**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**



**МОСКОВСКИЙ ПОЛИТЕХ**

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ  
КАФЕДРА «ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ»

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ №1**

На тему:

«ATT&CK Matrix for Enterprise. Изучение и описание информации с сайта OWAPS. Разбор инцидента на основании отчета аналитического центра»

**Выполнил**: ст. гр. 241-371 Вожаков М. А

**Руководитель**: Шорников А. В.

**Место проведения**: Московский Политех, лаборатория «Программно-аппаратных средств обеспечения информационной безопасности»

**Москва – 2025**

Оглавление

[ВВЕДЕНИЕ 4](#_Toc194941283)

[ГЛАВА 1. Исследование MITRE ATT&CK Matrix for Enterprise 5](#_Toc194941284)

[1.1 Общие сведения 5](#_Toc194941285)

[1.2 Reconnaissance (TA0043) 5](#_Toc194941286)

[1.3 Resource Development (TA0042) 8](#_Toc194941287)

[1.4 Initial Access (TA0001) 10](#_Toc194941288)

[1.5 Execution (TA0002) 13](#_Toc194941289)

[1.6 Persistence (TA0003) 14](#_Toc194941290)

[1.7 Privilege Escalation (TA0004) 16](#_Toc194941291)

[1.8 Defense Evasion (TA0005) 18](#_Toc194941292)

[1.9 Credential Access (TA0006) 20](#_Toc194941293)

[1.10 Discovery (TA0007) 22](#_Toc194941294)

[1.11 Lateral Movement (TA0008) 24](#_Toc194941295)

[1.12 Collection (TA0009) 25](#_Toc194941296)

[1.13 Command and Control (TA0011) 27](#_Toc194941297)

[1.14 Exfiltration (TA0010) 29](#_Toc194941298)

[1.15 Impact (TA0040) 31](#_Toc194941299)

[1.16 Сравнение матрицы MITRE с Cyber Kill Chain 33](#_Toc194941300)

[ГЛАВА 2. Изучение и описание информации с сайта OWASP 36](#_Toc194941301)

[2.1 Общие сведения о проекте 36](#_Toc194941302)

[2.2 Шапка 36](#_Toc194941303)

[2.3 Главная страница 37](#_Toc194941304)

[2.4 OWASP Top Ten 38](#_Toc194941305)

[2.5 OWASP Application Security Verification Standard (ASVS) 68](#_Toc194941306)

[2.6 OWASP SAMM 69](#_Toc194941307)

[2.7 OWASP Web Security Testing Guide 69](#_Toc194941308)

[2.8 The OWASP Gen AI Security Project 69](#_Toc194941309)

[2.9 About the OWASP Foundation 70](#_Toc194941310)

[2.10 OWASP CRS 71](#_Toc194941311)

[2.11 OWASP Mobile Application Security 71](#_Toc194941312)

[ГЛАВА 3. Разбор инцидента на основании отчета аналитического центра 73](#_Toc194941313)

[3.1 Выбор инцидента и сбор сведений 73](#_Toc194941314)

[3.2 Определение тактик и техник 74](#_Toc194941315)

[3.3 Один из возможных вариантов действий злоумышленников. 76](#_Toc194941316)

[3.4 Принятые компанией меры 77](#_Toc194941317)

[ВЫВОД 78](#_Toc194941318)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 79](#_Toc194941319)

# 

# ВВЕДЕНИЕ

Учебная практика по теме «ATT&CK Matrix for Enterprise. Изучение и описание информации с сайта OWAPS. Разбор инцидента на основе отчета аналитического центра», выполненная Вожаковым Михаилом, ст. гр. 241-371, включает в себя изучение и описание основных аспектов матрицы – Mitre Att&ck, изучение и описание информации с сайта OWASP, разбор реального инцидента произошедшего за последние год-полтора с обозначением использованных злоумышленниками тактик, техник и процедур на основе отчета аналитического центра об этом инциденте.

Цель учебной практики – получить новые знания и практические навыки в области ИБ, изучив основные техники, процедуры и тактики, используемые злоумышленниками для совершения кибер-атак.

Для достижения поставленной цели поставлены и решены следующие задачи:

1. Изучение предметных областей;
2. Анализ, структурирование и описание изученного материала;
3. Разбор инцидента на основе отчета аналитического центра.

