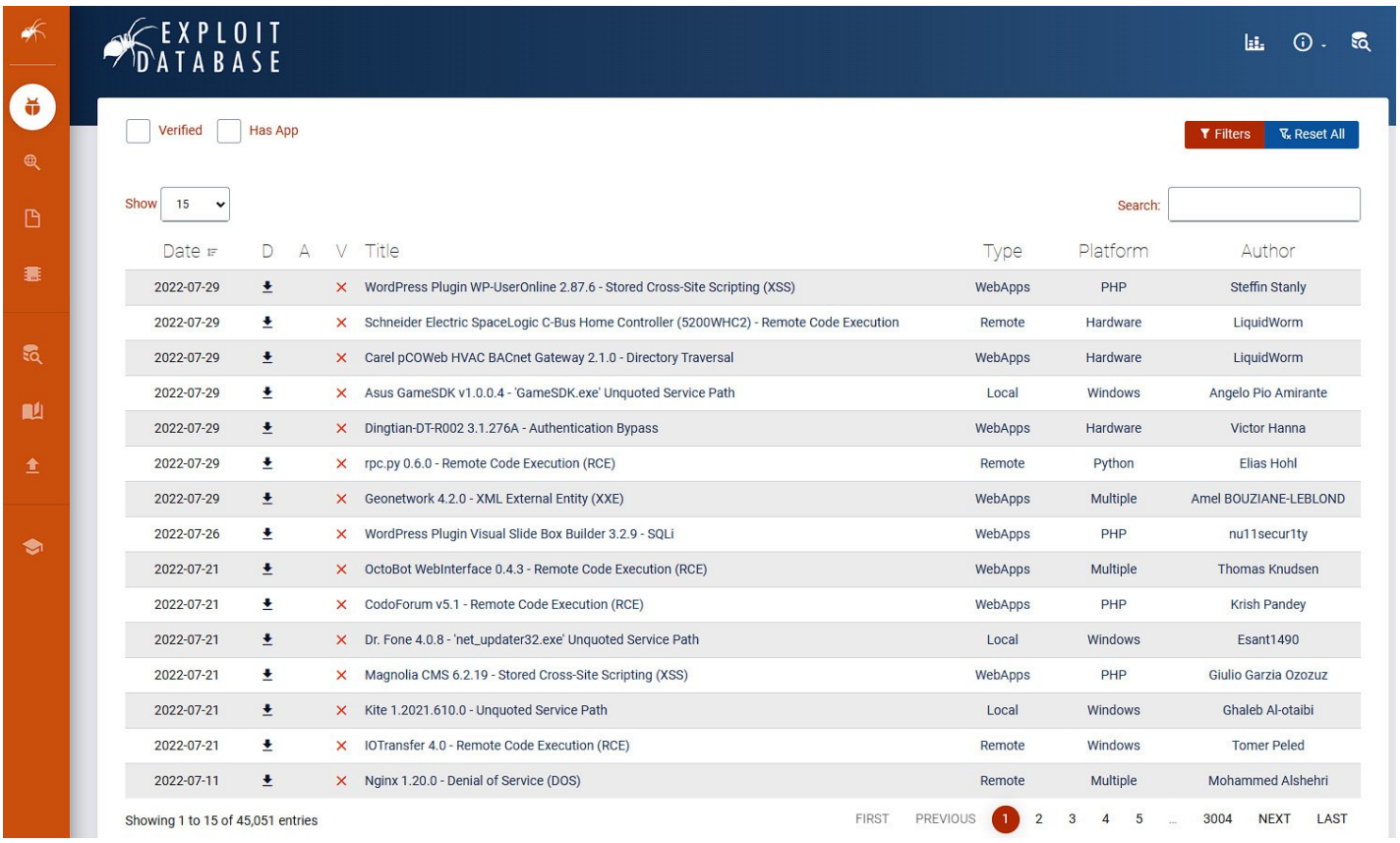


Рис. 5.8 – Домашняя страница ExploitDB со списком доступных эксплоитов

Веб-сайт содержит упорядоченную базу эксплоитов, а также коллекцию полезных инструментов и ресурсов для специалистов информационной безопасности. Когда вы находите эксплоит, который хотите использовать, вы можете или скачать чисто исходный код эксплоита или иногда установить их непосредственно в Metasploit framework.

Packet Storm

Известная площадка, где исследователи безопасности могли бы размещать код, это старейший ресурс Packet Storm. Packet Storm выступает в роли некоего архива исследований безопасности и инструментов. По сути, они собирают и хранят инструментарий, код эксплоитов и объявления о проблемах безопасности. Это прекрасное место, для рассмотрения материалов по исследованию безопасности, в исторической ретроспективе. Исследователи могут публиковать код эксплоитов и ссылки на инструменты для облегчения процесса непредвзятого представления информации. Вкладка Files, сайта Packet Storm, выглядит как:

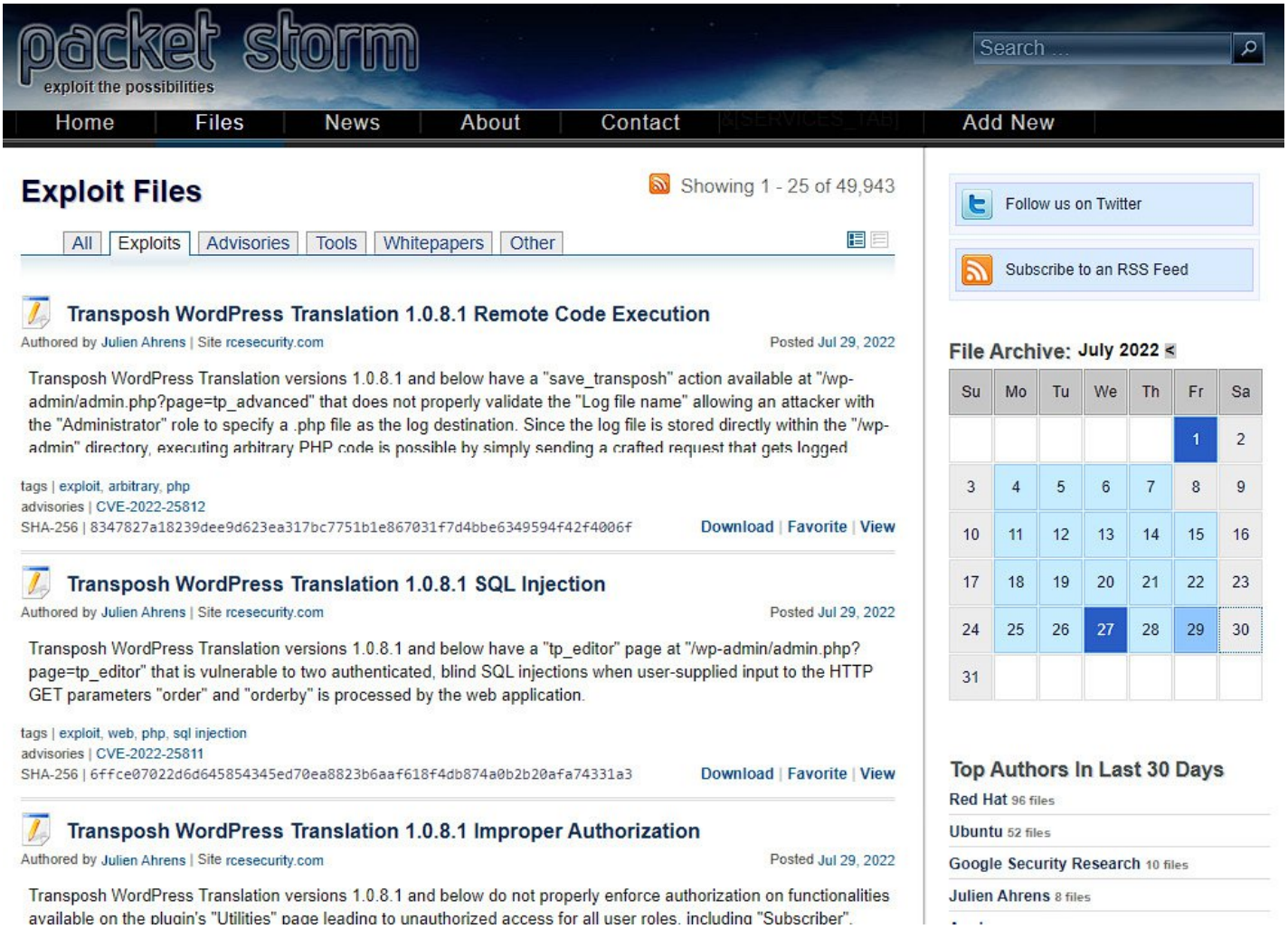


Рис. 5.9 – Раздел Exploit Files сайта Packet Storm, насчитывает тысячи архивных уязвимостей

Подобно большинству CNA провайдеров, которые открывают CVE, каждая база эксплоитов имеет свои собственные правила и рекомендации по подаче заявок. Перед отправкой чего бы то ни было в базу эксплоитов, вам следует ознакомиться с такими правилами, во избежание лишней путаницы или повторной подачи материала на рассмотрение.

Сейчас у вас должно быть хорошее понимание того, какие уязвимости можно публиковать в базах данных, с кем вы можете поработать для передачи результатов своих исследований в базы и, наконец, что такое базы данных эксплоитов и почему они могут быть полезны для ваших целей. С этими знаниями, давайте уделим некоторое время рассмотрению публикации CVE, на нескольких примерах раскрытия информации об уязвимостях.

Практические примеры публикации уязвимостей

В следующем разделе мы шаг за шагом рассмотрим, процедуру публикации двух отличающихся типов раскрытий информации в CVE, одно из которых сосредоточится на раскрытии CVE совместно с CNA, а другое на CNA-LR. Для этого примера, мы предложим произвольный образец уязвимости. Скажем так, мы описываем уязвимость XSS, в программном обеспечении связанном с платформой электронной коммерции для малого бизнеса, мы назовем ее *Ecom-o-matic* исключительно для этих примеров. Итак, начнем с раскрытий CVE продвигаемые CNA.

Примечание

Раскрытие информации, всегда предшествует публикации, однако в большинстве современных случаев, если вы уже начали процедуру раскрытия совместно с компанией, являющейся CNA – они могут раскрыть, найденную вами уязвимость на своем веб-сайте, в соответствии с процедурой раскрытия, определяемой CNA. Применение компанией этого механизма, в сущности объединяет раскрытие и публикацию в один большой шаг. Некоторые исследователи обнаружили, что так происходит не всегда – иногда случается первоначальное раскрытие информации и затем любые запросы по CVE, должны идти через уполномоченного представителя CNA. Крупнейшие корпорации, могут иметь отдельные подразделения реагирования на раскрытия информации и запросы по CVE, итак прикинем варианты и займемся тем, что должно быть в каждом конкретном запросе по CVE.

CVE продвигаемые CNA

Первым шагом, который вам необходимо предпринять, это определиться, необходимо ли вам (или желательно) применять CNA для раскрытия информации об уязвимости. Скажем для нашего примера, мы обратились к команде, управляющей Ecom-o-matic и они поделились с нами, что являются частью родительской компании – Logitech, поэтому нам следует работать с Logitech по любым задачам касающимся результатов исследования безопасности. В первом приближении для нашего раскрытия информации, нам нужно определить, есть ли Logitech в списке компаний, наделенных полномочиями CNA, которые могут резервировать и публиковать CVE. Мы можем сделать это, посетив www.cve.org/PartnerInformation/ListofPartners и поискав там компанию Logitech. В данном случае, нам повезло, потому что Logitech оказалась в списке CNA. Примите за правило, если конкретная CNA, плотно связана с компанией-владельцем программного обеспечения, вы должны использовать эту CNA для раскрытия информации об уязвимости:

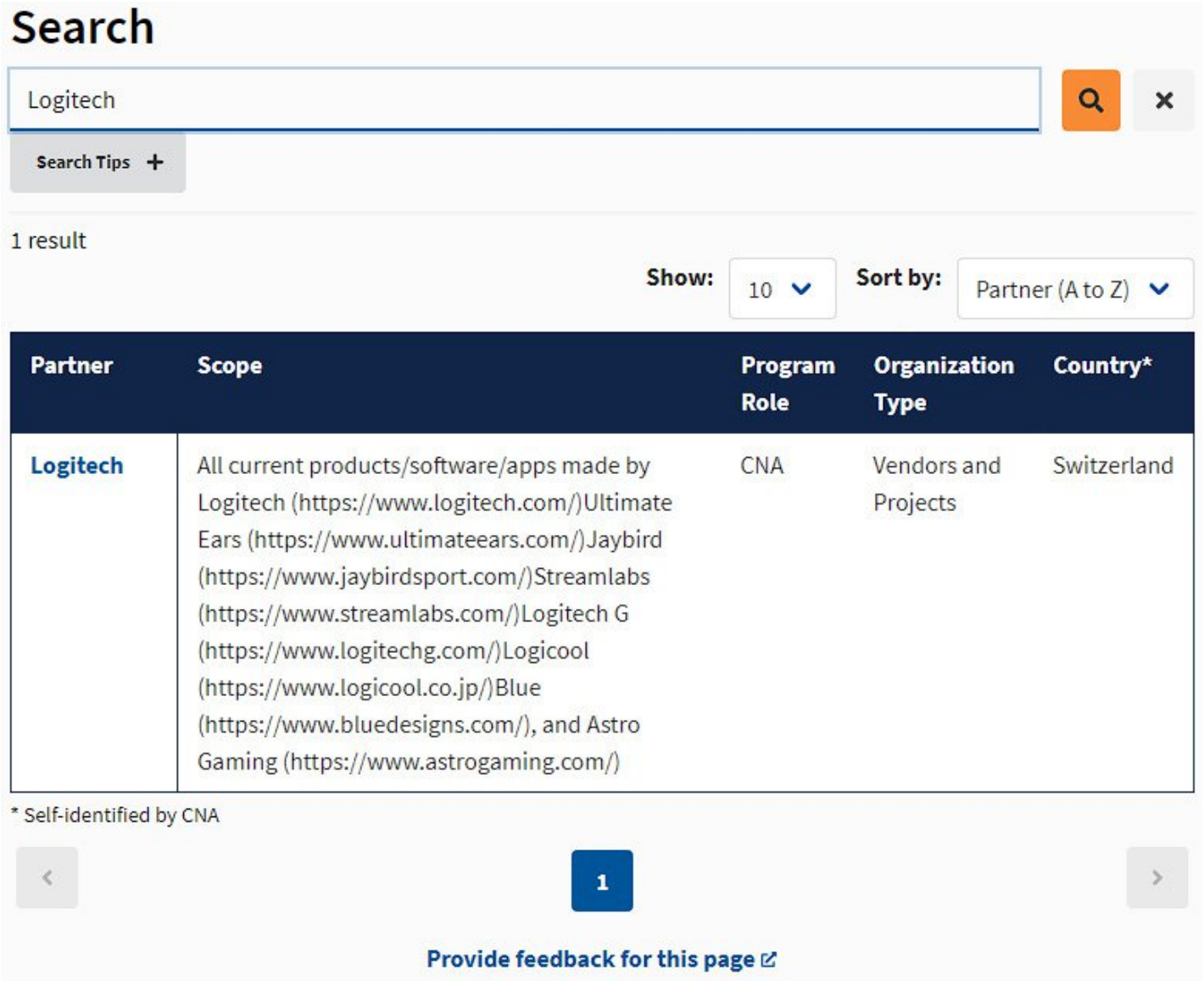


Рис. 5.10 – Поиск участников CNA

Теперь, если вы кликнете по ссылке участника, вы сможете узнать, каким образом компания Logitech, предпочитает получать информацию об уязвимостях от исследователей. Обычно это включает правила, которые будут предписывать, как уязвимость должна быть описана, какие программные продукты входят в область действия и какую-то форму связи. В нашем случае, мы имеем политику раскрытия и контактную информацию, как показано на рис. 5.11:

Logitech

The majority of the links on this page redirect to external websites; these links will open a new window or tab depending on the web browser used.

Steps to Report a Vulnerability or Request a CVE ID

Step 1: Read disclosure policy	Step 2: Contact
View Policy	**Email**

Scope	All current products/software/apps made by Logitech (https://www.logitech.com/)Ultimate Ears (https://www.ultimateears.com/)Jaybird (https://www.jaybirdsport.com/)Streamlabs (https://www.streamlabs.com/)Logitech G (https://www.logitechg.com/)Logicool (https://www.logicool.co.jp/)Blue (https://www.bluedesigns.com/), and Astro Gaming (https://www.astrogaming.com/)
Program Role	CNA
Top-Level Root	**MITRE Corporation**
Security Advisories	**View Advisories**
Organization Type	Vendors and Projects
Country*	Switzerland

* Self-identified by CNA

Рис. 5.11 – Подробности программ CNA, перечисленны для каждого участника

Если вы кликнете по ссылке **View Policy**, в нашем случае нас перенаправит на страницу правил, которые оказывается, размещены на платформе HackerOne. Если вы помните, HackerOne это платформа баг-баунти, которая позволяет исследователям раскрывать результаты исследований безопасности, при посредстве стороны, которая проверяет результаты этих исследований и может финансово вознаградить исследователей за отчеты об уязвимостях, как можно видеть на рис. 5.12:

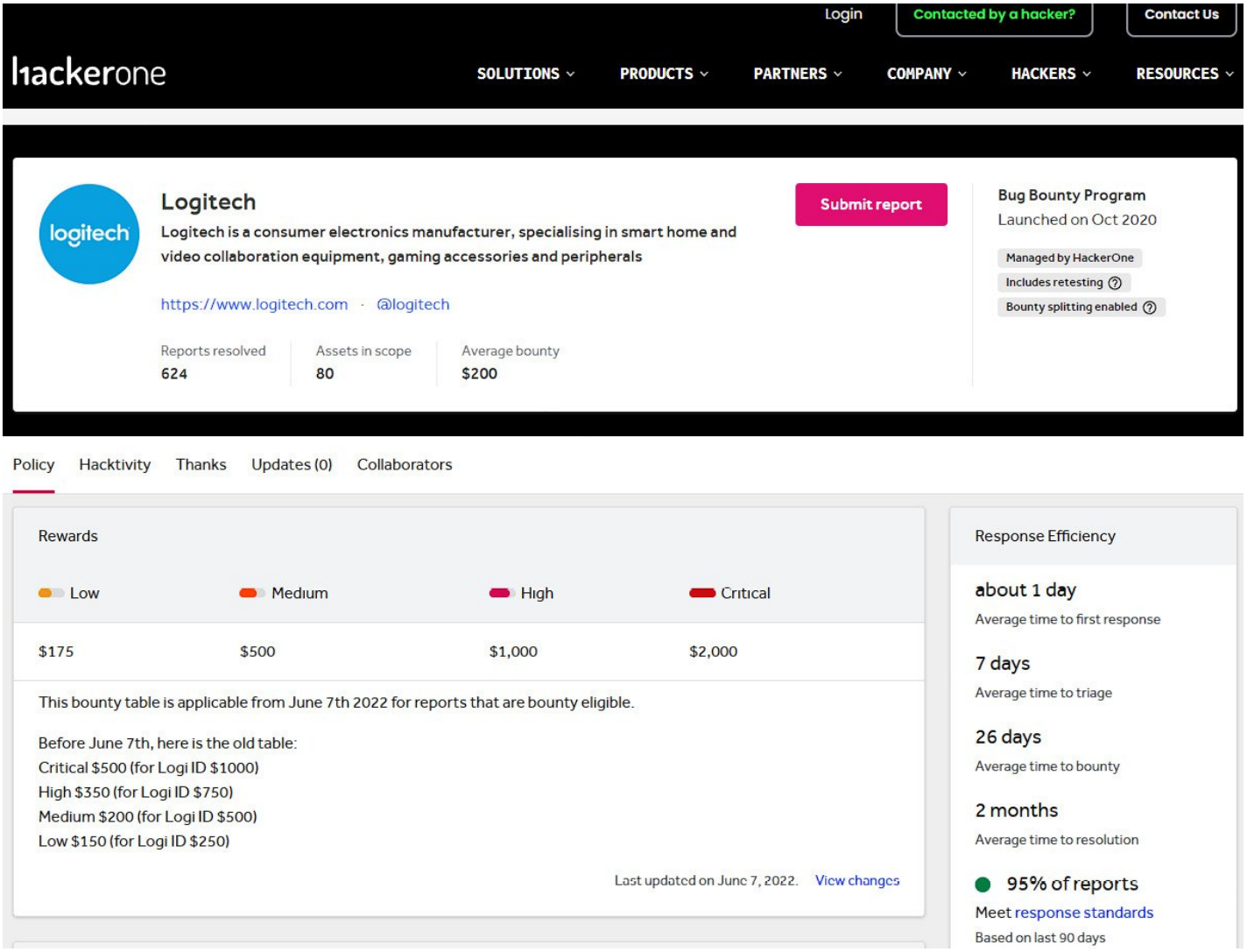


Рис. 5.12 – Ссылки со страницы политики CNA, могут привести вас куда угодно, включая, что неожиданно – подробности программы баг-баунти

Это нас интересует. Здесь похоже подразумевается, что мы можем поделиться результатами наших исследований уязвимостей, через платформу HackerOne. Если наш материал, окажется подходящим, мы сможем потом получить от них вознаграждение. Logitech не упоминал об CVE на этой платформе, однако стоит отметить, что HackerOne является CNA, и также точно HackerOne может открывать CVE на различных платформах, совместно с пользователями. Из этой политики неясно, как они управляют программным обеспечением и как нам действительно открыть CVE. Итак, что нам делать для того, чтобы понять, что делать дальше – отправить письмо по адресу указанному на сайте CNA. В этом письме, мы просто сообщим информацию, о том, что нашли уязвимость и поработали бы с ними над открытием CVE, а также получили рекомендации по дальнейшим действиям. Для этих сценариев, я советую поступить так же, как в нашем первоначальном раскрытии – вежливо, но коротко, по возможности не вдаваясь в подробности в первом письме. Это могло бы выглядеть, как что-то вроде:

Здравствуйте,

Меня зовут {Имя} и я недавно обнаружил уязвимость безопасности в приложении Ecom-o-matic. Я связался с командой EcomSolutions напрямую, чтобы раскрыть информацию об этой уязвимости, а они рассказали мне, что все CVE и раскрытие информации об уязвимостях – контролирует их родительская компания. Я хочу сообщить вам это, чтобы вы могли устранить проблему до того, как кто-либо ее использует со злым умыслом.

Детали уязвимости:

Тип: Кросс-сайт скриптинг

Связанная CWE: CWE-79

Версия ПО: 7.2.1

Функциональность, в которой найдена уязвимость: Во вкладке пользователя, в поле ввода текста, обозначенном “Comments“

Я бы хотел поработать с Logitech по открытию CVE для этой уязвимости, как только она будет подтверждена. Я прочел ваш документ о раскрытии информации, который ваша служба безопасности, разместила на платформе HackerOne. Вы выпускаете CVE, через платформу HackerOne для раскрытых там уязвимостей, или существует другая процедура регулирующая это? Когда вы подскажете мне как действовать дальше, я с радостью дам вам подробные инструкции по воссозданию условий для эксплуатации уязвимости.

С уважением,

{Имя}

Это вежливое письмо, скорее всего вызовет отклик и обычно, есть основания ожидать ответа. Если вы не получаете от них ответа, попробуйте написать снова или попытайтесь представить уязвимость на рассмотрение, через их программу раскрытия на HackerOne. Вам следует предполагать, что каждая программа CNA, будет иметь разные политики и разные требования, относительно раскрытия информации. В одних случаях, они просты и понятны, тогда как в других – намного более запутаны.

А что, если наше вымышленное программное обеспечение, не числится в CNA? Тогда нам нужно начать запрос по открытию CVE, или через CNA-LR или использовать какое-либо решение с участием посредника, как бы обсуждали в предыдущем разделе. Давайте посмотрим, что из себя представляет этот процесс.

CVE продвигаемые CNA-LR

Итак, в этом новом сценарии предположим, что наше программное обеспечение не принадлежит какой-либо родительской компании и в течение процедуры раскрытия информации, мы узнаем как взаимодействовать с компанией, а компанию убедить работать с нами. Если вы соберетесь искать CNA, для этой компании – вы его не найдете. Это подразумевает, что нам придется обрабатывать уязвимости с помощью CNA-LR. Мы сделаем это в два этапа, первый – это исходный запрос получения CVE ID и второй – публикация уведомления. Сейчас, давайте рассмотрим первый этап.

Запрос получения CVE ID

Варианты с CNA-LR, хороши тем, что они гарантируют вам работу напрямую с CISA или MITRE и ваша уязвимость, не будет обрабатываться сторонней компанией, которая могла бы иметь другие мотивы, в том числе отклонить или перепродать ваше исследование. Чтобы начать процедуру запроса на открытие CVE через CNA-LR, посетите следующий URL: https://cveform.mitre.org. Вы должны будете увидеть форму, как на рис. 5.13:

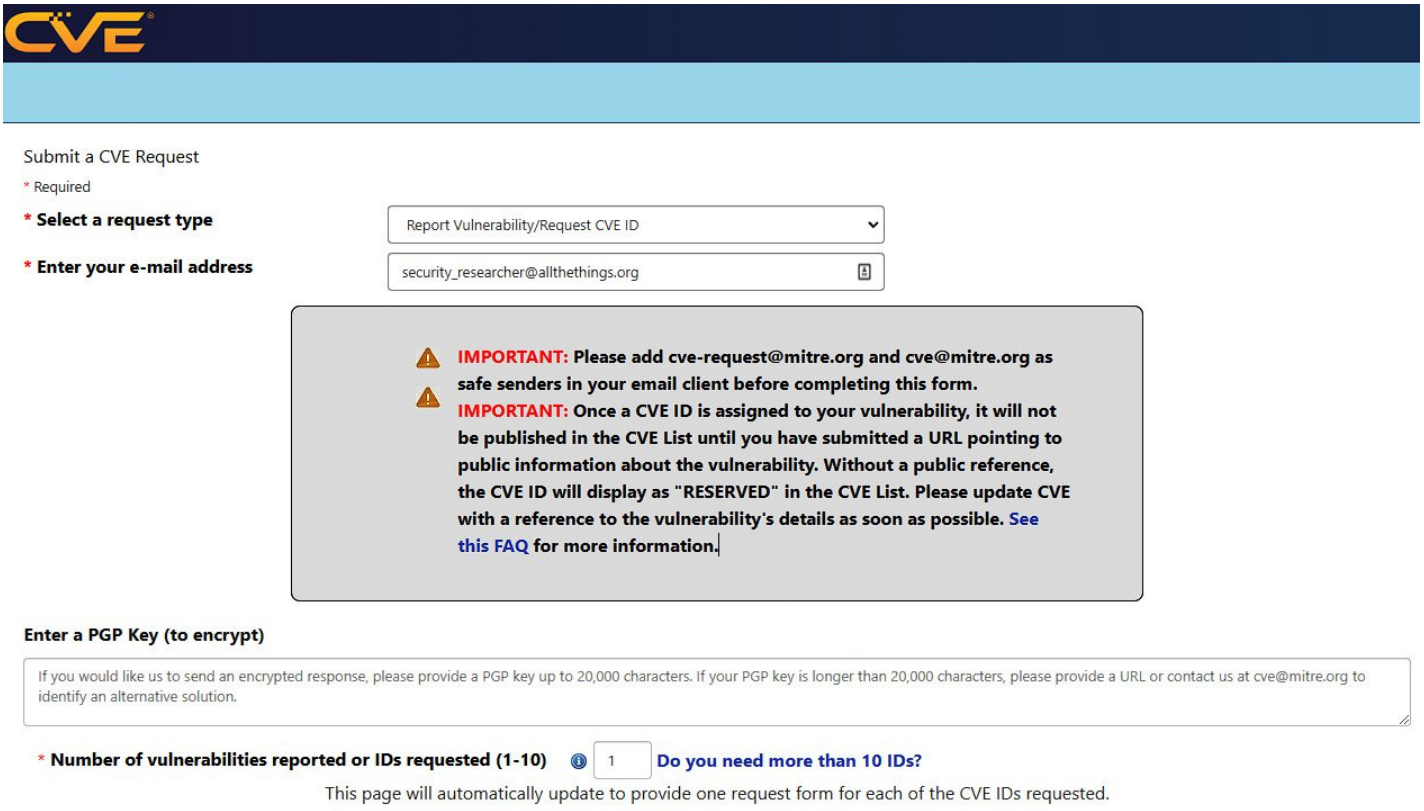


Рис. 5.13 – Начните с выбора типа запроса, ввода email и опционально ключей шифрования PGP

Вам нужно подтвердить ваш запрос, как **Report Vulnerability/Request CVE ID**, а затем подтвердить ваш адрес электронной почты. Дополнительно, вы можете поделиться своим открытым ключом PGP, для шифрования результатов своих исследований. Далее сайт запросит у вас, сколько уязвимостей вы хотите открыть. В нашем случае, это всего одна. На следующем шаге у вас запросят подтверждение, что этот программный продукт не поддерживается CNA напрямую и нет других CVE ID, открытых по этой уязвимости. Это отличная возможность, вернуться назад и убедиться, что для этого ПО нет организации CNA и нет опубликованных CVE по уязвимости, для продукта по которому вы предоставили информацию. Как только вы убедитесь, что оба пункта верны мы сможем заполнить раздел с информацией об уязвимости, изображенный на рис. 5.14:

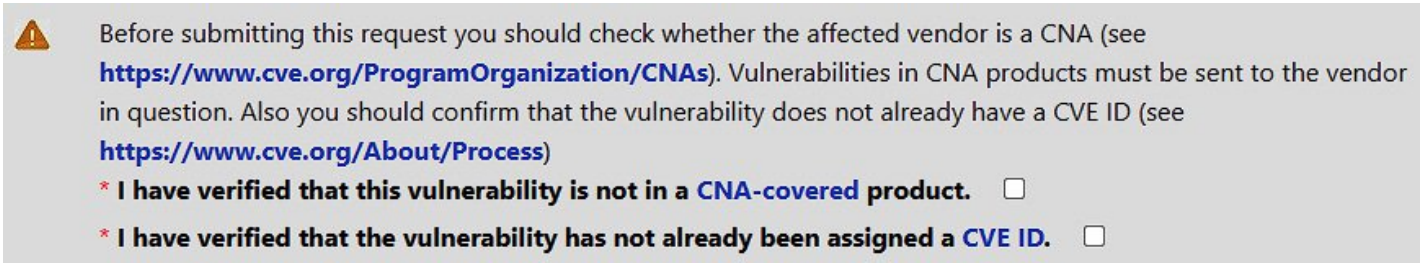


Рис. 5.14 – Подтверждение, что поставщик исследуемого ПО отсутствует в списке CNA и что для нашей уязвимости отсутствует, уже назначенный CVE ID

Теперь, нам необходимо заполнить поля: тип уязвимости, наименование поставщика ПО и номер версии:

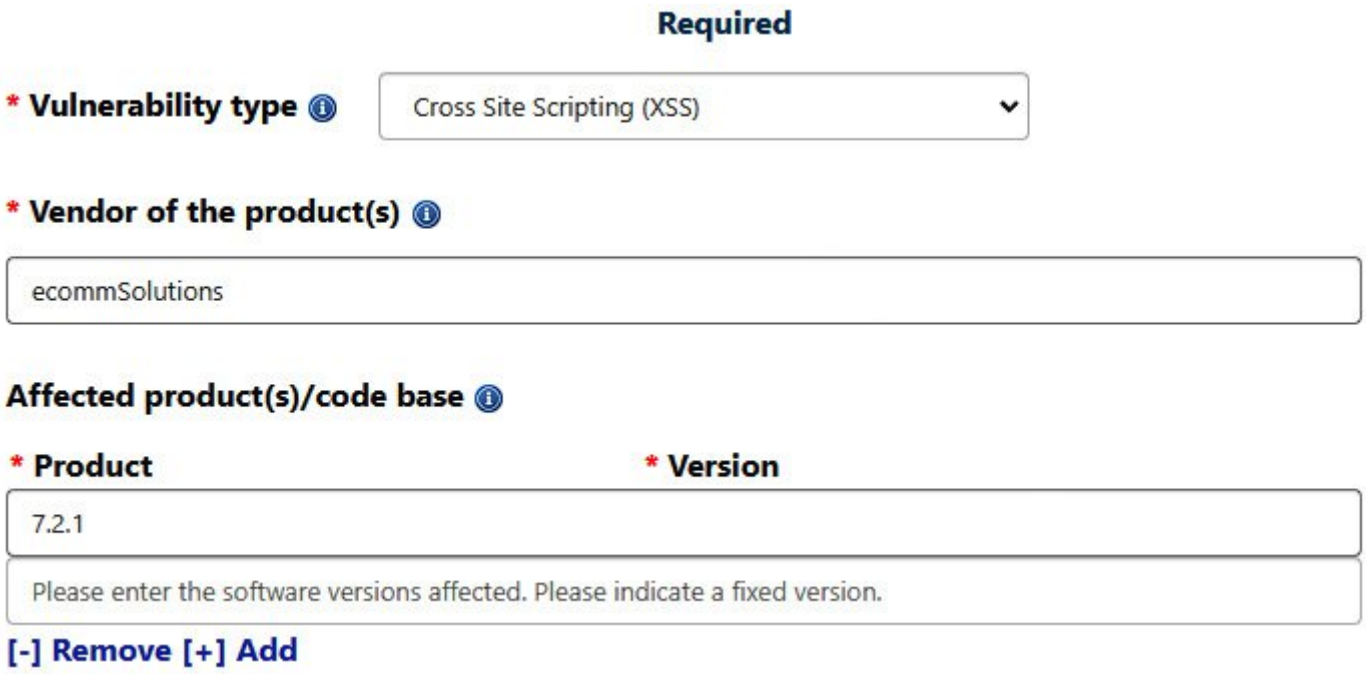


Рис. 5.15 – Ввод типа уязвимости, наименования поставщика ПО и номера версии программного обеспечения

Следующая секция является необязательной, однако мы рекомендуем вам ее заполнить, чтобы CNA-LR имела достаточно информации для назначения номера CVE. Это также поможет улучшить качество описания уязвимости, когда она в итоге будет опубликована. Здесь мы определяем, подтверждена ли уязвимость вендором, в чем заключается атака, что является уязвимым, какие элементы подвержены угрозе и возможные векторы атаки. Давайте сейчас заполним эту форму для лучшего описания уязвимости:

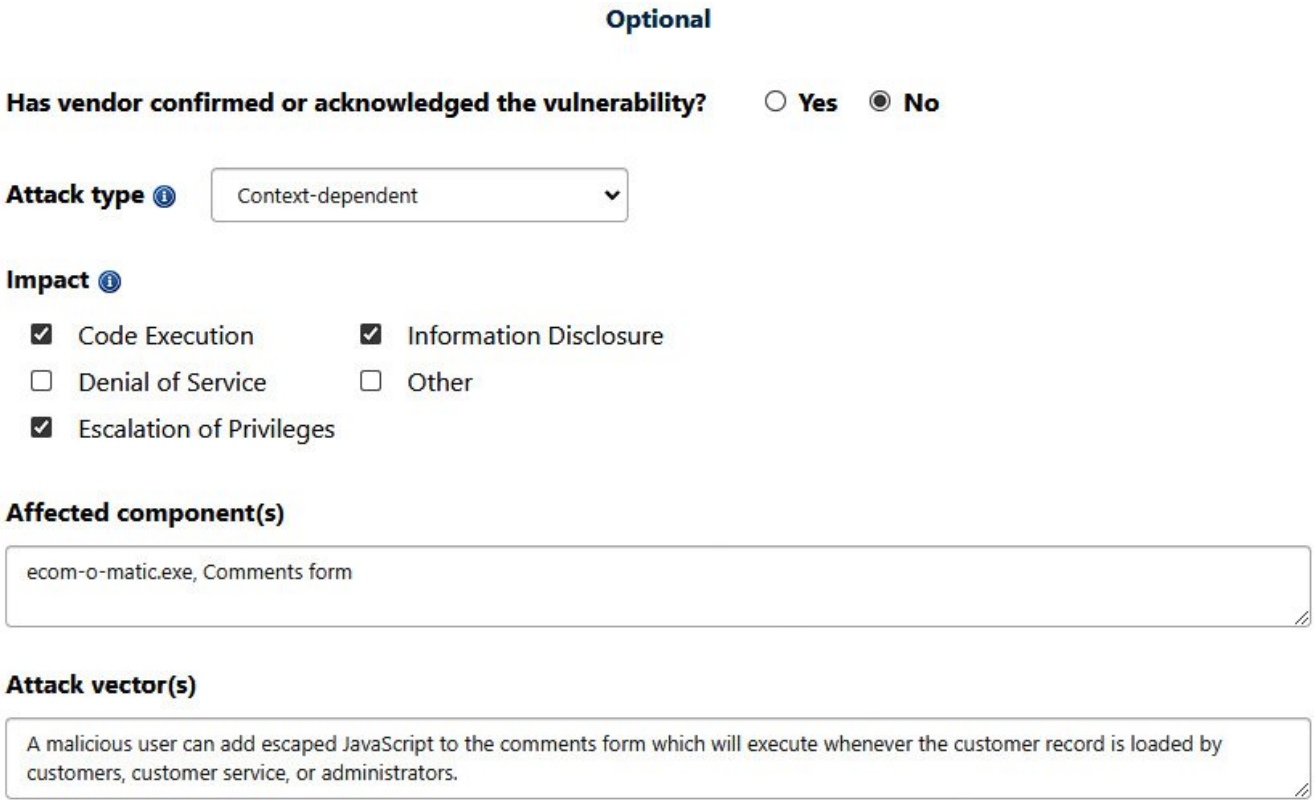


Рис. 5.16 – Определение параметров относящихся к информированности вендора об уязвимости, типе атаки, подверженных уязвимости компонентах и векторах атаки

Далее мы поработаем с рекомендованным описанием уязвимости, для использования в CVE. Это одна из областей, которая часто приводит людей к ошибочным действиям, поэтому давайте потратим некоторое время, чтобы разобраться с форматом. Специфический формат MITRE, представленный здесь – обычно используют чуть ли не все CNA. Если вы уделите некоторое время, рассмотрению прошлых CVE, вы смогли бы уловить смысл этого шаблона. Шаблон всегда включает тип уязвимости, в каких компонентах она была найдена, наименование поставщика ПО, наименование и версию продукта. В дополнение, можно также добавить информацию, которая поясняет каким образом злоумышленник может нанести какой-либо ущерб. MITRE предоставляет отличный шаблон, который содержит эту информацию:

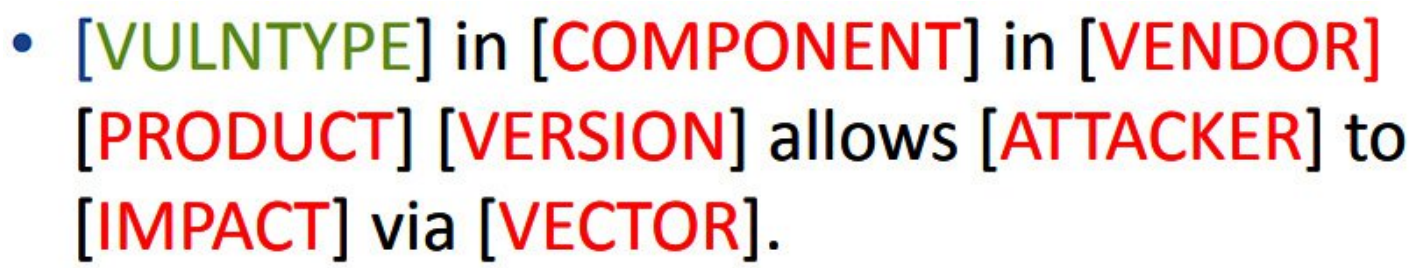


Рис. 5.17 – Рекомендованный MITRE шаблон описания уязвимости

Итак, с теми данными, что у нас есть на текущий момент, описание нашей уязвимости, могло бы выглядеть примерно так:

XSS в форме ввода комментариев пользователя в продукте EcomSolutions Ecom-o-matic v 7.2.1, позволяет легальному удаленному пользователю, запустить произвольный JavaScript, посетив страницу конкретного пользователя.

Тут не требуется совершенства, CNA-LR часто вносят изменения там, где считают это уместным, однако будет хорошей идеей получше подогнать все под их требования, для снижения возможных разногласий. В качестве прекрасного обзора вариантов изложения и прочих примеров, просмотрите этот источник реальных шаблонов и множества полезностей: http://cveproject.github.io/docs/content/key-details-phrasing.pdf

Следующая тема будет связанна с исследователем или признанием результатов исследования. Если вы хотите, чтобы ваш труд по обнаружению уязвимости был оценен, вы обязательно должны сказать об этом. Следующей порцией информации, которую вы должны предоставить, являются справочные материалы относящиеся к программному обеспечению (в виде URL и ссылок) и ресурсы связанные с результатами ваших исследований, если они уже опубликованы. В заключение, вы можете заполнить поле, предоставив любую дополнительную информацию, которую считаете нужной:

Рис. 5.18 – Последние поля описывают, кого можно отблагодарить, любые ссылки и дополнительную информацию

Мы советуем вам выбрать время и вернуться назад к тому, что вы уже добавили в описание уязвимости и дважды все перепроверить, перед отправкой этого в CNA-LR. Когда вы почувствуете, что у вас получилось наилучшее представление вашего исследования, нажмите **Submit Request**.

Резервирование CVE ID, в итоге распределится между исследователями, запросами отклоненными с замечаниями или запросом дополнительной информации. Вся эта информация, рассылается по электронной почте. В большинстве случаев, ответ от CNA-LR может занимать до 8 недель, однако нет точной информации, что ожидание не будет намного дольше. В этой и других главах, мы объясняли, что резервирование номера CVE не подразумевает того, что ваше CVE будет опубликовано, но как только вы получили номер, можно подавать следующий запрос – уже на публикацию. А сейчас, давайте рассмотрим этапы подачи материала на публикацию.

Уведомление MITRE о публикации

Как только вы получите зарезервированный CVE ID на свою почту, можете публиковать найденную вами уязвимость в базе данных и чтобы это сделать, вам понадобится заполнить отдельную форму на cveform.mitre.org.

Замечание о публикации

Публикация CVE требует, чтобы некая информация об уязвимости и подробностях о последствиях ее применения была сделана общедоступной. Публикация подразумевает, размещение информации где-нибудь в интернете там, где CNA-LR сможет на нее сослаться. В нашем случае, это мог бы быть простой пост в блоге, который вы ведете и делитесь там подробностями, публикация эксплоита на PacketStorm или раскрытие информации компанией, в виде заметок о выпущенном исправлении, после устранения уязвимости. В любом случае, нужно что-то опубликовать, перед выполнением следующего шага.