Объектом исследования являются MITRE ATT&CK Matrix for Enterprise, сайт OWASP, отчет об инциденте от аналитического центра.

# ГЛАВА 1. Исследование MITRE ATT&CK Matrix for Enterprise

1.1 Общие сведения

MITRE ATT&CK - это глобально доступная база знаний о тактике и технике противника, основанная на реальных наблюдениях. База знаний ATT&CK используется в качестве основы для разработки конкретных моделей и методологий угроз в частном секторе, в правительстве, а также в сообществе продуктов и услуг в области кибербезопасности.

Доступ к данной матрице предоставляется на безвозмездной основе.

Данная матрица включает в себя следующие компоненты:

* тактики – это цели, задачи, которых пытается достичь злоумышленник (Например, разведка или вывод данных) – всего 14 штук;
* техники – это конкретные методы, которые используют злоумышленники для реализации тактик (поставленных целей);
* подтехники – детальные варианты основных техник;
* процедуры – реальные примеры использования техник хакерскими группировками.

Далее приведены тактики и некоторые из их техник.

1.2 Reconnaissance (TA0043)

Reconnaissance (или разведка) – включает в себя техники активного или пассивного сбора противником информации, которая может быть использована для планирования будущих операций. Такая информация может включать сведения об организации, инфраструктуре, рабочем составе организации или определенной персоне.

Данная тактика включает в себя 10 техник:

1. *Активное сканирование* – активные действия противника для сбора информации, которая может быть использована для определения будущей цели злоумышленника.

При осуществлении данной техники противник проверяет инфраструктуру жертвы через сетевой трафик. Одним из отличий от других техник разведки является прямое воздействие.

Сюда можно отнести сканирование IP-блоков, сканирование уязвимостей, сканирование с использованием часто употребляемых слов или сочетаний знаков для идентификации контента и инфраструктуры.

В качестве примера реализации можно привести пример атаки Triton Safety Instrumented System Attack на систему приборов безопасности для сетевой разведки интересующих целей.

1. *Сбор информации о хостах жертвы*. В качестве цели могут выступать различные сведения, в том числе административные данные (например, имя, назначенный IP, функциональность), а также сведения о конфигурации хоста.

Сюда относится сбор информации об аппаратном и (или) программном обеспечении, сведения о прошивке или сведения о конфигурации клиента.

Пример процедуры: Volt Typhoon провел предварительную разведку для получения информации о жертве.

1. *Сбор информации о личности жертвы* – злоумышленники собирают сведения личного характера, в том числе имена сотрудников, адреса электронной почты, ответы на вопросы безопасности, конфиденциальные данные, такие как учетные данные или конфигурации многофакторной аутентификации, и др.

Пример процедуры: APT32 вел целенаправленную слежку за активистами и блогерами.

1. *Сбор информации о сетях жертвы* – поиск административных данных (сведения о DNS, свойствах домена и т.д.) специфических сведений о топологии и операциях.

Примером является случай, когда Indrik Spider загрузил такие инструменты, как утилита Advanced Port Scanner и Lansweeper, чтобы провести внутреннюю разведку сети жертвы.

1. *Сбор информации об организации жертвы* – сбор информации об организации, в том числе названия отделов/департаментов, особенности бизнес-операций, а также роли и обязанности ключевых сотрудников.

В качестве примера можно привести инцидент, когда Moonstone Sleet собирал информацию о пострадавших организациях по электронной почте и через социальные сети.

1. *Фишинг для получения информации о жертве* - попытки обманом заставить цель разгласить информацию, часто учетные данные или другую полезную информацию, путем рассылки специальных сообщений через сторонние сервисы, отправки фишинговых ссылок или писем, использования голосовой связи.

Процедура: Scattered Spider использовал комбинацию фишинга учетных данных и социальной инженерии для получения кодов одноразовых паролей.

1. *Поиск информации на закрытых ресурсах* – сбор информации из закрытых источников (платных, частных и других), а также через dark web или cybercrime blackmarkets.

Процедура: EXOTIC LILY искала информацию о целевых лицах в бизнес-базах данных, включая RocketReach и CrunchBase.

1. *Поиск в открытых технических базах данных* (далее – БД) – сбор сведений из открытых БД и других хранилищ, таких как регистрации доменов/сертификатов, а также публичные коллекции сетевых данных/артефактов, собранных в результате трафика и/или сканирования.

Сюда относят поиск через DNS, WHOIS, CDNs, цифровые сертификаты и другие источники.

1. *Поиск по открытым веб-сайтам и доменам* – поиск информации на свободно доступных веб-сайтах и (или) в доменах.

Поиск в социальных сетях, через поисковые системы (Например, через специальные запросы в поисковых строках), в публичных репозиториях кода.

1. *Поиск на веб-сайтах, принадлежащих жертве –* сбор различных сведений , таких как названия отделов, департаментов, физические адреса, данные о ключевых сотрудниках, сведения о деловых операциях и связях.

1.3 Resource Development (TA0042)

Resource Development (развитие/получение ресурсов) – попытка получить ресурсы, которые можно использовать для поддержки операций.

В данную тактику входит техники, которые содержат создание, покупку или компрометацию (кражу) ресурсов, которые можно использовать для поддержки действий по достижению цели.

Данная тактика включает в себя 8 техник:

1. *Приобретение доступа –* получение доступа к целевой системе или сети путем покупки или использования другого способа. Это можно сделать через различные онлайн-сервисы по продаже доступа ранее взломанных систем.
2. *Приобретение инфраструктуры –* покупка, аренда, получение инфраструктуры, которую можно использовать для осуществления атак. Сюда включаются физические или облачные серверы, домены и сторонние веб-сервисы, реклама для распространения вредоносного программного обеспечения (далее – ПО).

Процедура: в одном из инцидентов Ember Bear использовал такие сервисы, как IVPN, SurfShark и Tor, чтобы обеспечить анонимность операций.

1. *Компрометация учетных записей* – использование уже существующих личностей для вызова доверия у потенциальных жертв (Например, взломанные аккаунты социальных сетей, адреса почт и др.)
2. *Компрометация инфраструктуры* – компрометация инфраструктуры третьих сторон. К инфраструктурным решениям относятся физические или облачные серверы, домены, сетевые устройства, а также сторонние веб-сервисы и DNS. Вместо того чтобы покупать, арендовать или брать в аренду инфраструктуру, противник может скомпрометировать ее и использовать на других этапах.
3. *Развитие возможностей* – создание и разработка своих собственных решений, вместо покупки или кражи. Например, создание вредоносных программ, эксплойтов и самоподписанных сертификатов, определение требований к их разработке.

Процедура: Moonstone Sleet разрабатывал вредоносные пакеты npm для доставки вредоносных программ или получения информации о жертвах.

1. *Создание учетных записей –* создание и развитие аккаунтов для создания персоны, которая будет использоваться в дальнейших операциях. Это могут быть аккаунты в соц. сетях, email-аккаунты или аккаунты у облачных провайдеров.

Пример: Ember Bear создал аккаунты на темных веб-форумах, чтобы получить различные инструменты и вредоносное ПО.

1. *Получение возможностей* – покупка, кража, скачивание из свободных источников средств для дальнейших действий. Это включает приобретение вредоносных программ, программного обеспечения (включая лицензии), эксплойтов, сертификатов и информации об уязвимостях.
2. *Постановка возможностей* – загрузка, установка, настраивание возможностей для совершения дальнейших операций. Для поддержки своих операций противнику может понадобиться взять возможности, которые он разработал (Develop Capabilities) или получил (Obtain Capabilities), и разместить их на инфраструктуре, находящейся под его контролем. Эти возможности могут быть размещены на инфраструктуре, которая ранее была куплена (арендована) противником (Acquire Infrastructure) или была скомпрометирована им иным образом (Compromise Infrastructure).

Процедура: Mustang Panda использовала подконтрольные ей серверы для проверки пикселей отслеживания, отправляемых жертвам фишинга.

1.4 Initial Access (TA0001)

Initial Access (первичный доступ) – попытка проникновения в сеть жертвы. Сюда входят техники, которые используют различные векторы проникновения, чтобы закрепиться в сети.