Мы начнем с выбора пункта **Notify CVE about a publication**, заполнения адреса вашей электронной почты и опционально указания открытого ключа PGP, для шифрования сообщений:

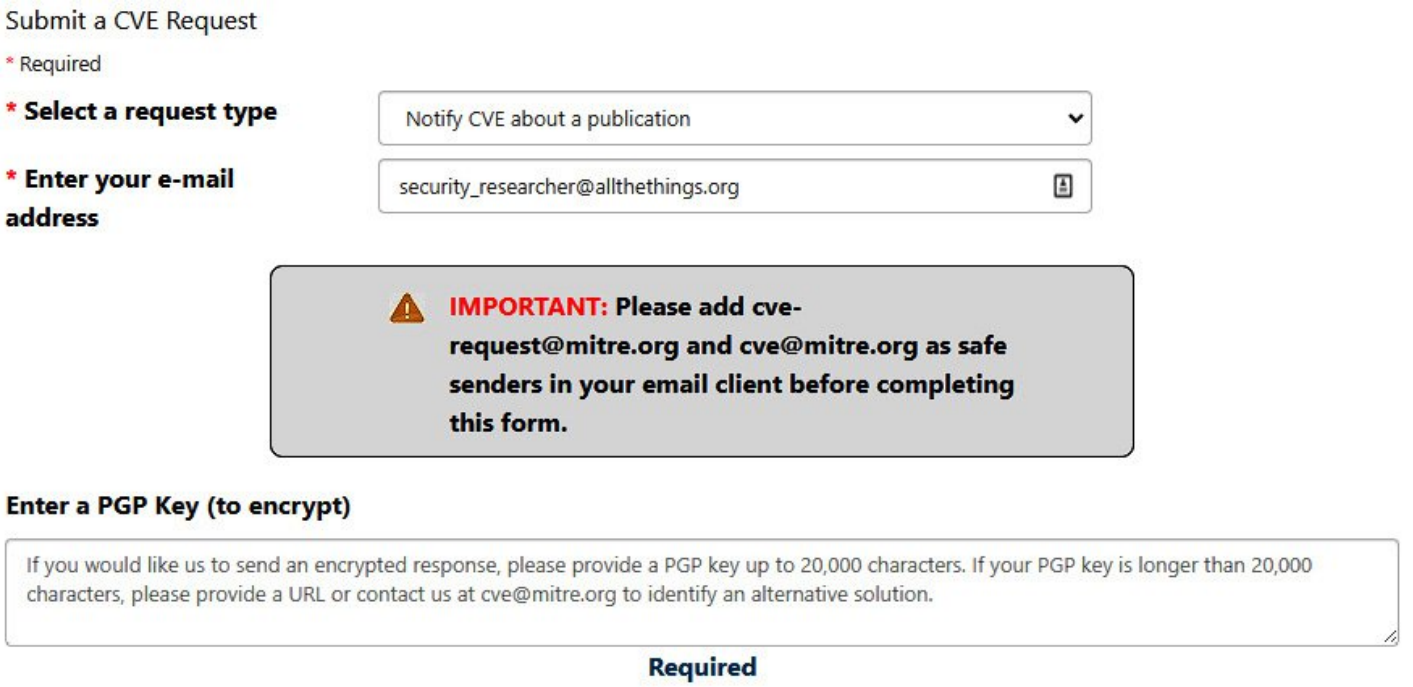


Рис. 5.19 – Для публикации потребуется заполнить похожую форму, в которой нужно указать свои контакты и тип запроса

После завершения, понадобится заполнить еще несколько полей. Это поле **Link to the advisory**, в котором можно указать блог в котором вы писали об уязвимости, раскрытие информации поставщиком ПО или публикацию об уязвимости третьей стороной. Вам потребуется указать CVE ID, который вам был отправлен, в поле **CVE IDs of vulnerabilities to be published** и любую дополнительную информацию о раскрытии уязвимости, а также дату публикации:

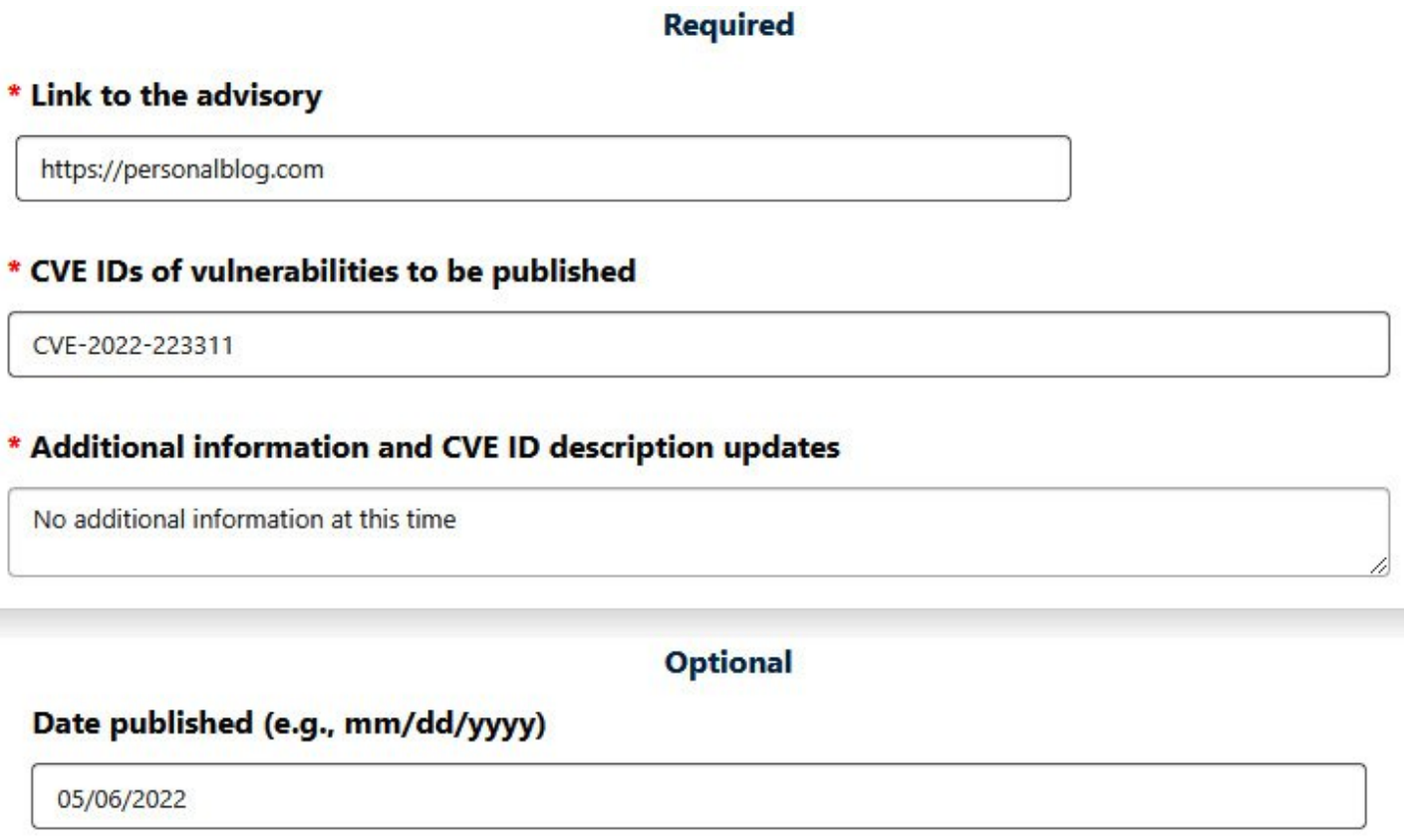


Рис. 5.20 – Этап публикации, требует меньше информации, но вы должны иметь описание (или опубликованную информацию), CVE ID и любые обновления, касающиеся ситуации с уязвимостью.

Когда закончите, подтвердите введенную информацию и ждите пока MITRE уведомит вас о том, что найденную вами уязвимость опубликовали. Это может занять несколько дней и обычно происходит заметно быстрее, чем начальный этап резервирования CVE ID. Редко, но иногда MITRE запрашивают более подробную информацию если источники или информация в описании, не удовлетворяет их критериям.

В предыдущем разделе, мы немного поговорили о роли CNA в качестве посредников, которые являются организациями или людьми курирующими мероприятия по открытию CVE. Давайте кратко рассмотрим это поближе.

CVE опосредованно продвигаемые CNA

Работая с CNA-LR, мы по видимому как будто избегаем легких путей подачи информации через такие организации, как Snyk, VulnDB или других вариантов предлагающих денежное вознаграждение, вроде ZDI. Каждая из этих программ, имеет свои специфичные требования, и вам следует уточнить, подходят ли они к вашему виду раскрытия информации. В нашем гипотетическом примере, давайте прикинем, какие свойства могло бы иметь наше программное обеспечение:

- Является ли оно OSS: Нет
- Является ли оно библиотекой: Нет
- Поддерживается ли оно производителем: Да
- Размещен ли код в репозитории GitHub: Нет
- Приблизительное число пользователей: 2000

Итак, в нашем случае – это не оперсорс, не библиотека или не проект на GitHub, что исключает такие варианты, как huntr.dev и Snyk, т. к. они ориентированы на OSS. Так как у нас небольшое число пользователей, около 2000, ZDI может быть не заинтересованно в нашем исследовании. Нашим единственным вариантом, представляется VulnDB. Далее, нам нужно обратиться к VulnDB и к их конкретным руководствам по подаче на рассмотрение информации об уязвимостях. В этом случае, VulnDB имеет упрощенную форму запроса наименования, резюме и ссылки на описание уязвимости:

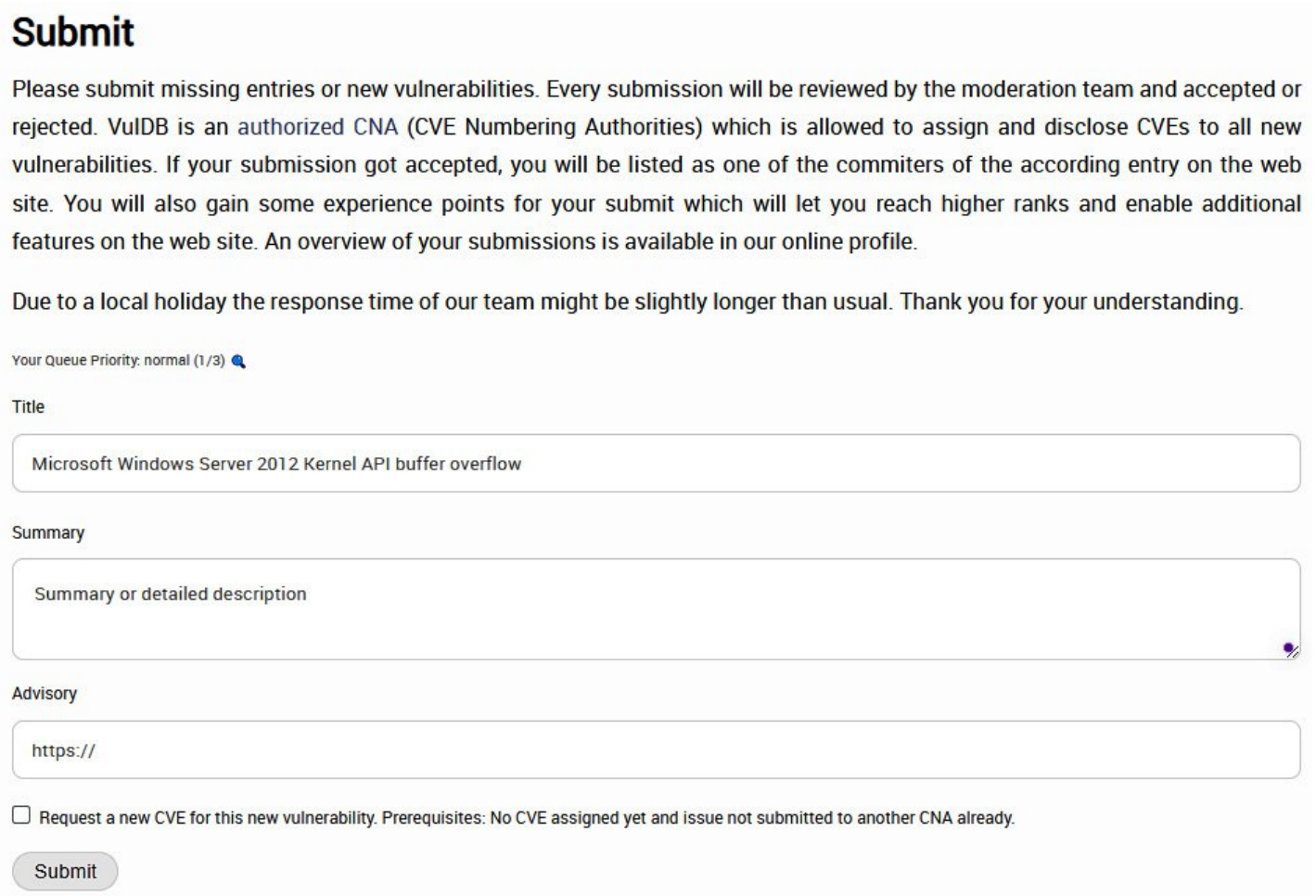


Рис. 5.21 – Форма VulnDB, для информации об уязвимости намного проще и требует меньше деталей, чем используемая MITRE CNA-LR

Предоставление информации об уязвимостях этих продуктов и программ, в конечном счете помогает их улучшать, и в свою очередь продавать пользователям. Если вы в курсе этого и не возражаете против их поддержки, то эти программы могут подойти для ваших нужд по отправке отчетов об уязвимостях.

К заключительной части этой главы, вы должны иметь прекрасные знания основ того, как подать CVE на рассмотрение в CNA, в организацию-посредника CNA а может и в CNA-LR. Каждая из этих организаций имеет собственный набор условий, которые необходимо соблюсти и какое направление вы бы не выбрали, они смогут влиять на вашу работу и координировать ваши действия.

Итоги главы

В этой главе, мы уделили время повышению нашего понимания процесса публикации уязвимостей в базах данных. Мы рассмотрели плюсы публикации уязвимостей, сопоставив их в исторической ретроспективе, когда они оказывали огромное воздействие на область исследований безопасности. Далее мы обсудили, выбор подходящей базы уязвимостей, поговорив об CVE и близких к CVE организациям, называемых CNA. Мы обсудили программное обеспечение, которое не удовлетворяет критериям публикации уязвимостей, базы эксплоитов и их взаимосвязь с CVE. В заключение, мы прошли несколько практических примеров публикации уязвимостей, тремя разными способами. И сейчас вы должны иметь хорошее понимание процесса публикации уязвимостей. А что, если в процессе публикации, все-таки что-то пойдет не так? Итак, в следующей главе мы приступим к обсуждению того, о чем вкратце упоминали в этой главе, под названием “арбитраж уязвимостей“, где

мы поговорим о том, как все может пойти наперекосяк и кто может помочь вернуть ваше раскрытие информации и публикацию в нужное русло.

Дополнительные материалы

- Доклад о BlueBorne – https://info.armis.com/rs/645PDC047/images/BlueBorne%20Technical%20White%20Paper_20171130.pdf
- Сводка ISC по MiCODUS MV720 GPS Tracker – https://www.cisa.gov/uscert/ics/advisories/icsa-22-200-01
- VulnDB – https://vuldb.com/
- Snyk – https://snyk.io/
- Huntr.dev – https://huntr.dev/
- ZDI – https://www.zerodayinitiative.com/
- 0day.today – https://0day.today/
- ExploitDB – https://www.exploit-db.com/
- PacketStorm – https://packetstormsecurity.com/
- База организаций CNA – https://www.cve.org/ProgramOrganization/CNAs
- Форма запроса CVE – https://cveform.mitre.org/
- Ключевые моменты описания CVE – http://cveproject.github.io/docs/content/key-details-phrasing.pdf
- Получение CVE ID – https://www.youtube.com/watch?v=JQYq-mxLo-U

Глава 6. Арбитраж уязвимости – что пошло не так и кто может помочь

Летом 2021 года, немецкая исследовательница Лилит Виттманн, оказалась фигурантом уголовного дела, после раскрытия результатов исследований безопасности. Правящей политической партией Германии, в то время Христианско-Демократическим Союзом (**Christian Democratic Union - CDU**), использовались мобильные приложения для сбора информации об общественном мнении, перед предстоящими выборами. Вместе со своей организацией Компьютерный Клуб Хаос (**Chaos Computer Club - CCC**), Виттманн ответственно раскрыла информацию CDU, Федеральному Управлению Информационной Безопасности и Берлинскому Комиссару по Защите Данных. Виттманн заявляла, что уязвимость настолько проста, что ее использование, даже трудно называть взломом. Уязвимость раскрывала данные полумиллиона граждан Германии и тысяч сотрудников избирательных комиссий. После раскрытия информации об уязвимости, CDU инициировал уголовное преследование Виттманн и начал писать в СМИ, что они стали жертвами кражи данных и сотрудничали с полицией, чтобы привлечь к ответственности за это нарушение конфиденциальности. Виттманн обнародовала свою неудачу в Твиттере, где другие исследователи сразу проявили к ней участие и оказали поддержку. К счастью Виттманн, до этого никогда не была замечена в противозаконной деятельности и как только СМИ начали раскручивать ее историю, CDU снял обвинения. Ее организация, выпустила уведомление, устанавливающее, что CCC не будет в дальнейшем раскрывать уязвимости относящиеся к CDU, получив от них такую враждебную реакцию.

В этом сценарии взаимодействие между исследователем и условной организацией быстро переросло в обвинения в преступлении, вместе с заявлениями политической партии о саботаже предстоящих выборов. Хотя CDU в конце концов, принесла извинения исследователю, ущерб был нанесен – исследователи безопасности сообщили, что в будущем не станут раскрывать информацию об уязвимостях для этой организации. Как показало это раскрытие информации об уязвимости, исследователи, работающие над раскрытием информации, могут быстро вызвать хаотичные враждебные реакции, даже действуя с благими намерениями.

В двух предыдущих главах, мы выучили некоторые основы по раскрытию информации об уязвимостях и ее публикации. В этой главе, мы подробнее поговорим об посредниках, которые могут помочь избежать неприятных ситуаций, вроде таких, с которой столкнулась Виттманн, во время раскрытия информации об уязвимости. Такие посредники могут снизить вероятность конфликтов, сохранить ваше инкогнито и в конце концов помочь, вполне законным образом.

Из этой главы, мы узнаем следующее:

- Арбитраж в раскрытии информации об уязвимостях
- Виды разногласий, которые могли бы возникнуть в процессе раскрытия информации и в каких из них разумно использовать арбитраж
- Разберем средства и инструменты, рекомендованные арбитражными организациями и научимся выбирать подходящие

Эти результаты будут представлены на протяжении следующих основных тем этой главы:

- Основы арбитража уязвимостей
- Решение споров путем арбитража уязвимостей
- Организации проводящие арбитраж

Основы арбитража уязвимостей

Во введении, мы узнали о раскрытии информации об уязвимости, которую Лилит Виттманн довела до сведения политической партии. К сожалению, несмотря на то, что Виттманн выбрала корректный способ взаимодействия с организацией в отношении уязвимости – оказалось невозможным

предугадать то, как организация это восприняла. Ошибки взаимодействия или непредвиденная реакция в ходе онлайн коммуникации, это один из наиболее частых рисков сопутствующих начальному этапу раскрытия любых уязвимостей. Мысль, бывает выражена словами, языком тела, мимикой и даже тоном голоса. Так как, мы не видим выражение лица собеседника или язык тела, его отношение или намерения, можно легко истолковать ошибочно. Дополнительно, напряженность может усугубляться языковыми и культурными различиями между сторонами. Все эти факторы, вносят вклад в разрушение взаимодействия и усложняют его из-за распространения напряженных моментов.

Поэтому, если вы оказались работающим с вендором, который ранее вел себя враждебно, скандалил или игнорировал вас – арбитр, который поможет вам в раскрытии информации об уязвимости, является отличным выходом из ситуации. Прежде всего, давайте детально рассмотрим, что такое арбитраж уязвимости.

Что такое арбитраж уязвимости

Арбитраж уязвимости, расширяет идею согласованного раскрытия информации об уязвимости, сосредоточиваясь на некоей беспристрастной третьей стороне, представляющей общие интересы. Как мы рассмотрели на примере жизненного цикла уязвимости в *Главе 1, Знакомство с понятием уязвимости*, когда исследователь находит новую уязвимость, он имеет несколько вариантов, как поступить дальше. Часто исследователи, будут работать с поставщиком ПО, чтобы исправить уязвимость и предотвратить эксплуатацию. С этим процессом мы познакомились в *Главе 4, Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности*, названном **ответственным раскрытием информации** и это в целом считается, установившейся практикой при раскрытии информации об уязвимостях. В большинстве случаев, исследователь связывается непосредственно с вендором и работает с ним напрямую, чтобы решить проблему, как показано на рис. 6.1:

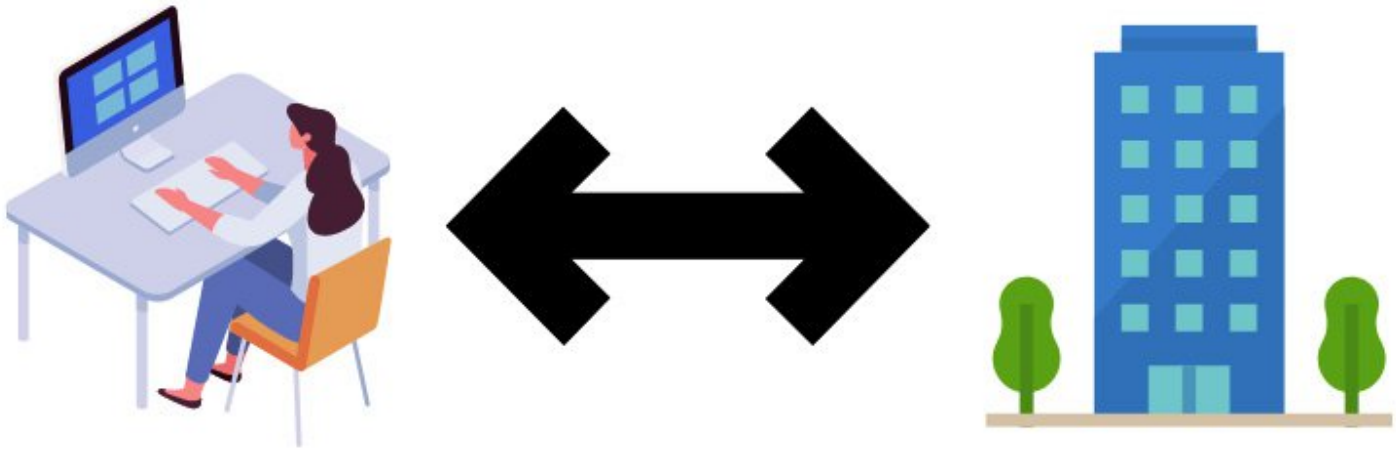


Рис. 6.1 – Непосредственное раскрытие информации, обычно происходит между исследователем и вендором

Арбитраж уязвимости, это процедура решения споров, которые могут возникать в ходе раскрытии информации, с помощью третьей стороны, в течение процесса раскрытия информации об уязвимости. Эта третья сторона, обычно выступает в роли передаточного звена, с одной стороны сохраняя конфиденциальность, с другой обеспечивая диалог между вендором и исследователем, как изображено на рис. 6.2:

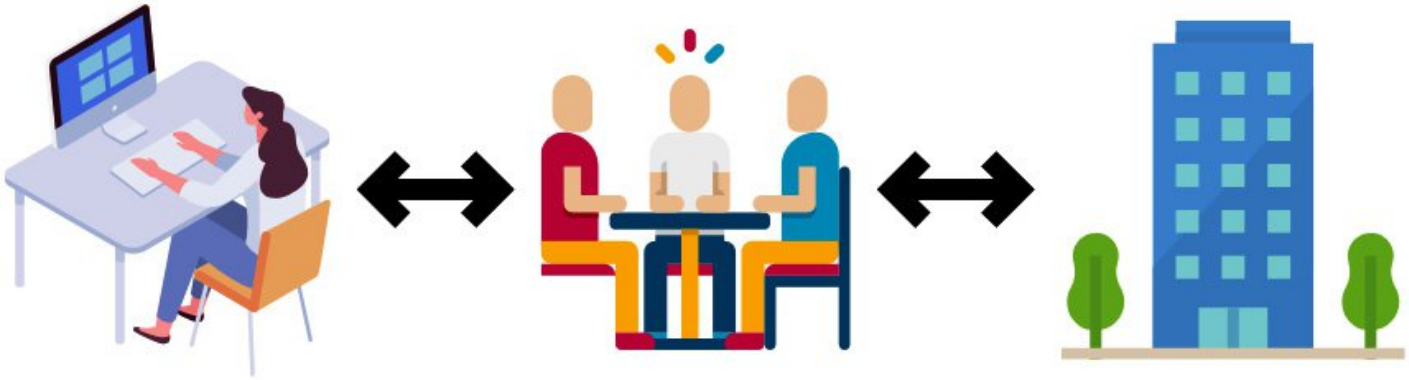


Рис. 6.2 – Опосредованные раскрытия информации, происходят с помощью брокера, который руководит взаимодействием и ожидаемым результатом

Как продемонстрировано в примере раскрытия уязвимости, начинавшем эту главу, поставщики программного обеспечения не всегда рады раскрытию уязвимостей. В этих случаях, арбитраж может оздоровить отношения между исследователем и вендором, разобраться в причинах разногласий и разработать план решения проблемы.

В конечном счете, арбитраж предоставляет участникам формализованные методы взаимодействия и достижения взаимопонимания. В большинстве случаев при арбитраже исследователь общается с этим беспристрастным участником. В большинстве случаев арбитр может помочь сохранить ваше инкогнито, если вы считаете, что будете работать с враждебно настроенной организацией, помочь вам обойти нежелательные сценарии и предоставить юридическое сопровождение, если потребуется.

Типы арбитров

Главным типом арбитров, являются **арбитражные организации**, которые в целом, считаются лучшим выбором для получения помощи по раскрытию информации. Эти организации, как правило работают с помощью волонтеров, ученых и штатных сотрудников, которые желают помочь в улучшении безопасности программного обеспечения и оборудования. Они часто связаны с компаниями, охотно принимающими отчеты об уязвимостях и если нужно, помогающих сохранять анонимность исследователя. Дополнительно, эти организации часто предоставляют такие возможности, как юридическое сопровождение, если компания угрожает исследователю, судебным преследованием. Помните мы обсуждали программы баг-баунти в предыдущей главе, *Публикация уязвимостей – публикуем свою работу в базах данных*? Они тоже могут выступать в качестве арбитра. Программы баг-баунти, запускаются компаниями, которые хотят улучшить свой подход в сфере информационной безопасности путем *привлечения всех желающих* к написанию (*в оригинале crowdsoursing*) отчетов об уязвимостях. Эти программы, обычно содержат четкие указания о том, какие типы уязвимостей следует искать, как исследователи должны подавать отчеты и на какое вознаграждение они могут рассчитывать.

Редко встречающимся источником арбитража, могут быть частные лица, действующие от имени исследователя. Такие арбитры, обычно защищают исследователя от разных неприятностей. Этот список может включать других исследователей, юристов, ученых и людей, которые могут иметь, а могут и не иметь опыта ведения арбитража.

В заключение отметим, что большинство организаций используют свои юридические службы для арбитража между своей компанией и остальными. Найм юридических услуг, не по карману

большинству исследователей, и как таковой, в основном применяется крупными поставщиками ПО в разрешении споров.

Когда задумываются об использовании арбитражных сервисов?

Существует масса причин, по которым исследователи, могли бы захотеть воспользоваться услугами арбитра при раскрытии информации об уязвимостях. Не каждые взаимоотношения между исследователем и вендором, заканчиваются враждой. Здравомыслящие исследователи знают, что арбитраж помогает защитить их от различных неприятностей могущих возникать в ходе раскрытия информации. Исследователи могут обратиться к арбитрам, чтобы избежать проблем при исследовании уязвимостей проприетарных программ определенных производителей. Это не является идеалом для всех исследователей, но арбитраж может помочь тем исследователям, которые хотят уклониться от подписания договора или соглашений о неразглашении.

В обратных случаях, работа с вендором могла бы не стоить времени и сил исследователя. Например, для исследователя может оказаться непреодолимым препятствием, если у компаний отсутствуют понятные правила в отношении раскрытия уязвимостей или доклады о них полностью игнорируются. В таких случаях, обращение к арбитру, поможет вам доставить ваш доклад нужным людям и повысит шансы на получение ответа, освободив вас от большей части хождения по инстанциям.

Если это не те факторы, которые вы обычно учитываете, может быть трудно определить, когда обращаться в сервис арбитража. Однако, обычно лучше рассмотреть арбитражный сервис, в качестве третьей стороны, если возникла напряженность между участниками и начали вырисовываться разногласия, в отношении вашего исследования. Множество разных видов противоречий могут возникать в течение раскрытия информации об уязвимостях. Некоторые из распространенных типов конфликтов, включают следующее:

- Поставщики ПО, отклоняют работу по уязвимостям по непонятным или неизвестным причинам
- Раскрытие информации, затрагивает несколько производителей, подвергающихся угрозе, но какой-нибудь заявляет, что не собирается ничего предпринимать.
- Несогласие отдельных личностей, по методам устранения или исправления уязвимости
- Поставщики ПО, искажают факты
- Вендоры не предоставляют графика или прогноза по исправлению
- Вендоры не взаимодействуют или прекращают общение с исследователями

Будучи исследователем, вы неизбежно окажетесь у черты, где ваши исследования встретятся с ситуацией, разрешить которую, будет по видимому непросто. Если это произошло, подумайте над началом согласованного раскрытия информации совместно с арбитром. Введение некоего непредвзятого участника, в неформальное обсуждение ваших исследований, может сильно помочь. Итак, какие же очевидные выгоды принесет такое раскрытие информации?

Выгоды использования арбитража уязвимостей

Если вы не уверены, следует ли вам обращаться к арбитру, в арбитраже есть определенные преимущества. Несмотря на то, что арбитраж, более растянут во времени, чем прямолинейные решения, некоторые его плюсы перевешивают неудобства и часто некоторые исследователи, в принципе могут раскрыть информацию об уязвимости, только с помощью арбитража. Некоторые из преимуществ, включают следующее:

- **Улучшенное взаимодействие**: Во многих случаях арбитр может помочь улучшить взаимодействие между исследователем и организацией. Это полезно в ситуациях, когда

организация оказывается неотзывчивой или присутствует языковой барьер. Например, если некая организация не отвечает на попытки исследователя раскрыть уязвимость, исследователь может связаться со стороны арбитража. Сторона арбитража, далее связывается с организацией от имени исследователя и пытается наладить взаимодействие. Если оба участника оказываются готовыми к общению, арбитр может содействовать проведению переговоров и сохранению атмосферы доброжелательности. Арбитр, также может помочь сохранению спокойного диалога и сосредоточению на решении проблемы. Повышение напряженности, может стать причиной словесной перепалки и общение через сторону арбитража предохраняет от такого развития событий.

- **Повышенные шансы получения вознаграждения**: В некоторых случаях, арбитр помогает увеличить шансы на получение вознаграждения за ваше исследование. Это особенно справедливо в отношении программ баг-баунти и zero-day брокеров, скупающих уязвимости, которые они потом раскрывают поставщикам ПО.
- **Защита от судебного преследования**: Арбитры могут помочь защитить исследователей от судебного преследования. Это чрезвычайно полезно, при работе с компаниями, которые могли бы угрожать судебным преследованием против исследователей, раскрывающих информацию об уязвимостях без соответствующих разрешений. Эта защита включает анонимность для исследователя и оказание юридической поддержки, когда это необходимо.
- **Создание личного бренда**: Вы, как исследователь, могли не иметь признанного "имени" в сообществе исследователей информационной безопасности, являясь слабо узнаваемым для компаний, с которыми вы связываетесь. Использование арбитражных сервисов, поможет узаконить ваши исследования, так как арбитры часто придают исследованию законную силу, перед тем как раскрыть его результаты, и как правило используют имена ассоциирующиеся с участниками исследования безопасности. Например университет Карнеги-Меллона, предлагает программы, которые сразу сделают вам "имя" и повысят авторитет ваших докладов. Это помогает добавить убедительности, избавиться от излишнего давления и создать благоприятные условия для сосредоточения на исправлении уязвимостей.
- **Беспристрастность**: Арбитры могут обеспечить объективный взгляд на ситуацию со стороны. Например, если вы не уверенны выпускать ли публично информацию об уязвимости, арбитр может помочь вам взвесить все "за и против", такого поступка.

Примечание

Держа в памяти все эти плюсы, пожалуйста отметьте, что в работе с арбитрами, предполагается, что вы позволите им руководить процессом и указывать вам, что делать и когда это делать. Деятельность выходящая за рамки инструкций, предоставленных арбитром, настоятельно не рекомендуется и даже может послужить причиной углубления разногласий во взаимоотношениях с любым вендором.

Теперь, когда мы разобрались, что такое арбитраж уязвимостей, в каких случаях обращаться к арбитру и ключевые преимущества арбитража, давайте рассмотрим, собственно процесс и покажем практический пример, системы арбитража в действии.

Разрешение споров посредством арбитража уязвимостей

Разрешение споров посредством арбитража, является относительно простым, как только вы нашли и вышли на контакт с арбитром. Давайте кратко рассмотрим процесс с самого начала, следуя по этапам арбитража уязвимостей.

Ход развития арбитража уязвимостей

Процесс арбитража уязвимостей, может варьироваться в зависимости от конкретных разногласий. Тем не менее, в целом есть три основных шага присутствующих в арбитраже:

- Одна из сторон, участвующих в споре, инициирует связь с арбитром и как правило это исследователь. На этом этапе, арбитр может согласиться или отвергнуть участие в раскрытии информации.
- Как только арбитр согласился заняться этим случаем раскрытия информации, он собирает контактные данные целевой организации. Арбитр, как правило на этом этапе просмотрит заявления, сделанные исследователем и подтвердит их достоверность. С этими контактными данными и подтвержденной информацией, он приступит к налаживанию взаимодействия, связи со всеми участниками, для обсуждения графика и уточнения ожиданий относительно решения.
- Арбитр помогает участникам достичь соглашения по урегулированию разногласий. Далее, арбитр ведет наблюдение за ходом процесса урегулирования и прекращает свое участие, только видя, что все возможные проблемы устранены или разрешились лучшим из доступных вариантов.

Арбитры будут отличаться своим опытом и арсеналом возможностей, в зависимости от того, к кому вы обратитесь за помощью. Учитывая вышесказанное, поскольку существует потребность в квалифицированном арбитраже, появились лучшие системы, чтобы помочь удовлетворить эту потребность. Один из прекрасных примеров это **Vulnerability Information and Coordination Environment** (**VINCE**), организованная **The Computer Emergency Response Team Coordination Center** (**CERT/CC**).

Нам нравится VINCE, потому что у нее куча замечательных функций. Думайте о VINCE, как о конфиденциальном email и системе планирования, которая помогает упорядочить информацию связанную с раскрытием информации об уязвимостях. Как и в большинстве систем описания уязвимостей, VINCE требует от исследователя, сначала создать описание раскрытия уязвимости, так чтобы они могли поскорее получить информацию по затрагиваемым системам, см. рис. 6.3:

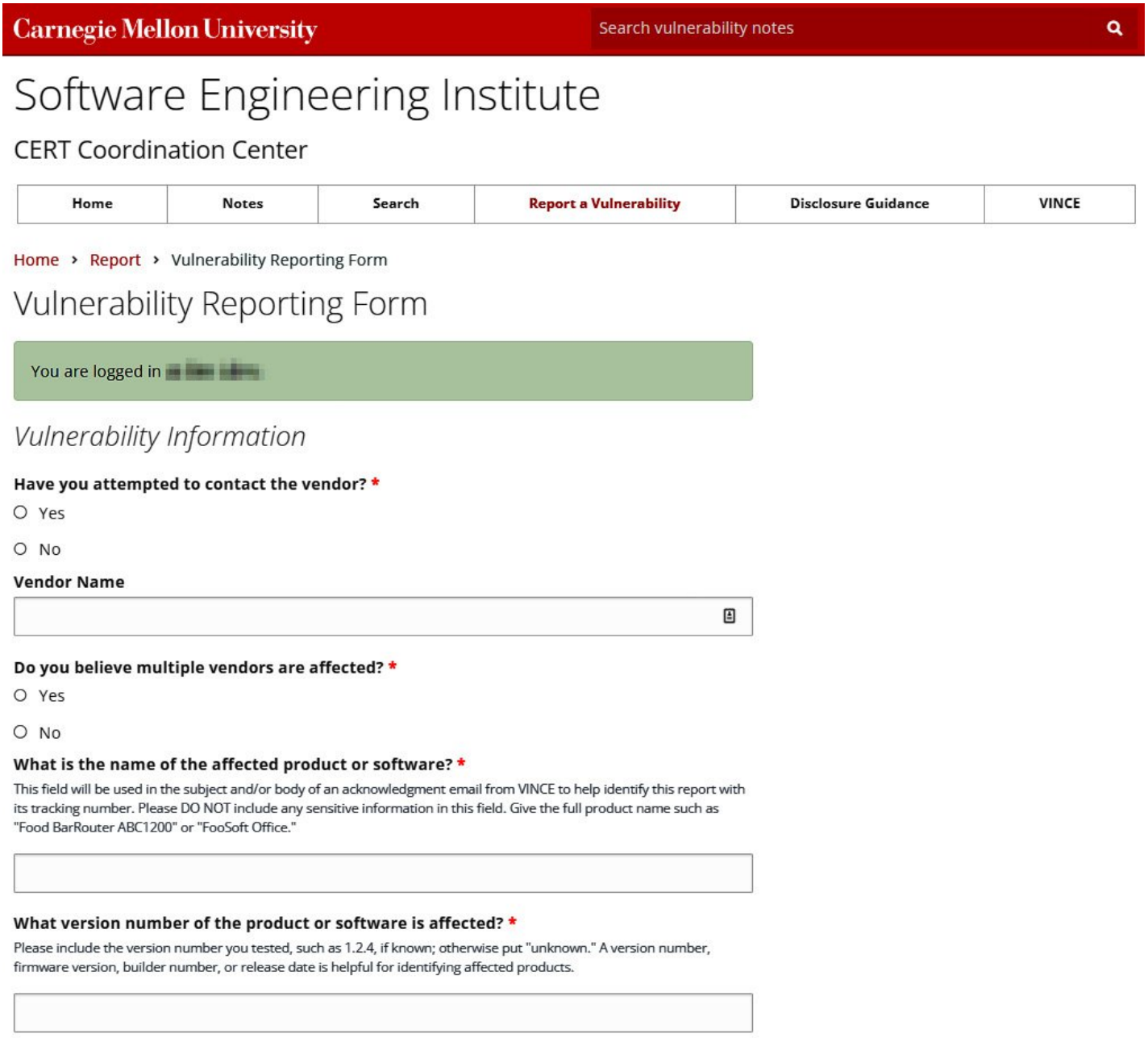


Рис. 6.3 – Форма первичной подачи информации VINCE

После предоставления первичной информации об уязвимости, данные получают ID и представителя от CERT/CC, который будет курировать процесс. Как только вам выделят ресурсы в системе VINCE, вы будете взаимодействовать с CERT/CC, которая будет запрашивать дополнительную информацию. Обычно, это подробности об уязвимости, контактная информация и что-нибудь еще, что могло бы понадобиться, как показано на рис. 6.4:

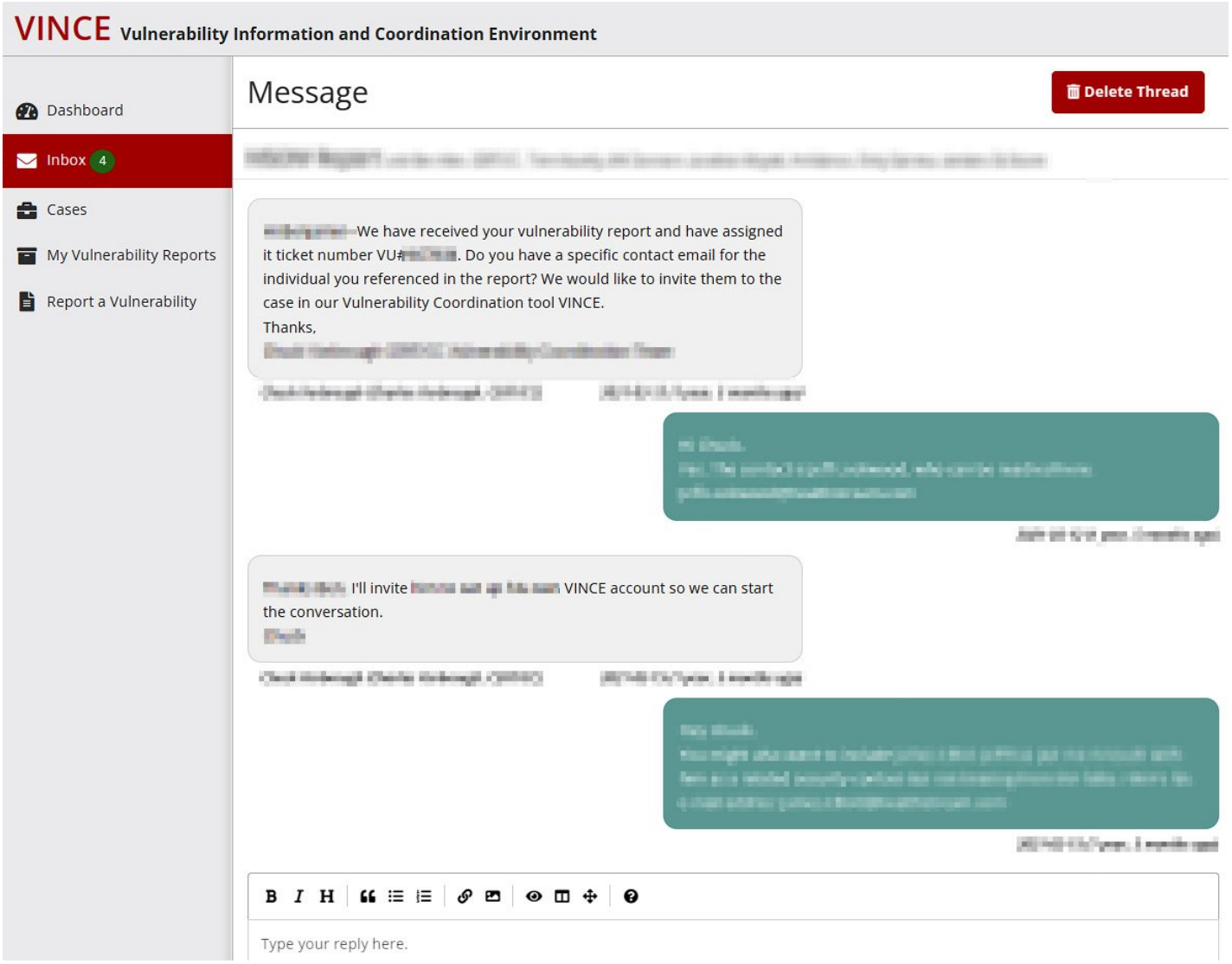


Рис. 6.4 – Портал связи VINCE, где вы можете вести конфиденциальные переговоры с CERT/CC

Как только арбитры начали общение с вендором, они помогают установить реалистичный график и предложат проверенный и эффективный способ взаимодействия для определения ожидаемых показателей закрытия уязвимости. Поставщики ПО, арбитры и исследователи совместно работают в ветке обсуждения касаемо запланированной даты раскрытия информации. В зависимости от серьезности уязвимости, CERT/CC может принять решение опубликовать раскрытие информации об уязвимости самостоятельно или оставить это на усмотрение исследователя, с учетом предоставленных ему сроков. Для вашего удобства, вся информация отображена в дашборде VINCE, как показано на рис. 6.5:

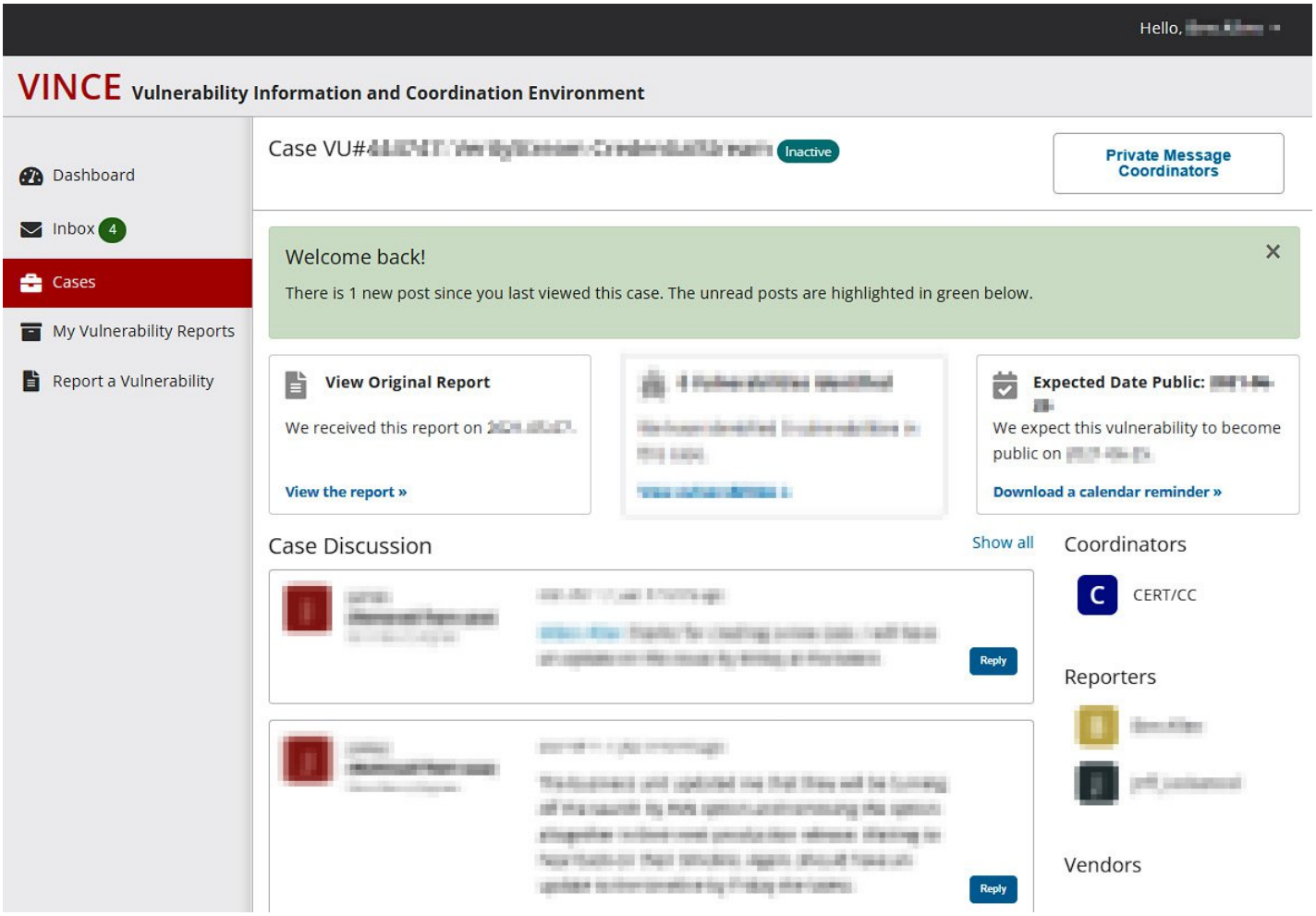


Рис. 6.5 – У VINCE, подробнейший дашборд, вмещающий всю информацию относящуюся к задаче.

Задачи доступные в вашем аккаунте VINCE, имеют префикс **VU#номер** и VINCE удобно сохраняет переписку, документацию и список участников в одном месте. Как только CERT/CC завершила арбитраж между сторонами, отчет закрывается и сохраняется в вашем дашборде, чтобы позднее можно было его просмотреть.

Большинство арбитражных ресурсов, не настолько всеобъемлющи или организованы, так как большинство работают по email. Даже если специфические нужды вашего арбитража, требуют работы CERT/CC, мы настоятельно рекомендуем использовать систему VINCE.

Теперь, когда мы разобрались, как производится раскрытие информации и рассмотрели этот процесс с применением такого средства CERT/CC, как VINCE, давайте поговорим про другие организации, которые мы можем использовать, если ищем конкретной помощи в арбитраже.

Организации проводящие арбитраж

К сожалению, когда доходит до организаций доступных публично, не существует исчерпывающего набора ресурсов, связанных с арбитражом. Однако, есть три крупных арбитражных организации, которые вам следует в первую очередь принять во внимание. Они считаются частью **Computer Emergency Response Team** или **CERT**. В основном CERT распространены в Соединенных Штатах, хотя могут быть найдены и в других частях света. Дополняя сказанное выше, даже если вы находитесь не в Соединенных Штатах, если в вашем исследовании есть компоненты поддерживаемые в США, использование одной из этих трех организаций, вероятно лучший способ обращения за помощью. А сейчас, давайте рассмотрим три наиболее известные организации.

Примечание

CERT, это акроним применяемый различными правительственными организациями, такими как Community Emergency Response Team и Correctional Emergency Response Team. Их часто путают между собой, так как они обе коллективно работают с инцидентами информационной безопасности.

CERT/CC

CERT/CC – это учреждение, помогающее координировать ответ на инциденты компьютерной безопасности. CERT является подразделением **Института программной инженерии** (**Software Engineering Institute - SEI**), университета Карнеги-Меллона. SEI сотрудничает с Министерством Обороны Соединенных Штатов и другими правительственными учреждениями. Программа CERT стартовала в ноябре 1988 года, в качестве ответа на инцидент с червем Морриса. Этот червь причинил значительный ущерб компьютерным системам по всему интернету. CERT/CC была создана, чтобы помочь согласованию ответных действий на будущие инциденты и предоставлять информацию об угрозах безопасности.

С момента своего создания, CERT/CC отреагировала на тысячи инцидентов. Они простирались от небольших DOS-атак до крупномасштабного распространения червей, таких как Code Red и Nimda. CERT/CC также, сотрудничали с правоохранительными органами по некоторым расследованиям.

Дополнительно к реакции на инциденты, CERT/CC участвуют в нескольких проектах по предупреждению угроз. Это включает разработку метрик безопасности, исследование новых техник атак и работу с поставщиками ПО над улучшением безопасности их продуктов.

CERT/CC играет важнейшую роль в работе по улучшению информационной безопасности. Другие организации, такие как **US Computer Emergency Readiness Team** (**US-CERT**) и **UK National Cyber Security Centre** (**NCSC**), играющие важнейшую роль в этой работе, вдохновлялись, главным образом CERT/CC.

Как обсуждалось в предыдущем разделе по *Разрешению споров посредством арбитража*, лучшим средством задействовать арбитраж CERT/CC, является программа VINCE. Этот сервис подойдет для большинства споров и предоставит наиболее удобный способ взаимодействия между арбитрами из CERT и проблематичным поставщиком ПО. Вот ссылки, которые вы можете использовать:

- **VINCE:** https://www.kb.cert.org/vince/
- **CERT/CC:** https://www.kb.cert.org/vuls/

US-CERT

US-CERT – это команда, которая координирует реакцию на инциденты информационной безопасности в Соединенных Штатах. Они также предоставляют руководство по устоявшимся практикам раскрытия информации об уязвимостях. US-CERT, являются подразделением Министерства внутренней безопасности (United States Department of Homeland Security) начали работать в 2003 году в качестве ответа на террористическую атаку 9/11 по Соединенным Штатам. С момента своего создания организация сосредоточила усилия на вопросах кибербезопасности. Миссия US-CERT – обеспечивать безопасность национальной ИТ-инфраструктуры, сотрудничая с партнерами по предотвращению, выявлению и реагированию на киберугрозы, защищая граждан Соединенных Штатов и всего мира. Для выполнения своей миссии US-CERT, сосредоточивается на следующих целях:

- Сбор и анализ информации о киберугрозах и уязвимостях

- Информирование партнеров о киберугрозах и уязвимостях
- Разработка и распространение обучающих материалов по информационной безопасности
- Координирование ответных действий на инциденты информационной безопасности
- Управление расследованием инцидентов и предоставление поддержки пострадавшим при инцидентах информационной безопасности

US-CERT также управляет National Cyber Awareness System, платформой информирующей общественность о ситуации в информационной безопасности. National Cyber Awareness System включает веб-сайт, сервис электронной рассылки и страницы в социальных сетях, все это используется, чтобы информировать общественность об уязвимостях. Дополнительно US-CERT сотрудничает со множеством контрагентов, как в открытых, так и в конфиденциальных сегментах, включая следующие:

- Федеральные, региональные и городские власти, а также руководство автономий
- Правоохранительные органы
- Разведка
- Владельцы и сотрудники объектов критической инфраструктуры
- Исследовательское сообщество
- Сообщество разработчиков программного обеспечения
- Интернет-провайдеры
- Поставщики продуктов и услуг в сфере информационной безопасности

US-CERT нацелены на улучшение состояния информационной безопасности страны, совместно с коллегами работая над предотвращением, выявлением и реагированием на киберугрозы. В контексте раскрытия информации об уязвимостях, US-CERT работают непосредственно с поставщиками программных продуктов согласовывая раскрытия информации. Это включает работу с вендорами по разработке патчей или альтернативных решений. US-CERT также распространяет информацию об уязвимостях и угрозах через свой общедоступный веб-сайт, списки почтовой рассылки и страницы в социальных сетях. Подытоживая, US-CERT является отличным партнером, который может помочь в арбитраже раскрытия информации об уязвимостях безопасности или, по крайней мере, точно предоставит вам наиболее подходящие средства, конкретно для ваших нужд. Вот ссылка, которую вы можете использовать:

- **US-CERT:** https://www.cisa.gov/uscert/

Команда реагирования на кибер-инциденты в промышленных системах управления ICS-CERT

ICS-CERT – это отдел Министерства внутренней безопасности Соединенных Штатов. Миссией ICS-CERT, является снижение угроз в области систем промышленной автоматики. Единственный способ, которым ICS-CERT выполняет эту задачу, является совместная работа с вендорами и исследователями по согласованному раскрытию информации об уязвимостях в изделиях и комплектующих систем промышленной автоматики.

Когда обнаружены изъяны в безопасности изделия или составляющей какой-либо системы промышленной автоматики, ICS-CERT работает с производителем или исследователем над согласованным раскрытием информации об этой уязвимости. Это согласование, включает разработку плана, когда и как, информацию об уязвимости сделают общедоступной. Целью всего этого, является предоставление пользователям и организациям, времени на исправление уязвимости или мероприятия по снижению ущерба, до того, как злоумышленники смогут ее использовать.

ICS-CERT также консультирует пользователей и организации, как справляться с уязвимостями. Эти консультации включают информацию о том, что делать, если вы предполагаете, что ваша организация оказалась мишенью злоумышленников и советы по защите от предстоящих атак.

Предположим, вы поставщик оборудования или исследователь, который обнаружил дефект безопасности в изделии или составляющей какой-либо системы промышленной автоматики. В таком случае, вы можете связаться с ICS-CERT для согласованного раскрытия информации об уязвимости. Вот ссылка, которую вы можете использовать:

- **ICS-CERT:** https://www.cisa.gov/ics

Другие CERT-организации

Есть множество различных арбитражных организаций, доступных исследователям. Программы CERT доступны по всему миру, а более 75 партнерских организаций – в разных странах. Один из лучших способов отыскать, есть ли ваша страна среди участников программ CERT - это через справочник команды **Forum of Incident Response and Security Teams** (**FIRST**), как показано на рис. 6.6:

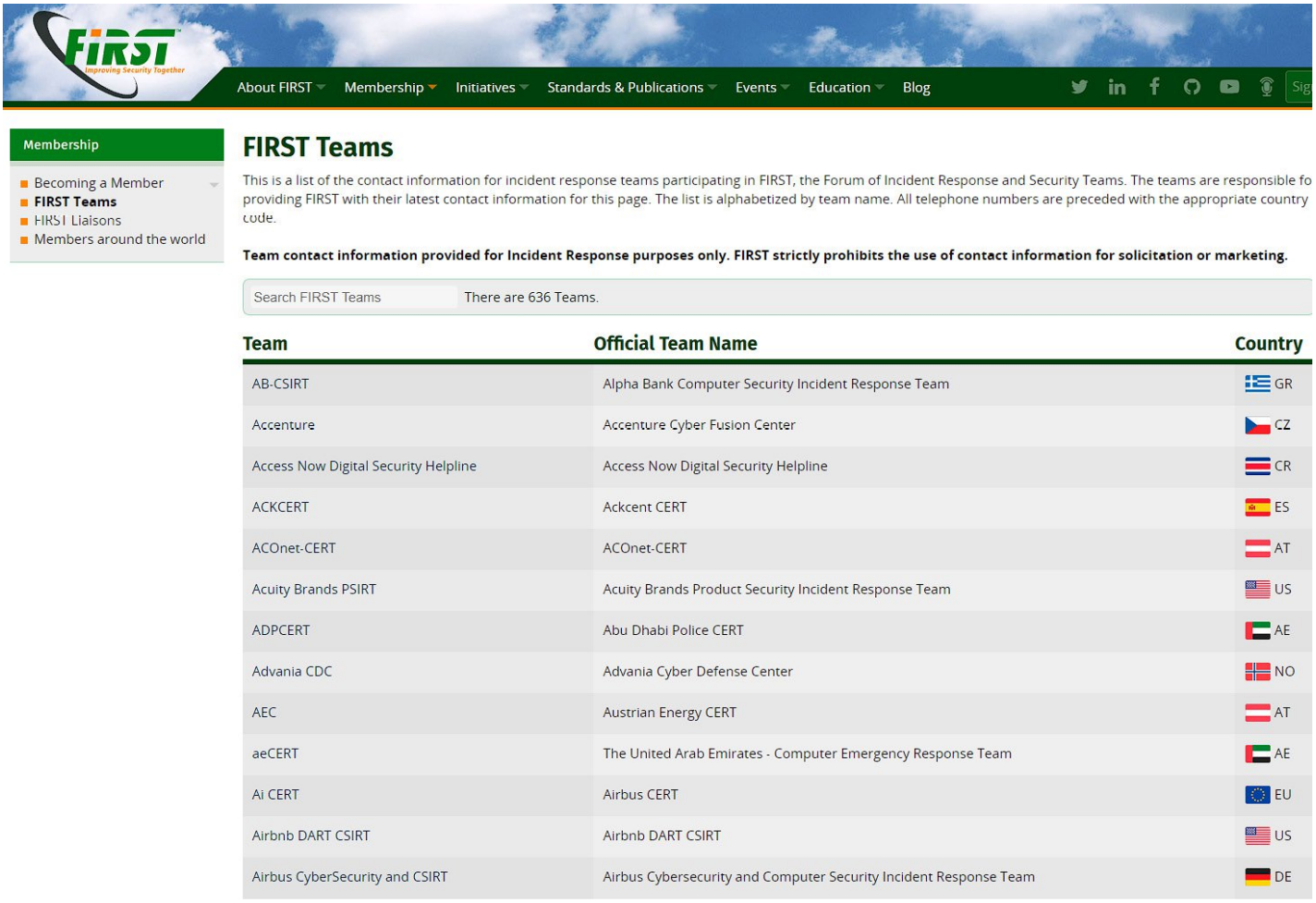


Рис. 6.6 – Справочник команды FIRST, поможет вам найти другие программы CERT

Хотя FIRST, является отличной организацией, в качестве отправной точки – она не будет управлять раскрытием информации об уязвимостях. *Организация-участник FIRST, это не обязательно CERT, однако организации CERT, обычно состоят в FIRST.* Не забывайте об этом, просматривая их справочник, CERT обычно будет отображаться с одноименной аббревиатурой. Вот ссылка, которую вы можете использовать:

- **Справочник FIRST team:** https://www.first.org/members/teams/

Программы баг-баунти

Программы баг-баунти, были рассмотрены в главе, посвященной процессу раскрытия информации об уязвимостях, *Глава 4, Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности.* Обычно эти программы, действуют как арбитражный сервис, это еще одна возможность поиска организаций проводящих арбитраж.

Хотя большинство организаторов баг-баунти, применяют модель, где вознаграждения выплачиваются за конкретные цели исследования, альтернатива есть. Например платформа Open Bug Bounty, следует стандартам ISO по согласованному и этичному раскрытию информации об уязвимостях и позволяет исследователям, предлагать на рассмотрение баги веб-сайтов, которые не участвуют в программе, см. на рис. 6.7 их статистику и заявленные условия. Хотя Open Bug Bounty замечательная платформа, она имеет свои ограничения и принимает только отчеты по определенным классам уязвимостей веб-приложений:

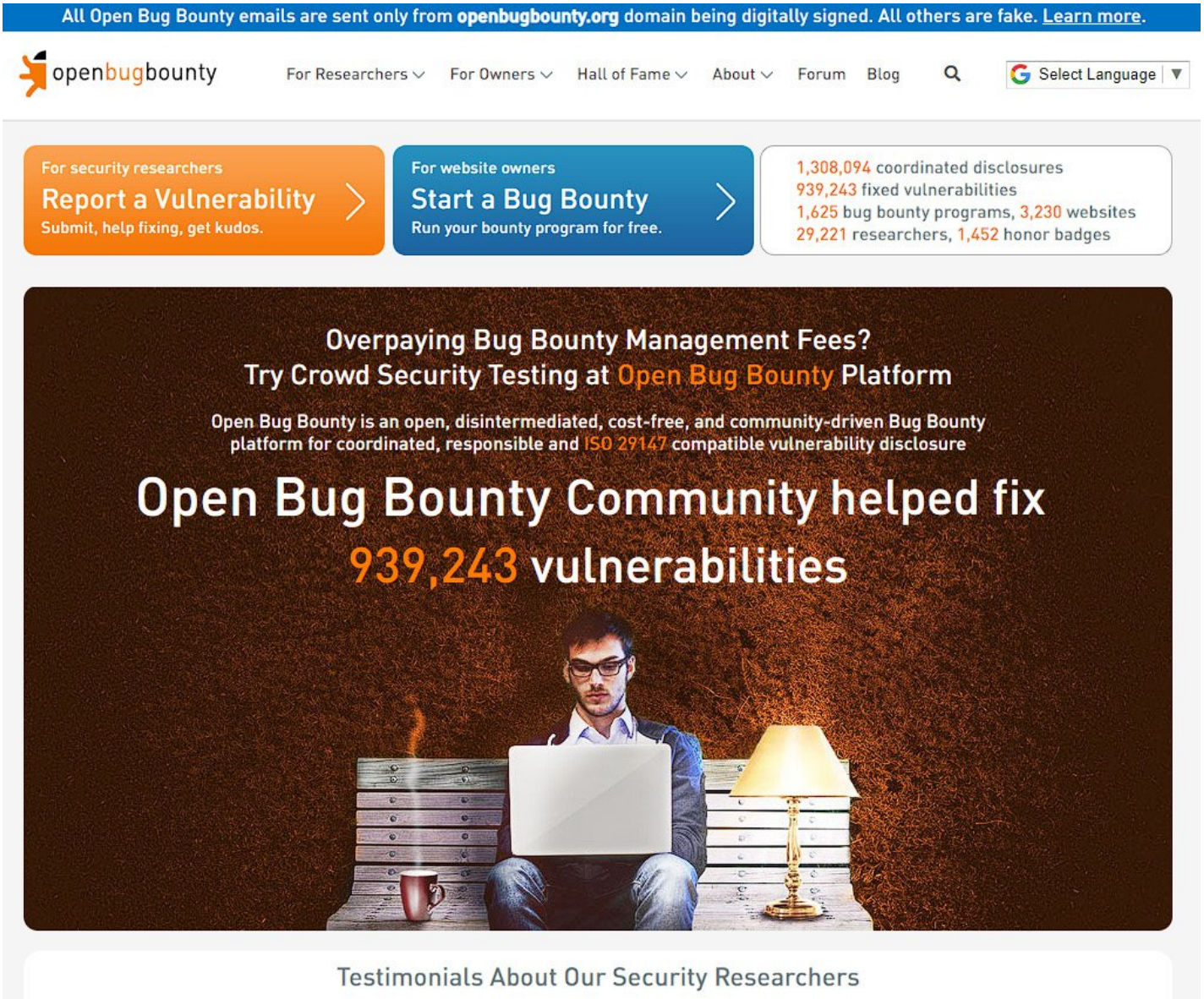


Рис. 6.7 – Open Bug Bounty соответствует ISO 29147, редкая фича среди программ баг-баунти

Вот ссылка, которую вы можете использовать:

- **Open Bug Bounty:** https://www.openbugbounty.org/

Юридическое сопровождение

Раскрытие информации об уязвимостях, может спровоцировать судебные претензии и исследователям грозят проблемы с законом. Когда риск судебного преследования, приобретает реальные очертания, исследователи безопасности часто обращаются за поддержкой в **Electronic Frontier Foundation** (**EFF**). EFF, является некоммерческой организацией, которая борется в сети за гражданские свободы, с акцентом на защите прав хакеров и исследователей безопасности.

EFF помогла множеству хакеров, которые оказались под угрозой судебного преследования из-за раскрытия информации об уязвимостях. Во многих случаях EFF выступала представителем хакеров, которые нашли изъяны безопасности в программном обеспечении. Софтверные компании, угрожали хакерам масштабным преследованием, если они раскроют информацию об уязвимости.

EFF вмешалась и помогла договориться о решении, которое позволило хакеру, раскрыть информацию об уязвимости, не опасаясь привлечения к ответственности. EFF является приверженцем защиты хакеров, которые участвуют в раскрытии информации об уязвимостях. Поступая так, они помогают делать мир безопаснее для каждого. Вот ссылка, которую вы можете использовать:

- **EFF:** https://www.eff.org/

Другие варианты арбитража

Арбитры не всегда появляются в виде официальных организаций. Не существует ни каких правил оговаривающих, кто может быть арбитром. Например, исследователи могут обратиться к следующим лицам, если обращение в какую-либо организацию невозможно:

- Юрист, знакомый с информационной безопасностью и относящимся к ней законодательством
- Ученый или исследователь специализирующийся на компьютерной или информационной безопасности
- Какой-нибудь этичный хакер или консультант в сфере безопасности, имеющий опыт раскрытия информации об уязвимостях
- Любые незаинтересованные третьи лица, которые как вы предполагаете, могли бы облегчить взаимодействие и понимание между сторонами вовлеченными в процесс раскрытия информации об уязвимостях

Важно отметить, что кто угодно может вести арбитраж, безотносительно квалификации и опыта. Однако учтите, что обращение к арбитрам не имеющим опыта может сильно затянуть арбитраж и усложнить его, по сравнению с опытным арбитром. Главное найти кого-то, кому вы доверяете и не напрягаетесь, по поводу его руководства. Четко понимайте свои цели, чего хотите вы и что требуется от арбитра, привлеченной организации или частного лица, представляющего ваши интересы.

Завершая тему об организациях проводящих арбитраж – вы теперь знаете ключевые арбитражные организации используемые исследователями, как найти международные организации и прочие сопутствующие средства.

Итоги главы

Разногласия в течение раскрытия информации об уязвимостях, могут требовать напряжения сил для их разрешения. Однако арбитраж – полезный инструмент для улаживания таких споров. Арбитраж улучшает коммуникацию между участниками, предотвращает будущие разногласия и выстраивает доверие между ними. Множество различных организаций, могут помочь, если все пошло

наперекосяк в течение раскрытия информации. При выборе арбитражной организации, главное учесть конкретные пункты ваших разногласий, которые необходимо разрешить, стороны вовлеченные в спор, место проведения арбитража и его издержки. Действовать необходимо быстро, но выбор верных методов работы, также важен. Лучший образ действий, зависит от характера разногласий и участвующих сторон.

Использование арбитража считается одним из лучших методов раскрытия информации об уязвимостях. В следующей главе, мы рассмотрим одно из последних средств, часто воспринимаемое, как одно из наихудших в раскрытии информации – полностью самостоятельное раскрытие информации.

Глава 7. Независимая публикация уязвимостей

В течение нескольких часов после публикации своего исследования, озаглавленного *Множество пользователей Talkspace восприимчивы к фроду* в личном блоге, Джексон Джексон получил письмо от юридического отдела Talkspace, информирующее его, что он столкнется с юридическими последствиями, если не удалит пост об их компании со своего веб-сайта. Джексон выполнил требование Talkspace, но обратился в СМИ со своей историей. TechCrunch запостили материал, описав Джексона, как исследователя безопасности обнаружившего, возможность использования промокодов компании, для регистрации и бесплатного использования Talkspace, находя особые выражения на веб-сайтах проиндексированных Google. Затем они поделились тем, что смогли зарегистрироваться на Talkspace используя промокоды, которые принадлежали другим компаниям, оплатившим сервис. Зак Уитакер, журналист опубликовавший материал на TechCrunch, составил надлежащее уведомление, подведя итог по ряду случаев, когда исследователи подвергались угрозам крупных технологических компаний.

Хотя точно известно, что исследователи безопасности (и даже работники прессы), в основном предвидят юридическое противодействие, разглашению своих исследований безопасности, были ли претензии к Джексону обоснованы? В письме Talkspace Джексону, они сообщили, что не согласны с изложением проблем описанных в блоге, относительно обокраденных клиентов Talkspace. Ассоциация клиентов Talkspace с воровством и мошенническими действиями, была посчитана Talkspace недопустимой, и они направили Джексону предписание о запрете продолжения исследования.

Исследователи безопасности, во многом похожи прессу, потому что они распространяют подробности о результатах своих исследований и доносят их до широкой публики. Однако в отличие от СМИ, некоторые исследователи обнаруживают свои высказывания в разгоряченных постах блогов и иногда спекулятивные утверждения о своих исследованиях. В дополнение скажем, что исследователи безопасности часто, работают в условиях ограниченного обзора, как кто-то не в теме, заглядывающий снаружи, из-за этого многие исследователи, заполняют свои пробелы в знаниях допущениями и гипотезами.

Если взаимодействие с вендором нарушается, исследователи могут искать альтернативные способы разделить свои исследования с сообществом. Это тот случай, когда происходит независимая публикация уязвимостей, и часто это применяется, если поставщик ПО и исследователь не могут достичь согласия или арбитраж между сторонами не принес положительного результата.

Сворачивая раздел с главами, включающими исследование уязвимостей, раскрытие информации, публикацию и арбитраж, мы завершим его рассмотрением **независимой публикации уязвимостей**. Вы можете вспомнить, мы обсуждали публикацию уязвимостей в контексте независимого раскрытия информации в *Главе 4, Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности* и *Главе 5. Публикация уязвимостей – публикуем свою работу в базах данных*. Однако мы хотели бы взять немного дополнительного времени для обсуждения независимой публикации уязвимостей, начиная с того момента, как она появилась и продолжит оставаться насущным инструментом исследователей, по раскрытию результатов исследований, если вендоры отказываются в этом участвовать. Дополнительно, мы продолжим рассматривать спор Джексона, раскрытие результатов исследования Talkspace и обсуждать, насколько вероятно попасть под какой-нибудь из шести рисков, которых вам следует избегать, чтобы предотвратить юридическое преследование, могущее начаться против вас. В заключение, мы поделимся чек-листом с выжимкой советов из этой главы, в виде простого в применении набора ключевых моментов.

В этой главе, мы займемся обсуждением следующих вопросов:

- Независимая публикация уязвимостей и присущие ей риски
- Как представить себя в лучшем свете и способы, которые могут помочь вам предотвратить получение судебного предписания
- Шесть сценариев, которых исследователям необходимо избегать, при независимой публикации
- Как независимо опубликовать, раскрытие информации об уязвимости

Эти результаты будут представлены на протяжении следующих основных тем этой главы:

- Независимые раскрытия информации и их место в жизненном цикле уязвимости
- Плюсы независимой публикации
- Риски независимой публикации
- Как независимо публиковаться избегая рисков

Примечание

Перед тем, как мы начнем эту главу, давайте кое-что проясним. Это не рекомендации автора или издателя, которых вы должны придерживаться в любых действиях по публикации ваших раскрытий информации об уязвимостях. Когда вы решаете, следует ли вам раскрывать информацию об уязвимости, честно определите сами для себя ваши цели. Чего вы пытаетесь достичь этим раскрытием информации? Использовали ли вы возможности арбитража или сотрудничества с представителем, который возьмет ваши риски на себя, вместо вас? Будьте очень внимательны и скрупулезны , когда выбираете, могли ли быть лучшие способы достичь ваших целей, без самостоятельного раскрытия информации. Мы досконально проанализируем, каким образом минимизировать любые риски, с которыми вы можете столкнуться, но даже с этими минимальными рисками, все еще есть существенные угрозы могущие проявиться, если вы в одиночку примете на себя все последствия независимого раскрытия информации. Воспринимайте это предупреждение серьезно, перед тем как начнете действовать.

Независимые раскрытия информации и их место в жизненном цикле уязвимости

В традиционном полном раскрытии информации, как только исследователь обнаруживает новый баг, он докладывает об этом компании или организации, отвечающей за проблемное ПО. Компания или организация, оценивает этот баг, чтобы определить является ли он проблемой безопасности, и если это так – они работают над созданием исправления. Как только исправление готово, они согласовывают с исследователем его выпуск, параллельно с объявлением деталей проблемы и шагов по снижению ее последствий.

Эта процедура, нормально работает, когда все вовлеченные стороны сотрудничают делясь друг с другом всей необходимой информацией. К сожалению, не каждый, особенно со стороны поставщика ПО, следует этим базовым принципам в процессе раскрытия уязвимостей, сохраняя его функциональность и разумный подход. Иногда компании скрывают уязвимости, оставляя их без исправления, месяцы или даже годы. В одних случаях, это происходит из-за отсутствия у компании нужных ресурсов. Однако в других, это от того, что компания не желает знать, что их программное обеспечение имеет изъяны в безопасности. Чтобы бороться с этим, как исследователю – вам следует инициировать переговоры совместно с посредником и компанией, действиями которой вы недовольны, как описано в *Главе 6. Арбитраж уязвимости – что пошло не так и кто может помочь*. А что, если компания, не садится за стол переговоров? А может они и сели, но переговоры провалились и ни одна из сторон не была удовлетворена результатами. Если это произошло с вами, независимое раскрытие

информации, могло бы быть последним шагом в вашей последовательности действий по информированию общественности.

Интересным фактором, могущим подтолкнуть исследователей к независимой публикации, является то, что некоторые вендоры, могли не реагировать на раскрытие информации об уязвимости, без изначального юридического соглашения с исследователем. Прекрасный пример такого случая – когда Modzero, компания по информационной безопасности, раскрыла информацию о незначительных проблемах, найденных в агенте Falcon **endpoint detection and response** (**EDR**), производства CrowdStrike 8 августа 2022 года. CrowdStrike's Falcon agent, это вроде продвинутого антивируса, который отслеживает события и останавливает вредоносную активность. Для работы, агент должен быть установлен на компьютер, так он сможет мониторить систему. В информации раскрытой исследователями безопасности, они выяснили, что могут обойти защиту удаляя ПО, способом доступным исследователю, так как с началом процедуры удаления – подразумевается остановка работы программы. Когда Modzero раскрыла информацию об уязвимости, они использовали для раскрытия информации, непосредственно платформу HackerOne. Как обсуждалось в *Главе 4*, программы баг-баунти, часто имеют условия использования сервиса и определенные ограничения. Modzero нашла условия неприемлемыми (особенно **соглашение о неразглашении** - **non-disclosure agreement NDA**) и хотели раскрыть информацию, непосредственно CrowdStrike. После нескольких перекидываний мяча туда-сюда, более 3 месяцев, Modzero опубликовала код эксплоита в своем блоге, т. к. CrowdStrike былы не склонны принимать результаты исследования безопасности без подписания Modzero некоего NDA.

Как исследователю безопасности, вам очень важно помнить, что если вы разглашаете результаты своего исследования и процесс раскрытия информации, оказывается чрезмерно затрудненным, вам следует поделиться этим с сообществом. Если достигнуто общее мнение, что процесс идет настолько плохо, что вызывает возмущение в исследовательском сообществе, вендор мог бы изменить свои правила для улучшения результатов исследования. После того, как Modzero опубликовали свой доклад, истории нескольких других исследователей о своем неудачном опыте, вышли на свет и в конце концов CrowdStrike, начали исправлять уязвимости (даже, хотя их обвиняли в безответственности) и обещали в будущем пересмотреть свои политики. Прискорбно, когда поставщики ПО создают сценарии требующие излишних усилий, вроде этого – это подрывает доверие к ним, как коллегам по будущим раскрытиям информации.

В ответ на такие ситуации, некоторые исследователи начали независимо публиковать результаты своих исследований, большей частью как Modzero. Вкратце это подразумевает, что исследователь, пишет подробный доклад об уязвимости и публикует его в сети. К сожалению, это обычно происходит, по прошествии многих месяцев, с момента изначального раскрытия информации и отсутствия реакции вендора.

В таких раскрытиях информации об уязвимостях, опубликованных независимо, исследователи могли выпускать код **proof-of-concept** (**PoC**) или пошаговое описание эксплуатации, так чтобы остальные смогли воспроизвести проблему. Однако если уязвимость представляет высокий риск нанесения ущерба пользователю или рабочему окружению, исследователь может только констатировать факт существования уязвимости или продемонстрировать результаты ее использования. В целом, задача состоит в оказании давления на компанию, путем разъяснения им, что уязвимость существует и то, что кто-нибудь со злыми намерениями найдет и использует ее – только вопрос времени.

Вместе со всем сказанным выше, крайне важно понять, что независимое раскрытие информации, нежелательно задействовать в поиске уязвимостей и публикации упомянутой уязвимости. Вместо этого, было бы лучше, в первую очередь попытаться связаться с компанией, информируя их о проблеме. Если согласованное раскрытие оказалось невозможным, компания прекратила сотрудничество или, что еще хуже арбитраж с участием третьей стороны провалился, что

ж – это оправдывает публичное раскрытие информации. В дополнение, большинство команд реагирования на компьютерные ЧС (**computer emergency response teams - CERTs**), рекомендуют давать поставщикам ПО временной промежуток в 45-90 дней на ответ и устранение уязвимости.

В целом независимое раскрытие информации, может быть мощным инструментом, чтобы добиться от вендора обещания закрыть уязвимость. Однако, оно также приносит с собой определенные риски. Эти риски могут включать уголовную и гражданскую ответственность, которая может послужить причиной судебного преследования, если вендор предъявит вам иск. Позже мы обсудим эти риски более подробно, в разделе *Риски независимой публикации*. Но в первую очередь, давайте взглянем на плюсы независимой публикации результатов ваших исследований.

Плюсы независимой публикации

Одним из ключевых плюсов независимой публикации, является принуждение компании к действиям. К примеру, если исследователь публикует доклад по серьезному изъяну в безопасности, а компания ничего не делает, это отразится на них плохо. В результате, клиенты могут перевести свой бизнес в куда-нибудь в другое место, а другие исследователи станут менее расположены сотрудничать с этой компанией в будущем. Поставщики ПО это знают, и это подталкивает их действовать побыстрее. Хотя в сообществе информационной безопасности давно подозревают, что публикация уязвимостей подобным образом, является причиной лучшего отклика на уязвимости. Это изучалось в исследовании от **2010** года, озаглавленном ***Практический анализ жизненного цикла выпуска патча поставщиком ПО: Влияние раскрытия информации***. Согласно исследованиям и анализу, проведенному Ашишем Аророй, Рамайем Кришнаном, Рахулом Телангом, Юбао Яном из университета Карнеги-Меллона – уязвимости закрывались значительно быстрее. Они выяснили, что вендоры были в два и более раза расторопнее с исправлением ПО, и в среднем раскрытие информации приводило к исправлению на 35 дней раньше, чем если информация не обнародовалась.

В дополнение к этим плюсам, независимая публикация, также позволяет исследователям следить за графиком и комментировать ход работы. Например, как мы обсуждали в *Главе 4*, это может быть полезно, чтобы дать компании время на исправление дефекта, до публикации чего бы то ни было. Тем не менее в других сценариях, выпуск информации в общий доступ при первой же возможности, может оказаться решающим, чтобы пользователи успели себя защитить. С помощью независимой публикации вы можете делать то, что считаете наилучшим, исходя из имеющейся у вас информации.

Нередко можно услышать от исследователей истории, в которых поставщик ПО преуменьшал серьезность, предоставленных ими результатов исследования, обесценивал исследование или не давал их труду справедливой оценки. Когда исследователи публикуются независимо, они могут выбирать, что включать в свой доклад и насколько подробна будет эта информация. Независимая публикация, также позволяет вам решать, будете ли вы использовать свое настоящее имя или псевдоним, если публикуете демо, чтобы показать насколько пугающе ваше открытие, в чем его воздействие и даже поделиться кодом эксплоита, если желаете. Просто, когда вы независимо публикуете свое раскрытие информации, вы можете убедиться, что люди знают о проблеме и владеют всей необходимой им информацией, чтобы понять и снизить возможные последствия.

В конце концов, независимая публикация позволяет вам укреплять свою репутацию исследователя безопасности. Подготовьте свое исследование скрупулезно и содержательно, следуйте лучшим из общеизвестных методов и это будет полезным, если вы надеетесь вкатиться в инфосек. Вместе с вышесказаным, отслеживайте, что в опубликованном вами раскрытии информации, является основным. Теперь, давайте рассмотрим некоторые из рисков сопутствующих независимой публикации ваших исследований.

Риски независимой публикации

Независимая публикация уязвимостей, может быть отличным способом привлечь внимание к результатам ваших исследований и быть уверенным в том, что люди осведомлены об угрозах. Однако перед принятием решения, важно взвесить все плюсы и минусы.

Исследователи могут столкнуться с рядом юридических последствий, связанных с независимой публикацией исследований. Например, если вы выпустили код **proof-of-concept** (**PoC**) или пошаговое описание эксплуатации вместе со своей публикацией, в Соединенных Штатах – вы можете нарушить законодательство об авторском праве или **Акт о мошенничестве и злоупотреблении компьютерными технологиями** (**Computer Fraud and Abuse Act - CFAA**). На программный код, например – распространяется действие законов об авторском праве. В результате, вы можете подвергнуться преследованию за нарушения авторских прав, если опубликуете код вендора, как составную часть вашего PoC, без соответствующего разрешения.

Для сравнения, CFAA также запрещает несанкционированный доступ к компьютерным системам. Это часто используется, для предъявления исков хакерам, которые проникают в системы или воруют данные. Однако, это также применимо к исследователям, выпустившим код PoC или пошаговое описание эксплуатации. Так принято потому, что выпуск такой информации может быть рассмотрен, как провокация и подстрекательство кого-то еще ко взлому системы.

Совершение преступления, это еще не все подводные камни, о которых стоит беспокоиться при независимой публикации. Например, простое написание о компаниях в негативном ключе (если не сделать это корректно), может вызвать реальные проблемы с законом и привести к обвинению вас в клевете.

В дополнение к юридическим рискам, независимая публикация исследований может разрушить вашу репутацию, если она не согласуется с установившейся практикой. Например предположим, что вы раскрыли информацию об уязвимости довольно подробно, до того как она была исправлена, и без всякого предупреждения. В таком случае, вы могли бы быть обвинены в том, что подвергаете пользователей угрозе – или даже привлечены к ответственности за ущерб, причиненный в связи с раскрытием вами информации. Еще одним возможным риском, является публикация секретной информации, такой как работоспособный код PoC, который подвергает опасности информацию о клиентах поставщика ПО, которую можно идентифицировать, и вы можете быть обвинены в безответственном раскрытии информации о дефекте.

Как уже говорилось, независимая публикация может стать мощным инструментом в руках исследователей, желающих убедиться, что результаты их исследований увидят свет. Кроме того, если действовать ответственно, это поможет держать компании в тонусе и защитить пользователей от серьезных угроз безопасности.

Нам следует отметить, что хотя некоторые из описаных рисков звучат устрашающе и потенциально опасны для благосостояния исследователя, юридические последствия можно пересчитать по пальцам одной руки, да и наступают они не часто. Мы приравниваем это к опасности управления автомобилем. Да, это может быть страшно и неприятности происходят, однако если вы хорошо подготовлены и управляете машиной аккуратно, угрозы значительно снижены. Следовательно, важнейшей вещью, которую вы должны делать как исследователь – это быть осведомленным о рисках и обходить их настолько, насколько это возможно.

Помня об этих рисках, давайте уделим некоторое время обсуждению того, как вам их избежать и как независимо публиковать уязвимости, одновременно обходя все шесть наиболее распространенных рисков, с которыми сталкиваются исследователи, раскрывая информацию об уязвимостях независимо.

Как независимо публиковаться избегая рисков

Когда исследователи делают свои первые шаги в раскрытии информации об уязвимостях, путем самостоятельных публикаций, для них не редкость переживать по поводу того, как им следует поступать. К сожалению, мы скорее всего не успокоили всех, кто прочел риски раскрытия информации в начале этой главы. Обычно исследователи, будучи новичками в независимой публикации, волнуются по поводу возникновения неприятностей, проявления агрессии недоброжелателей или любого негативного воздействия, которое иссследование могло бы оказать на их жизнь.

Недавно, обсуждая публикацию уязвимостей с коллегами и другими исследователями, один исследователь (не важно кто) пыранул, что публикация уязвимостей в Соединенных Штатах, защищена поправкой к конституции о свободе слова. Это наводит на мысль, что такая деятельность, защищена законом и сообщение правдивой информации о компании вполне законно, если вы не связаны с этой компанием соглашением о том, что не будете эту информацию сообщать. Они сравнивают публикацию уязвимостей с оставлением отрицательных отзывов о бизнесе в Google, Yelp или чем-то подобном – скажем, об одном из ваших местных ресторанов. Если вы получили негативный опыт, и ресторан не пытается исправить ситуацию (или даже, если попытался), вы можете говорить все, что вы думаете об этом – в Соединенных Штатах это ваше право.

Прослеживая аналогию с рестораном, критик свободен в своих высказываниях. А еще, если эти высказывания не достоверны или в какой-то части сфальсифицированы, а бизнес может предположить, что такой обзор приведет их бизнес к убыткам, они могут подать на вас в суд. Разумеется ресторан, тоже может подать на вас иск по какому-то поводу, но это не обязательно подразумевает, что они его выиграют.

Давайте вернемся к истории Джексона Джексона, с которой мы начали и его посту в блоге насчет Talkspace. К сожалению у нас нет оригинального содержания этого поста. В настоящее время документ, которым он поделился в сети, представлен в виде цитат конкретных частей публикации, на которые ссылаются юристы Talkspace. Разделы, на которые ссылается Talkspace не содержат никаких утверждений относительно их клиентов о том, что их корпоративные клиенты были ограблены, и что клиенты Talkspace уязвимы к мошенничеству. Основываясь на этой информации, несложно увидеть, почему Джексон Джексон, получил юридическое уведомление от Talkspace.

Как некий сторонний исследователь безопасности, Джексон нашел купон с промокодами и вход на сайты, позволяющие получить бесплатный доступ к услугам. Его мнение по поводу своего исследования, выражается во фразе – *факты, вещь упрямая*. При этом наоборот, он скорее всего не владел информацией о том, как клиенты оплачивали услуги и проводилась ли верификация со стороны Talkspace или нет. Эти излишне эмоциональные выводы, были ни адекватны, ни проверены в свете фактов и доказательств, как приведено в пресс-релизе, после отправки юридического уведомления Джексону.

Как же именно, вам поступать, чтобы избежать этих факторов риска? Мы выясним это, но в первую очередь, давайте уделим время определению того, как подготовиться к противостоянию шести самым распространенным рискам, с которыми сталкиваются исследователи.

Как избежать распространенных рисков при публикации?