Данная тактика включает в себя 10 техник:

1. *Инъекция содержимого* – получение доступа к жертвам и постоянное общение с ними для внедрения вредоносного контента в системы через сетевой трафик.

Злоумышленники могут получить доступ к жертвам через взломанные каналы передачи данных, где они могут манипулировать трафиком и (или) внедрять свой собственный контент. Также скомпрометированные сетевые каналы могут использоваться для доставки дополнительной полезной нагрузки.

Процедура: MoustachedBouncer внедрял содержимое в DNS, HTTP и SMB-ответы, чтобы перенаправить жертв на поддельную страницу Windows Update для загрузки вредоносного ПО.

1. *Компрометация через зараженный сайт* – получение доступа к системе через пользователя, посещающего веб-сайт в ходе обычного просмотра. Обычно используют взломанные веб-сайты*.*
2. *Эксплуатация публичного приложения –* попытка использования слабых мест в хосте или системе (программных ошибок, временных сбоев, неправильной конфигурации), выходящих в Интернет, для получения первичного доступа к сети.

Процедура: Axiom был замечен в использовании SQL-инъекций для получения доступа к системам.

1. *Внешние сервисы удаленного доступа* – использование внешних удаленных служб для первичного доступа и (или) закрепления в сети. Например: VPN, Citrix и других механизмов доступа, позволяющих пользователям подключаться к внутренним ресурсам корпоративной сети из внешних мест.

Процедура: Akira использовал взломанные VPN-аккаунты для первоначального доступа к сетям жертв.

1. *Аппаратные дополнения* – внедрение в систему или сеть компьютерных аксессуаров, сетевого оборудования или других вычислительных устройств, которые можно использовать в качестве вектора получения доступа.
2. *Фишинг* – рассылка фишинговых сообщений для внедрения вредоносных программ и закрепления в системе или сети.
3. *Распространение через съемные носители –* копирование вредоносного ПО на съемные носители и использование функций автозапуска при внедрении носителя в аппаратное обеспечение.

Процедура: Agent.btz проникает на съемные носители и создает файл autorun.inf с инструкцией по запуску этого файла. Когда устройство вставляется в другую систему, открывается autorun.inf и загружается вредоносная программа.

1. *Компрометация во время поставок* – манипулирование продуктами или механизмами их доставки до получения конечными потребителями с целью компрометации данных или систем.

Процедура: Raccoon Stealer распространялся через загрузки взломанных программ.

1. *Использование каналов, которым доверяет жертва –* взлом или иной способ использования организаций, имеющих доверительные отношения (доступ) к жертве.
2. *Действующие учетные записи –* получение учетных данных существующих учетных записей и злоупотребление ими для получения первичного доступа и закрепления в системе или сети, для повышения привилегий или уклонения от защиты.Злоумышленники могут не использовать вредоносное ПО или инструменты в сочетании с законным доступом, который предоставляют учетные данные, чтобы затруднить обнаружение их присутствия.

Процедура: Для удаленного доступа к сетям жертв Akira использует достоверную учетную информацию, например учетные данные VPN.

1.5 Execution (TA0002)

Execution (исполнение) – попытка запуска вредоносного кода. Данная тактика состоит из техник, которые приводят к выполнению контролируемого противником кода в локальной или удаленной системе. Часто используются в паре с техниками из других тактик для достижения более широких целей.

Некоторые из техник:

1. *Команды облачного администрирования* – использование облачных служб управления для выполнения команд на виртуальных машинах. Удаленный запуск сценариев на виртуальных машинах с использованием установленных агентов виртуальных машин.

Процедура: Pacu может выполнять команды на экземплярах EC2 с помощью команды AWS Systems Manager Run Command.

1. *Интерпретатор команд и сценариев* – использование интерпретаторов команд и сценариев для выполнения команд, скриптов или двоичных кодов. Данные интерфейсы и языки обеспечивают взаимодействие с компьютерными системами и являются общими для различных платформ.

Процедура: Bandook может поддерживать команды для выполнения полезной нагрузки на основе Java.

1. *Развертывание контейнера* - развертывание контейнера в среде для облегчения выполнения кода или обхода защиты.

Процедура: Peirates могут развернуть стручок, который монтирует корневую файловую систему узла, а затем выполнить команду для создания обратной оболочки на узле.