Существует шесть основных вещей, которые могут доставить исследователю проблем с независимой публикацией и раскрытием информации об уязвимостях в Соединенных Штатах. Отметьте пожалуйста, что эти случаи специфичны для Соединенных Штатов. Однако, последствия ваших действий и риски, скорее всего будут отличаться, в зависимости от страны в которой вы публикуете свои исследования. Например, в Японии формально противозаконно, распространять

правдивую информацию, которая могла бы повредить компании, согласно их строжайшим законам о клевете. Так, что убедитесь, что учитываете все необходимые законы, действующие в соответствующем регионе.

Нарушение договора

Многие компании имеют строгие соглашения о конфиденциальности, которые сотрудники, контрагенты и все остальные должны подписать, до получения доступа к корпоративной информации. Такие соглашения, обычно запрещают людям разглашать корпоративную информацию, кому бы то ни было, за пределами компании. В результате исследователи получают иски за нарушение договора, после опубликования информации, которую они согласились не разглашать.

Выполняя исследование, вы можете обнаружить себя ограниченным договорами, которые устанавливают ограничения по исследованию и раскрытию информации. Как исследователю, вам важно читать договоры, которые вы подписываете и быть очень внимательным к тому, что вы можете говорить, а что нет. К сожалению, этот совет относится к тем, которые многие исследователи усваивают, лишь сами набивая шишки. Опытные исследователи, нередко избегают программного обеспечения, которое требует подписания договоров или NDA (соглашение о неразглашении). Если вы нарушаете договор – подготовьтесь к последствиям. Это серьезные юридические соглашения, которые могут привести к суровым последствиям в случае нарушения.

Нарушение Акта о мошенничестве и злоупотреблении компьютерными технологиями

CFAA, является законом Соединенных Штатов, который запрещает несанкционированный доступ к защищенным компьютерам. CFAA используется для предъявления исков исследователям, получившим доступ к корпоративным системам, не имея на это разрешения.

Серверы стоят дорого, программы стоят дорого, а исследователи могут не захотеть вкладываться в покупку софта самостоятельно, вместо этого они могут их арендовать. В дополнение, многие исследователи будут тестировать свое ПО на серверах, которые им не принадлежат. Это общепринятая практика, если вы имеете явное разрешение от владельца сервера или участвуете в соглашении какой-то программы баг-баунти.

Если у вас нет разрешения на тестирование ресурсов, можно утверждать, что вы совершаете преступление и против вас могут быть возбуждены судебные иски. Следовательно, вы должны подключаться только к системам, для которых вам предоставлены разрешения на доступ и тестирование.

Нарушение авторских прав

Закон об авторском праве в цифровую эпоху (**Digital Millennium Copyright Act** – **DMCA**), это закон Соединенных Штатов, объявляющий противозаконным выпуск или распространение любых технологий, устройств или услуг, в обход средств контроля доступа к материалам, защищенных авторскими правами. Он также объявляет противозаконными, действия по обходу таких средств, даже если это не является нарушением авторских прав.

Исследователи, опубликовавшие код PoC, сталкивались с проблемами из-за DMCA. Например в 1999 году, DMCA стал орудием против хакеров, разместивших копии программы дешифрования **DeCSS**, применяемой для расшифровки DVD. В конечном итоге, высшие судебные инстанции решили, что распространение программы DeCSS и кода, было в конечном счете разновидностью защищенной законом в Соединенных Штатах свободы слова, однако это произошло после массы ходатайств, поданных ответчиками.

Во избежание обвинений в нарушении авторских прав, вам следует публиковать только тот код PoC, который вы написали лично. Если вы используете чей-либо код – в начале убедитесь, что получили разрешение.

Использование уязвимостей в преступных целях

Здесь все просто – не делайте этого. Если вы нашли и использовали уязвимость в преступных целях, а затем раскрыли информацию о ней, это называется “и рыбку съесть и на качелях покататься“, и это плохая идея. Когда власти или вендор неизбежно это выяснят, для вас это скорее всего закончится разбирательствами, как по уголовным, так и по гражданским делам, возбужденным против вас. Этот тип сценария, происходит не часто, однако когда это случается, это делает вам антирекламу, из-за вопиющего злоупотребления силами исследователя.

Клевета

Клевета, это осознанное или неосознанное высказывание ложных утверждений о людях или организациях, которые могут повредить их репутации. В Соединенных Штатах, гарантии свободы слова, обеспечиваются первой поправкой к конституции. И все же если то, что вы говорите – является ложью и подрыватет репутацию другой стороны, эта сторона может подать на вас в суд за клевету. В Соединенных Штатах, выделяют две разновидности клеветы: *клевета* (письменные утверждения) и *злословие* (устные утверждения).

Клевета, обычно относится к письменным заявлениям, в которых исследователь ошибочно утверждает, что определенный продукт, является уязвимым, в то время, как никаких фактов подтверждающих это – в публикации исследователя нет. Такие публичные заявления, в результате приводят к большой потере времени и сил части пользователей, которые следили за отчетом исследователя, только чтобы выяснить, что никакой уязвимости нет. В конце концов, недостаточно проверенные уязвимости, становятся причиной множества проблем для исследователей, так что, когда публикуете ваши исследования, будьте внимательны и предоставляйте четкие факты, подтверждающие ваши заявления. Избегайте спекуляций, понятно излагайте, в чем заключается ваше мнение и какие факты его подтверждают. Компании, которые стремятся преследовать вас по закону, за недостоверные утверждения, могут ссылаться , что вы сообщили клевету, которая послужила причиной бессмысленных трудозатрат и привела к убыткам.

Исследователи, получают судебные иски за клевету, после ложных или бездоказательных утверждений о компаниях или их продуктах. Однако на продуктах, клевета не заканчивается. Вспомним обсуждение ситуации с Talkspace в начале главы, где исследователь сделал определенные утверждения о том, что клиенты компании были обокрадены. Это не отметает, его подтвержденных фактами заявлений, о возможности применить промокоды для получения доступа к сервисам, без соответствующей оплаты. В то время как, сказанное им о Talkspace, могло бы быть правдой, но не имело под собой никаких фактов и следовательно было, чисто гипотетическим.

Во избежание обвинений в клевете, вам следует делать, только такие утверждения, которые подтвержденны фактами. Будьте на 100% уверены в том, что вы собираетесь сказать, до того как скажете и будьте готовы подтвердить это фактами. Не болтайте о том, о чем не знаете. Придерживайтесь фактов.

Непредоставление уведомления

Независимая публикация любой уязвимости без предупреждения, весьма сомнительна и мы явно и неоднократно утверждали в этой книге до сих пор, что вы должны не раз и не два уведомить поставщика ПО, перед тем, как публично поделиться информацией об уязвимостях в их продуктах. Повторим это: не раскрывайте информацию об уязвимостях непосредственно, без предварительного

уведомления вендора. Как у исследователя, у вас есть право на ошибку по невнимательности, но вы должны давать компании разумное время на закрытие уязвимости, перед тем как донести ее до широкой публики. Обеспечение, ясности, полезности и открытости для общения с поставщиком ПО, поможет вам избежать любых возможных юридических последствий в будущем, и создать лучшую репутацию для всех будущих исследователей безопасности, сотрудничающих с этой компанией.

Как независимо публиковать уязвимости

Есть несколько способов, публиковаться независимо. Самый простой способ, это написать доклад, который включает результаты ваших исследований и ваше (изложенное литературно) взаимодействие с поставщиком ПО, предложить ему исправления и разместить это в сети. Вы можете использовать личный блог или веб-сайт, позволяющий людям выкладывать свой контент, например такой, как Medium. Доклад необязательно ограничивать, непосредственно главными результатами исследований, исследователи могу также решить, выложить демо-версии PoC, оригинальный код эксплоита и что угодно, что с их точки зрения могло оказаться полезным читателям.

Если у вас нет личного блога, где вы могли бы публиковаться, мы советуем задействовать GitHub Pages. GitHub Pages, это бесплатный и доступный способ установить независимый блог, поверх репозитория по разработке ПО. Затем, используя простой язык шаблонов, вроде Jekyll, вы сможете собрать нечто, выглядящее "вылизано" и профессионально за отличную цену, аж в ноль баксов:

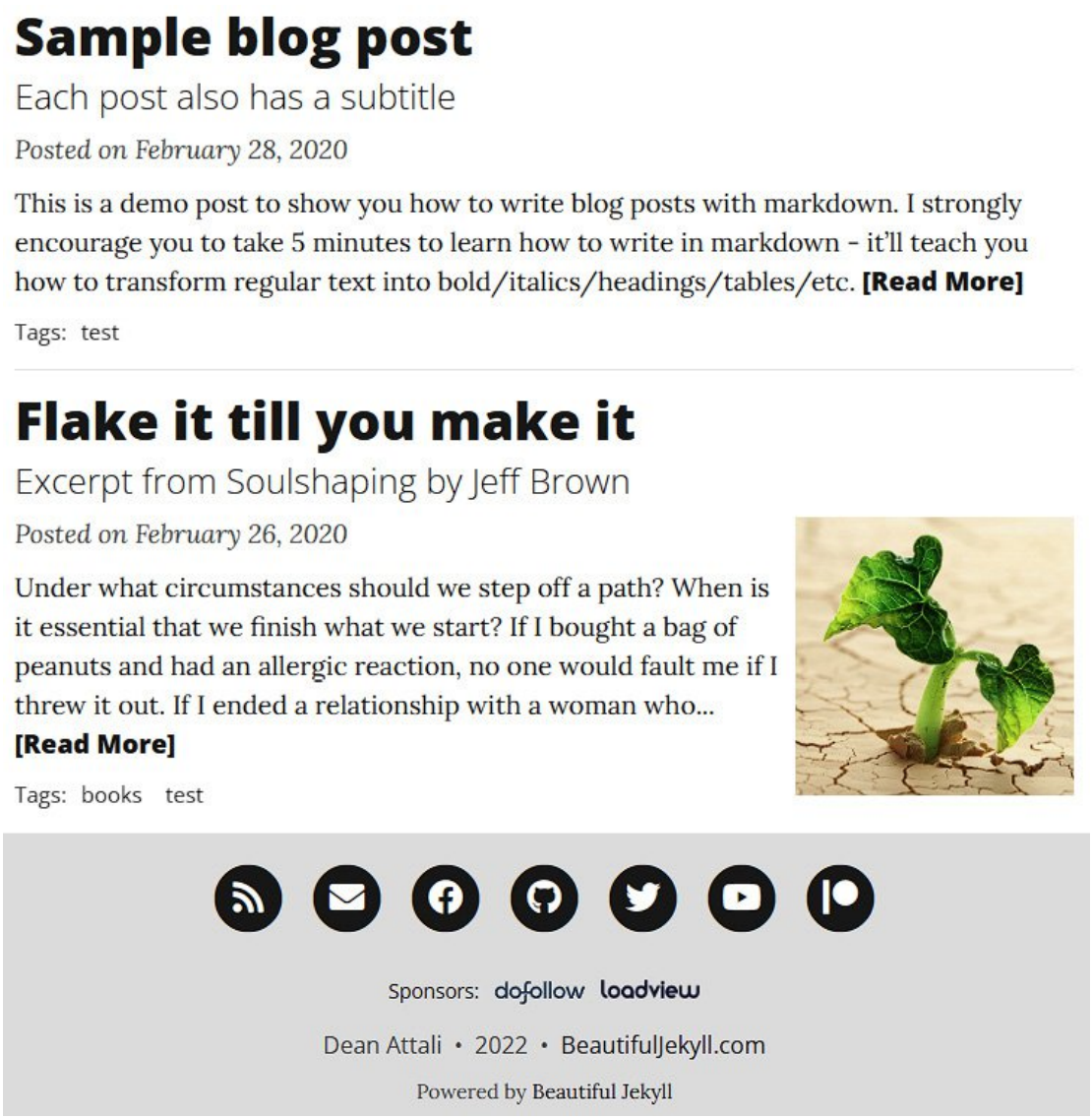


Рис. 7.1 – Пример страницы GitHub, сформированной Jekyll

Вы можете выбирать из примерно 1000 с хвостиком тем, имеющихся на GitHub, с воспользовавшись вот этим топиком Jekyll-theme: https://github.com/topics/jekyll-theme.

Еще одним популярным вариантом, для исследователей, которые подходят к теме независимой публикации, является отправка отчетов через **список почтовой рассылки** по информационной безопасности или аналогичные форумы. К сожалению, эти средства сейчас не так полезны, как когда-то были и некоторые из культовых списков рассылки, прекратили существование. Например, **Bugtraq** – один из наиболее популярных списков рассылки, прекратил работу после 27 лет существования, в 2021 году после некоторых изменений на платформе. Однако, несмотря на снижение популярности этого способа, он продолжает оставаться ценным источником, касательно не обработанных цензурой комментариев об исследованиях безопасности. Вероятно некоторые из лучших исследований, до сих пор, находятся на таких форумах, призывающих исследователей к обсуждению уязвимостей.

Лучший из доступных ресурсов, чтобы познакомиться со списками рассылки по информационной безопасности, это **SecLists** (https://seclists.org). Это веб-сайт, созданный Гордоном Лайоном, создателем и мейнтейненром популярного сетевого сканера Nmap. Просматривая этот сайт, вы увидите ряд списков рассылки, их архивы и информацию о списках, как видно на следующем скриншоте:

Рис. 7.2 – SecLists.org, содержит несколько списков рассылки, на которые вы можете подписаться и почитать

Если вы заинтересованны в использовании списков рассылки, таких как найденные на SecLists.org, уделите немного времени, тем из них, которые вам интересны и как следует разберитесь, как с ними обращаться. Вы могли бы найти тот, который наиболее подходит именно для вашего случая, лучше других.

Примечание

Списки рассылки по информационной безопасности, часто имеют правила, которых потребуется придерживаться, чтобы быть уверенным, что ваше исследование дойдет до подписчиков. Хотя, эти списки рассылки, обычно не слишком строго отслеживаются и модерируются, они все же управляются администраторами, которые обычно решают, что можно, а что нельзя публиковать в списке. Убедитесь, что прочитали правила, перед тем, как что-то отправлять.

Наконец, одним из жизнеспособных вариантов может быть привлечение средств массовой информации. К сожалению, это часто выглядит, как последнее средство, когда журналисты, которым нужен сюжет, могут ловко сделать из этого сенсационный материал. Одним из способов преодолеть возможное отсутствие интереса со стороны ведущих СМИ, является выход на прямую работу с технологически-ориентированными медиа-компаниями, такими как TechCrunch, которые напечатали историю, с которой мы начали эту главу. Другие представители СМИ, такие как Ars Technica, Wired, Dark Reading, Tom's Hardware и The Verge, также являются отличными источниками, которые регулярно публикуют сюжеты об исследователях и их работе. Нет ничего необычного в том, что материал в первую очередь попадает на один из этих ресурсов, а если он достаточно интересен или описанные в нем события, интересны широкой публике, ведущие СМИ – растиражируют эту историю.

Теперь, лучше понимая, где может быть опубликована информация об уязвимостях, давайте завершим обсуждение процесса публикации, чек-листом, который вам следует дважды перепроверить, перед нажатием кнопки **Отправить**.

Чек-лист перед публикацией

Мы уделили весьма много времени, обсуждению различных способов публикации информации об уязвимостях. Кроме того, мы сделали это, чтобы помочь вам понять, что если вы публикуете информацию об уязвимостях независимо, то неизбежно подвергаетесь риску. Если этот риск для вас приемлем, мы надеемся, что вы последуете этим советам, если решите опубликовать свое исследование. Чтобы помочь свести все это воедино, в виде более удобном для практического применения, мы составили чек-лист, который поможет ограничить возможность возникновения любых неприятностей, с которыми вы могли бы столкнуться. Хотя, соблюдение этих рекомендаций поможет избежать рисков, связанных с независимой публикацией уязвимостей, это не защитит вас полностью, даже если все, что вы выложите на публику, подтверждается фактами и обосновывается множеством доказательств. Возвращаясь к нашей аналогии с неодобрительным обзором ресторана, владелец бизнеса может добиваться преследования вас в судебном порядке, какую бы причину он ни выбрал, точно также это применимо и к любым исследованиям, которые вы публикуете про компанию или ее продукты. Это не подразумевает, что они обязательно выиграют, но скорее всего последствием судебной тяжбы, будет заметное облегчение вашего кошелька. Итак, с учетом этих оговорок, получился такой чек-лист:

- Если это возможно, поищите любые контакты, по которым вы или ваша компания, можете обратиться через программу раскрытия информации или прямо к поставщику ПО.
- Дважды перепроверьте, что компания или какой-нибудь исследователь уже не опубликовал информацию об уязвимости.

- Информируйте вендора о своих намерениях, касательно публикации, согласовав с ним четкий график.
- Раз уж вы пишете, проверяйте факты. Юмор и подколки, это не проблема, однако если ваше раскрытие информации содержит бездоказательные предположения о компании, их взаимодействии с вами или уязвимостей с ними, вы в кратчайшие сроки, окажетесь с предписанием от юристов в руках.
- Удалите из результатов исследования, любую конфиденциальную информацию о компании или их пользователях, если вдруг ее узнали.
- Удалите из раскрываемой вами информации, любой код или в особенности лицензионный контент.
- Благодарите всегда, когда это уместно. Если вам помогли другие исследователи или блоги, не забудьте упомянуть это и поблагодарить иследователей за их вклад.
- Имейте кого-то, кому вы доверите проверку корректности своей работы, на предмет грамматических ошибок, возможной предвзятости, недостатка доказательств и правдивости приведенных фактов.
- Опубликуйте раскрытие информации об уязвимости.
- На свое усмотрение, можете дать вендору знать, где он может прочесть ваше раскрытие информации.

Если вы не уверенны, как сделать свое первое раскрытие информации об уязвимости в виде поста в блоге, мы предоставим вам шаблон в *Главе 10, Шаблоны, ресурсы и заключительные наставления*, который вы сможете использовать, чтобы начать свои независимые публикации, раскрытий информации об уязвимостях.

Итак, это завершает главу о независимой публикации. Давайте повторим, что мы в ней прошли.

Итоги главы

В этой главе, мы рассмотрели тему, о независимой публикации раскрытия информации об уязвимостях. В первую очередь мы рассмотрели, где она находится в жизненном цикле уязвимости и почему вы могли бы захотеть, поделиться информацией с другими исследователями, и что более важно с общественностью – именно этим способом. Затем мы обсудили различные плюсы и минусы, присущие таким раскрытиям информации. После этого, мы исследовали шесть ключевых моментов, которые как правило, приносят исследователям проблемы, когда они решают публиковаться самостоятельно и поделились ключевыми стратегиями по обходу этих препятствий. Потом, мы сжато изучили СМИ, подходящие для публикации исследований и в заключение, рассмотрели чек-лист, который вы можете скорректировать под себя и использовать, будем надеяться для ограничения своих рисков, когда вы решите публично поделиться информацией об уязвимостях. На всем протяжении этой книги, мы делились реальными историями о результатах работы исследователей и раскрытии компаниям информации об угрозах безопасности, что является эффективным способом научиться делать это самостоятельно. В следующей главе, исследуем процесс раскрытия информации значительно глубже (иногда обезличивая контент) по сравнению с тем, что мы уже узнали и извлечем квинтэссенцию из успехов и неудач.

Дополнительные материалы

- Talkspace угрожает исследователю судебным иском, после баг-репорта, TechCrunch – https://techcrunch.com/2020/03/09/talkspace-cease-desist/
- GitHub Pages – https://pages.github.com/

Часть 3 – Изучение конкретных примеров, ресурсы исследователя и ресурсы вендоров

Цель этого раздела – предоставление средств для раскрытия информации об уязвимостях. Это включает информацию по конкретным примерам, в которых реальные исследователи достигли успеха в своей работе, или потерпели поражение. Затем мы поменяемся ролями и проведем беседу с вендорами о взаимодействии с исследователями безопасности. Завершим мы этот раздел полезными шаблонами и средствами управления исследованием.

Эта часть, включает следующие главы:

- *Глава 8, Изучаем примеры из жизни – подробно разбираемся в удачных (и неудачных) отчетах об исследовании*
- *Глава 9, Взаимодействие с исследователями в области безопасности – руководство для вендоров*
- *Глава 10, Шаблоны, ресурсы и заключительные наставления*

Глава 8. Изучаем примеры из жизни – подробно разбираемся в удачных (и неудачных) отчетах об исследовании

На протяжении этой книги, мы обсуждали истории исследователей безопасности, находящих и раскрывающих информацию об уязвимостях. Мы также говорили о том, что эти истории часто происходят по похожей схеме: кто-то находит изъяны в безопасности, рассказывает об этом компании, ответственной за программное обеспечение и затем (если все идет хорошо) – компания выпускает патч, устраняющий проблему. В других случаях, бывают сложности. Например, исследователь и компания могли не достичь согласия, является ли дефект проблемой. Компания могла не реагировать, на раскрытие информации исследователем. Или, как в некоторых наиболее известных случаях компания могла попытаться, заставить исследователя замолчать законными средствами.

Прекрасно, когда раскрытие информации об уязвимостях, получилось как надо и все быстро и аккуратно пропатчено. Тем не менее, конфликты между исследователями и поставщиками программного обеспечения распространены, даже при высочайшей компетентности и вложениях в безопасность. Эти конфликты, часто переходят из сценария в сценарий, которые приводят к трате ресурсов, таких как время, деньги и ваше терпение. Это ставит под угрозу плодотворные отношения между компаниями и исследователями безопасности. Эта глава нацелена помочь вам узнать об этих проблемам, рассказывая истории об исследователях и сложностях в работе, с которыми они сталкивались в прошлом. В заключение, мы обдумаем, чему мы могли бы научиться из этих случаев и обсудим возможные усовершенствования, которые можно применить для будушего взаимодействия с вендорами.

В связи с секретным характером этих взаимодействий, мы сохраним инкогнито всех задействованных исследователей и компаний. Эти истории невозможно рассмотреть во всех подробностях – по разным причинам, некоторые из которых будут пояснены в нашем обзоре. Тем не менее, изучая эти уязвимости, мы должны лучше разобраться, каким образом исследовали передавали информацию и получали ответы от вендоров, возможные проблемы при раскрытии информации и методы, которыми мы можем добиться наилучших результатов наших исследований.

Резюмируя содержание этой главы, мы будем рассматривать следующее:

Мы рассмотрим пять примеров из жизни, по исследованию уязвимостей и раскрытию информации

Мы подробно разберем уроки, изученные нами в каждом случае

В заключение, мы кратко обсудим подходящие ситуации, по применению этой информации для совершенствования наших будущих исследований

Эти примеры из жизни, будут представленны в следующих основных темах этой главы:

- Пример 1 – ну что, приехали?
- Пример 2 – условия договора
- Пример 3 – несговорчивые клиенты
- Пример 4 – крупные корпорации и вы
- Пример 5 – я хотел бы поговорить с вашим менеджером

Пример 1 – ну что, приехали?

В этом примере, мы рассмотрим поставщиков программного обеспечения, которые не применяют политик ответственного раскрытия информации, затягивают раскрытие и как одна малюсенькая ошибка в вашей публикации, может задержать ее выход на месяцы.

В 2020 году, некий исследователь уязвимостей был нанят одной компанией, для проведения обычного исследования и составления отчета о безопасности, которые они часто получают. Это программное обеспечение, состояло из веб-приложения, толстого клиента и серверных приложений. В процессе изучения безопасности приложения, исследователь нашел несколько уникальных уязвимостей, о которых необходимо было сообщить вендору, создавшему и обслуживающему этот продукт. Некоторые их этих уязвимостей, были оценены, как имеющие высокую степень угрозы, а родительская компания, запросившая отчет по безопасности – была заинтересованной стороной.

Исследователь подготовил отчет о результатах своих исследований и послал его своему нанимателю. Наниматель, в свою очередь поделился этой информацией с вендором, создавшем это ПО и передал его уже своему нанимателю, тесно связанному с вендором. Это была компания-мультимиллиардер, с программой ответственного раскрытия информации и отделом безопасности, ответственным за отслеживание уязвимостей. Исследователь безопасности, ознакомился с их программой и поделился информацией об уязвимости, согласно требований политики ответственного раскрытия информации.

Вендор, начал говорить и компании исследователя, и самому исследователю, что уязвимости могли бы получить высший приоритет и быть исправлены. Обе стороны договорились, в ближайшие две недели определить дальнейшие шаги, после того как группа разработчиков выберет время и проанализирует продукт. Недели прошли и исследователь, поинтересовался у вендора, как движется дело. Вендор сообщил, что не знает, когда уязвимости могли бы быть исправлены, но что он оповестит исследователя и связанные контакты.

Прошло несколько месяцев с того времени, как исследователь обратился к вендору за обновлениями. Запрос некоторое время оставался без ответа, поэтому исследователь попробовал еще раз. После многочисленных попыток, уже в течение года с момента первоначального раскрытия информации, поставщик ПО ответил и сообщил, что продукт более не обновляется и не поддерживается. Уязвимости никогда не будут исправлены.

Исследователя, такой результат разочаровал. Исследователь решил предать огласке результаты своих исследований, для повышения осведомленности о рисках, связанных с использованием этого ПО, которое активно продавалось, хотя уже не поддерживалось.

Уязвимости были подходящими для публикации CVE, в соответствии с правилами обозначенными MITRE и исследователь начал рассматривать доступные варианты. После сбора всей требуемой информации об уязвимости, исследователь подал заявку на шесть уязвимостей, которые MITRE зарезервировала за исследователем, через месяц после запроса.

Исследователь уведомил поставщика программного обеспечения о своих намерениях опубликовать CVE и ознакомил их с подробностями. Вендор, запросил дополнительное время на решение этих проблем, он заявил, что они не хотели этим заниматься, просто из-за отсутствия установленного срока или графика. В качестве демонстрации добропорядочности, исследователь предоставил 3 дополнительных месяца до публикации. Никогда не было, и вот опять – после контакта с вендором и ожидания ответа, исследователь не получил никаких обновлений. Он опубликовал пост в открытом блоге о результатах своего исследования и затем запросил выпуск CVE у MITRE.

К удивлению исследователя, его первоначальный запрос на публикацию, был отклонен MITRE. Вместо этого, MITRE отправила ему шаблонный ответ, в котором говорилось следующее:

CVE Team, требуется общедоступная ссылка на URL, содержащий минимально необходимую информацию о типе уязвимости, включающем те уязвимости, которые вы желаете раскрыть. Если у вас ее нет, быстрее всего будет, создать ее самостоятельно в виде сообщения в форме подтверждения информации, содержащей минимально необходимые данные по адресу https://gist.github.com, доступно изложив суть. Вы можете использовать и другие сторонние ресурсы

(напр. какой-то блог или архив списков рассылки https://seclists.org/fulldisclosure или https://pastebin.com).

Пожалуйста, не стесняйтесь связываться с CVE Team, в ответ на это письмо, если у вас есть любые вопросы или более детальная информация.

Исследователь отправил им озадаченное электронное письмо, задавая вопросы о том, зачем им могло потребоваться опубликованное раскрытие информации, ведь оно уже было предоставлено ранее. По прошествии нескольких недель исследователь, не получив ответа от MITRE, начал писать новый запрос. Через несколько дней, запрос был отправлен обратно с утверждением, что отстутствует доступная ссылка на материалы и информацию об исследователе.

Исследователь написал снова, спрашивая, что он мог сделать не так. В итоге, спустя 5 недель, MITRE отписалась шаблонным ответом, снова указав, что они не получали запрошенных материалов:

Благодарим вас за подачу заявки на публикацию [номера CVE], однако общедоступная ссылка на описание подробностей уязвимости, все еще необходима. Так как, она не была своевременно предоставлена, запрос на обслуживание [номер запроса] – отклоняется. Пожалуйста, не стесняйтесь подать его еще раз, как только захотите.

Пожалуйста, не стесняйтесь связываться с CVE Team, в ответ на это письмо, если у вас есть любые вопросы или более детальная информация.

Исследователь был в замешательстве. Он консультировался с коллегами по индустрии, которые считали, что могло бы помочь открытие отдельного запроса на публикацию, для каждой из уязвимостей. Тем не менее, он подумал, что отправить все сразу (как допускается, в качестве опции на сайте), могло быть более эффективным методом решения этой задачи, так как MITRE не отвечала на письма. Тогда, вместо отправки всех пяти уязвимостей, которые он изначально пытался отправить одним запросом на публикацию – он сделал 5 отдельных запросов на публикацию, по одному на каждую уязвимость.

После такой попытки, исследователь получил четыре подтверждения и один отказ. Пересмотрев отказ, исследователь отметил в материалах своего блога, что отклоненная уязвимость имела некорректно присвоенный CVE ID. В результате он перепроверил опубликованный материал. Затем он повторно отправил отчет об уязвимости в MITRE, завершив процесс, который от начала, до полного завершения – занял почти два года.

Извлеченный урок

Этот пример из жизни, иллюстрирует сложности, с которыми столкнулись исследователи, работая с вендорами, которые казались способными и готовыми к сотрудничеству с исследователями, однако не сдержали своих обещаний по поддержанию безопасности своих продуктов.

В этом примере исследователь, в конце концов опубликовал свои уязвимости с помощью MITRE после того, как вендор долго и нудно водил исследователя за нос. Задумаемся на минутку, над важностью того факта, насколько легко исследователь безопасности может оказаться в растерянности и просто на все плюнуть, когда столкнулся с чередой непрекращающихся препятствий. Два года, это слишком долго, чтобы сосредоточиваться на таком маленьком наборе уязвимостей, выявленных всего за несколько дней исследования.

Кроме того, это подчеркивает некоторые недостатки программы CVE, вскрывшиеся при принятии отчетов. С момента написания этих строк, ежегодное число сообщений об уязвимостях выросло на пугающие величины, за последние несколько лет. Коллектив, работающий над этим проектом, постоянно засыпан запросами – в 2021 году, исследователями публиковалось в среднем 56

уязвимостей в день (это не считая запросов). В программе CVE, используются почтовые боты, там где это возможно, но иногда, как выяснилось на нашем примере, они оказываются не слишком полезны или информативны.

Возможные усовершенствования

Исследователь, поступил правильно, сообщив компании об уязвимости, а возможным улучшением могло бы быть, если бы исследователь задействовал возможности арбитража уязвимостей. Кроме того, взаимодействие с арбитром поможет определить сферу ответственности поставщика программного обеспечения, который не спешит выполнять обещанное или задерживает график. Если вендор отказался работать с арбитром, это достаточная причина, раскрыть информацию об уязвимости не растягивая процесс на два года.

В заключение, исследователю было бы неплохо рассмотреть еще одно усовершенствование: разделить результаты исследований, по отдельным раскрытиям информации. Команда MITRE очень занята и автоматизированные способы управления некоторыми запросами иногда приводит к накладкам, а повторная отправка, на первый взгляд занимающая больше времени – доставит ваш отчет во входящие.

Теперь, давайте рассмотрим наш следующий пример из жизни, в котором мы исследуем, как контракты ограничивают исследователей.

Пример 2 – условия договора

В этом примере, мы изучим, как поставщики программного обеспечения, составляют договоры и **соглашения о неразглашении** (**NDA**), которые иногда ограничивают исследователей в возможностях там, где исследователи этого не ожидают.

В 2021 году, к одному исследователю обратились за оценкой состояния безопасности некой платформы аназиза данных, используемой компанией, в которой он работал. Продукт был лицензирован и обслуживался поставщиком ПО, продавшим его организации.

Когда исследователя пригласили для изучения приложения, бизнес уже начал использовать этот продукт. Он был развернут без промежуточной среды, непосредственно в интернете, подразумевая, что любые изменения или задачи сделанные на платформе – происходят "на лету". Исследователь был сильно заинтересован в том, чтобы не проводить тестов, которые могли нарушить функционирование платформы. Поэтому он решил, что мог бы протестировать приложение в каком-либо окружении, настроенном в соответствии с инструкциями из документации поставщика ПО. Исследователь создал свое окружение, с пробной версией ПО, которую он нашел на веб-сайте вендора. Исследователь отлично справился с изучением действующих задач и клонировал их в свое тестовое окружение, после длительных консультаций с пользователями работодателя.

В ходе анализа безопасности, исследователь нашел несколько уязвимостей, которые позволяли использовать ПО для получения удаленного доступа к машине, на которой оно было установлено. Хотя эти уязвимости, были ограничены пользователями прошедшими проверку подлинности, сама проверка подлинности была проблемой. Вендором использовались логины и пароли по умолчанию, и это вкупе с одно-факторной аутентификацией. Это было особенно проблематичным, из-за того, что это программное обеспечение было создано для работы через интернет, как общедоступное веб-приложение.

Исследователь изначально сделал доклад для внутреннего пользования, владельцам бизнеса, применявшим это программное обеспечение, однако в первую очередь ПО обслуживал вендор. В связи с этим, исследователь следующим этапом, должен был связаться с поставщиком ПО для решения

проблем. Исследователь добыл контакты техподдержки вендора у владельцев бизнеса и связался с ним. Поставщик программного обеспечения, сначала ответил на письмо исследователя, но оказался незаинтересован в результатах исследований. В исследователь добивался от вендора сотрудничества течение нескольких месяцев, но получил минимум информации с его стороны о подтверждении, что вендор получил отчет об уязвимости и изучает его.

В конце концов, после нескольких запросов, поставщик ПО рассказал исследователю, что он не смотрел его работу по уязвимости. Он объяснил это тем, что вряд ли кто-то мог бы использовать их ПО эксплуатируя уязвимости, потому что их пользователи обычно не были хакерами. Исследователь, нашел это заявление идиотским и повторил, что уязвимость может привести к компрометации всех взаимосвязанных систем. Хотя хакеры, могли и не быть пользователями системы изначально, они могли быть однозначно заинтересованы данными содержащимися в системе в особенности видя, что они открыто доступны из интернета.

После неудавшегося общения с вендором, исследователь решил, что из-за расхождения во мнениях, нужно использовать средства арбитража уязвимостей, он не думал, что сможет решить проблему самостоятельно. Исследователь открыл задачу в CERT-программе VINCE и координатор арбитражного сервиса связался с вендором, после предоставления исследователем исходных данных. Как только арбитр обратился к поставщику ПО, тот сразу начал показывать, что не собирается с ним разговаривать. Они сразу подняли этот вопрос с компанией, где работал исследователь.

Исследователю было неизвестно, что договор обслуживания, изначально подписанный при передаче программного обеспечения, включал формулировку однозначно утверждающую, что уязвимости связанные с этим ПО – не должны разглашаться никаким третьим лицам. Хотя само по себе ПО, вы могли загрузить, установить и использовать, договор подписанный компанией, которая приобрела права на использование программного обеспечения, по сути сделал невозможным для общественности узнать об уязвимостях, без нарушения контракта между огранизацией исследователя и поставщиком ПО.

Вендора, настолько не обрадовало то, что исследователь захотел раскрыть информацию об уязвимостях, что он начал распространять сообщения, в которых предупредил несколько аффилированных групп о нарушении договора из-за раскрытия результатов исследования. В своих сообщениях, он утверждал, что любая публикация или распространение сведений о уязвимости – будет считаться нарушением условий договора, с которыми обе стороны выразили согласие. Для исправления этого нарушения, компания потребовала чтобы упоминания об уязвимости были удалены из CERT и в дальнейшем не производилось раскрытия информации или обсуждения этих уязвимостей. Исследователь был уведомлен об удалении информации из CERT, своим работодателем.

Уязвимости не были опубликованы в широком доступе и скорее всего, до сих пор присутствуют в программном обеспечении, которое крупные корпорации используют для доступа к своим данным. В результате, знания об этих уязвимостях, скорее всего ограничатся теми, кто нашел их изначально, пока другой исследователь не обнаружит и не сообщит о них.

Это тот случай, где соображения этики, вступили в противоречие с интересами бизнеса. Исследователь оказался в затруднительном положении, ведь его работодатель согласился с условиями, запрещающими раскрывать информацию об уязвимостях. С одной стороны, он хотел защитить людей, которые могли быть подвержены этим уязвимостям, однако с другой стороны – его полномочия были ограничены неким соглашением, подписанным его работодателем. В конце концов, исследователь подчинился и не стал раскрывать информацию об уязвимостях.

Извлеченный урок

Этот пример, прекрасно иллюстрирует, как информация об уязвимостях и ее публикация, может быть задушена NDA и договорами, о которых вы, как исследователь – можете быть не осведомлены. К сожалению, приходится нередко слышать, что некоторые корпорации поступают, как в описанном случае, чтобы извлекать прибыль за лицензирование программного обеспечения.

Каким бы тревожным ни было это открытие для пользователей, это реальность, с которой часто сталкиваются исследователи. К примеру на конференциях по информационной безопасности, нет тех, кто не знаком с препятствиями раскрытию информации об исследовании. Бывало множество случаев, когда исследователи не могли представить результаты исследования на конференции. Например, поставщик ПО препятствует выпуску результатов исследования, через такое средство, как принуждение организаторов конференции, прекратить выступление их докладчика и угрожая сторонам судебной тяжбой. В редких случаях, вендоры даже физически задерживали исследователей в гостиничных номерах, чтобы воспрепятствовать им рассказать о результатах своего исследования.

К счастью принуждение, с которым столкнулся наш исследователь, не было физическим. Угрозы судебного преследования утихли, после того как исследователь отозвал, раскрытую им информацию из CERT. При этом, исследование не было опубликовано, а дефекты безопасности найденные в этом ПО, скорее всего существуют и сегодня. Такова ужасная действительность и мы одни из тех, кто прилагает усилия, чтобы это изменить.

Возможные усовершенствования

Как же все-таки мало исследователей, вышедших за пределы указанные в договорах, подготовились к возможным последствиям. Это могло бы быть возможным для исследователей, опубликовавших информацию об уязвимостях, как независимый исследователь, сохраняя инкогнито или используя псевдоним. Тем не менее, в этой стратегии есть изъяны. Будет справедливым сказать, что исследователь изначально действовал добропорядочно и раскрыл информацию об уязвимости. Однако, если поставщик программного обеспечения, предпочел бы ничего не делать, любые раскрытия информации после такого взаимодействия, могли быть связаны с ее инициатором, даже если информация была раскрыта анонимно. Для некоторых исследователей, это провальный сценарий, угрожающий потерей работы.

Кроме той защиты пользователей, которую по видимому должно предоставлять правительство, не существует никакой защиты от того, что покупатели ПО, иногда соглашаются с условиями договоров, которые они могут даже не понимать. Было бы полезно обсудить с партнерами и попытаться включить в новые договоры получения прав на использование программного обеспечения – пункт о раскрытии информации об уязвимостях. Этот пункт, определит основные права компании для ситуации, когда в продукте найдена уязвимость, а поставщик ПО, решает ничего не предпринимать. Если вендор не расположен, принимать такие условия, это может указывать на то, как он в будущем может реагировать на доклады по безопасности.

Не забывая об этом примере, давайте рассмотрим следующий случай в нашем списке, который описывает последствия плохой коммуникации между исследователями и поставщиками программного обеспечения.

Пример 3 – несговорчивые клиенты

Давайте сейчас рассмотрим проблемы касающиеся уязвимостей, найденных ранее в старших версиях кода, плохой коммуникации и того, как публичное раскрытие информации, может послужить причиной быстрого исправления со стороны вендора, но может подвергнуть угрозе пользователей ПО.

В конце 2017 года, один исследователь безопасности нацелился на популярный программный комплекс, который организации использовали для облегчения онлайн коммерции. Исследователь специализировался на реверс-инжиниринге патчей безопасности и поиске изъянов в исправлениях. Он начал свое исследование с поиска публично известных уязвимостей, исправленных на этой платформе.

Сосредоточившись на одной недавней уязвимости, которую поставщик ПО, как утверждалось исправил, исследователь подумал, что там могли быть ошибки в реализации их первоначальных выводов. После более тщательного изучения он выяснил, что патч технически предназначен защитить от одного вектора атаки, оставляя еще один не закрытым. Он решил попытаться эксплуатировать этот неполноценный патч, который если повезет – позволит исслезователю захватить аккаунты пользователей и похитить информацию, такую как данные кредитных карт.

После нескольких часов экспериментов, он добился успеха в своих попытках и смог эксплуатировать уязвимость. Исследователь был уверен, что эта уязвимость будет представлять ценность для вендора и надеялся получить компенсацию за свой труд. Этот исследователь нашел подходящие контакты поставщика ПО и начал налаживать связь. При первой попытке общения, исследователь не стал делиться всеми подробностями и умышленно не конкретизировал, надеясь уболтать вендора выплатить исследователю, какое-то вознаграждение. Этот подход, неожиданно привел к обратному результату.

Во-первых, поставщик ПО не поверил исследователю, а подумал, что исследователь пытался вымогать у него деньги, за уязвимость, которую как думал вендор – они успешно исправили. Вендор имел предыдущий опыт такого рода взаимодействий с исследователями и не хотел стать жертвой, возможного преступного замысла. Исследователь и поставщик ПО, потратили несколько дней переписываясь по электронной почте, обмениваясь враждебными выпадами, пока вендор в конце концов, не рассказал исследователю, что не может воспроизвести проблему с той информацией, что ему предоставлена и следовательно он не может работать над исправлением или выплачивать исследователю вознаграждение.

К несчастью для нашего исследователя, в своем общении с вендором он предоставил технически не завершенный proof-of-concept эксплоит. Это сделало не возможным проверку вендором, заявлений исследователя и в результате привело к недопониманию между вендором и исследователем. Хуже всего то, что исследователь начал это исследование с запроса финансового вознаграждения. В результате поставщик программного обеспечения, не воспринял исследователя всерьез и имел достаточную причину отвергнуть заявления, сделанные исследователем.

В этом случае, исследователь все-таки не думал, что ответ поставщика был приемлемым, поэтому он решил обнародовать результаты своих исследований, непосредственно. В первую очередь, он опубликовал в блоге подробности об уязвимости, как ее можно эксплуатировать и предоставил захватывающее видео, наглядно демонстрирующее эксплуатацию в действии. Затем исследователь написал в технически-ориентированные агрегаторы новостей, чтобы информировать их о серьезности проблемы и о том, насколько вендору на нее плевать.

История получила некоторые шансы на успех и была распространена по нескольким известным техническим блогам, а также новостным сайтам. К сожалению, поставщик программного обеспечения, оказался осведомлен об статьях в СМИ так, как журналисты обращались к нему за комментариями.

В процессе изучения поста в блоге исследователя, вендор обнаружил, что публикация содержит все подробности, которых он не получил общаясь с исследователем – это позволило ему воспроизвести проблему. В результате, поставщик ПО быстро выпустил патч и убедил своих клиентов, установить его настолько быстро, насколько это возможно. Тем не менее, работа исследователя и

подробная информация по использованию уязвимости, закончилась компрометацией нескольких предприятий использовавших это приложение и привела к неисчислимым убыткам.

Извлеченный урок

Этот пример из жизни, является прекрасной иллюстрацией того, как может прерваться общение, с крайне невыгодными для обеих сторон последствиями. Этот мелкий, бессмысленный конфликт закончился для конечных пользователей, похищением их данных преступниками, которые взяли подготовленную информацию об эксплоите и тут же применили ее.

Вне всяких сомнений, обеим сторонам следовало получше наладить взаимодействие. К сожалению, исследователь изначально слишком расплывчато и невразумительно обозначил проблему, а поставщик ПО был настроен критически и наделал безосновательных предположений. Если бы и та и другая сторона, имели бы чуть более терпения или взаимопонимания, такого развития событий, можно было не допустить.

Однако, справедливо будет отметить, что исследователь, выбрав момент, чтобы раскрыть информацию об уязвимости, при этом оставил поставщика программного обеспечения в неведении, относительно своих намерений поделиться информацией со СМИ. В итоге, СМИ помогли распространить материал, чем вызвали нездоровый интерес, закончившийся кражей личной информации, вроде номеров кредитных карт, домашнего адреса и паролей. Как бы вы не были разочарованы, этот случай – не пример для подражания, в описанной ситуации исследователь действовал все хуже и хуже в геометрической прогрессии.

Возможные усовершенствования

Важно запомнить, что мы не всегда можем контроллировать, как ведут себя остальные, однако управлять собой, мы в состоянии. Исследователи уже бывали замечены в недружественной, так скажем активности, возможно этот пример поможет закрепить, некоторые из таких представлений. Вероятно исследователь, должен был изъясняться понятнее и предоставить вендору больше подробностей. Вендор же, в свою очередь мог быть бы более расположен к сотрудничеству и отзывчив к предложениям исследователя, вместо изначальной оборонительной позиции и неприятия. В конечном счете, улучшение взаимодействия, могло бы помочь предотвратить усугубление ситуации.

В качестве некоторого улучшения: следует быть безусловно компетентным как исследователь и обязательно соблюдать меры предосторожности, делясь с кем-либо результатами исследований. Предоставление пре-релизов эксплоитов, в то время как не существует известных исправлений, явно безответственная и опасная практика, которой следует избегать. Для раскрытия информации в CVE, не требуется исчерпывающих подробностей.

В конечном счете, в ходе этого сценария ошибочные действия вендора, в целом не были неизбежными, но он поступил бы лучше, тоже занявшись уязвимостями. Многие поставщики программного обеспечения отдают это на аутсорсинг, принимая результаты исследований через программу баг-баунти, они охотно сотрудничают с исследователями и выплачивают вознаграждения за их труд. В качестве предварительного условия, исследователи должны продемонстрировать уязвимость и пояснить в чем ее угроза, до получения вознаграждения. Однако, как обсуждалось в *Главе 7, Независимая публикация уязвимостей*, некоторым исследователям не интересны программы баг-баунти. По хорошему, вендору следовало бы учитывать возможные исключения и давать исследователям пространство для маневра, при принятии результатов их исследований.

В следующем примере мы обсудим, как исследователи могут законтактировать с крупными организациями, которые могут принять и опубликовать информацию о найденных ими уязвимостях.

Пример 4 – крупные корпорации и вы

Этот пример, иллюстрирует работу совместно с крупной корпорацией, оказавшейся "поставщиком" CVE и возможные подводные камни, на которые вы можете наткнуться, работая с крупными организациями.

В 2019 году некий исследователь, неожиданно докопался до уязвимости, которую не искал. Во время изучения другого программного обеспечения на предмет возможных уязвимостей, он отметил одну странность в работе ПО своего исследовательского окружения, которая оказалась весьма серьезной уязвимостью в программном продукте, поддерживаемом компанией-мультимиллиардером.

Исследователь, ответственно раскрыл информацию об уязвимости этой организации. Организация выразила признательность за отчет и быстро начала работать над исправлением. Однако соответственно огромному размеру организации, исправление уязвимости и развертывание обновлений – заняло несколько месяцев. Тем временем, исследователь продолжил искать другие возможные уязвимости, и нашел еще одну.

Он снова, ответственно раскрыл результаты своего исследования организации. Однако, на этот раз вместо выражения благодарности, контора ушла в глухую оборону и начала задавать вопросы, в первую очередь о том, почему исследователь вообще ковыряется в их софте. Они требовали объяснить, с какой целью велось исследование, угрожали судебным преследованием и в конце концов потребовали от исследователя, подписать NDA до того, как они предоставят любую дополнительную информацию об уязвимости или ее исправление.

Исследователь был ошеломлен таким ответом и не понимал, что делать дальше. Он хотел помочь этой организации исправить уязвимость, но не собирался подписывать никаких NDA, которые ограничивали бы его возможности по обсуждению результатов своих исследований.

В этом случае, исследователь начал рассматривать возможные пути, которыми можно было бы опубликовать информацию об уязвимости. Рассматривая варианты, он обнаружил, что эта компания, была **организацией регистрирующей CVE** (**CNA**). Исследователь перечитал, соответствующую политику раскрытия информации CNA и информацию, предоставленную службой безопасности вендора – он отметил, что письмо о раскрытии CVE сильно отличалось от того, что он получил.

С этой новой информацией, исследователь начал копировать всю переписку на адрес электронной почты, относящийся к поставщику ПО. Это значительно улучшило взаимодействие и подключило отдел безопасности этой организации к процессу устранения проблемы, которая первоначально была передана только команде по разработке ПО.

С того момента, как об уязвимости было сообщено в CNA, исследователю нужно было просто ждать выпуска патча и публикации исследования. После того, как информация об уязвимости была опубликована, исследователь был несколько удивлен, отсутствием благодарностей, относившихся к его работе, несмотря на имеющуюся переписку, которая подтверждала взаимодействие с отделом безопасности. При дальнейшем рассмотрении материалы, на которые ссылались в CVE, преуменьшали важность результатов исследования, по сути обесценивая его труд.

К сожалению, независимый исследователь решил смириться с неудачей и не продолжать переписку с вендором или планирование дальнейших исследований с этой компанией. В конце концов, по этой программе не предлагалось денежного вознаграждения, демонстрировалась враждебность, хоть это была и CNA, активно преуменьшалась ценность исследования, безо всякой благодарности за его усилия, сделавшие продукт вендора безопаснее.

Извлеченный урок

Этот пример, отличный образец того, как CNA не должно управлять раскрытием информации об уязвимостях. Есть несколько ключевых моментов, которые нужно вынести из этой истории. Во-первых, подумаем о тщательной перепроверке данных, когда вы раскрываете информацию об уязвимостях. Попытайтесь определить, является ли компания CNA или нет, и хорошо ли мы разобрались в их правилах. Наконец, всегда перечитывайте политику раскрытия информации, перед тем, как отправляете любые данные о потенциальных уязвимостях.

Второй момент – будьте поосторожнее с NDA. У вас будут часто требовать подписать подобные соглашения, когда вы делаете первые попытки работы с компанией, чтобы снизить их риски. Во многих случаях, для компании это просто способ выиграть время, пока они определяют мероприятия по снижению ущерба. Если вы подписали NDA, будьте уверены – это ограничит область ваших исследований и установит временные рамки. Вы же не хотите, оказаться в рамках какого-то соглашения, которое запрещает вам обсуждать свою работу или ее результаты в будущем?

В конце концов, запомните получше – работая с крупными корпорациями, можно сильно разочароваться. Не забывайте тот основной факт, что они часто пытаются защитить свою репутацию и могут быть не так отзывчивы и готовы к сотрудничеству, как вам того бы хотелось. Это не всегда так, но слишком уж обыденно для исследователей обжигаться на работе с вендорами, которые оказываются безответственными, пренебрежительными и обесценивающими результаты их исследований, в конечном счете улучшающих продукт, за который они прямо не отвечают. Запомните обязательно, что не все компании созданы одинаковыми – в то время как одни, могут быть неотзывчивыми и затрудняющими работу с ними, другие вполне могут быть невероятно полезными и способными оценить ваши усилия. Чрезвычайно важно отложить это в памяти, во время своих занятий исследованиями и раскрытием информации об уязвимостях, потому что это поможет вам поддерживать позитивный настрой, даже когда вы сталкиваетесь с трудностями.

Возможные усовершенствования

В этом примере, для улучшения результата, несколько вещей можно было сделать по другому. Первое, с чего исследователю нужно было начать, это сразу определить с кем лучше связаться по поводу уязвимости, что было ясно определено в политике CNA. В результате, исследователь прекратил бы тратить время на разговоры с людьми, плохо подготовленными для работы с ним и потребовавшими от исследователя подписать NDA, которое затормозило бы раскрытие информации еще больше.

Во-вторых, нужно было уделить время и разобраться, как работают CNA и что представляют из себя их правила касательно раскрытия информации об уязвимостях. Это помогло бы избежать недоразумений насчет того, как могла быть использована работа исследователя и как она могла быть вознаграждена.

И в заключение, некоторым улучшением со стороны поставщика ПО, возможно было бы предотвращение замалчивания вклада исследователей, чтобы у них была уверенность, что их публично поблагодарят за труд и достоверность представленных фактов. В главе 7, мы рекомендовали избегать излишней витиеватости в раскрытии информации, которая также поступает и к вендорам. Придерживайтесь фактов, человека поступающего порядочно, будет сложно выставить виноватым.

В заключительном примере из жизни, мы рассмотрим некомпетентность и то, как настойчивость иногда может привести ваше раскрытие информации к лучшим результатам.

Пример 5 – я хотел бы поговорить с вашим менеджером

Этот пример иллюстрирует безвыходные ситуации и лютейшую неспособность к диалогу между людьми, которые принимают и обрабатывают информацию от исследователей.

В 2017 году, один исследователь рассматривал программный продукт, используемый некоммерческими организациями, в составе исследования безопасности для одного из своих клиентов. Программное обеспечение, использовалось для определения того, как некоммерческие организации распределяют фонды по проектам и содержало секретную банковскую информацию по сбору средств.

В ходе своего исследования, он обнаружил метод захвата любой системной учетной записи, используя слабые места в конфигурации платформы. Исследователь быстро связался с организацией, нанявшей его протестировать это ПО. Компания отправила поставщику программного обеспечения контакты исследователя, чтобы они могли держать с ним связь и с надеждой на получение исправления для уязвимости. В разговоре с инженерами техподдержки, исследователя проинформировали, что компания уже знает об уязвимостях и что они были исправлены в самой последней версии программного обеспечения. Они посоветовали исследователю рассказать своему заказчику про обновление ПО.

Далее, исследователь потратил время на составление отчета и перечня работ, которые необходимо выполнить для заказчика, включая обновление устаревшей версии программного обеспечения. Его заказчик был в замешательстве, потому что обновился до новейшей версии программы за пару месяцев до того, как исследователь приступил к работе.

Исследователь, тут же связался с компанией вендора и в беседе с тем же самым сотрудником, с которым разговаривал изначально, задал вопрос – зачем он советовал обновиться, если у заказчика уже была установлена последняя версия. Сотрудник техподдержки извинился, но сказал, что он уверен в том, что уязвимости были устранены. Он уточнил подробности у разработчиков и сообщил, что исправление будет в предстоящем релизе. После этого исследователь отослал все подробности, сотруднику с которым общался по этому вопросу и настроил напоминание, чтобы в течение трех месяцев отслеживать появление следующего выпуска ПО.

Как только три месяца прошло, исследователь опять связался с тем же сотрудником поддержки, который сообщал, что уязвимости были устранены ранее. Специалист техподдержки, был рад сообщить, что уязвимости будут исправленны в следующей версии – после того как ее запланируют к выпуску в течение пары недель. Исследователь понимал, что в конце концов, три месяца было довольно коротким сроком для обработки результатов исследования уязвимостей и рассылки патчей клиентам, поэтому он снова поставил напоминание.

При очередном срабатывании напоминания, исследователь снова общался с тем же человеком к которому он обращался изначально и кому раскрыл информацию об уязвимостях и был проинформирован, что уязвимости были действительно устранены, а все сделанные изменения – содержатся в Release Notes программного обеспечения. Исследователь связался с заказчиком и поделился хорошими новостями, а заказчик запланировал обновление программного обеспечения. Исследователь хотел убедиться, что уязвимости были исправлены и согласовал со своим заказчиком время на подтверждение, что все уязвимости были закрыты. Как только заказчик обновил свое ПО до последней версии, исследователь протестировав его обнаружил, что каждая из уязвимостей о которых заявлялось изначально – до сих пор присутствует в программном обеспечении. Исследователь вспомнил, что все исправления должы были быть отражены в заметках об изменениях к выпуску новой версии. К удивлению исследователя заметки об изменениях, содержали единственное предложение:

В этом выпуске, отсутствуют значительные изменения.

Исследователь обратился к вендору еще раз, чтобы навести справки. На этом этапе, исследователь рассказал о программном обеспечении, которое вроде уже было исправлено, затем исправлялось в будущем релизе, позднее оно было протестировано и как выяснилось – исправлено не было. Тот же инженер технической поддержки, который помогал ему с самого начала, пояснил, что там могла быть некоторая путаница относительно того, что было или не было исправлено. Тем не менее, он сообщил, что они выйдут на связь с исследователем, после переговоров со своими разработчиками.

Прошло несколько месяцев, но вендор на связь не выходил. В конце концов исследователь, еще раз связался с поставщиком ПО, переговорил с тем же сотрудником и опять саппорт извинился, что не выходил на связь и сообщил, что с выходом обновлений в следующем месяце, все багфиксы которые были обещаны, будут включены в программное обеспечение.

Проводя со своим заказчиком, работы по обновлению до последней версии ПО спустя месяц, исследователь выяснил, что уязвимости не были закрыты, так же как и в прошлый раз, когда он повторно исследовал ПО. Понятное дело, исследователь был крайне огорчен и чувствовал, что тип из техподдержки с которым он связывался, либо совершенно не компетентен, либо просто бездельник. Исследователь решил, что следующим шагом могло быть открытие CVE, получение зарезервированных номеров, а затем информирование компании, что если уязвимости не будут закрыты в течение следующих 90 дней, он выпустит эту информацию в широкий доступ. Как только исследователь получил зарезервированные номера CVE, он запланировал созвать переговоры для обсуждения ситуации. Разговаривая со своим старым знакомым из техподдержки, исследователь изложил причины своего недовольства, заключавшиеся во вранье и обещаниях исправления уязвимостей, которые так никогда и не были претворены в жизнь. Инженер техподдержки, предсказуемо извинился и сказал, что впредь станет работать лучше и еще раз повторил, что уязвимости будут закрыты в следующей версии.

Тогда исследователь, потребовал разговора с руководителем этого сотрудника. Он сообщил, что уверен, в том что инженер техподдержки хотел как лучше, но в связи с отсутствием результатов нужных для разрешения ситуации, вопрос нужно решать на более высоком уровне. Немного поупиравшись, тип из саппорта все-таки дал исследователю контакты своего начальника. Исследователь составил письмо руководителю, с разъяснением сиутации, своего недовольства бездействием поставщика ПО и тем, что информация об уязвимостях, будет раскрыта в ближайшие недели.

Руководитель ответил в течение дня и попросил о телефонном разговоре. После разговора с ним, исследователь обнаружил, что информация об уязвимостях, вообще не попадала к нужным людям. Инженер техподдержки, переправлял отчеты в службу эксплуатации, которая прогоняла серверы с установленным софтом антивирусом и заявляла, что никаких уязвимостей нет. Руководитель понял, что проблема выстрелит не один раз в следующие несколько недель, в виде вопросов о методах снижения или предотвращения ущерба для их клиентов. В итоге, после нескольких недель хождения взад-вперед, менеджер назначил дату, когда исследователь еще раз протестирует программное обеспечение, для проверки, что все уязвимости исправленны, перед рассылкой обновления клиентам – прямо перед крайним сроком, установленным исследователем для раскрытия результатов исследования широкой публике.

Исследователь последовал предложению менеджера и согласился с предложенной датой проверки исправления уязвимостей. Он поблагодарил менеджера за потраченное время и силы на исправление ПО и предотвращение публичного раскрытия информации при отсутствии методов снижения ущерба. Хотя эта ситуация закончилась хорошо, несложно видеть, что события могли закончиться совсем по другому, особенно если бы исследователь отказался бы следовать предложению вендора.

Извлеченный урок

В этом примере, взаимодействие шло просто ужасно. Наш исследователь совершил ошибку, думая, что говорит с нужным человеком. В свою очередь инженер технической поддержки, имел крайне низкую квалификацию, не понимал технического исследования и соответственно не знал как отвечать. Обратись исследователь, непосредственно к руководителю, это помогло бы избежать огорчений и пустой траты времени.

В данном случае, важно также отметить, что исследователь дал поставщику ПО максимум возможностей исправить уязвимости, до того, как они уйдут в народ. Сверх того, пытаясь работать с вендором, он пошел на обострение, только когда не было другого выхода. Однако срыв и затягивание планов, часто является установившейся практикой при работе с уязвимостями – убедитесь, что вы дали поставщику ПО возможность исправить проблему до того, как она будет обнародована, но подготовьтесь к публикации, если это не сработает.

Будьте уверенны и терпеливы, а если раз за разом результат, оказывается не таким как вам хотелось или некорректен, иногда правильным будет привлечь больше людей, которые могли бы распутать этот клубок и были бы полезны вам в достижении цели. Следовательно, требование поговорить с руководителем, в данном случае оказалось именно той вещью, которая была необходима, чтобы нормально обсудить результаты исследования.

Возможные усовершенствования

Для улучшения результата, некоторые вещи в этом примере, можно было бы сделать по другому. Во-первых всегда стоит убедиться, что вы работаете больше, чем просто с одним человеком, если есть возможность, включите в свою аудиторию доступных вам экспертов.

Еще одним доступным улучшением, является то, что исследователь мог обострить ситуацию раньше, когда стало ясно, что инженер техподдержки оказался бесполезен. Поднятие вопроса на высший уровень, случись оно раньше, сохранило бы кучу времени и обеспечило ускоренное получение патчей, заказчиком исследователя.

И в заключение, рекоменуется установка конкретных дат раскрытия информации об уязвимостях, в самом начале работы с поставщиком программного обеспечения. Хотя это идеально для исследователя, поговорить с вендором и просто ждать патча, вендоры не всегда действуют ожидаемо. В первом приближении, набросок вашего плана для компании и утверждение, что вы раскроете информацию о результатах исследований, в течение X дней, привлечет особое внимание к вопросу. Сообщение, вроде этого, может послужить причиной привлечения сотрудниками своего руководства и выносом ситуации на более высокий уровень, без вмешательства исследователя. Все-таки в целом, исследователь проделал прекрасную работу, пытаясь сотрудничать с поставщиком ПО, до обострения ситуации.

Будем надеятся, что вы нашли эти пять примеров из жизни, поучительными. Теперь, давайте завершим эту главу, вкратце подытожив чему мы научились.

Итоги главы

В этой главе мы потратили время на рассмотрение пяти реальных случаев, погрузившись в прецеденты с раскрытием информации об уязвимостях, демонстрирующие разные проблемы и обстоятельства, которые к ним привели. Подготовленное вами исследование и дальнейшее раскрытие информации, будут не всегда идти по плану. В таких сценариях, это в основном проявляется в виде халатного и некомпетентного поведения, демонстрируемого поставщиками программного

обеспечения. В каждом рассмотренном случае, различные изъяны в диалоге межу исследователем и вендором, привели к результатам далеким от идеала. В одних случаях, исследователь сделал неверные предположения о том, с кем разговаривает и его профессиональном уровне. А в других, вендор бездействовал, даже после массы предоставленных ему возможностей или, что еще хуже – проявлял враждебность к исследователям.

Чтобы защитить плоды своего труда, крайне важно запомнить несколько ключевых вещей:

- Всегда общайтесь в вендорами професионально, во время изложения им своих требований
- Подумайте об использовании арбитража уязвимостей, при первых признаках трудностей или задержек
- Давайте поставщику ПО, разумное время на исправление уязвимостей, до их публикации
- Имейте четкий план по раскрытию информации, включая конечные сроки завершения работы и резервные планы
- Будте готовы следовать вашему плану раскрытия информации, без сотрудничества с вендором

Держа все это в уме, вы будете в лучшем положении, чтобы разрулить любые ситуации во время ваших будущих авантюр по исследованиям и раскрытию информации.

В следующей главе мы сместим фокус в область советов, непосредственно производителям и поставщикам ПО, чтобы помочь им лучше понять, почему исследователи могли с ними связываться, как с лучшей стороны представить свои бизнес-интересы и использовать исследование с выгодой для себя.

Глава 9. Взаимодействие с исследователями в области безопасности – руководство для вендоров

Когда Барнаби Джек выступил со своим печально известным докладом на конференции Black Hat в 2010 году, он покинул сцену под восторженные аплодисменты. Обладая талантом произвести впечатление, он представил технику эксплуатации **банкоматов** (**automated teller machines** – **ATM**), названную им **Jackpotting**. Используя вредоносное ПО, нацеленное на уязвимости обнаруженные в функционале банкоматов, он провел наглядную демонстрацию этой техники, собранию хакеров и специалистов по ИБ. В своей демонстрации, он удаленно активировал уязвимость, которая захватывала дисплей банкомата и запускала ролик с анимацией джекпота, пока из банкомата непрерывно сыпались на пол деньги.

Перед своей смертью в возрасте 35 лет, он продемонстрировал перспективное исследование, подробно описывающее уязвимости в медицинской технике, такой как инсулиновые помпы и кардиостимуляторы. Жизнь и работа Барнаби Джека были чем-то невероятным. Благодаря своей любознательности, он развил такие способности и навыки, которые дали ему, что-то вроде сверх-возможностей в мире современных технологий. Он мог заставить банкомат выплевывать деньги, практически по команде. Он мог вызывать у людей самопроизвольные сердечные приступы или инсулиновый шок у диабетиков, транслируя специально подготовленные радиосигналы. Задумайтесь на минуту о том, насколько немыслимо, что бы кто-нибудь сделал подобное, но вместо того, чтобы продать результаты исследования или злоупотребить этими знаниями для извлечения личной выгоды, он поделился ими, чтобы сделать мир лучше. Подобно большинству исследователей, Джек испытывал чувство глубокого морального удовлетворения в том, чтобы делать мир лучше и безопаснее, погружаясь в изучение технологий в различных их проявлениях.

На протяжении этой книги, мы уделили значительное время, рассказам о вендорах, создающих программное обеспечение. Читая этот материал, вы на определенном этапе своей карьеры, могли быть в разных ролях связанных с эксплуатацией, разработкой или даже безопасностью какой-то компании. Несмотря на освещение поставщиков ПО в негативном ключе, на протяжении этого руководства, вендоры нанимают специалистов, которые (в основном), хотят качественно выполнять работу, подразумевая поддержку их программных продуктов. В дополнение к сказанному, большинство вендоров и людей, которые у них работают – не понимают, кем являются исследователи безопасности, чем они занимаются или что их к тому подвигло. Исследователей безопасности, считают супергероями технологий и как большинство вымышленных персонажей, эти герои часто оказываются непоняты – и даже оклеветаны – теми, кто их не понимает. Точно также, фундаментальное непонимание поставщиками ПО исследователей безопасности, приводит вендоров к устойчиво негативным ассоциациям связанных с исследователями. Такие негативные ассоциации, приводят поставщиков ПО к упущению удобных случаев для выстраивания конструктивного диалога и обращения к собственной выгоде возможностей исследователей безопасности, добровольно предлагающим услуги организациям.

Эта глава, представляет собой руководство для вендоров по общению с исследователями безопасности на тему раскрытия информации об уязвимостях. В первую очередь, мы нацелены пролить свет на то, кем же являются исследователи в отношении поставщиков программного обеспечения. Далее, мы обсудим важность построения здоровых взаимоотношений с исследователями, т. к. это гарантирует, что раскрытие информации об уязвимостях, изначально пойдет ответственно. Мы предложим руководство по взаимодействию с исследователями, включая подсказки по реагированию на запросы о раскрытии информации и обеспечению обратной связи по предполагаемым исправлениям. В заключение, в этой главе будет представлен обзор различных политик раскрытия информации, которые вендоры смогут взять на вооружение вместе с рекомендациями по созданию эффективной политики для своей организации.

Итак, чтобы подытожить, что вы увидите в этой главе – заниматься мы будем следующим:

- Дадим определение термину "исследователи безопасности", изучив их характеристики, навыки и мотивацию
- Подумаем над тем, как организации могут конструктивно сотрудничать с исследователями, чтобы создавать улучшенные, более безопасные продукты и услуги.
- Прольем свет на правильную линию поведения, для построения доверительных отношений с исследователями и обрисуем, что следует делать, а чего избегать
- Изучим, как создать эффективную политику ответственного раскрытия информации, исследуя полноценный пример такой политики, который вы сможете использовать, как образец

Эти результаты будут представлены на протяжении следующих основных тем этой главы:

- Кем является исследователь информационной безопасности?
- Использование возможностей исследователя
- Построение доверия и сотрудничества с исследователями
- Разрабатываем политику ответственного раскрытия

Кем является исследователь информационной безопасности?

Вы можете иметь об этом хорошее представление, если читали эту книгу, начиная с *Главы 1, Знакомство с понятием уязвимости*, через которую проходит это понятие. Все-таки кратко повторим – исследователь информационной безопасности, это тот, кто ищет уязвимости в программном обеспечении и различных системах. Исследователи могут работать над своими исследованиями полностью независимо или в коллективе. Исследователи имеют разную образовательную базу, некоторые официально не заканчивали каких-то учебных заведений. Компании нанимают исследователей, чтобы изучать их продукцию и программное обеспечение, некоторые являются профессиональными баг-хантерами, а другие трудятся в конторах, которые барыжат результатами исследований. Мы считаем, что диалог и налаживание взаимоотношений с исследователями – даст эффект и вы должны первыми попытаться понять это. Одним из способов, которыми вы можете улучшить свое представление об исследователях, это разобраться в их основных характеристиках, навыках которыми должен владеть исследователь и тем, что его мотивирует. Итак, давайте приступим к изучению общих характеристик исследователя.

Характеристики исследователя

В целом, предполагается, что некоторые из наиболее успешных исследователей обладают ключевыми характеристиками: они творчески подходят к делу, любознательны и упорны. Эти характеристики дополняют друг друга и помогают исследователям понять то, что они хотят освоить. Начав с творческого подхода, мы расмотрим, что делает исследователей безопасности, теми кто они есть, обсудив эти характеристики.

Творческий подход

Лучшие исследователи информационной безопасности, умеют размышлять об абстрактных задачах оригинальными способами. Они способны видеть закономерности, там где остальные видят хаос и они способны изобретать новые методы достижения целей, выходя за рамки привычных шаблонов. Творческий подход, выражается в умении взглянуть на проблему под разными углами зрения и находить свежие и инновационные решения.

Творческая сущность, совершенно ясно прослеживается в их подходе к работе. Они постоянно проверяют свои новые идеи и рисуют в воображении новые подходы к решению старых проблем. Эта

тяга к экспериментам и испытание новых методов, позволяет им находить наилучшие решения трудноразрешимых задач.

Любознательность

Лучшие исследователи информационной безопасности, обладают ненасытной жаждой знаний. Они хотят понимать суть и решительно подходят к решению проблем. Эти черты, позволяют им раздвинуть границы возможного и обнаружить слабые места в логике.

Мысленно возвращаясь к некоторым из наиболее резонансных исследований, проведенных в последние 10 лет, выясняется, что обычно они начинались с пустяков. Для исследователей не является чем-то удивительным добиваться лучшего понимания технологий, путем досконального изучения **Request for Comments** (**RFC**), утвержденных стандартом **Institute of Electrical Electronic Engineering** (**IEEE**) или томов исследований опубликованных до них. Они путешествуют через океан неизведанного и такая любознательность, часто приводит их к всеобъемлющему пониманию того, как что-то функционирует. Любознательность, приводит к постижению объекта внимания.

Упорство

Большинство талантливых исследователей – упорны. Исследователи обычно считают, что если что-то не возможно, то просто еще не пришло время для этого. В итоге, они периодически возвращаются к таким сценариям, когда представляется возможность и отказываются от дальнейших попыток, только после большого количества постоянных неудач.

Такие упорство и настойчивость, очень пригодятся исследователям, когда они столкнутся со множеством препятствий в случаях, когда их идеи не срабатывают. Это нужно им, чтобы суметь оправляться от неудач и продолжать движение к цели, чтобы в конце концов достичь успеха. Такую целеустремленность, можно рассматривать, как разновидность способности быстро приходить в норму, еще одну важнейшую черту исследователей. Изрядный запас упорства, обычно отличает лучших исследователей информационной безопасности от остальных. Они непрерывно самосовершенствуются и никогда не пасуют перед трудностями.

Эти отличительные особенности не единственное, что характеризует исследователя безопасности. Квалифицированные исследователи обладают широким спектром навыков и познаний в технических дисциплинах. Давайте обсудим это сейчас.

Ключевые навыки исследователя

Успешные исследователи безопасности, должны опираться на прочное основание разнообразных навыков. Лучшие исследователи глубоко понимают, насколько взаимосвязаны технологии применительно к вычислительным системам. Они могут превосходно усваивать дисциплины, в большей степени сосредоточенные на логике, таких как математика. Дополнительно, они могут владеть навыками коммуникации, включая письменную, устную и влияние на людей. Давайте поговорим об этих навыках.

Технические навыки

Лучшие исследователи информационной безопасности, обладают мощными техническими навыками. Они умеют разбираться в сложных системах и в совершенстве владеют языками программирования и инструментами разработки. Эти технические навыки разнообразны, но обычно состоят из следующих категорий:

- **Реверс-инжиниринг:** Умение разбираться, как работает железо и софт, без доступа к исходному коду или проектной документации.
- **Криптография:** Умение понимать и применять на практике, алгоритмы шифрования.
- **Программирование:** Умение писать код, и что более важно понимать уже написанный, на одном или более языках программирования.
- **Системное администрирование:** Умение разбираться, как администраторы конфигурируют информационные системы, и как эксплуатировать такие конфигурации.
- **Сетевые технологии:** Умение разбираться во взаимосвязи систем, соединенных с использованием различных версий стандартов и протоколов. Это может включать понимание физических и эфирных сред передачи сигнала, таких как радио в беспроводных сетях.

Аналитические навыки

Часто исследователи обладают чрезвычайно развитым навыком обнаружения закономерностей там, где у других это не получается. Они способны быстро анализировать огромные объемы данных. Этот набор аналитических способностей, также полезен во время поиска источника проблем при исследовании. Исследователям, часто необходимо быстро определить основную причину сложившейся ситуации, чтобы прикинуть доступные варианты решения.

Навыки коммуникации

Хотя это не всегда так, но первоклассные исследователи могут обладать или со временем приобретать склонность к распространению своих работ, часто для продолжения своих собственных исследований или же, исследований проводимых в ходе своей трудовой деятельности. Часто это приводит исследователей, движимых жаждой совершенствования безопасности информационных систем, к участию в работе по неформальному освещению результатов своих исследований. Такие навыки коммуникации, могут проявляться в стиле эпатажного выступления Барнаби Джека на Black Hat или в более спокойном виде обмена идеями. Исследователи вместе с такими навыками, могут также обладать врожденным талантом находить общий язык с самыми разными людьми.

Вы можете сделать вывод, что исследователи обладают крайне развитым интеллектом и разрушительными способностями, позволяющими им бросать вызов общепринятым нормам поведения. К примеру, некоторые исследователи могли бы поддаться искушению, использовать свои навыки в преступными целях, в то время как остальные могли их использовать для решения сложнейших задач и во благо общества. Чтобы лучше понять, почему исследователи делают то, что они делают, нам потребуется изучить мотивацию исследователей.

Мотивы движущие исследователем

Подобно всем людям, жизнь исследователя управляется их движущими мотивами, которые обусловлены культурными, общественными и личными факторами. Множество различных мотивов могут привести исследователей к увлечению этим занятием. Некоторые из этих мотивов, могут быть скорее личными, такие как желание внести свой вклад в развитие человеческих знаний или желание изменить мир. Давайте все это рассмотрим.

Тяга к открытиям

Стремление к открытию неизвестного, является мощнейшим движущим мотивом для большинства исследователей. Увлекательные задачи по поиску чего-нибудь нового, или раскрытию чего-то, что было скрыто – мотивирующий фактор для множества людей. Стремление к открытиям, может также быть мотивирующим фактором для исследователей, пытающихся искать новые способы решения старых проблем. Например, исследователь пытающийся найти нестандартный способ

эксплуатации программного обеспечения, может быть сподвигнут к этому стремлением поиска новых способов добиться уже известного эффекта.

Дух соперничества

Стремление к соперничеству, является еще одним распространенным мотивом исследователей. Стремление быть лучшим или первооткрывателем, может сподвигнуть людей на многое. По одним направлениям информационной безопасности, можно найти соперников среди изучающих техники защиты и нападения, на соревнованиях **capture-the-flag** (**CTF**), на которых лучшие умы сходятся в битве друг против друга в погоне за денежними призами, присуждаемыми лучшим хакерам. Соответственно, в других – исследователи могут состязаться за право считаться лучшим в реверс-инжиниринге патчей безопасности, вскрывая информацию об уязвимостях, о которой по другому не узнаешь.

Тяга к знаниям

Еще один известный мотив, это тяга к знаниям. Исследователи хотят знать, как все работает и часто движимы желанием познавать мир вокруг себя. Стремление к знаниям у разных людей, проявляется по разному. Например, одни исследователи могут хотеть разобраться, как работают отдельные системы, тогда как другие хотят понять, как работают отдельные атаки.

Жажда признания

Потребность в одобрении своих действий, это тоже признанный мотив для многих исследователей. Удовлетворение высокой оценкой своего труда или признание коллегами, вполне себе фактор мотивации для многих людей. Одни исследователи, занимаются этим для повышения собственной репутации, в то время как другие могут использовать свои исследования, как ступеньку карьерного роста или для других профессиональных выгод.

Власть

Можно считать доказанным, что темная сторона искушает исследователей властью. Некоторые исследователи получают огромнейшее удовольствие, захватывая контроль над окружающим миром или влияя на него по своему желанию. Такие мотивы, часто выражаются в **хактивизме**, когда люди совершают взломы, чтобы продемонстрировать свои гражданские, этические, религиозные или политические воззрения или насильно донести их до кого-либо.

Деньги

Разумеется деньги, весьма распространенный мотив для многих исследователей. Желание хорошо зарабатывать или необходимость добывать средства к существованию для себя и своей семьи, сильнейший фактор мотивации. С такими способностями, исследователи могут получать намного больше средне-мировой зарплаты за свою работу. Этот доход, может быть получен вполне законными методами, типа работы в качестве штатного исследователя на какую-либо контору или в качестве баг-хантера. Или наоборот, исследователи могут действовать методами, сомнительными с точки зрения этики – толкая уязвимости перекупам или занимаются, откровенно преступной деятельностью.

Помощь другим

Потребность помогать другим, это еще один мотивирующий фактор, который можно заметить у некоторых исследователей. Этот мотив может проявляться в действиях, определяемых их мировоззрением. К примеру, исследователь придающий большое значение социальной справедливости, мог бы попытаться найти способы помочь угнетенным из-за их взглядов или из-за

того, кто они есть. Однако, это тонкая грань между помощью нуждающимся и злоупотреблением своей силой.

Например, исследователь может быть интересоваться конфиденциальностью и исследованием мобильных приложений, которые нарушают конфиденциальность пользователей. Обнаружение таких дефектов и передача этой информации общественности, так чтобы люди могли принимать более информированные решения, является прекрасным примером помощи другим. Однако, точно не будет полезно, если такие конфиденциальные данные, окажутся использованными со злым умыслом, приводя к нарушению прав человека, скажем, авторитарными режимами, стремящимися угнетать недовольных.

Теперь, имея базовое представление о том, что делает исследователей теми, кто они есть, давайте уделим время обсуждению того, какую выгоду могут извлечь организации из работы с такими талантливыми личностями.

Использование возможностей исследователя

Прогрессивно мыслящие организации знают, что они могут улучшить состояние информационной безопасности своих продуктов, задействуя возможности исследователей. Теперь, когда мы примерно представляем, чем занимаются исследователи безопасности, их характеристики и что ими движет, давайте поговорим о том, как огранизации могут применить все это к собственной выгоде.

Из-за постоянного спроса на zero-day эксплоиты и их высокой стоимости, некоторые исследователи могут быть более склонны к продаже результатов своих исследований вместо того, чтобы поделиться ими с компанией поставщика ПО напрямую. Такие исследователи, обычно могут идти несколькими путями. Они могли бы приложив усилия, поделиться своими исследованиями напрямую с вендором не ожидая награды, использовать уязвимости самостоятельно или продать их перекупам. Когда исследователь связывается с компанией поставщика ПО, воспринимайте это как хороший знак, такие исследователи поступают добропорядочно.

Как обсуждалось в *Главе 4. Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности*, исследователи в основном связываются с компаниями, если обнаруживают, что их исследование пересекается с бизнесом и его продуктами. Исследователи, действующие из лучших побуждений, движимые наиболее конструктивными намерениями, должны приветствоваться в любой организации. И наконец, они держат руку на пульсе, относительно открытий, улучшающих ситуацию в сфере безопасности программного обеспечения.

Одной из лучших вещей, которые может сделать компания, чтобы подтолкнуть исследователей к сотрудничеству, является наличие опубликованной **политики ответственного раскрытия информации**, которую легко найти и содержащую все, что нужно для взаимовыгодного сотрудничества. К сожалению, большинство вендоров не имеют налаженной процедуры приема и управления поступающими отчетами об уязвимостях. Поставщики программного обеспечения, не имеющие таких политик, рискуют упустить подходящую возможность поработать с исследователями. Исследователи могут связывать компании без подобных политик с прошлым негативным опытом, разочарованием и ощущением обесценивания, с которыми они могли сталкиваться, работая с другими компаниями без политики ответственного раскрытия информации. Подобные чувства, толкают исследователей на публикации, даже не пытаясь связаться с вендором, что приводит к полному раскрытию информации или к перепродаже результатов исследований тем, кто готов их купить. Процедура согласованного раскрытия информации, помогает справляться с такими недочетами. Такие политики устанавливают, каким образом исследователи должны сообщать об обнаруженных ими уязвимостях, и как организации будут такие сообщения обрабатывать. Создавая понятные правила и

методики, организации могут быть уверены, что раскрытие информации пройдет со всей ответственностью и в разумное время. В главе 4, мы советовали исследователям, пытаться находить политики раскрытия информации о безопасности и считать их наличие – признаком того, что компании готовы к сотрудничеству с ними.

Предупреждение

Хотя здесь это подразумевается, стоит выделить это явно. Когда поставщики программного обеспечения решают, не идти на контакт, вести себя пренебрежительно или угрожать исследователям, они порождают сценарии, в которых они теряют контроль над уязвимостью. Исследователи, действующие из лучших побуждений, предлагают вендорам кое-какую ценность – контроль над тем, насколько подробно раскрывать информацию. Исследователи могут обратиться и к другим методам, которые могут представлять для вендоров угрозу, источник неприятностей и возможно финансовые потери, в зависимости от ситуации. Вам следует избегать этого и добиваться более благоприятных результатов, применяя разумный подход, а политики согласованного раскрытия информации – это то, что позволит вам контролировать, как именно раскрывать информацию.

Во многих случаях политику ответственного раскрытия информации, можно найти на веб-сайте поставщика ПО, как правило по ссылке *безопасность*. Однако, новые веяния, такие как спецификация формата файла по раскрытию информации об уязвимостях **RFC9116** от **Internet Engineering Task Force** (**IETF**), предлагают, компаниям размещать текстовый файл **security.txt** в корневом каталоге их веб-сайта. Этот текстовый файл, должен включать такую информацию, как email, который исследователь сможет использовать для первого контакта, ключи шифрования, которые можно использовать для шифрования переписки, копия политики раскрытия и страница с отчетами уже принятыми к рассмотрению. Этот простенький текстовый файл, может быть и таким сжатым, как в следующем примере:

```
# Our security address
Contact: mailto:security@example.com

# Our OpenPGP key
Encryption: https://example.com/pgp-key.txt

# Our security policy
Policy: https://example.com/security-policy.html

# Our security acknowledgments page
Acknowledgments: https://example.com/hall-of-fame.html

Expires: 2021-12-31T18:37:07z
```

Рис. 9.1 – Пример того, как обычно выглядит применение RFC9116

Принятие стандартов, подобных этим, помогает легче находить политики безопасности, а также дает исследователям знать, что вы готовы сотрудничать с ними в области улучшения своих программных продуктов. В разделе *Разрабатываем политику ответственного раскрытия информации* мы обсудим, такую политику.

Использование широких возможностей исследователей в области раскрытия информации об уязвимостях, создает беспроигрышный сценарий и для компаний и для исследователей. Компании получают выгоду в виде повышения безопасности и доверие сообщества исследователей, а исследователи – признание и награду за свой труд.

Также важно помнить, что раскрытие информации об уязвимостях, это дорога с двухсторонним движением. В добавок к получению отчетов об уязвимостях, поставщики ПО в свою очередь, должны предоставлять исследователям обратную связь по предполагаемым исправлениям. Обратная связь, должна быть своевременной и конструктивной, учитывающей точку зрения исследователя.

Вендор, который эффективно управляет взаимодействием с исследователями или заботится о построении серьезных, взаимовыгодных отношений – получит взамен, множество выгод от этого. В добавок, когда эти отношения эффективно управляются, они предоставляют существенные выгоды, при низких затратах. Это включает раннее обнаружение и предотвращение атак, улучшение безопасности продукта, глубокое понимание положения с безопасностью и повышение доверия клиентов.

А сейчас, давайте тщательно разберем типичные вещи, которые вы должны стараться делать и те, которые вы должны стараться **не делать**, когда выстраиваете серьезные и взаимовыгодные отношения, которые укрепят безопасность ваших продуктов, задействуя труд исследователей.

Построение доверия и сотрудничества с исследователями

Вендоры, при первом знакомстве с исследователями безопасности, могли испытывать опасения, касающиеся мотивации исследователей. К сожалению, это часто приводит организации к совершению ошибок при работе с исследователями. Они могли избегать всяческих контактов, думая что любое взаимодействие, закончится либо атакой, либо публичным раскрытием информации. Поставщики ПО, могли попытаться купить молчание исследователей, с помощью баг-баунти и принудительного соглашения о неразглашении. Или, что еще хуже – они могли угрожать исследователям судебным преследованием. Такая далеко не оптимальная реакция, может повредить возможности создания ситуации, в которой можно было бы спокойно добиться результатов, выгодных и вам, и вашим клиентам и исследователям.

Для достижения положительных результатов, мы должны в первую очередь, проанализировать ошибки при работе с исследователями и способы их избежать. Давайте рассмотрим эту тему сейчас.

Обходим распространенные ошибки во взаимоотношениях

Сейчас мы уделим некоторое время, обсуждению распространенных способов, которыми вендор может повредить, возможности конструктивного взаимодействия с исследователями. Хотя это не самый исчерпывающий список, есть распространенные темы, которые мы охватим, беседуя с исследователями, плотно работающими с поставщиками программного обеспечения.

Не относитесь к исследователям с пренебрежением

Работая с исследователями важно помнить, что они являются экспертами в своей области и обладают множеством ценных качеств. К сожалению, многие вендоры совершают ошибку, отклоняя доклады об уязвимостях, даже не взглянув на них или отмахиваются от них, как от *хакеров*, которые просто пытаются что-то поломать. Пренебрежительное отношение, показывает отсутствие уважения к исследователям и абсолютное непонимание того, чем они занимаются. Помните, что эти люди пытаются помочь сделать ваш софт безопаснее и с ними нужно обращаться как с коллегами, а не как с возможными преступниками. Крайне необычно для преступников, связываться с вами напрямую, чтобы предложить улучшить ваше программное обеспечение. Помните, что эти люди о которых мы сейчас говорим – эксперты, пытающиеся помочь сделать ваши продукты защищеннее, иногда не требуя ничего взамен.

Не стройте бессмысленных предположений

Крайне важно избегать излишних фантазий насчет того, чего добиваются исследователи или почему они на вас вышли. Например многие компании полагают, что все исследователи заинтересованы исключительно в получении прибыли и будут сотрудничать с вами, только если вы предложите им вознаграждение. В некоторых случаях, это может и так, но в большинстве – это исключение, а не правило. Множество исследователей, просто заинтересованы в том, чтобы сделать интернет безопаснее и будут с вами сотрудничать, даже если вы не сможете предложить им взамен, каких-либо денег. Предположения о чьих-либо движущих мотивах, часто приводит к напряженности и недоверию, так что лучше их не делать вообще.