1. *Взаимодействие с родным API* – данные API обеспечивают контролируемые средства вызова низкоуровневых служб ОС в ядре, таких как аппаратные средства или устройства, память и процессы. Также они используются ОС во время загрузки системы, а также для выполнения задач и запросов во время рутинных операций.
2. *Утилита Cron* - использование утилиты Cron для планирования задач для первоначального или периодического выполнения вредоносного кода (Утилита cron - это планировщик заданий на основе времени для Unix-подобных операционных систем).

Процедура: Exaramel для Linux использует crontab для сохранения, если у него нет привилегий root.

1. *Средства развертывания программного обеспечения* – получение доступа к централизованным наборам программного обеспечения, установленным на предприятии, и использовать их для выполнения команд и перемещения по сети.

Процедура: Команда Sandworm Team использовала коммерческий инструмент RemoteExec для удаленного выполнения кода без агентов.

1. *Windows Management Instrumentation* – использование Windows Management Instrumentation для выполнения вредоносных команд и полезных нагрузок. (WMI - это функция администрирования, которая обеспечивает единую среду для доступа к компонентам системы Windows).

1.6 Persistence (TA0003)

Persistence (настойчивость, стойкость) – попытка удержания позиции.

Данная тактика заключается в техниках, которые злоумышленник использует для сохранения доступа к системам после перезагрузок, изменения учетных данных и других прерываний, которые могут лишить их доступа. Техники включают любые изменения доступа, действий или конфигурации, которые позволяют удерживать свои позиции в системах.

Некоторые из техник:

1. *Манипулирование аккаунтом* - манипулирование учетными записями, чтобы сохранить и (или) расширить доступ к системам жертвы. Примеры: изменение учетных данных или групп разрешений, итеративное обновление пароля для обхода политик длительности пароля и сохранения срока действия скомпрометированных учетных данных.
2. *Выполнение автозапуска при загрузке или входе в систему* - настройка системных параметров на автоматическое выполнение программы при загрузке системы или входе в систему для сохранения доступа или получения привилегий более высокого уровня на взломанных системах.
3. *Модификация ярлыков* - создание или модификация ярлыков, которые могут запускать программу во время загрузки системы или входа пользователя в систему.

Процедура: Comnie устанавливает постоянство через .lnk-файл в пути запуска жертвы.

1. *Активная установка* – сохранение доступа за счет добавления ключа реестра в Active Setup в local machine. Значение, хранящееся в ключе реестра, будет выполняться после входа пользователя в компьютер.
2. *Сценарии инициализации загрузки или входа в систему* - использование сценариев, автоматически выполняемых при загрузке или инициализации входа в систему, для сохранения доступа. Могут использоваться для выполнения административных функций, которые часто могут запускать другие программы или отправлять информацию на внутренний сервер регистрации.

Процедура: Rocke установливает скрипт запуска «init.d» для поддержания постоянства.

1. *Создание или модификация системных процессов* – создание или модификация процессов системного уровня для многократного выполнения вредоносной полезной нагрузки в рамках персистенции.

Процедура: Exaramel для Linux имеет жестко закодированное местоположение, которое он использует для сохранения доступа, если система запуска - Upstart или System V, и он запущен от имени root.

1. *Шимминг приложений* – выполнение вредоносных программ или занесение вредоносного контента с помощью шимов приложений для сохранения доступа или повышения привилегий.

1.7 Privilege Escalation (TA0004)

Privilege Escalation (Повышение привилегий) – получение разрешения более высокого уровня. Включает в себя техники, которые злоумышленники используют для получения разрешений более высокого уровня в системе или сети. Основная угроза состоит в получении злоумышленниками непривилегированного доступа с большими возможностями.

Некоторые из техник:

1. *Обман механизмов контроля повышения привилегий* – злоумышленники обходят механизмы, предназначенные для контроля повышения привилегий, чтобы получить разрешения более высокого уровня.

Некоторым сотрудникам, задачи которых отнесены к категории повышенного риска, предоставляются определенные полномочия с помощью таких механизмов. Злоумышленник может использовать некоторые методы для обмана этих механизмов.