Не игнорируйте отчеты об уязвимостях

Одной из самых частых жалоб исследователей является то, что поставщики ПО, просто игнорируют их доклады. Это оставляет у исследователей ощущение недооцененности их участия и часто приводит к полной потере интереса к сотрудничеству с такой компанией. "Радиомолчание" вендоров, может также вызвать у исследователей негодование, побуждая их обнародовать результаты исследования и как правило поделиться своим опытом насчет не реагирующей на обращения компании. В конечном счете, это вызывает у клиентов сомнения в качестве программного обеспечения, которое им продали и возможно вопрос, а так ли уж вендор заботится о безопасности своих продуктов?

Чтобы избежать этого, вендорам необходимо каждый отчет об уязвимости, воспринимать серьезно. Серьезное восприятие информации об уязвимости, подразумевает быстрый ответ исследователю, подтверждение получения отчета и обещание ознакомиться с ним. Если информация представляет ценность, свяжитесь с исследователем, поблагодарите его за содействие и займитесь исправлением выявленной проблемы. Такие действия покажут, что компания дорожит безопасностью своих продуктов и с готовностью сотрудничает с исследователями, чтобы гарантированно соответствовать ожиданиям клиентов.

Когда компания игнорирует отчеты или слишком затягивает с ответом на них, это не радует исследователей и снижает желание продолжать сотрудничество с компанией. Более того, это подвергает риску безопасность программного обеспечения и вредит репутации компании. Такого ужасного результата, легко избежать своевременно реагируя отчеты об уязвимостях, даже в виде короткой благодарности и слов, что ваша команда, как раз изучает проблему.

Не обрывайте связь

Сотрудничество исследователей и поставщиков ПО, часто разрывается из-за трудностей коммуникации, когда вендоры не держат исследователя в курсе, о ходе работы. То, что исследователю подтвердили принятие его отчета об уязвимости не подразумевает, что его работа завершена. Исследователь, часто хочет опубликовать результаты своего труда и, разумеется будет наблюдать за вашим продвижением в исправлении найденных им уязвимостей.

К примеру исследователи, в основном продолжат общение с вендором, по ходу выработки мер по снижению ущерба от уязвимости, предлагая дополнительную информацию, давая примеры того, как вендор может сам проверить уязвимость, воссоздать ее для разработчиков и даже провести повторное тестирование, после исправления. Во многих случаях, исследователи также помогут вашей команде понять основную причину возникших проблем и предложат рекомендации по предотвращению подобных ситуаций в будущем. Советы экспертов в этих вопросах, могут быть неоценимым источником улучшения безопасности ваших продуктов в целом, помогая вашим сотрудникам понять, в каких направлениях работать.

Если исследователь связался с вами после передачи отчета, постарайтесь ответить сразу и держать его в курсе, о ходе работ по его отчету. Раз уж вы приняли обозначенную проблему, дайте исследователю знать, что собираетесь предпринимать и полезен ли его доклад в решении проблемы. Такая обратная связь, чрезвычайно важна для поддержания хороших отношений с исследователями и побуждению их к дальнейшему сотрудничеству с вами.

Не тяните с исправлением уязвимости

Как только вы получили отчет об уязвимости, вы должны сразу на него отреагировать. Исследователи, часто оказываются крайне огорчены, когда они находят какую-нибудь уязвимость и озвучивают ее, а взамен получается вендор, что возится с ее исправлением месяцы или даже годы. Растянутые сроки исправления уязвимостей, показывают исследователю, что вы не придаете безопасности первостепенного значения. Снижение приоритета таких задач, заставляет исследователя чувствовать, что его работа бессмысленна, и в итоге на долго оставить продукты вендора уязвимыми. Вместо этого, побыстрее набросайте график исправления, равняясь на лучшие практики, принятые в индустрии, касательно закрытия уязвимостей (напр., 30 дней для критических уязвимостей и 90 дней для средней степени опасности), и передайте его исследователю.

Если у вас не получается исправить, какую-то проблему в рамках заявленного срока, свяжитесь с исследователем и объясните ситуацию. В большинстве случаев, они поймут вас и может даже помогут приблизить решение. Однако, если вы неоднократно нарушаете установленные сроки или не держите исследователя в курсе о ходе вашей работы, он скорее всего утратит к вам доверие и обнародует результаты своих исследований.

Не занимайтесь отчетами единолично

Важно помнить, что когда некий исследователь, передал вам отчет об уязвимости, он не станет пытаться атаковать вашу компанию. Большинство исследователей, связываются с вами явно потому, что движимы желанием повышения безопасности программных продуктов. Если вы получили отчет об уязвимости, подойдите к этому конструктивно и используйте как подходящую возможность для улучшения безопасности своих продуктов. Для исследователей не редкость, получать от прижимистых вендоров комментарии не по делу, так или иначе ранящие их профессиональную гордость. Тем не менее, исследователи желают вам помочь и вам следует принимать эту помощь, когда ее предлагают. Альтернативы – гораздо хуже.

Не опускайтесь до обмана

Это бывает редко, но иногда мы видели и такое. Поставщики ПО, могут лукавить насчет своих взглядов, планов и возможностей, сильно их преувеличивая или вводить исследователя в заблуждение. К несчастью для вендоров, исследователи редко отступаются и с присущей им настойчивостью продолжают размышлять об уязвимости. Если они узнают, что оказались обмануты в ходе сотрудничества, не удивляйтесь, что исследователь отвернется от вашей компании и в итоге обнародует результаты исследования, в которых несомненно будут подробности о том, как вы их обманули.

Не начинайте с возбуждения судебного дела

Классическим примером общения между поставщиками ПО и исследователями, часто оказывается его начало, с угроз судебного преследования. Если исследователь связывается с вами, рассчитывая на сотрудничество, иногда без явной выгоды, он скорее всего плохой кандидат на такое жестокое обращение. Это не способ начать диалог, а только ухудшить дальнейшие взаимоотношения. Если исследователь чувствует, что скорее всего подвергнется угрозам, он вероятно полностью прекратит общение и поведает свою историю СМИ. Хотя определенные сценарии оправдывают

привлечение юристов или правоохранительных органов, уведомление о юридических последствиях – не лучшее начало разговора.

Теперь, когда мы рассмотрели некоторые из вещей, которые не нужно делать во время работы с исследователями, давайте поговорим о том, как следует поступать для построения взаимовыгодных отношений.

Выстраиваем взаимовыгодные отношения вендор-исследователь

Вместе с основами того, что мы не должны делать в нашей области, давайте поговорим о действиях, которые помогают укрепить взаимоотношения и согласовать работу исследователя и усилия вендора по созданию и поддержке отличных продуктов.

Будьте пунктуальны и вежливы

Как мы упоминали ранее, одним из наиболее важных условий, которые вы должны соблюдать работая с исследователями, это быстрая реакция на их отчеты об уязвимостях. Подход подобный описанному, демонстрирует вашу благодарность за их труд и то, что вы воспринимаете их всерьез. Учитывайте тот факт, что исследователь потратил время и силы на подготовку отчета, его отправки и ожидание вашей реакции.

Когда вы впервые получили от исследователя отчет об уязвимости, вы должны немедленно подтвердить получение и поблагодарить его за работу. Это сразу задает тон будущего взаимодействия и демонстрирует, что вы цените его вклад. Вы можете это сделать, отправив простенький ответ, вроде *"Спасибо за ваше письмо. Мы ценим это. Мы с ним ознакомимся и ответим вам в течение нескольких дней."* Следуйте этому совету и для других предложений исследователя, отвечая на любые дополнительные вопросы без задержек.

Даже если отчет об уязвимости требует дополнительного изучения, ответ следует дать. Дайте исследователю знать, что его вклад оценен, даже если вы не можете предоставить детального ответа прямо сейчас. Такой подход, закладывает начало долгой работы по выстраиванию доверия и взаимопонимания.

Дайте понять, чего от вас стоит ожидать

Одна из вещей, которые мы ранее советовали исследователям, в *Главе 4, Раскрытие информации об уязвимостях — сообщение о результатах анализа безопасности*, касательно раскрытия информации об уязвимостях, это определиться с ожиданиями от сотрудничества с вендорами. Это точно так же работает и для вендоров. Ситуации, где исследователи оказались отвергнуты, часто возникают из-за обманутых ожиданий и непонимания. Как производитель программного обеспечения, вы должны вместе с исследователем обозначить реальные границы, относительно того, что вы сможете сделать, а что нет. Ожидаемые результаты должны быть сформулированы в рамках согласованных сроков.

Например, если исследователь направил вам отчет о критической проблеме безопасности, которая требует немедленного исправления, дайте ему знать о следующих этапах и когда ему ждать вашего ответа. С другой стороны, если исследователь сообщил о не слишком серьезной проблеме или о чем-то, что невозможно исправить немедленно, все равно ответьте ему, чтобы он не напрягался в ожидании ответа.

Еще одним критичным моментом, о котором следует помнить, является то, что не все исследователи хорошо знакомы с внутренней работой компании. И кстати, поясняйте любые задержки с ответом или причины по которым отдельная проблема не может быть исправлена в принципе.

Давайте понять, чего от вас стоит ожидать и соответствуйте этому, и вы заложите прочный фундамент, который несомненно поспособствует доверию и построению взаимопонимания с исследователями.

Сохраняйте каналы связи открытыми

Как мы упоминали ранее, одной из лучших вещей, которую вы можете сделать, как поставщик ПО, является поддержание открытыми каналов связи с вашими исследователями. Это подразумевает быть доступным для ответов на вопросы, информирование исследователя о прогрессе в ходе работ и обсуждение с ним новых идей.

Отличным способом поддержания открытыми каналов связи, является наличие контактов, предназначенных именно для исследователей, будь то отдельный человек или команда. Это доверенное лицо, должно быть постоянно на связи и быстро отвечать на запросы. В добавок, доверенное лицо должно быть уполномоченно принимать решения и действовать от имени исследователя.

В зависимости от серьезности проблемы и средств ее решения, доступных и исследователю, и вендору, еще одним способом поддержания коммуникации, является периодическая организация собраний или аудиоконференций, один на один или в составе группы. Это используется для информирования исследователей о ходе работ по их отчетам, ответам на любые вопросы или обсуждению новых идей.

Исправляйте уязвимости сразу же

Как только вы получили отчет об уязвимости, крайне важно действовать незамедлительно. Исследователи, как правило сильно огорчаются, если они обнаружили уязвимость и информировали о ней вендора, а он растянул ее исправление на месяцы или даже годы. Инертность вендора вызывает у исследователей ощущуние бессмысленности их работы и на продолжительный срок оставляет программное обеспечение уязвимым. Если ваша компания, не может исправить уязвимость незамедлительно, или в течение разумного времени, держите исследователя в курсе о продвижении работ и давайте реальные прогнозы. Исследователь может сообщить о своих ожиданиях, когда захочет передать вам результаты своих исследований.

Предлагайте вознаграждение и признание заслуг

Хотя не все поставщики ПО, обладают одинаковыми ресурсами, одним из способов простимулировать исследователей, является денежное вознаграждение. Одни из ключевых мотивов для исследователей – признание приложенных ими усилий и финансовая выгода. Множество известных вендоров, реализовали это с помощью программ баг-баунти или договоренностей с отдельными исследователями.

На всем протяжении исследования материалов для этой книги и изучения хороших взаимоотношений между вендорами и исследователями, мы выслушали некоторые рассказы, в которых поставщики ПО иногда могли предлагать оплату консультаций исследователя. Это привлекает исследователей, рассказывать сотрудникам компании об исследованиях в области информационной безопасности.

Вознаграждение, необязательно нужно выражать в виде денег или трудоустройства. Вместо этого, некоторые исследователи рассчитывают на благодарность в виде разных фишек, представляющих для них ценность, включая такие вещи как хавчик на всю толпу, напитки, футболки, шляпы и кружки.

Хотя вообще-то, найм или оплата труда исследователей необязательны. Методы нематериального поощрения, также могут эффективно мотивировать исследователей. Например

публичное признание результатов работы исследователя, отличный способ выразить благодарность и подстегнуть интерес к дальнейшему сотрудничеству. Признание заслуг исследователя, может выражаться в публикации в блоге, разрывном сюжете в соцсети, актуальных списков контрибьюторов в сфере безопасности или даже простое "Спасибо" по электронной почте, где вендор поблагодарит за поддержку.

Другим нестандартным способом выстраивания взаимоотношений и стимуляции исследователей, является предоставление доступа к новым функциям или программным продуктам, до официального релиза и просьба поучаствовать в оценке безопасности продукта. Такой вид раннего доступа, очень мотивирует исследователей и показывает, что вы цените их вклад.

Успешные компании, которым удалось построить определенно выгодные взаимоотношения с исследователями, использовали несколько методов, если вообще не все. Однако, если вы помните, что мотивирует исследователей – награждайте их деньгами, плюшками и что боле важно, признанием их заслуг в самых разных видах.

Будьте открыты

Еще одним основополагающим условием при работе с исследователями, является открытость. Открытость, подразумевает быть честным относительно ситуации с вашим программным обеспечением и публично делиться инфомацией о проблемах безопасности. Такая открытость, показывает, что вы приглашаете исследователей к сотрудничеству и она помогает построению доверия между вами. Дополнительный ключевой момент, это держать исследователя в курсе о ходе работ по исправлению уязвимости. Привлечение исследователей, на всем протяжении процесса исправления уязвимости, поможет им прочувствовать, какую пользу они приносят и подтолкнет к продолжению сотрудничества с вами.

Признание независимости

Исследователи обладают независимостью, и частью поддержания прочных взаимоотношений вендор-исследователь, является уважение этого. Признание независимости исследователя показывает, что вы уважаете его труд и охотнее готовы работать с ним, чем против него. Один из способов сделать это – позволить исследователям выбирать, каким образом их имена будут перечисленны в публичных отчетах об устранении уязвимости. Если исследователь, предпочитает остаться инкогнито, уважьте такую просьбу. Дополнительно, попробуйте дать исследователю возможность контролировать, когда и каким образом будет публиковаться их отчет. Предоставление исследователям права голоса показывает, что вы цените их вклад и заинтересованны в сотрудничестве.

Если между исследователем и вендором, существуют разногласия, признайте, что у исследователя свое видение проблемы и попытайтесь посмотреть на нее с его точки зрения. Также признавайте, что исследователь может делать и говорить то, что считает нужным. Например, распространенным поводом для конфликта, является, когда исследователь собирается опубликовать результаты своих изысканий. Какой-то вендор может хотеть, чтобы исследователь либо подождал с публикацией, пока не исправят баг, либо вообще никогда опубликовал результаты исследований. С другой стороны, исследователи могут почувствовать, что раскрытие информации о результатах исследований, является лучшим вкладом в защиту общества. В этом случае, при отсутствии юридического соглашения, важно помнить, что в вопросе раскрывать ли информацию о результатах исследования, и каким именно образом – последнее слово остается за исследователем. Излишнее ограничение свободы действий, создает неприятные прецеденты и исследователи, скорее всего опубликуют результаты исследования, параллельно с красочными подробностями о том, как вы пытались замять их исследование.

В основном в интересах, как поставщиков ПО, так и исследователей – согласовать процедуру публикации, до того, как работа будет завершена. Однако, если это не представляется возможным, попытайтесь понять позицию исследователя и помнить, что он скорее всего действовал из лучших побуждений.

Теперь мы имеем неплохой свод правил, что делать, а что нет, во время совместной работы с исследователями. Однако, как мы установили в предыдущем разделе по использованию возможностей исследователя, создание ответственной политики раскрытия информации – один из лучших путей установления хороших отношений. Применение того, что мы только что узнали, прибавит нашей политике конструктивности и выстроит позитивные отношения с исследователями. А сейчас, давайте посмотрим, что из себя представляют такие политики и как их создавать.

Разрабатываем политику ответственного раскрытия информации

Политика ответственного раскрытия информации, это набор методических рекомендаций, которые устанавливают, каким образом компания будет поступать с отчетами об уязвимостях. Например, политика должна устанавливать, кто может докладывать о проблемах, каким образом им следует это делать и какую информацию необходимо включить в отчет. Следует также определить, какие дефекты принимаются для дальнейшего раскрытия информации и в какой срок, компания планирует закрывать уязвимости.

Существует множество способов, сформулировать политику раскрытия информации, однако все политики должны стремиться к балансу между необходимостью защиты пользователей и достойной оценкой труда исследователей.

Политика ответственного раскрытия информации, поможет выстроить доверительные отношения между компанией и сообществом исследователей в сфере безопасности. Это также в целом улучшит безопасность программных продуктов компании, гарантируя, что все отчеты об уязвимостях воспримут серьезно и отреагируют своевременно.

Выстраивание доверительных отношений с исследователями, часто предусматривается политикой компаний, действующими в духе сотрудничества, уважения и открытости, которые чаще награждают исследователей, в наиболее удобной для них форме и воздерживаются от порицания. Имея политику раскрытия информации, вы можете дать понять, чего вы хотите от сотрудничающих с вами исследователей, и какого вида информацию, вы хотите от них получать. Политики подобные этой, помогают избежать недопонимания и ненужных конфликтов во взаимоотношениях вендор-исследователь.

Существует несколько ключевых компонентов, которые должны быть включены в каждую политику ответственного раскрытия информации:

- Четкая формулировка политики компании в отношении безопасности и конфиденциальности. Эта политика, должна демонстрировать, что компания серьезно относится к безопасности своих пользователей и их данных.
- Описание типов уязвимостей, попадающих под эту политику.
- Четкое утверждение, что компания не приемлет противозаконной деятельности, типа взломов без явно предоставленного разрешения. Если подразумевалось, что исследователям предоставлено разрешение в этом документе, этот пункт может включать перечень разрешенных действий.
- Общее описание процедуры подачи отчета об уязвимости, включая контактную информацию для заинтересованных лиц. Исследователи должны знать, с кем связаться и как предоставить отчет на рассмотрение, без хождения по инстанциям.

- Качественная политика раскрытия информации, как правило имеет раздел с перечнем разрешений, в котором определяется, что именно исследователи могут делать, не опасаясь судебного преследования за передачу компании, результатов своих исследований.
- Лучшие политики, включают графики реагирования на отчеты об уязвимостях и принятия соответствующих мер. Исследователи должны знать, когда ждать ответа и когда проблема будет исправлена.
- Сформулируйте, какого рода информацией вы будете делиться с исследователем после того, как вы подтвердили наличие проблемы. Например, это может включать информацию о том, когда будет выпущено исправление или будут ли отмечены заслуги исследователя при публичном раскрытии информации.

Помните, что ваша политика раскрытия информации должна быть сбалансированной между привлечением исследователей и защитой интересов вашей компании. Важно также запомнить, что ваша политика не "отлита в бронзе". Вы можете корректировать ее, когда понадобится, основываясь на отзывах исследовательского сообщества.

С пониманием основополагающих принципов ответственной политики раскрытия информации, ее предназначения и того, что в нее должно быть включено, давайте уделим некоторое время анализу примера такой политики.

Пример политики – политика ответственного раскрытия информации Acme Logistics

Есть много чего, что могла бы содержать ответственная политика раскрытия информации. До сих пор, лучшие политики – являются доступными пониманию людям без знания юриспруденции, состоят всего из нескольких листов и четко определяют параметры взаимодействия исследователя с компанией. В нашем следующем примере, мы исследуем воображаемую политику **Acme Logistics**, компании с работающей политикой ответственного раскрытия информации, которая демонстрирует лучшие практики, рассмотренные в этой главе ранее. Отметим, что примечания автора, выделены курсивом, чтобы дополнительно пояснить контекст раздела, который вы читаете.

Введение

Каждая политика раскрытия информации, должна начинаться с введения. Оно должно отражать ценности, представляемой организации. Этот раздел должен быть выдержан в целеустремленном, заинтересованном и восприимчивом тоне.

Мы, Acme Logistics – ценим безопасность наших клиентов и граждан, больше всего остального. Поэтому, мы применяем строгие меры по защите секретной информации и ценим любое участие, которое сделает их более эффективными для тех, кому мы предоставляем услуги. Назначение этого документа, предоставить исследователям безопасности, фреймворк для проведения исследовательской деятельности, вроде сканирования на уязвимости или тестирование программ на баги. Эта политика описывает, как правильно связаться с нами по любой информации, относящейся к уязвимостям и дает рекомендации, как и когда результаты исследования должны, обнародоваться исследователем, если он того желает. Мы приглашаем всех исследователей связываться с нами, чтобы сообщать о возможных уязвимостях в наших системах и с нетерпением ждем сотрудничества с исследователями, которые помогут закрыть незамеченные баги в наших системах.

Разрешенное применение

Этот раздел подразумевает установление положения, что исследователь, действующий с благими целями, может рассчитывать, что не столкнется с юридическими последствиями своих действий, в виде судебного преследования или гражданских исков.

Если исследователи безопасности, следуют рекомендациям из этого документа, мы предоставим им разрешения в разделах *Рекомендации по применению* и *Определение области действия*. Исследователи должны прилагать все возможные усилия, по соблюдению этих ограничений и сообщать нам, если их исследование начинает от них отклоняться.

Acme Logistics обещает помочь разрешить любые проблемы обнаруженные исследователями и не будет возбуждать против них уголовных дел. Если другая компания предъявит вам юридические претензии за ваши действия, в рамках разрешенных этой политикой, мы подтвердим ваши полномочия по требованию исследователя.

Рекомендации по применению

Рекомендации по применению должны обрисовать исследователям, что они должны оставаться в границах своих полномочий, чтобы проводить исследование.

Согласие с данной политикой, подтверждает, что вы придерживаетесь в своем исследовании следующих принципов и обязуетесь выполнять следующее:

- Уведомлять нас, используя методы изложенные в политике, относительно любых уязвимостей безопасности или угрозы их обнаружения в течение вашего исследования.
- Предоставлять Acme Logistics, разумное время на исправление любых уязвимостей и угроз до публичного раскрытия информации или публикации в базах уязвимостей, типа списка CVE MITRE и **national vulnerability database** (**NVD**).
- Применять методики исследования, не наносящие ущерба, то есть избегать возникновения прецедентов повреждения систем.
- Воздерживаться от нарушения конфиденциальности пользователей, в плане использования их идентификационных данных, паролей или конфиденциальной информации, потенциально небезопасными способами.
- Не продавать уязвимости или информацию, связанную с нашими системами и предоставляемыми услугами третьим лицам, типа брокеров уязвимостей.
- Не применять без нужды, техники эксплуатации, могущие нанести ущерб или слишком перегрузить сотрудников техподдержки Acme Logistics.
- Воздерживаться от обмана, применяя социальную инженерию на работниках Acme Logistics или ее клиентах.

По сути проверка безопасности, в которой установлены жесткие границы, запрещает такие техники, как:

- Социальная инженерия
- DOS-атаки
- Брутфорс паролей
- Применение глючных эксплоитов, которые, как известно могут стать причиной нестабильной работы системы

Если вы в ходе своего исследования, предполагаете, что добрались или увидели любую секретную информацию, вроде коммерческой тайны, финансовой информации, персональных данных или чего угодно, имеющего владельца – исследование должно быть прекращено и вы обязаны незамедлительно уведомить об этом Acme Logistics. Ни при каких обстоятельствах, вы не должны

обнародовать или распространять информацию о деталях или содержании исследования под угрозой ответственности за нарушение требований данной политики.

Определение области действия

Этот раздел обозначает область, в которой могут быть протестированны взаимосвязанные системы (или любые), если это предписывается политикой.

Наша политика, ограничивает исследование тем, что определено, как *внутренняя область для тестирования и оценки*. Область, предназначенных для исследования сервисов, доступных из интернета, включает следующие домены:

- ***.acme-logistics.com**
- ***.acme-mail.com**
- ***.acme-delivery.com**

Сервисы и хосты, обнаруженные во время изучения доменов обозначенной области, так же будут считаться находящимися в области исследования. Эта политика также применима, к нашим коммерческим продуктам, поставляемых в виде программного обеспечения и к нашим компонентам с открытым исходным кодом, входящим в различные библиотеки для разработки.

Любые другие обнаруженные ресурсы, которые могут принадлежать к Acme Logistics будут считаться находящимися за пределами области исследования. Если вы желаете обсудить, объекты не входящие в область исследования, пожалуйста сделайте это, до проведения тестирования и исследований, написав нам на **security@acme-logstics.com**. Если отдельная система не входит в область исследования, которую вы считаете нужным протестировать, пожалуйста, свяжитесь сначала с нами, чтобы это обсудить. Мы распространим область исследования, еще на одно наше подразделение, для этой политики в целом или в рамках дополнительного соглашения с отдельным исследователем.

Предоставление информации об уязвимости

Этот раздел описывает, как бы вы принимали информацию об исследованиях. Убедитесь, что в этом разделе вы четко определили, чего вы ожидаете и, что вам требуется от исследователей, для подтверждения результатов их исследований.

Мы принимаем отчеты об уязвимостях, на наш основной электронный адрес для любых вопросов связанных с безопасностью: **security@acme-logistics.com**. Что касается вашего обращения, оно не предполагает, что исследователь обязан втягиваться в дальнейшее обсуждение по любым раскрытиям информации и по сути, отчет может быть предоставлен анонимно. Если вместе со своим отчетом об уязвимости, вы решили поделиться своей контактной информацией, мы подтвердим получение вашего отчета в течение 14 календарных дней с даты отправки.

Информация поданная, в соответствии с данной политикой, будет использоваться только для защитных мер по снижению возможного ущерба или исправлению уязвимостей. Если вы присылаете отчет об уязвимости, которой подвержен партнер Acme Logistics, мы можем поделиться результатом ваших исследований с подверженным уязвимости партнером. В процессе нашего общения по уязвимостям, мы не будем разглашать ваше имя или публиковать содержание вашей работы, за исключением явно предоставленных Acme Logistics разрешений.

Мы ожидаем, что ваш отчет должен четко и понятно описывать уязвимость, формулировать представляемые ей угрозы и места, где можно увидеть уязвимые продукты. Ваш отчет об уязвимости, также должен включать инструкции по воссозданию условий для проявления уязвимости со скриншотами и соответствующими наглядными примерами.

Какой реакции ждать исследователю

Этот раздел, поможет исследователю узнать, чего ждать от организации после того, как они отправили вам результаты исследований. Эта информация должна соответствовать реальным практическим наработкам, которые строго соблюдаются.

Уважаемый исследователь, мы принимаем перед тобой обязательства, что означенная политика гарантирует, наше серьезное отношение к вашему исследованию и вам следует ожидать соблюдения нами графиков работ по любым исследованиям, которыми вы с нами поделитесь:

- Мы подтвердим получение вашего отчета об уязвимости в течение 14 календарных дней с даты отправки. В ответном письме, вы можете ожидать предположительный срок, когда исследование будет изучено и проверено Acme Logistics.
- Мы будем делиться информацией о нашей работе по проверке вашего отчета, запрашиваать помощь и дополнительные пояснения, если вы изначально сообщили нам свои контактные данные.
- Как только ваш отчет будет проверен, мы определим степень серьезности найденой вами уязвимости, с помощью **Единой Системы Оценки Уязвимостей** (**Common Vulnerability Scoring System - CVSS**). Если угроза оценена, как ***informational***, мы не будем предпринимать каких-либо действий и информируем вас об этом.
- Как только степень серьезности уязвимости будет определена, мы будем работать с вами над созданием эффективного плана ее исправления, который будет включать ожидаемые даты по каждому этапу.
- Процесс подтверждения, проверки и планирования исправления, займет не более 60 календарных дней.
- Мы остаемся открытыми для желающих принять участие и ответим на любой вопрос или просьбу освежить информацию в течение 14 календарных дней с момента первого контакта.

Вознаграждение исследователя и признание его заслуг

Исследователи, часто хотят знать, в чем конкретно будет выражаться признание их заслуг компанией, когда они опубликуют результаты своих исследований, и будут ли они вознаграждены вами за их участие. Крайне важно, установить график который показывает, когда ожидать публикации и была ли она признана полезной. Возьмите на себя ответственность за это, предоставляйте исследователю разрешение на публикацию, если организация упустила этот момент в своей политике. Исследователи будут очень благодарны за это.

Acme Logistics осведомлены о времени и силах исследователей, затраченых на изучение наших продуктов, чтобы сделать их безопаснее для наших клиентов. Помня об этом, мы поддерживаем публикацию всех исследований в сторонних базах уязвимостей или на общедоступных веб-сайтах, рассчитывая, что исследователь согласен выдержать следующие отрезки времени до публикации:

60 суток, включая изначальное подтверждение получения отчета, его проверку и планирование этапов исправления, которые могут быть продлены следующим образом:

- 30 дополнительных дней на решение критических уязвимостей
- 60 дополнительных дней на решение серьезных уязвимостей
- 90 дополнительных дней на решение уязвимостей средней степени опасности
- 180 дополнительных дней на решение уязвимостей представляющих небольшую опасность

Такое распределение времени, предоставит Acme Logistics достаточное количество времени на ответ и исправление уязвимостей, следуя нашим внутренним правилам. Acme Logistics оставляет за собой право запрашивать дополнительное время, в зависимости от объема работ, в размере равном

изначально установленному времени. Если Acme Logistics не выполнят эти обязательства, исследователь в праве опубликовать результаты своих исследований, когда сочтет нужным, без каких-либо последствий.

Мы просим, чтобы при любом раскрытии информации, вы согласились продумать ограничения, в том плане, чтобы не раскрывать уязвимость во всех подробностях.

Все раскрытия информации об исследованиях, будут подтверждены в замечаниях к релизу. В добавок, перед публикацией, мы уточним, хотите ли вы скорректировать свое имя или любую другую информацию, относящуюся к вашему исследованию. Acme Logistics, в рамках своих правил, не выплачивает денежного вознаграждения исследователям. Однако, в исключительных случаях, когда исследование приносит большую пользу организации, Acme Logistics может наградить исследователя фирменной одеждой, памятными знаками за проведение исследования и иногда единовременной выплатой за полезное исследование.

Вопросы и обратная связь

В этом разделе, организация заканчивает описание политики с искренней благодарностью, предлагает задавать вопросы, для получения более подробной информации и просит предоставить отзывы об этой политике.

Огромное вам спасибо, что вы прочитали нашу политику и если вы планируете исследовать наши ресурсы в будущем, мы с нетерпением ожидаем построения более безопасного будущего для наших программных продуктов и организации с вашей помощью. Если у вас есть любые вопросы, связанные с этой политикой, вы можете их прислать на наш основную почту, по вопросам безопасности **security@acmelogistics.com**. В добавок, мы приветствуем рекомендации по совершенствованию нашей политики. Если вы видите полезный для нас вариант улучшить эту политику, пожалуйста, дайте нам знать, как мы можем выстроить лучшие отношения с вами и замечательным сообществом исследователей информационной безопасности.

Заключительные мысли по эффективным политикам

Мы надеемся, что этот пример политики ответственного раскрытия информации, пролил свет на что-то, что вы могли видеть в вашей собственной компании. Мы разработали эту политику, проанализировав политики, существующие в Соединенных Штатах, в которых созданы эффективные комплексы предложений и рекомендаций для исследователей безопасности. Не стесняйтесь копировать и подгонять ее под себя, когда будете разрабатывать свою политику ответственного раскрытия. Однако продвигаясь в создании своей политики или ее совершенствовании, консультируйтесь по политике со своим руководством. Политики опубликованные, отделом информационной безопасности самостоятельно, редко оказываются эффективными без поддержки людей, занимающих руководящие должности. Слушайте и принимайте советы сотрудников, которые могут видеть потенциальные угрозы или советы юридического отдела, желающего снизить возможную ответственность организации. Это те лица, которые могут помочь оценить и изменить политику ответственного раскрытия информации в сторону, лучше соответствующую целям вашей организации и гарантированно подходящую для всех подразделений компании.

Наша задача по изучению политики ответственного раскрытия информации завершена, теперь давайте подытожим, чему мы научились в этой главе.

Итоги главы

В этой главе, мы уделили время рассмотрению исследований в сфере безопасности, глазами компаний, которые могли многого не знать о том, кто такие исследователи безопасности. В первую

очередь, мы помогли подробно разобраться в характеристиках исследователей, их навыках и мотивации. Затем мы рассмотрели, каким образом мы могли бы применить знания и опыт исследователей, чтобы помочь некоей организации улучшить свои продукты и сервисы, сделав их более безопасными. После этого мы охватили некоторые из наиболее распространенных способов выстраивания доверительных отношений с исследователями и вещей которых стоит избегать, во избежание разрушения взаимовыгодного сотрудничества. И наконец, мы изучили пример политики ответственного раскрытия информации, разработанной на базе ряда политик, используемых крупными организациями, правительственными учреждениями и некоммерческими организациями. Закрывая эту главу и двигаясь дальше к заключительной части этой книги, мы уделим время предоставляя исследователям шаблоны для использования во взаимодействии с вендорами и базами уязвимостей. В конце, мы затронем кое-какие заключительные рекомендации и справочные материалы, и завершим нашу книгу напутствием автора.

Глава 10. Шаблоны, ресурсы и заключительные наставления

На протяжении этой книги, мы изучили и создали базу знаний об уязвимостях, их жизненном цикле, а также каким образом начать заниматься исследованиями. Затем мы рассмотрели методы, применяемые для передачи и опубликования данных по раскрытию информации об уязвимостях. Продвигаясь дальше, мы обсудили, как нам найти посредника и привлечь для помощи третьих лиц, а также как опубликовать информацию об уязвимости самостоятельно. Позже, мы рассмотрели несколько реальных сценариев, указав на трудности, с которыми столкнулись исследователи. И в заключение, мы обсудили раскрытие информации с точки зрения вендора и то, как лучше создать атмосферу взаимовыгодного сотрудничества.

Я искренне надеюсь, что вам понравился этот материал. Я желаю, чтобы это помогло вашему пониманию того, как просто исследование можно превратить в опубликованную информацию об уязвимости. Сейчас, вооружившись этими знаниями, вы должны быть лучше подготовлены к началу своего пути в качестве исследователя и к трудностям, с которыми вы возможно столкнетесь.

Эта заключительная глава, задумывалась, как приложение к книге, в виде дополнительных материалов, таких как, шаблоны исследований, шаблоны переписки и организационные средства, которые помогут вам управлять и отслеживать обнаруженные вами уязвимости. Окончательно завершая эту главу, мы предложим вам несколько слов напутствия от автора этой книги.

Итак, чтобы подытожить, что вы увидите в этой главе – мы рассмотрим в ней следующее:

- Шаблоны тестов для исследования уязвимостей
- Шаблоны для связи по электронной почте
- Шаблоны и средства для публикации уязвимостей и резервирование CVE ID
- Шаблоны организационных моментов
- Заключительные положения и размышления о трудностях связанных с этой работой

В этой главе, мы рассмотрим шаблоны, которые могут быть использованы, чтобы помочь вам, когда вы соберетесь проводить ваше исследование, передавать его поставщику ПО и затем опубликовывать информацию об уязвимости. Эти результаты будут представлены на протяжении следующих основных тем этой главы:

- Шаблоны тестов для выполнения исследований
- Шаблоны для связи с вендором по электронной почте
- Шаблоны CVE
- Организационные моменты

Давайте начнем эту главу с шаблонов тестов для выполнения исследований, подробно рассмотрев, как это систематезировано в OWASP Testing Guide (v 4.2).

Шаблоны тестов для выполнения исследований

Поскольку эта глава, включает уже рассмотренные исследования, нам нет нужды слишком углубляться в изучение тестов. В *Главе 3, Исследование уязвимостей - начнем с проверенных стратегий*, мы обсуждали и показывали, как создавать тесты в **CherryTree**. Наши тесты, содержали небольшую базу заметок по тесту, описания и статусы. Мы хотели бы подготовить вас к тестированию наиболее распространенных типов веб-приложений, чтобы вы могли воспользоваться информацией, систематизированной в OWASP Testing Guide (v 4.2) непосредственно в CherryTree. Как видно на следующем скриншоте, мы разбили тесты на категории в порядке выполнения тестирования. Мы включили информацию о тестах и методики тестирования, выработанные лучшими экспертами

индустрии, а также добавили разделы заметок по тесту, чтобы вы могли сверяться с тем, как ваши тесты завершаются. Мы настоятельно рекомендуем, взять этот шаблон и применять к своим задачам:

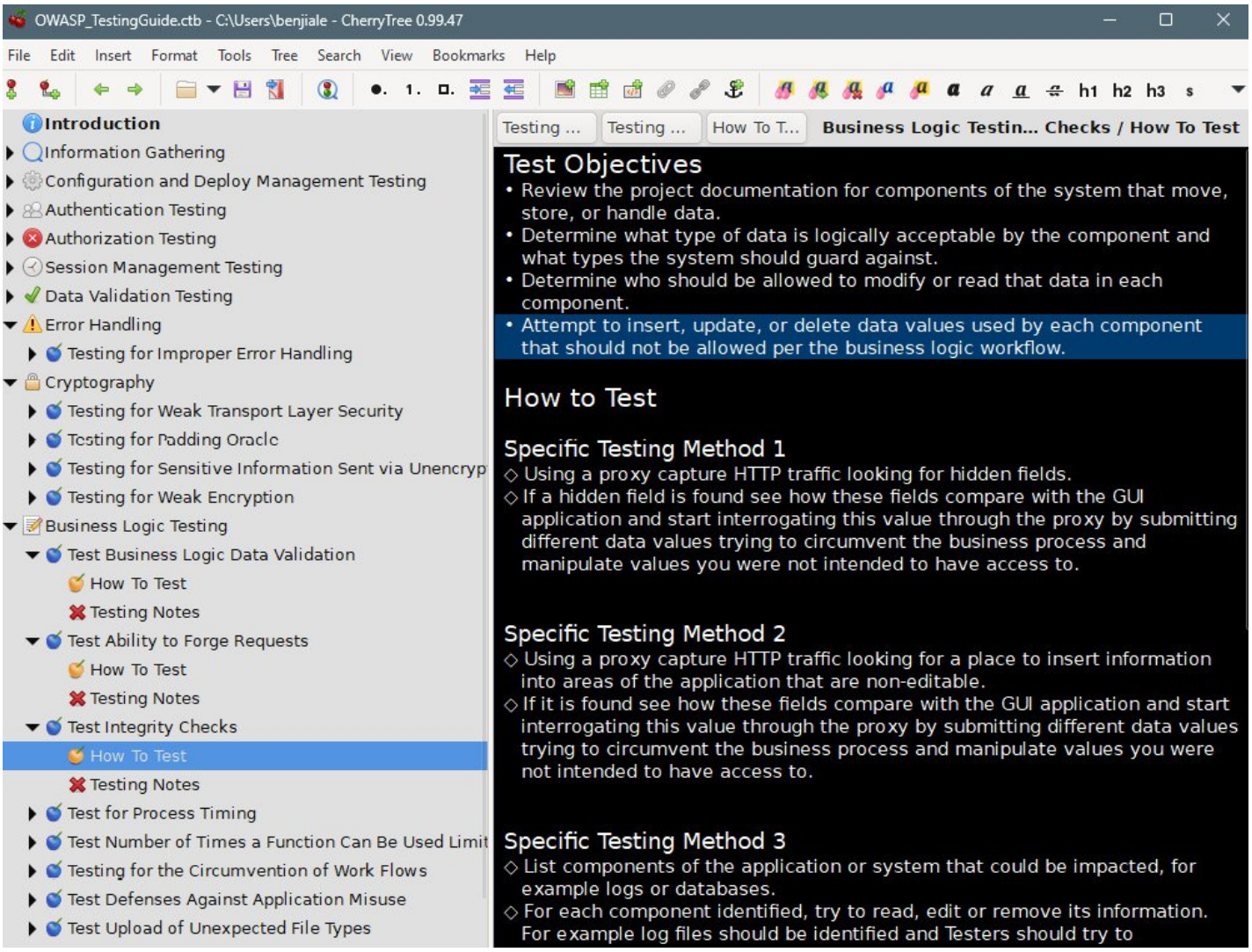


Рис. 10.1 – Обзор шаблона тестового пакета для веб-приложений

Здесь вы можете забрать копию этого шаблона и начать конфигурировать его как вам нравится.

В этом примере, мы будем использовать Linux, однако вы можете использовать Windows или macOS. Нам нужно клонировать следующий репозиторий, вот этой командой:

```
git clone https://github.com/BallinBallen/OWASP-Testing-Guide-Cherry-Tree-Template.git
```

Либо, вы можете зайти в репозиторий, прямо из веб-браузера, и загрузить копию шаблона, отсюда: https://github.com/BallinBallen/OWASP-Testing-Guide-Cherry-Tree-Template.

**Ссылки на скачивание шаблона битые, оставляю для справки. Если этот шаблон, вообще существовал и кто-то успел его скачать, выложите его пожалуйста в теме с книгой.*

Вот несколько подходящих шаблонов по теме:

https://guide.offsecnewbie.com/cherrytree-oscp-template

https://411hall.github.io/assets/files/CTF_template.ctb

https://github.com/CyberSecurityUP/Template-CherryTree-PenTest

https://github.com/securiters/CherryTree_Templates

Для использования этого шаблона, запустите программу CherryTree. Если вы используете, что-то вроде Kali Linux, эта программа скорее всего предустановлена в операционной системе:

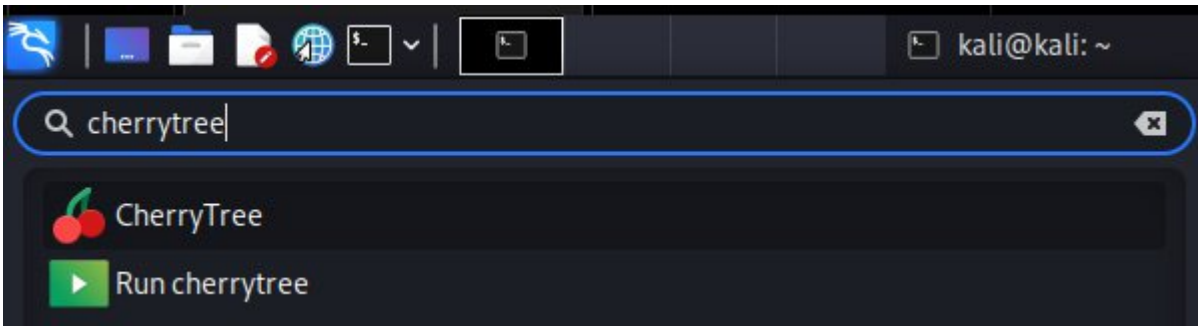


Рис. 10.2 – Открытие CherryTree

Примечание

Если вы используете Windows, macOS или другой дистрибутив Linux, вы можете загрузить и сконфигурировать CherryTree, посетив репозиторий GitHub, где хранятся исходники и сборки приложения: https://github.com/giuspen/cherrytree.