Процедура: В Raspberry Robin реализована разновидность метода ucmDccwCOMMethod, использующая бэкдор Windows AutoElevate для обхода UAC с повышением привилегий.

1. *Манипуляции с маркерами доступа* – изменение маркеров доступа для работы под другим контекстом безопасности пользователя или системы для выполнения действий и обхода контроля доступа.

Windows использует маркеры доступа для определения права собственности на запущенный процесс. Пользователь может манипулировать маркерами доступа, чтобы заставить запущенный процесс выглядеть так, будто он является дочерним процессом другого процесса или принадлежит не тому пользователю, который запустил процесс. Когда это происходит, процесс также принимает контекст безопасности, связанный с новым маркером.

Процедура: AppleSeed может получить привилегии системного уровня, передав SeDebugPrivilege в API AdjustTokenPrivilege.

1. *Выполнение автозапуска при загрузке или входе в систему* – настройка системных параметров на автоматическое выполнение программы при загрузке системы или входе в систему для сохранения доступа или получения привилегий более высокого уровня на скомпрометированных системах.

Злоумышленники могут добиться постоянства, добавив программу в папку запуска или указав на нее ключ запуска в реестре. Использование Security Support Provider.

1. *Создание или модификация системного процесса* - созданиеь или модификация процессов системного уровня для многократного выполнения вредоносной полезной нагрузки в рамках персистенции.
2. *Изменение политики домена или арендатора* - изменение параметров конфигурации домена или арендатора идентификационных данных, чтобы обойти защиту и/или повысить привилегии в централизованно управляемых средах.

Такие службы предоставляют централизованные средства управления ресурсами идентификации, такими как устройства и учетные записи, и часто включают параметры конфигурации, которые могут применяться между доменами или арендаторами.

1. *Побег на хост* – выход из контейнера и получение доступа к базовому хосту. Это позволяет злоумышленнику получить доступ к другим контейнерным ресурсам с уровня хоста или к самому хосту.

Процедура: Hildegard использует инструмент BOtB, с помощью которого можно вынимать контейнеры.

1.8 Defense Evasion (TA0005)

Defense Evasion (Уклонение от защиты) – попытка злоумышленника избежать обнаружения в ходе компрометации. К таким техникам относят деинсталляцию (отключение) защитного ПО или обфускацию (шифрование) данных и скриптов. Использование доверенных процессов для маскировки вредоносного ПО.

Некоторые техники:

1. *Обман механизмов контроля повышения привилегий* – злоумышленники обходят механизмы, предназначенные для контроля повышения привилегий, чтобы получить разрешения более высокого уровня.

Некоторым сотрудникам, задачи которых отнесены к категории повышенного риска, предоставляются определенные полномочия с помощью таких механизмов. Злоумышленник может использовать некоторые методы для обмана этих механизмов. Например, кэширование sudo и (или) использовать файл sudoers для повышения привилегий.

Процедура: В Raspberry Robin реализована разновидность метода ucmDccwCOMMethod, использующая бэкдор Windows AutoElevate для обхода UAC с повышением привилегий.

1. *Создание образа на хосте* - создание образа контейнера непосредственно на хосте для обхода средств защиты, отслеживающих получение вредоносных образов из публичного реестра. В качестве примера можно привести API Docker, в который может быть отправлен удаленный запрос на сборку, содержащий Dockerfile, который извлекает из публичного или локального реестра ванильный базовый образ, например alpine, а затем создает на его основе собственный образ.
2. *Уклонение от отладчиков* – использование различных средств для обнаружения и обхода отладчика (отладчики обычно используются blue team для отслеживания и (или) анализа выполнения потенциальных полезных нагрузок вредоносного ПО).

Процедура: DRATzarus может использовать IsDebuggerPresent, чтобы определить, присутствует ли отладчик у жертве.

1. *Деобфускация/декодирование файлов или информации* - использование обфусцированных файлов или информации, чтобы скрыть артефакты вторжения от антивируса и затруднить анализ вредоносного кода.
2. *Прямой доступ к томам* - получение прямого доступа к томам, чтобы обойти контроль доступа к файлам и мониторинг файловой системы.