Как только CherryTree откроется, укажите ей файл, который только что получили клонировав репозиторий, как показано на следующем скриншоте:

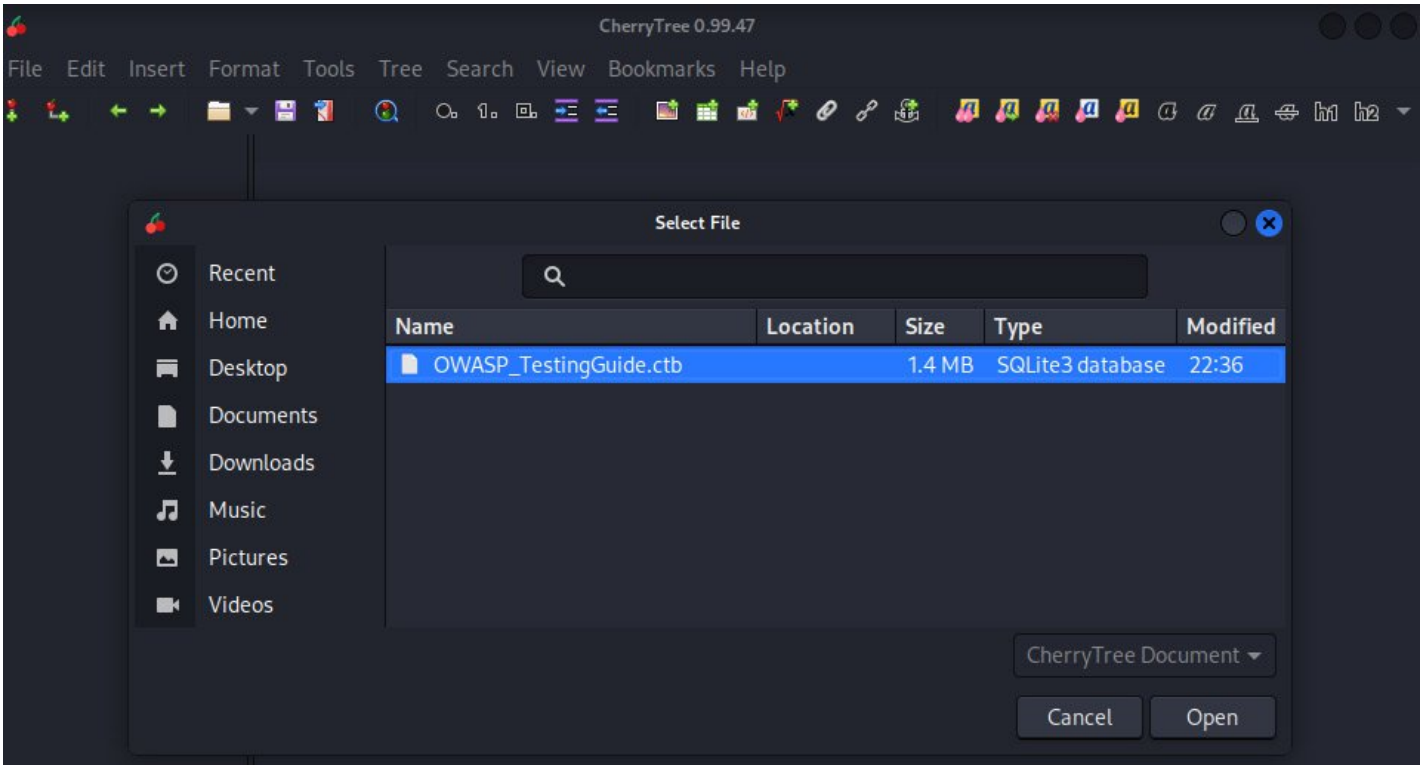


Рис. 10.3 – Открытие шаблона

Как только он откроется, начните с изучения вводной части, в которой рассматриваются основы использования и даются рекомендации тестировщику по использованию шаблона для создания тестового пакета, соответствующего именно вашим целям, как изображено на Рис. 10.4:

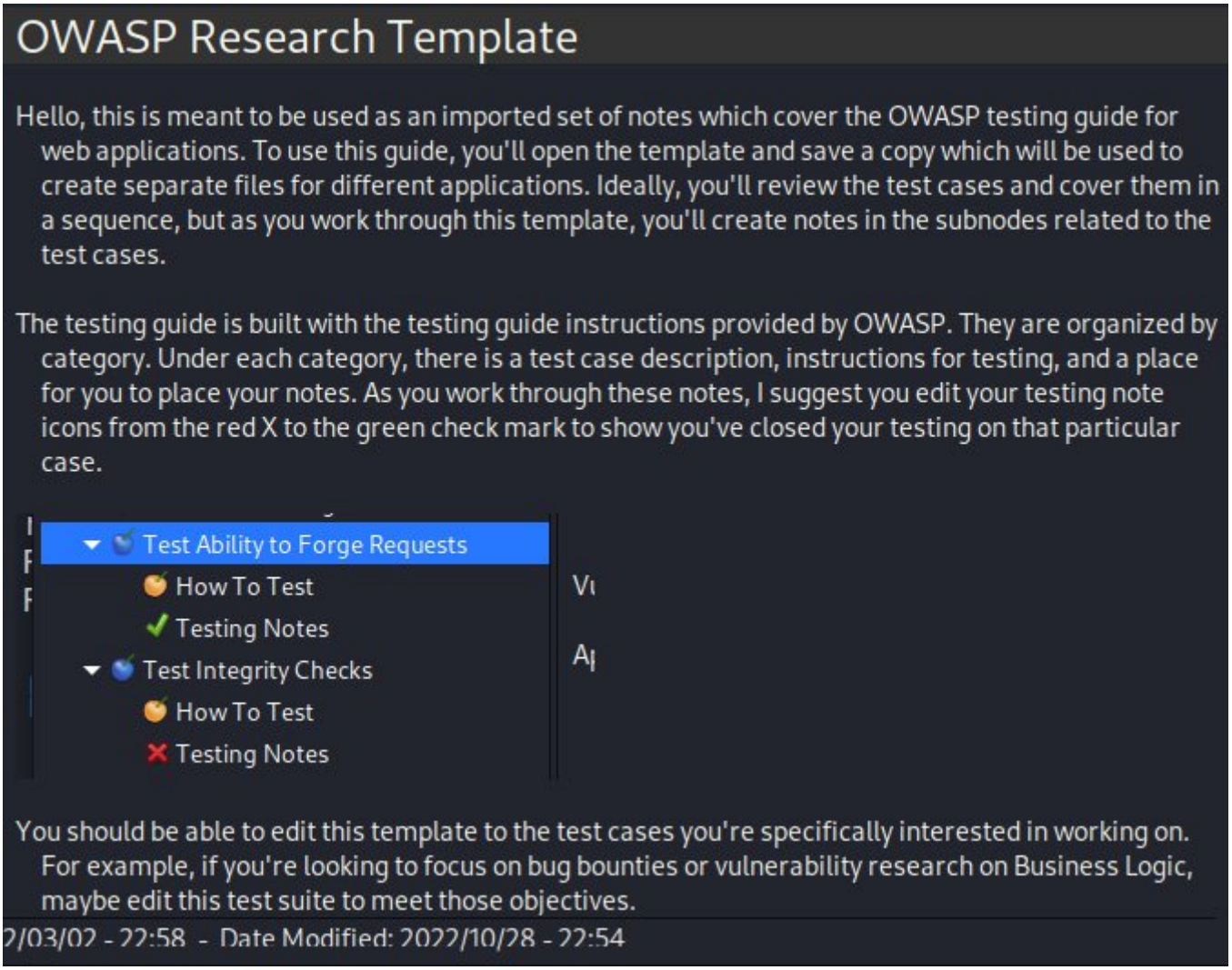


Рис. 10.4 – Стартовая страница с информацией об использовании шаблона

Далее, мы рекомендуем вам уделить время, детальному рассмотрению тестов, чтобы получше ознакомиться, как они выглядят и какова их структура:

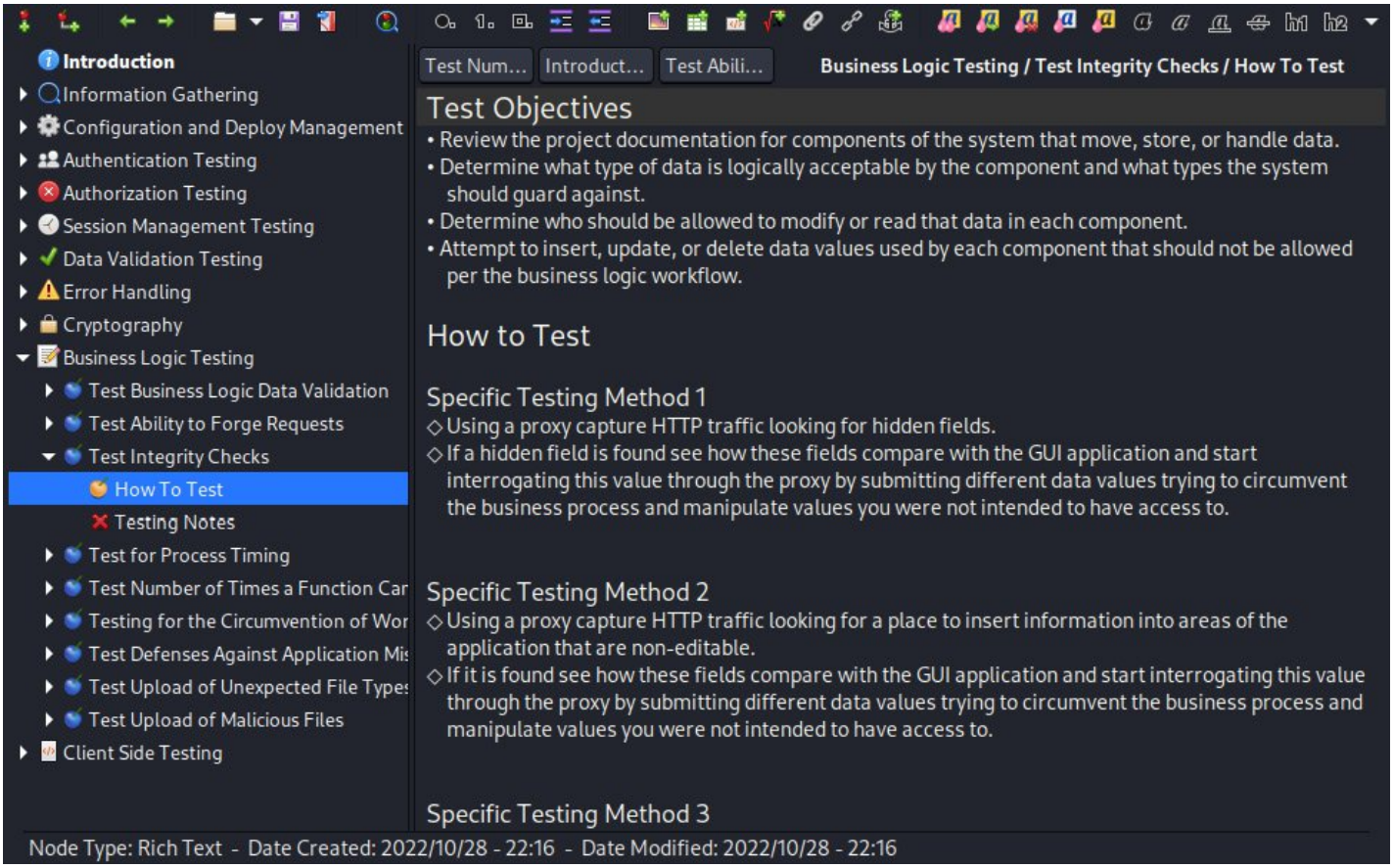


Рис. 10.5 – Инструкции по проведению тестов, можно найти для каждого из них

Когда вы будете готовы, начать использовать этот шаблон на практике, сохраните его копию в отдельный файл, под именем приложения, которое вы будете тестировать. Заметки к тестам – не заполнены, общая последовательность действий, заключается в изучении тестов и инструкций, и заполнении заметок к тесту – уже вашими данными.

Если вы ищете небольшие, возможно более сжатые шаблоны OWASP, мы советуем просмотреть список, опубликованный **tanprathan**. Это прекрасная краткая таблица, содержащая все отдельные низкоуровневые случаи, которые могло бы понадобиться отработать. Однако, она не предоставляет места для сохранения заметок об исследовании:

47	Authorization Testing	Test Name	Description	Tools	Result	Rema
48	OTG-AUTHZ-001	Testing Directory traversal/file include	*dot-dot-slash attack (../), Directory traversal, Local File inclusion/Remote File Inclusion.*	*Burp Proxy, ZAP, Wfuzz*	Not Started	
49	OTG-AUTHZ-002	Testing for bypassing authorization schema	*Access a resource without authentication?, Bypass ACL, Force browsing (/admin/adduser.jsp)*	*Burp Proxy (Autorize), ZAP*	Not Started	
50	OTG-AUTHZ-003	Testing for Privilege Escalation	*Testing for role/privilege manipulate the values of hidden variables. Change some param groupid=2 to groupid=1*	*Burp Proxy (Autorize), ZAP*	Not Started	
51	OTG-AUTHZ-004	Testing for Insecure Direct Object References	*Force changing parameter value (?invoice=123 -> ?invoice=456)*	*Burp Proxy (Autorize), ZAP*	Not Started	

Рис. 10.6 – Некоторые исследователи, предпочитают облегченные способы документирования, типа таблиц Excell

Вы можете найти этот шаблон, в следующем репозитории GitHub: https://github.com/tanprathan/OWASP-Testing-Checklist.

Хотя эти шаблоны исследований и не охватывают всех возможных вариантов по конкретному ПО, вы можете почерпнуть из них здравые идеи, которые можно применить конкретно в ваших исследованиях. Не забывайте об этом, создавая свой план, применяйте эти инструменты для составления плана действий, а потом действуйте.

Закончив с шаблонами тестов, давайте теперь обратимся к шаблонам коммуникации, которые нацелены помочь упорядочить, возможности по взаимодействию с вендорами.

Шаблоны для связи с вендором по электронной почте

При первоначальном раскрытии информации об уязвимостях поставщикам ПО, мы можем столкнуться с различными сценариями, в случае отсутствия политик раскрытия. В этом разделе, мы дадим вам несколько шаблонов составления писем, которые вы сможете взять за основу для создания своих.

Представляемся по email компании, не предоставляющей политики раскрытия проблем безопасности

В этом примере шаблона, мы представимся компании, не предоставляющей политики раскрытия информации на своем веб-сайте/в контактной информации. Это тот случай, когда краткость – сестра таланта, здесь не нужно раглашать все подробности по факту того, что вы обнаружили:

Здравствуйте,

Меня зовут {имя исследователя}, и я исследователь в сфере информационной безопасности. Недавно я обнаружил потенциальную уязвимость в программном обеспечении/веб-сайте вашей компании. Я захотел связаться с вами и дать вам об этом знать, а заодно и посмотреть, обрабатывает ли компания такие обращения.

Я был бы рад, предоставить дополнительные подробности и поработать с вашей командой над решением этой проблемы. Из-за деликатного характера этой информации, я хотел бы передать ее вашей службе информационной безопасности, для первоначальной проверки. Есть ли у вас сотрудник или отдел, которому было бы лучше передать эту информацию?

Надеюсь на ваш ответ, в ближайшее время. Я свяжусь вами через 30 дней, если не получу ответа от ваших сотрудников, насчет наших дальнейших действий. Если вам необходимы любое содействие или помощь, свяжитесь со мной пожалуйста по электронной почте {адрес вашей почты}.

С уважением,

{имя исследователя}

Как только, вы получите от вендора ответ с нужными контактами, либо они захотят получить от вас детали непосредственно – вы можете взять за основу, шаблон раскрытия информации об уязвимости применяемом в политиках безопасности:

Здравствуйте,

Я пишу, после получения вашей контактной информации от вашего коллеги, с которым я переписывался изначально. Меня зовут {имя исследователя}, и я хотел бы поделиться информацией о некоторых уязвимостях, которые я обнаружил, рассматривая {наименование ПО}, с точки зрения безопасности.

В {наименование ПО}, а именно в версии {номер версии} – существует {кол-во уязвимостей} уязвимостей. В следующем разделе, я опишу эти уязвимости:

Уязвимость 1

{Наименование и ID, указанное в CWE(Common Weakness Enumeration – Список Распространенных Уязвимостей)}

Эта уязвимость найдена в {где была найдена уязвимость} части приложения. Результатом этого дефекта, может оказаться {описание угрозы из CWE}, который может повредить вашим клиентам в {сценарий, как эта уязвимость могла бы повредить пользователям}.

Для воссоздания этой уязвимости, вы можете проделать следующие шаги, которые я сопроводил скриншотами, чтобы помочь проиллюстрировать proof-of-concept.

{Действия по воссозданию условий для уязвимости}

Уязвимость 2

{Дополнительные уязвимости}

В качестве части моего исследования, я планирую {написать в блоге / опубликовать CVE}, поэтому я бы с удовольствием принял любую помощь от участников, когда вы исправите и информируете об уязвимости своих клиентов. Кроме того, я с радостью предлагаю любое содействие, которым я могу помочь исправить уязвимость угрожающую {наименование ПО}.

Надеюсь на ваш ответ в ближайшее время. Я знаю, это длительный процесс, так что, я напомню об этом через 30 дней, если не получу ответа от вашей команды по нашим дальнейшим действиям. Если вам требуется любое содействие или помощь, пожалуйста держите со мной связь по электронной почте {email}.

С уважением,

{имя исследователя}

Теперь, давайте рассмотрим возможные стратегии общения с вендором, которые применимы к организациям, имеющим определенную политику в области безопасности.

Образец шаблона раскрытия информации с учетом политики безопасности

Вы должны стремиться к общению в доброжелательном русле и с готовностью к сотрудничеству, которое включает ваши намерения и всю существенную и полезную информацию о вашем исследовании:

Здравствуйте {наименование компании вендора},

Меня зовут {имя исследователя} и я являюсь независимым исследователем безопасности. Я недавно заинтересовался вашим программным обеспечением {название ПО} и потратил определенное время на изучение безопасности этой платформы.

Это письмо, для того, чтобы информировать вас об уязвимостях безопасности, которые я обнаружил в течение моего исследования. Я делюсь этим с вами, потому что я тоже хочу помочь в повышении безопасности вашего замечательного продукта и снизить шанс того, что ваши клиенты пострадают от угрозы безопасности, если уязвимость используют преступники.

Существует {перечисление уязвимостей} в {наименование ПО}, конкретно в версии {номер версии}. Я опишу эти уязвимости:

Уязвимость 1

{Наименование и ID, указанное в CWE(Common Weakness Enumeration – Список Распространенных Уязвимостей)}

Эта уязвимость найдена в {где была найдена уязвимость} части приложения. Результатом этого дефекта, может оказаться {описание угрозы из CWE}, который может повредить вашим клиентам в {сценарий, как эта уязвимость могла бы повредить пользователям}.

Для воссоздания этой уязвимости, вы можете проделать следующие шаги, которые я сопроводил скриншотами, чтобы помочь проиллюстрировать proof-of-concept.

{Действия по воссозданию условий для уязвимости}

Уязвимость 2

{Дополнительные уязвимости}

В качестве части моего исследования, я планирую {написать в блоге / опубликовать CVE}, поэтому я бы с удовольствием принял любую помощь от участников, когда вы исправите и информируете об уязвимости своих клиентов. Кроме того, я с радостью предлагаю любое содействие, которым я могу помочь исправить уязвимость угрожающую {наименование ПО}.

Надеюсь на ваш ответ в ближайшее время. Я знаю, это длительный процесс, так что, я напомню об этом через 30 дней, если не получу ответа от вашей команды по нашим дальнейшим действиям. Если вам требуется любое содействие или помощь, пожалуйста держите со мной связь по электронной почте {email}.

С уважением,

{имя исследователя}

А сейчас, давайте закончим с шаблонами переписки, парой примеров составления писем-напоминаний, которые вы могли бы отправить людям, с которыми пытаетесь наладить контакт.

Повторная попытка связи с вендором (если он не ответил с первого раза)

Основываясь на этом шаблоне, мы отправим вежливые напоминания людям, которые могли уже получать от вас сообщения, но не ответили на них. В большинстве случаев, мы советуем давать вендору на ответ 30 дней (за исключением критических уязвимостей). Письмо, в качестве напоминания, может выглядеть, как показано ниже. Заметьте, что мы информируем поставщика ПО о том, каким образом мы намерены действовать в направлении раскрытия информации в ближайшие 90 дней:

Здравствуйте {наименование компании вендора},

Я по поводу письма, отправленного мной вам 30 дней назад {дата}. Я хочу разобраться, каким образом ваша организация, планирует поступить с информацией об уязвимостях, которую я вам отправил. Напомню, что в {наименование ПО}, а именно в версии {номер версии} – существует {кол-во уязвимостей} уязвимостей.

Как сообщалось ранее, я планирую {написать в блоге / опубликовать CVE}, поэтому я бы с удовольствием принял любую помощь от участников, когда вы исправите и информируете об уязвимости своих клиентов. Я также напоминаю вам {наименование вендора}, что я с радостью предлагаю любое содействие, которым я могу помочь исправить уязвимость угрожающую {наименование ПО}.

Надеюсь на ваш ответ в ближайшее время. Отмечу, что установившейся практикой, является предоставление любой организации 90 дней, до публикации результатов исследования. Не получив от вас ответа, я буду считать, что ваша организация, не возражает, насчет выпуска результатов этого исследования {дата = 90 дней с момента письма}. Если вам требуется любое содействие или помощь, или есть любые вопросы, пожалуйста свяжитесь со мной по электронной почте {email}.

С уважением,

{имя исследователя}

Иногда, даже после попыток сотрудничества с поставщиком ПО, он остается безучастным. Мы рассмотрим это в следующем сценарии.

Уведомление неотвечающего вендора, о предстоящей публикации

Это шаблон электронного письма, в котором мы дадим поставщикам ПО знать о наших планах, относительно того, чего им стоит ожидать от публикации нашего исследования:

Здравствуйте {наименование компании вендора},

Я направлял вам несколько писем, насчет уязвимостей в {наименование продукта}. В {наименование ПО}, а именно в версии {номер версии} – целых {кол-во уязвимостей} уязвимостей.

Как ранее упоминалось, я обнародую результаты своих исследований {выпустив блог/опубликовав CVE}. Как сообщалось ранее, установившейся практикой, является предоставление любой организации 90 дней, до публикации результатов исследования. В связи с тем, что я так и не получил от вас ответа, считаю, что ваша организация, не возражает, насчет выпуска результатов этого исследования {дата публикации}. Поэтому, я собираюсь опубликовать их в виде {способ публикации/блог/CVE, и. т. п.}, при котором они будут общедоступны с этой даты или немного позднее.

Если у вас есть любые вопросы – смело связывайтесь со мной по email {ваш email}.

С уважением,

{имя исследователя}

Хотя мы не в состоянии охватить все возможные варианты, эти шаблоны должны помочь вам, если вы собираетесь работать с вендорами напрямую и не хотите прибегать к любому посредничеству третьих лиц. На этом мы заканчиваем раздел по общению с вендорами. Теперь мы поговорим о шаблонах CVE в том виде, в котором MITRE примет их описание и приведем непредвзятый пример самостоятельной публикации.

Шаблоны CVE

Публикация уязвимостей, может быть значительным шагом вперед, если у вас уже достаточный опыт в раскрытии информации. Сейчас мы рассмотрим шаблон резервирования CVE ID и шаблон CVE по раскрытию информации. Проект MITRE CVE четко определяет, как нужно правильно заполнять эти два шаблона, в разделе *Рекомендованное описание уязвимости, для использования в CVE*, формы CVE.

Шаблон резервирования CVE ID

Шаблон для резервирования CVE ID, должен содержать минимальную информацию:

{Уязвимость класса} в {компоненте}, {продукта} {версии} {вендора}, позволяет {тип атакующего} {тип атаки} путем {вектор атаки}.

Если вы плохо знакомы с терминологией, мы рекомендуем свериться с документацией по корректному заполнению этих полей, по следующему URL: http://cveproject.github.io/docs/content/key-details-phrasing.pdf

Шаблон раскрытия информации об уязвимости CVE

Самостоятельная публикация раскрытия информации об уязвимостях в форме CVE, может быть довольно простой задачей. Вот, отличный пример, как нужно общаться с командой MITRE, кратко и придерживаясь фактов:

CVE-{RESERVED ID}: {Наименование продукта} – {Наименование компонента}

Уязвимость, была обнаружена в {Наименование продукта} {версия} в {наименование компонента}. {Краткое описание дефекта}, который {краткое описание угрозы}.

Для воссоздания условий для проведения эксплуатации, сделайте следующее:

Шаг 1: {Начальные действия по воссозданию}

Шаг 2: ...

Для исправления этой проблемы, {определите шаги по исправлению проблемы}.

Вам нет надобности, излишне фантазировать при раскрытии информации об уязвимости, помните, что это необходимый минимум, который вы собираетесь отправить MITRE, для подтверждения наличия материала для публикации. Короче, придерживайтесь фактов и не городите теорий. Если вы желаете разработать развернутый отчет по этой уязвимости, это прекрасно. Возможно для этого, вам стоит опубликовать эти два документа поотдельности, сославшись на развернутый отчет, в своем кратком обращении к MITRE.

Мы завершаем главу о шаблонах, разделом по организационным моментам, которые можно использовать для обеспечения эффективной оценки хода исследования.

Организационные моменты

Важно содержать дела в порядке, в особенности касательно исследовательских проектов, в результате которых будут выявлены уязвимости, требующие дальнейших действий. Когда, по мере выполнения тестирования, вы отметили, уязвимости, требующие создания CVE, то этот процесс нужно контролировать месяцы, а иногда и годы. Применяя такой подход, вы можете работать над несколькими приложениями одновременно, а отсутствие системы предполагает возможность утраты каких-то деталей, полученных в ходе исследования. Разумеется у вас может быть своя система по организации труда, но давайте рассмотрим вариант, который мы настоятельно рекомендуем и нашли очень полезным для мониторинга исследования уязвимостей: **Доски Канбан**.

Представьте доски Канбан, как группу задач распределенных по карточкам или колонкам, отображающим выполнена ли задача. Доски Канбан иногда физически представленны в виде стикеров на доске для фломастеров, разлинованной на колонки, также называемые **дорожки** (**swimlanes**):

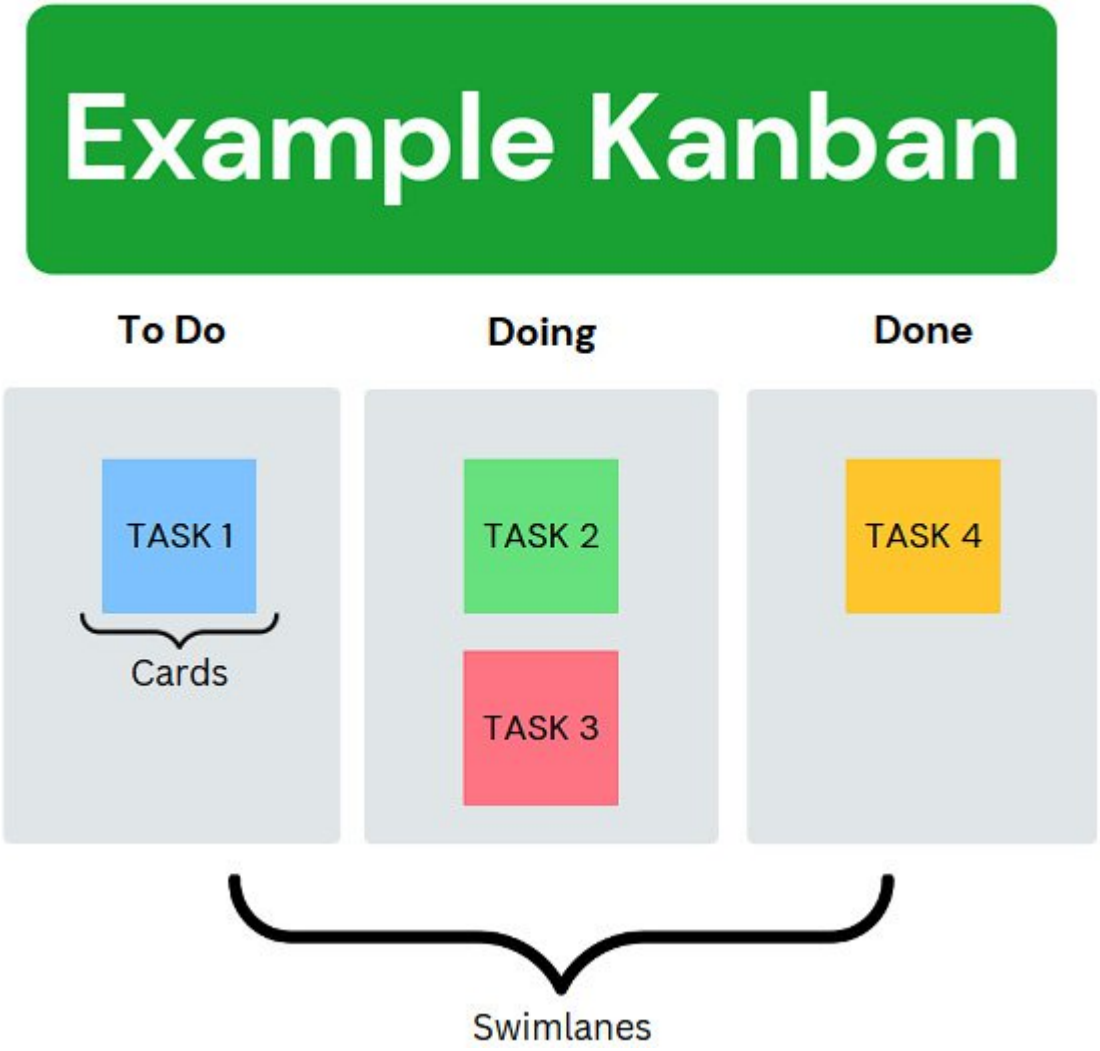


Рис. 10.7 – Иллюстрация Канбан

В этой книге, мы поделимся четырьмя шаблонами Канбан, созданными в программном обеспечении Trello. **Trello**, это облачное приложение, которое можно использовать бесплатно и мы сделали эти шаблоны общедоступными для всех. Теперь, давайте рассмотрим эти шаблоны и то, как они работают. Хотя мы здесь используем Trello, эти же идеи можно применить и на физической доске или в том ПО, которое вы предпочитаете.

Рабочая область

В Trello есть концепция **рабочих областей**, которые схожи с досками Канбан по принципу действия и содержимому. В нашем случае, мы создадим рабочие области и далее, добавим наши шаблоны в каждую из них. Чтобы создать рабочую область, нам понадобится зарегистрироваться в Trello и выбрать пункт **Create Workspace** в выпадающем меню кнопки **Create**, вверху экрана:

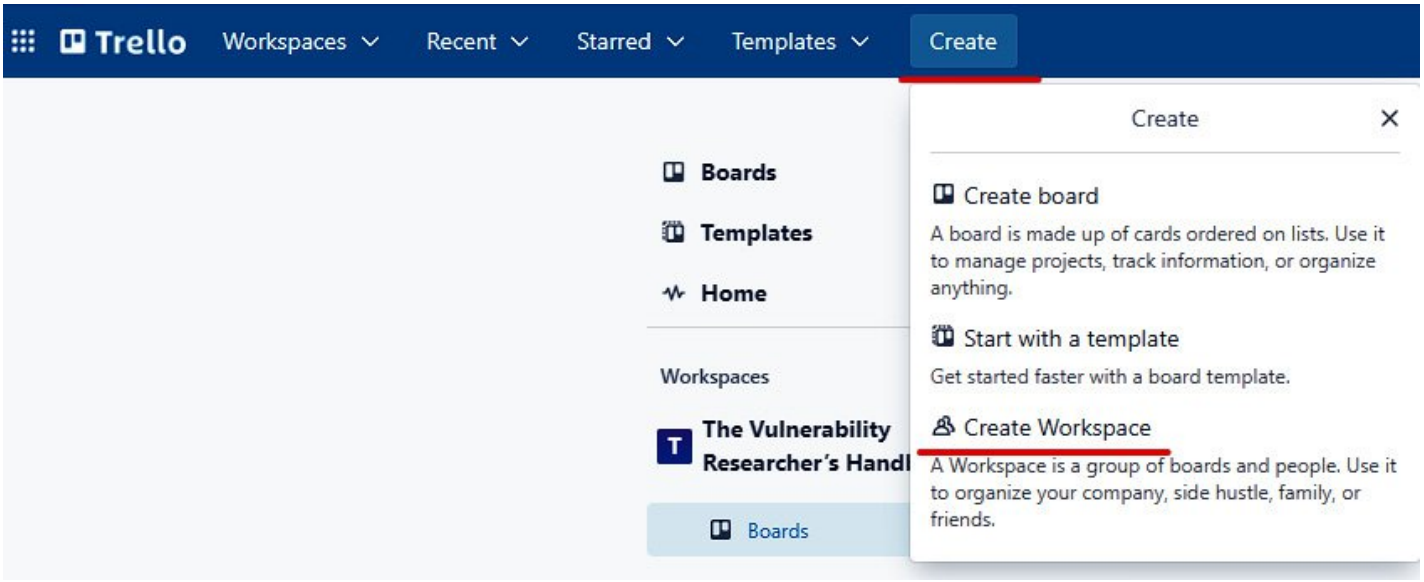


Рис. 10.8 – Создание рабочей области в Trello

После выбора, появится меню создания рабочей области. Укажите предпочитаемое название рабочей области и, если хотите - добавьте ее категорию или описание:

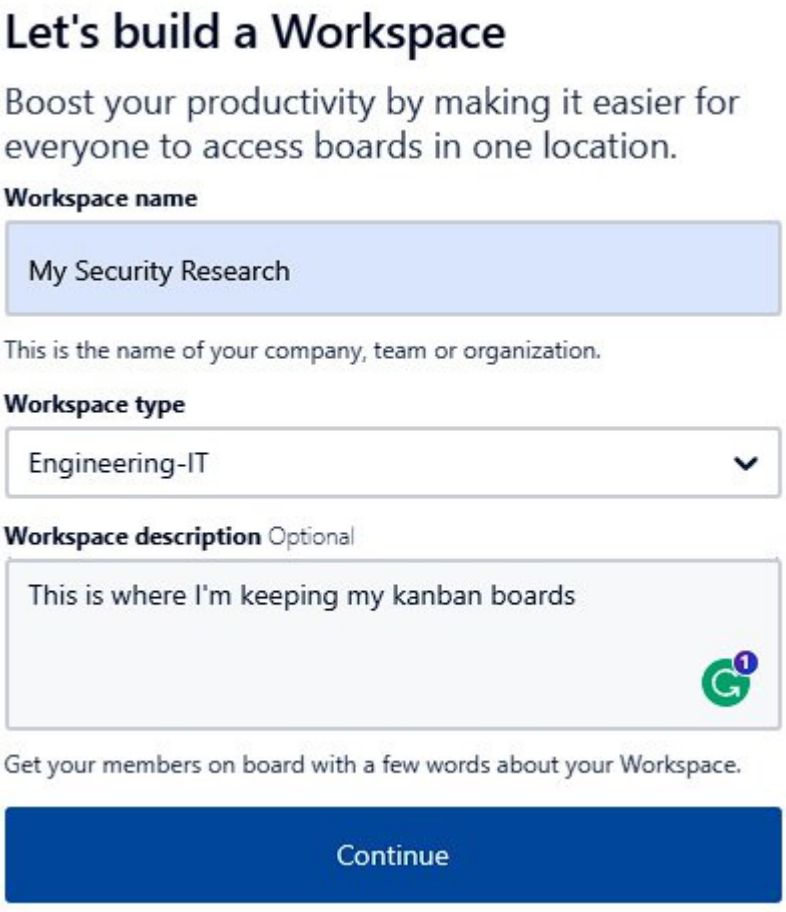


Рис. 10.9 – Стартовое окно создания рабочей области

Теперь, когда вы создали рабочую область, давайте скопируем в нее несколько "досок". Мы начнем с "доски" исследования программного обеспечения. Вы можете найти использованную в примере "доску", здесь: https://trello.com/b/4VVGFBky/software-research.

Давайте откроем этот URL и посетим "доску", которая должна походить на Канбан, изображенный на следующем скриншоте:

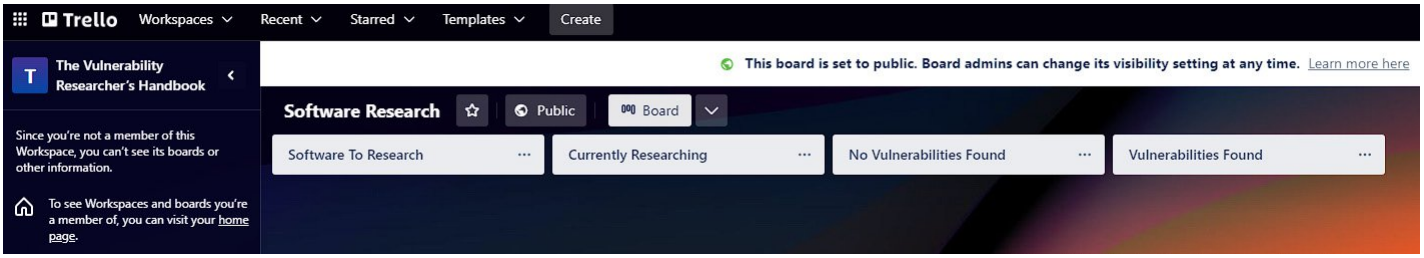


Рис. 10.10 – Обзор шаблона доски, который нам надо скопировать

Как только, мы открыли "доску", просто скопируем ее в нашу рабочую область, потому что мы не можем изменять сам шаблон. Чтобы скопировать нужную нам "доску", нажмите **Board Settings** | **More**, а затем выберите **Copy board**, как видно на следующем скриншоте:

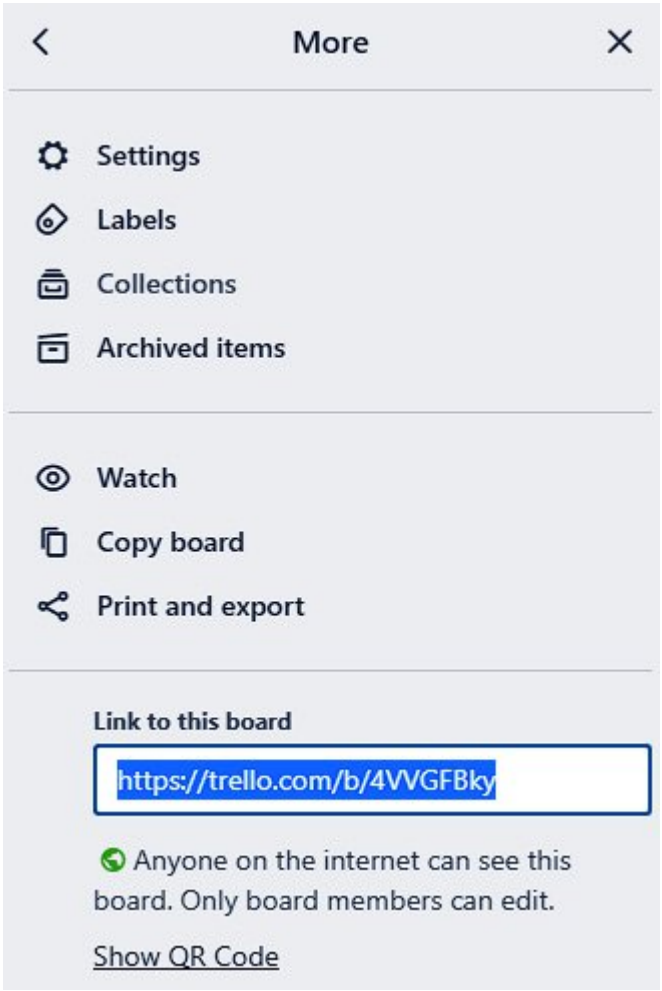


Рис. 10.11 – Открытие меню "Дополнительно"

Как только вы кликнете на **Copy board**, вам нужно присвоить ему такой же заголовок (как в шаблоне, открывшемся по ссылке выше) и выбрать собственную рабочую область. Возможно вы также захотите изменить права доступа по-умолчанию, чтобы гарантировать, что ваше исследование останется конфиденциальным. Щелкните по ссылке **Change** рядом с текстом, This board will be **Public**. Измените значение на **Private**, как показано на следующем скриншоте:

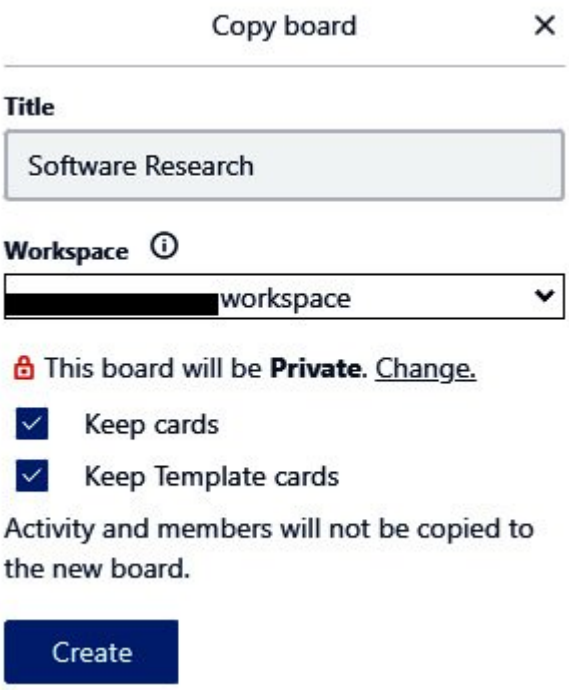


Рис. 10.12 – Окно копирования доски

Вместе с копированием "доски", вы можете настроить ее, как вам нравится. Давайте все-же перед этим, немного поговорим об этой "доске". "Доска" имеет четыре колонки, **Software To Research**, **Currently Researching**, **No Vulnerabilities Found** и **Vulnerabilities Found**:

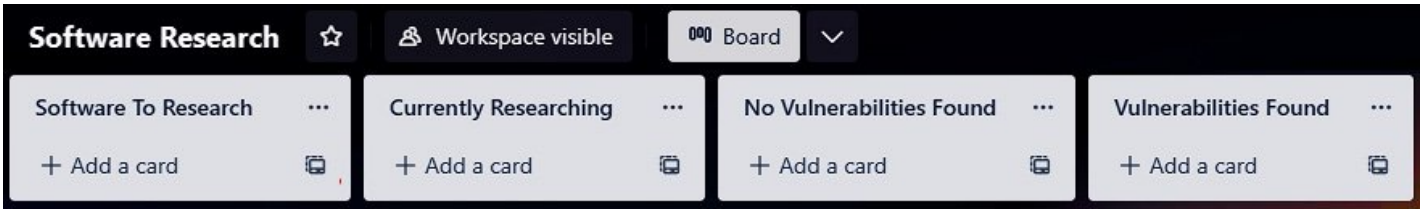


Рис. 10.13 – Создание нашей первой карточки, на основе полученого шаблона, используя кнопку "создать из шаблона"

Планируя свое исследование, вы должны подготовить список тем, которые собираетесь исследовать. Мы подготовили шаблон, который поможет вам собрать эту информацию в одном месте. Давайте создадим новую карточку на "доске" **Software Research**, используя маленькую иконку снизу и справа, колонки **Software To Research**:

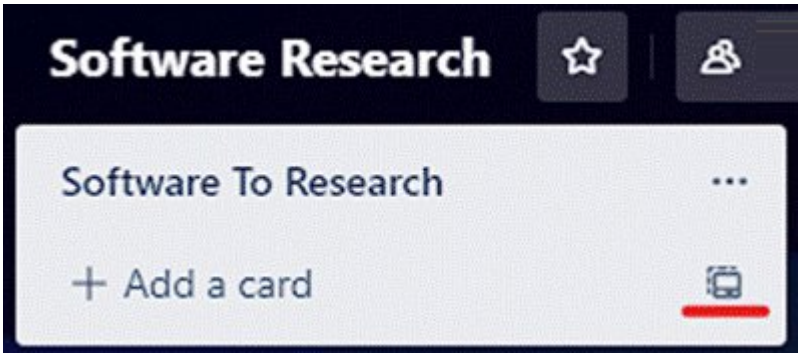


Рис. 10.14 – Кликните по иконке, для добавления карточки

Когда мы кликнем на ней, должно отобразится меню, с заголовком **Card templates**, предлагая нам, только выбрать шаблон. Кликните на этом шаблоне, чтобы создать на его основе новую карточку:

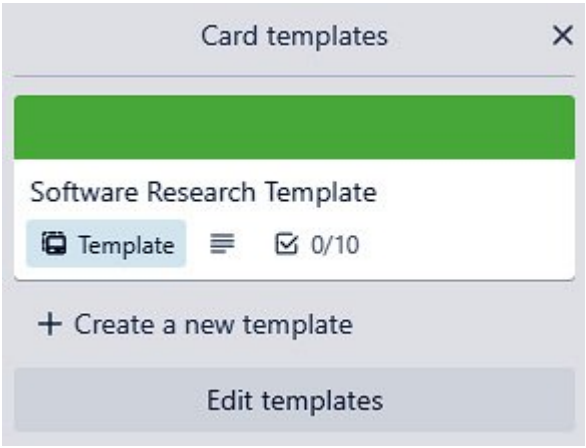


Рис. 10.15 – Окно с шаблонами карточек

Как только, вы кликнете на шаблоне, вам будет предоставлена возможность, назвать карточку иначе, чем **Software Research Template**. Мы советуем использовать название программного обеспечения и следовать чек-листам, прикрепленным к шаблону. В данном примере, мы будем использовать в качестве объекта исследования – программное обеспечение **Poodle Groomer 5000**:

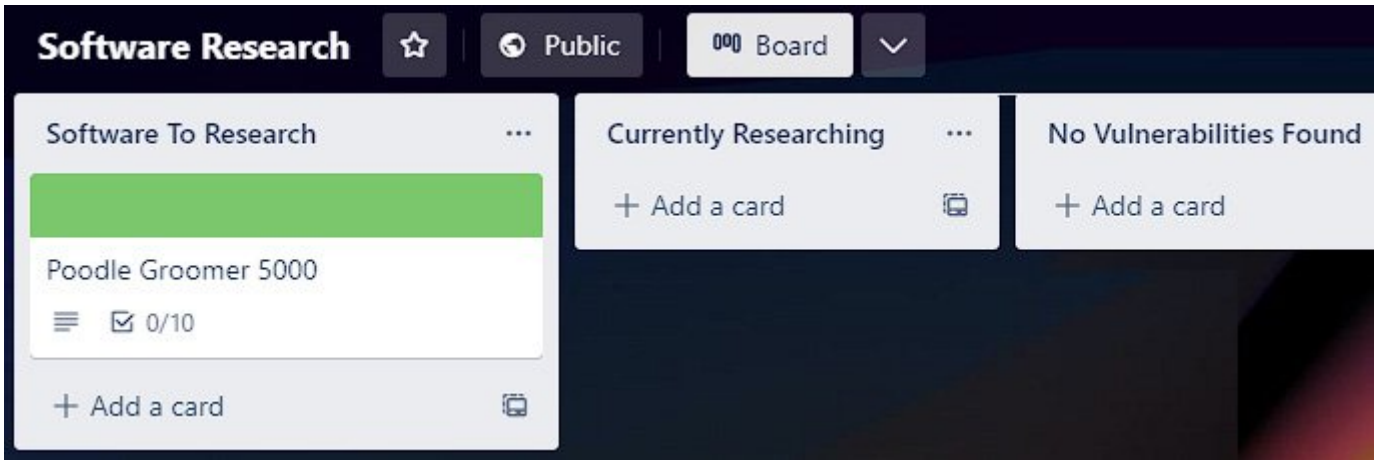


Рис. 10.16 – Обзор нашей первой карточки

Теперь, давайте кликнем по нашей новой карточке. По клику на **Poodle Groomer 5000**, нам откроется раздел описания, который нужно заполнить, как показано на Рис. 10.17:

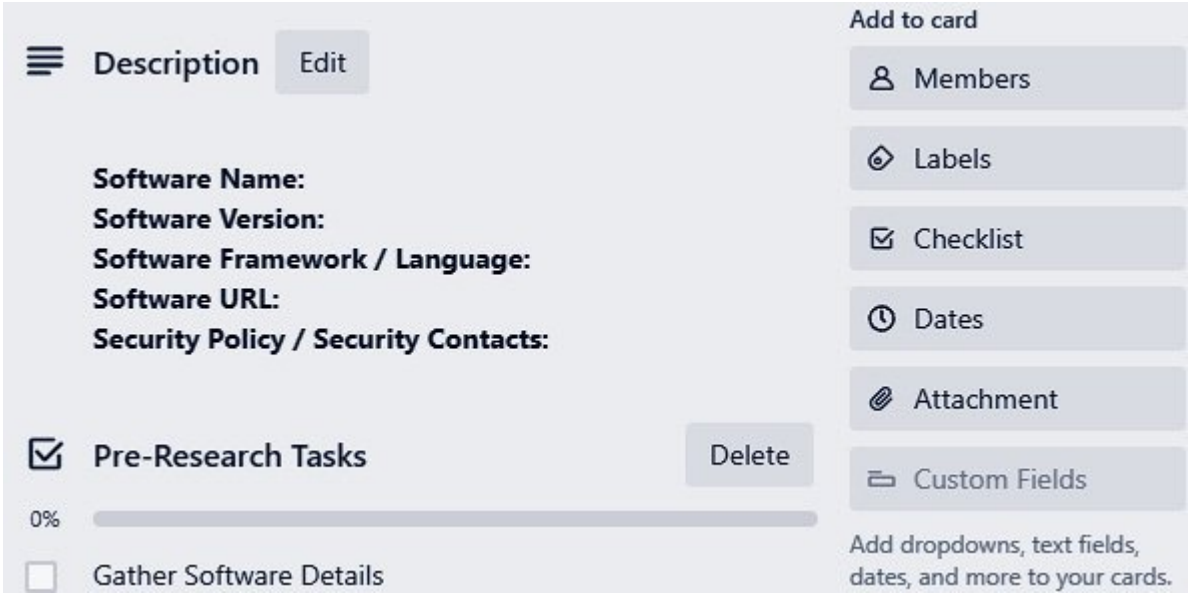


Рис. 10.17 – Раздел описания для Poodle Groomer 5000

Обратите внимание, на редактирование набора чек-листов, которые мы будем заполнять продвигаясь по ходу исследования, как показано на Рис. 10.18:

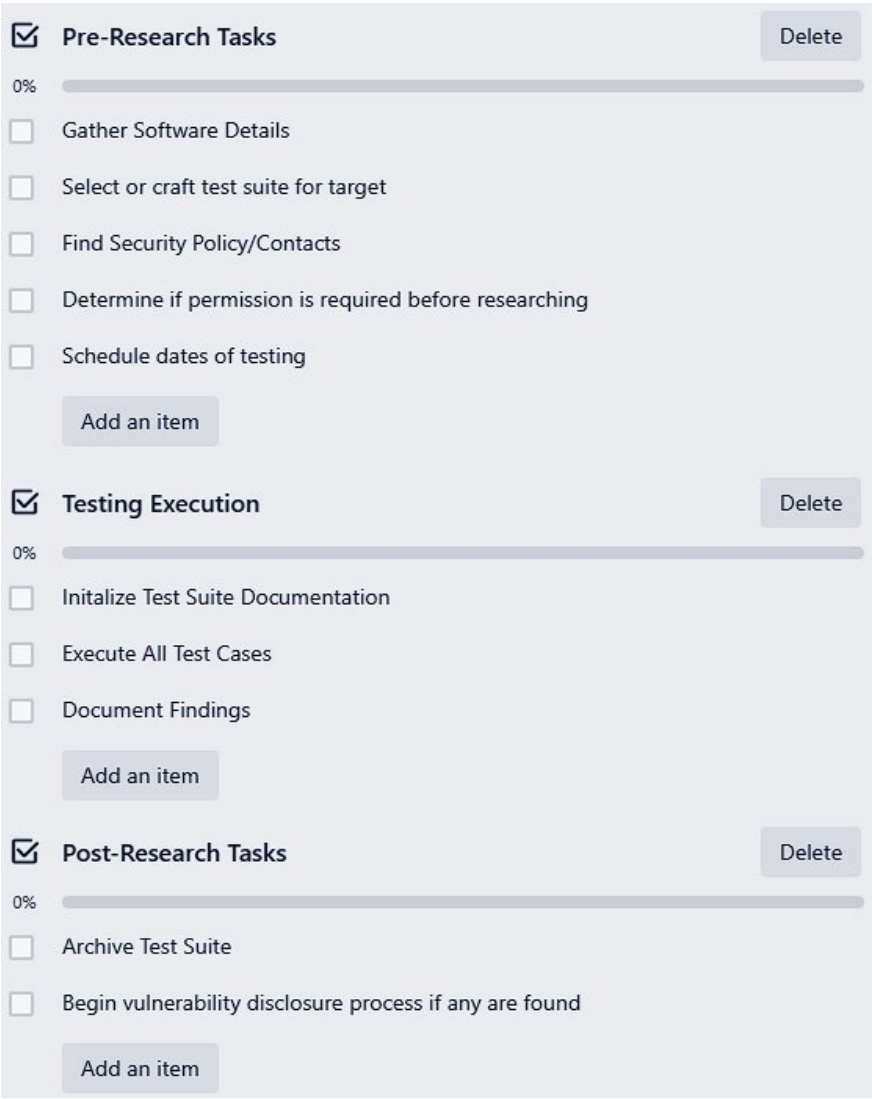


Рис. 10.18 – Каждая "дорожка", имеет свой набор задач в чек-листе, чтобы отслеживать прогресс

Как и с любым чек-листом, вы начинаете сверху и выполняете работу по порядку. Каждый чек-лист, представляет этапы задачи, которые должны быть выполнены. Закрывая их, вы можете передвигать карточку в следующую "дорожку". Итак, отработав наш первый чек-лист, мы собрали информацию о софте, и определились с графиком тестирования, как вы можете видеть – получилось, что-то вроде этого:

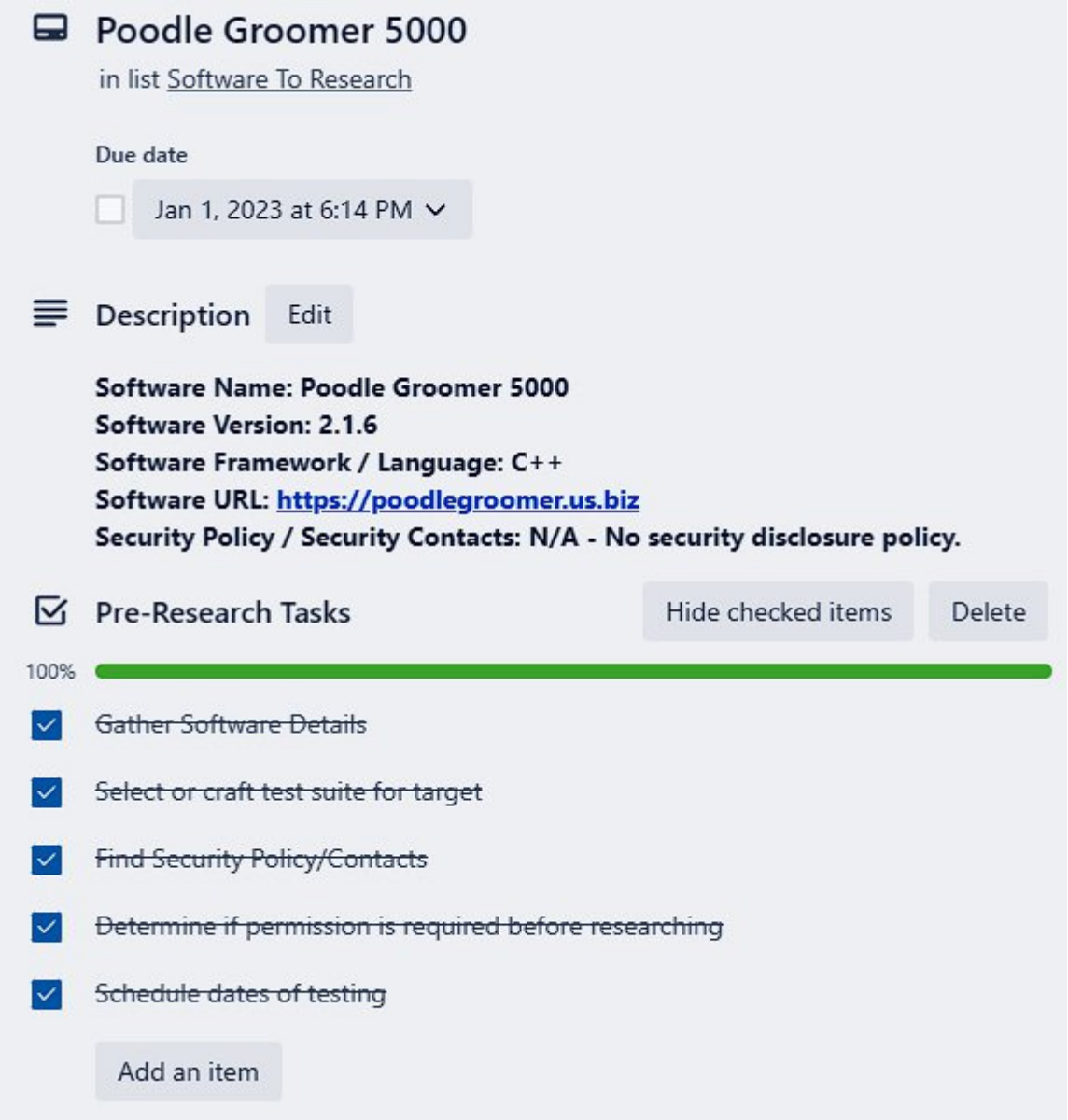


Рис. 10.19 – Завершение перечисленных задач

В таком случае, мы можем переместить нашу карточку из "дорожки" **Software To Research** в "дорожку" **Currently Researching**. Вы можете описать, каким образом вы продвигали карточки по ходу работы и оставить комментарии, которые могут оказаться полезны, чтобы зафиксировать как вы работали над этой задачей.

Для этой общедоступной рабочей области, есть еще три "доски". Теперь, когда мы готовы к их использованию, давайте скопируем их в нашу рабочую область. Они включают в себя следующее:

- **Публикация уязвимостей**
- https://trello.com/b/RF4Wrbi8/vulnerability-publishing
- **Раскрытие информации об уязвимостях**
- https://trello.com/b/ANRLGfgO/vulnerability-disclosure
- **Публикация CVE**

- https://trello.com/b/DbZgiL4T/cve-publishing

Скопируйте эти шаблоны, таким же образом, как и нашу первую "доску". Все они нам понадобятся, чтобы рассмотреть различные этапы при тестировании, раскрытии информации об уязвимостях и ее публикации. А сейчас, давайте рассмотрим эти "доски" и как мы будем их использовать.

Исследование до раскрытия

Как уже говорилось, мы обнаружили уязвимости в приложении **Poodle Groomer 5000**. Раз уж мы это сделали, то на следующем шаге, нужно поделиться этой информацией с поставщиком ПО. Но сначала, нам надо полностью упорядочить сам процесс. Мы переносим наше исследование на "доску", созданную специально для отслеживания прогресса в раскрытии информации об уязвимостях. Лучший способ это сделать – копировать карточку на новую "доску", выбрав **Copy** в свойствах карточки:

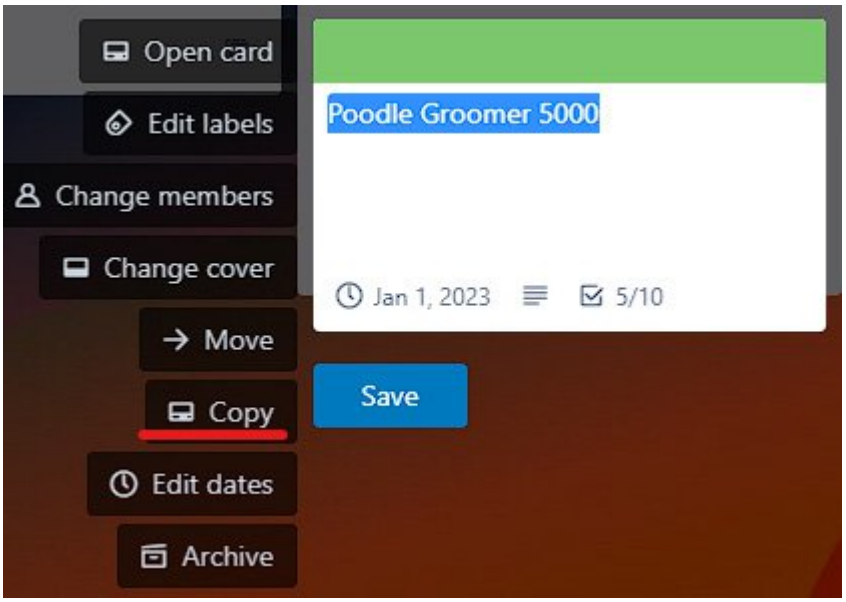


Рис. 10.20 – Выбираем "Копировать" в свойствах карточки

По нажатию **Copy**, появится меню, показанное на следующем скриншоте. Я предпочитаю не прикреплять старых чек-листов, но нам нужно будет сразу создать новые – для процедуры раскрытия информации:

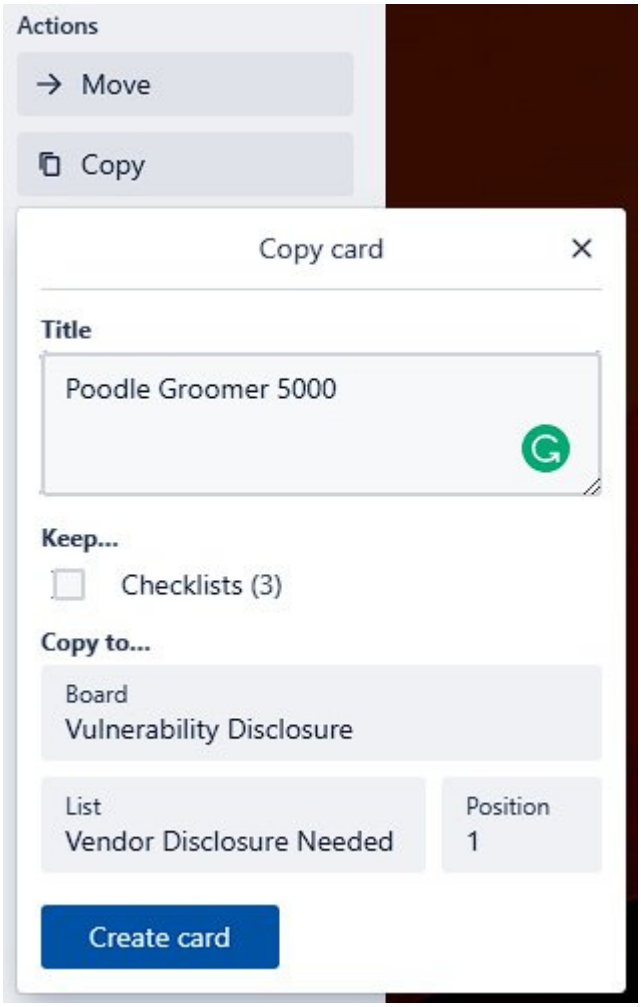


Рис. 10.21 – Перенос карточки на другую доску

Нажмите **Create card**, чтобы копировать вашу новую карточку на другую "доску" по раскрытию информации. Как только она скопируется, вы можете перейти на эту "доску" и увидеть, что появилась новая карточка:

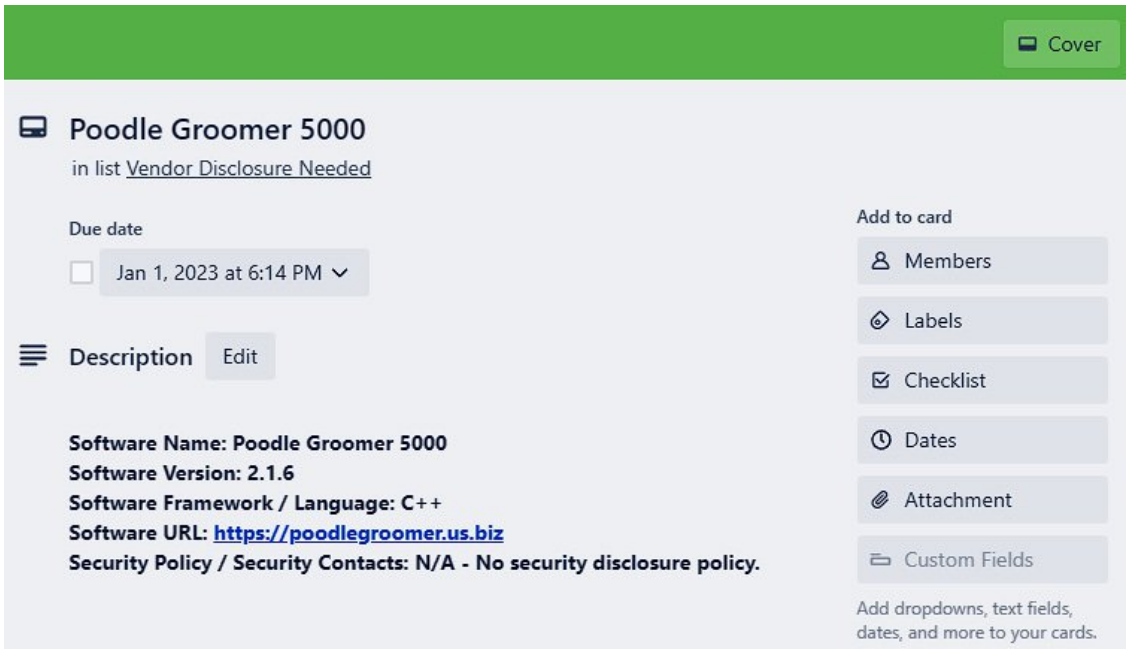


Рис. 10.22 – Наша новая карточка, содержит все значения предыдущей, за исключением заполненных чек-листов

Сейчас мы добавим наши чек-листы по раскрытию информации из шаблона соответствующей "доски".

Нажав **Checklist** мы добавим чек-листы из шаблона этой "доски". Вы можете добавить несколько чек-листов, как показано на следующем скриншоте:

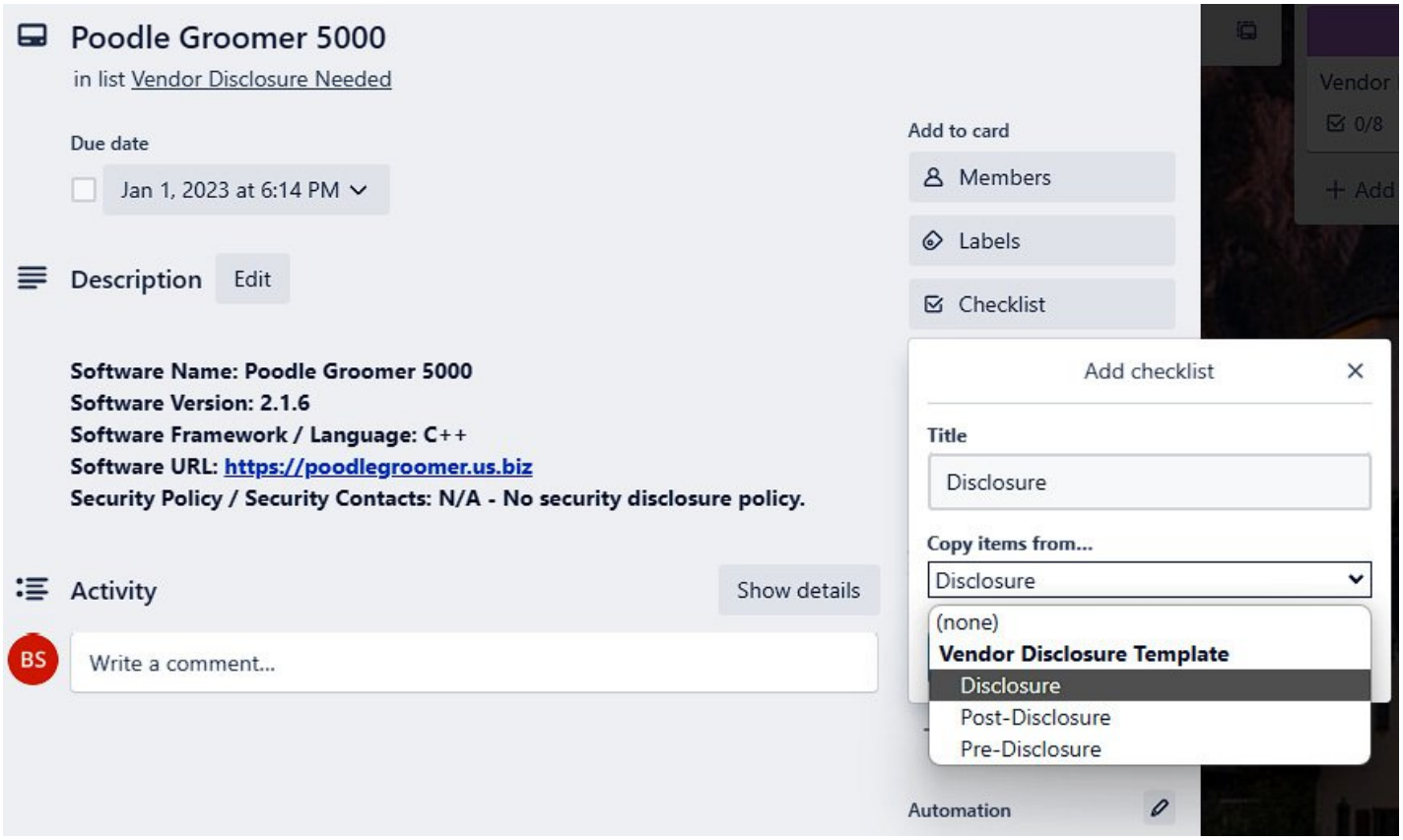


Рис. 10.23 – Копирование чек-листов подготовки раскрытия, процедуры раскрытия и завершения раскрытия

Как только вы добавили карточки, вы сможете увидеть чек-листы, которые нужно заполнять на данном этапе исследования:

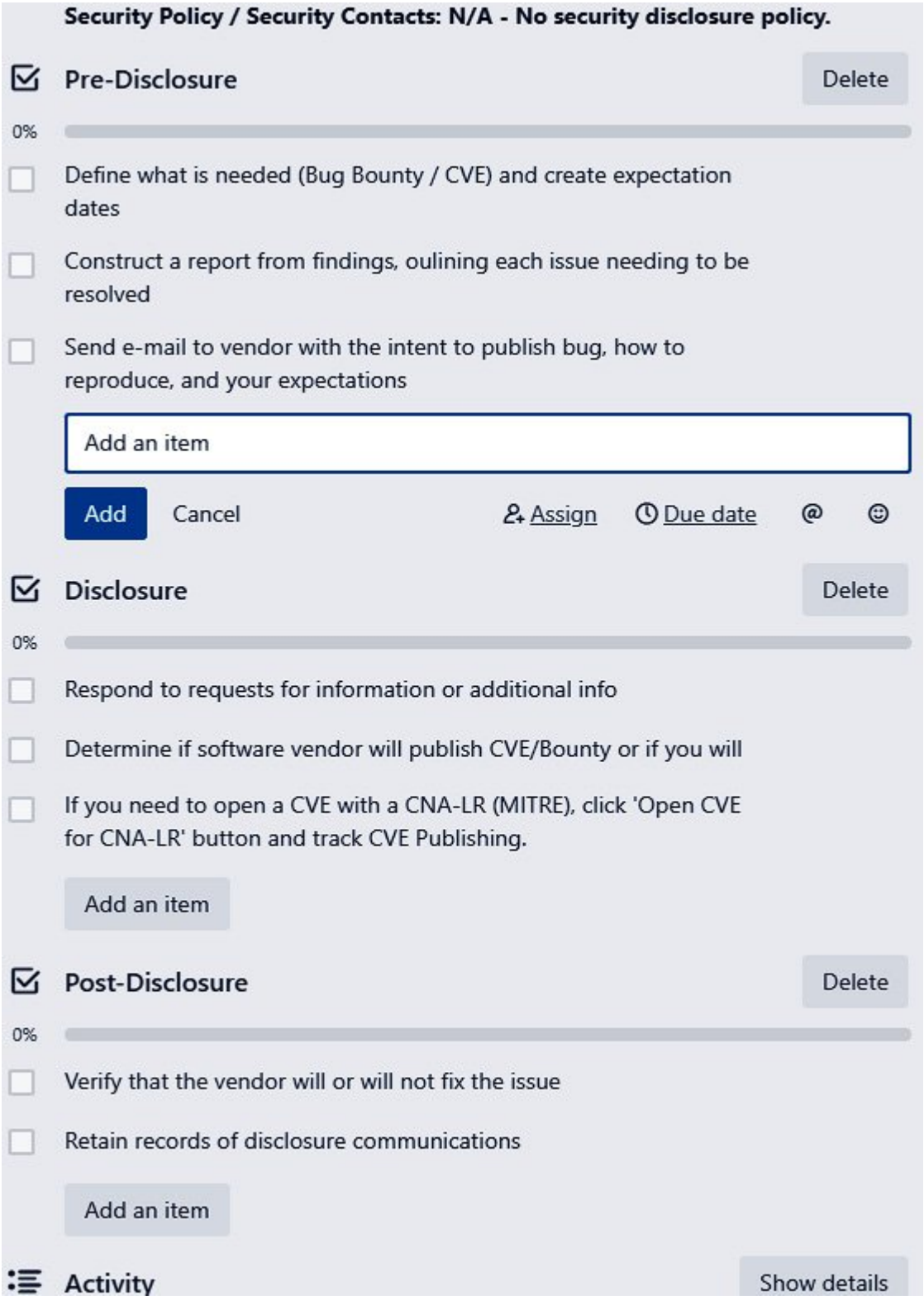


Рис. 10.24 – Наши новые чек-листы, теперь готовы для очередного набора задач по раскрытию информации об уязвимости

Этот процесс аналогичен, применяемому в случае открытия CVE, как видно на втором чек-листе. Вы просто копируете карточку на "доску" **Публикация CVE**. Похожим образом, выполняется и аналогичная процедура для шаблона **Публикация уязвимостей**. Чек-листы из шаблона **Публикация уязвимостей** можно увидеть здесь:

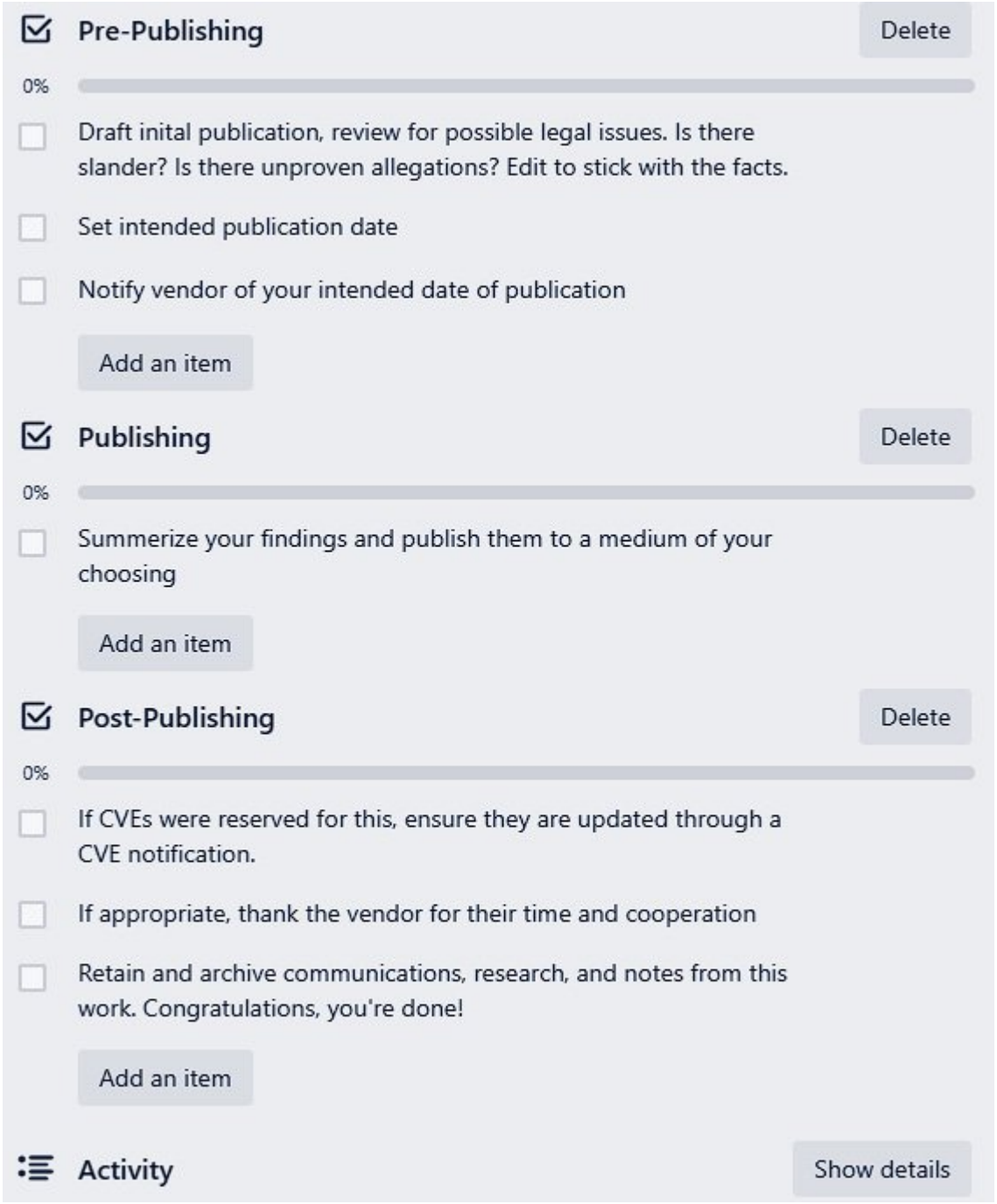


Рис. 10.25 – Чек-листы, относящиеся к публикации уязвимости

Чек-листы из шаблона **Публикация CVE**, показаны на следующем скриншоте:

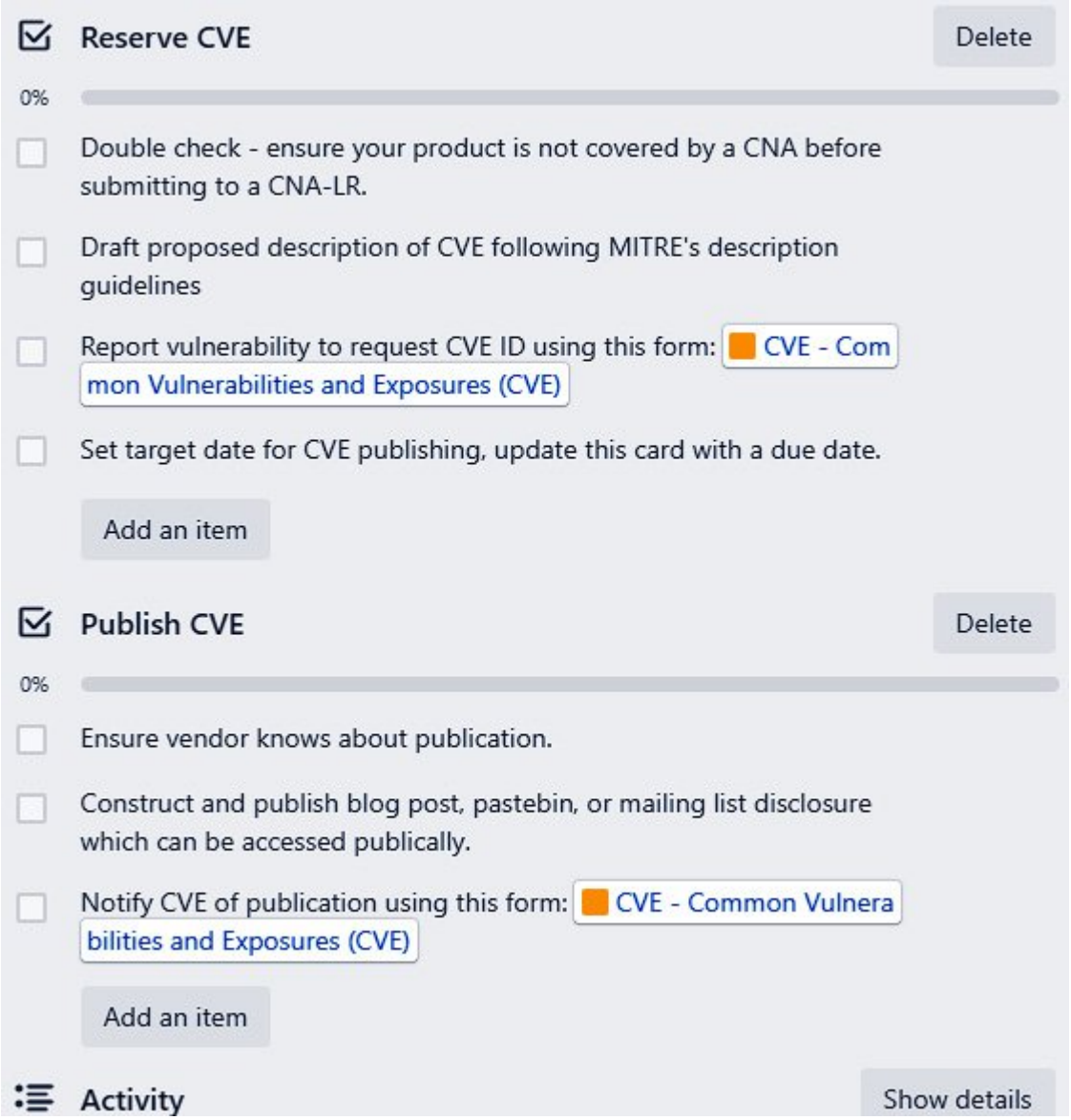


Рис. 10.26 – Чек-листы, относящиеся к публикации CVE

Эти шаблоны, можно сделать даже быстрее, используя возможности автоматизации, предоставляемые Trello. Например, вы могли бы автоматически запускать определенные процедуры, описанные раннее, по срабатыванию заданного условия. Не стесняйтесь это использовать, рассматривая варианты организации вашего исследования или черпайте из этого вдохновение и создавайте свои инструменты. За более подробной информацией по началу использования Trello, обратитесь к разделу *Дополнительные материалы* этой главы.

На этом мы закругляемся и завершаем эту главу заключительным напутствием и подведением итогов прочитанного.

Итоги главы и заключение

Здрав будь, ресерчер!

Если ты до сюда дошел, полагаю ты прочел эту книгу. Во время создания этой книги, моим искренним желанием было создать нечто такое, что дало бы тебе понимание сути исследования уязвимостей и процесса раскрытия информации о них. Я также хотел осветить различные опасности, которые ты можешь ожидать на своем пути становления, как исследователя и в ходе публикации своих работ. Этот концентрированный набор знаний именно то, что мне было нужно, когда я только вступал

на путь публикации результатов исследований и совместной работы с вендорами любого сорта. Я надеюсь, что ответил на вопросы, возникавшие у тебя во время исследований и сподвигнул на поиски своего места в исследовательском сообществе. Оно объединяет людей с разными знаниями и опытом для улучшения программного обеспечения и систем, которыми мы все пользуемся.

Исследования в сфере информационной безопасности, занимают значительное время и я аплодирую, даже если ты просто рассматриваешь возможность своего участия. Ты непременно столкнешься с препятствиями и разочарованиями, на всем протяжении своего пути от обнаружения до публикации уязвимостей. Многие поставщики ПО, будут пытаться помешать твоей работе или отрицать ценность твоего участия. Они будут дискредитировать тебя, лгать тебе и испытывать каждую деструктивную технику, которая есть в их арсенале, чтобы обесценить твой труд и подавить свободный обмен идеями, помогающий приносить пользу людям. Изменение порядка вещей в этом мире является сложной задачей и изменение того, как программное обеспечение создается и обслуживается вендорами, не является исключением. Твоя разоблачительная деятельность, неизбежно пойдет вразрез с инициативами корпораций, личными амбициями, корпоративными стандартами, новыми фичами и потенциальными источниками дохода. Несмотря на все эти факторы, которые безусловно усложнят твою работу, я призываю тебя не сдаваться.

Механизмы управляющие современной индустрией программного обеспечения, имеют множество областей для совершенствования, так же как и ее продукция повсеместно применяемая многими людьми, изо дня в день. Невозможно ничего улучшить, без внимания людей, которые могут показать эти проблемы и найти более совершенные методы разработки таких систем, тесно интегрированных в современное общество. Поэтому, все зависит от нас.

Исследователи, одни из тех, кто откликается на эти проблемы и продолжит делать это в обозримом будущем. Не существует никакого государственного контроля, который мог бы послужить причиной коренных изменений в этих системных проблемах. Отступать некуда и не стоит ожидать, что организации сейчас же изменят свою политику. Их проблемы в разработке, являются признаком упущений, но сквозь призму корпоративной ответственности то, что их клиенты подвергаются опасности, считается неким допустимым риском. Несмотря на это, я полон оптимизма. Публикация исследований, растет год за годом и это показатель, который корпорации собирают, отбрасывают с негодованием, но принимают тот факт, что исследователи существуют и будут распространять свои работы. Некоторые организации, даже соглашаются, что труд исследователей мог бы быть ими оплачен. Они используют свои отношения с сообществом информационной безопасности, в традиционной для бизнеса форме: как некий товар. Все это верные признаки улучшения ситуации, но для этого требуется прилагать усилия и я призываю тебя разделить этот труд, как важную часть этого движения.

В качестве поддержки на твоем пути, я рекомендую общаться с другими исследователями, обмениваться знаниями и возможно выражать сочувствие некоторым поставщикам ПО, в связи с проблемами, которые ты им недавно создал. Эти взаимоотношения, помогут облегчить бремя самостоятельной работы, помогут в ваших дальнейших совместных исследованиях и обогатят твой опыт бесчисленными способами. В границах доступного тебе, в сообществе информационной безопасности не забывай, что множество исследователей (кого-то уже больше нет с нами), иногда с риском для себя – внесли свой вклад в хранилище знаний , которыми мы все пользуемся. Уважай их вклад, где бы то ни было – беря за основу их наработки, применяя их с лучшими намерениями и применяя на пользу другим. Люди рассчитывают на тебя.

Заканчивая это руководство, я от души желаю тебе удачи в карьере исследователя. Благодарю, что уделяешь время участвуя в чем-то нужном и рассчитываю в будущем прочесть твои публикации.

Дополнительные материалы

Начало работы с Trello: https://trello.com/en/guide

Использование Trello для управления проектами: Пошаговое руководство: https://blog.hubstaff.com/trello-project-management/

Как пользоваться Trello (видео): https://www.youtube.com/watch?v=geRKHFzTxNY

Интенсивный курс управления проектами и основы Trello (видео): https://www.packtpub.com/product/product-management-crash-course-and-trello-fundamentals-video/9781803235646

Алфавитный указатель

A - Z

A

B

C

D

E

F

G

H

I

J

K

M

N

O

P

Q

R

S

T

U

V

W

Z

А - Я

А

Б

В

Г

Д

З

И

К

Н

О

П

Р

С

У

Э

Другие книги, которые могут вам понравиться

Джошуа Пикколет, Hash Crack: Password Cracking Manual v3 ru. https://xss.is/threads/82201/

Справочное руководство по техникам и инструментам взлома хешей и сопутствующим утилитам, для взлома паролей распространенных приложений.