Windows позволяет программам иметь прямой доступ к логическим томам. Программы с прямым доступом могут читать и записывать файлы непосредственно с диска, анализируя структуры данных файловой системы.

Процедура: Scattered Spider создавал теневые копии томов на дисках виртуального контроллера домена, чтобы извлечь файл NTDS.dit.

1. *Execution Guardrails* – использование Execution Guardrails для ограничения выполнения или действий на основе предоставленных злоумышленником специфических условий, которые, как ожидается, будут присутствовать у цели. Защитные рельсы обеспечивают выполнение полезной нагрузки только против намеченной цели и уменьшают сопутствующий ущерб от злоумышленников.
2. *Модификация разрешений файлов и каталогов* - изменение разрешений (атрибутов) файлов и каталогов для обхода списков контроля доступа (ACL) и получения доступа к защищенным файлам.

1.9 Credential Access (TA0006)

Credential Access (Доступ к учетным данным) – попытка кражи имен учетных записей и паролей. Состоит из техник кражи учетных данных (имен учетных записей, паролей). Данный техники включают в себя кейлоггинг или сброс учетных данных (Keylogging — это процесс перехвата нажатий клавиш на компьютере или мобильном устройстве без ведома пользователя). Использование легитимных учетных данных может дать злоумышленникам доступ к системам, затруднить их обнаружение и дать возможность создать больше учетных записей для достижения своих целей.

Некоторые техники:

1. *Adversary-in-the-Middle* – попытка занять позицию между двумя или более сетевыми устройствам для поддержки последующих действий, таких как сетевой сниффинг, манипулирование передаваемыми данными или атаки повторного воспроизведения.

Сюда входят отравление ARP-кэша, подмена DHCP, создание «злого» двойника» (размещение у себя кажущиеся настоящими точек доступа Wi-Fi, чтобы обманом заставить пользователей подключиться к вредоносным сетям).

1. *Брутфорс* – попытка получения доступа к учетным записям при неизвестных паролях или наличии хэшей паролей путем угадывания (перебора). Особенности: использование итерационного механизма.

Использование списка с часто используемыми паролями против многих различных учетных записей; данных из репозитория конфигурации, чтобы получить хэшированные учетные данные сетевых устройств; учётных данных, полученных из дампов не связанных между собой учетных записей, чтобы получить доступ к целевым учетным записям путем дублирования учетных данных.

1. *Принудительная аутентификация* – сбор учетных данных с помощью вызова и принуждения пользователя к автоматическому предоставлению информации об аутентификации через механизм, который злоумышленники могут перехватить.

Процедура: DarkHydrus использовал Template Injection для запуска окна аутентификации, в котором пользователи должны были ввести свои учетные данные.

1. *Подделка веб-удостоверений* - подделка учетных данных, которые можно использовать для получения доступа к веб-приложениям или интернет-сервисам (Web Cookies, SAML Tokens).
2. *Перехват ввода* - использование методов перехвата пользовательского ввода для получения учетных данных или сбора информации.

При обычном использовании системы пользователи часто вводят учетные данные в различных местах. Механизмы перехвата ввода могут быть прозрачными для пользователя (например, Credential API Hooking) или полагаться на то, чтобы обманом заставить пользователя ввести данные в службу, которую он считает подлинной.

1. *Модификация процесса аутентификации* – модификация механизмов и процессов аутентификации (За сбор, хранение и проверку учетных данных отвечают такие механизмы, как процесс Local Security Authentication Server (LSASS) и Security Accounts Manager (SAM) в Windows, подключаемые модули аутентификации (PAM) в системах на базе Unix и плагины авторизации в системах MacOS), чтобы получить доступ к учетным данным пользователей или открыть необоснованный доступ к учетным записям.
2. *Генерация запросов многофакторной аутентификации* - попытка обхода механизмов многофакторной аутентификации (MFA) и получения доступа к учетным записям путем генерации запросов MFA, отправляемых пользователям.

Процедура: LAPSUS$ рассылал целевым пользователям запросы MFA в надежде, что легитимный пользователь даст необходимое разрешение.

1.10 Discovery (TA0007)

Discovery (Обнаружение, разбор) – попытка разбора в окружении жертвы. Иными словами, получение знаний о системе и внутренней сети. Техники, относящиеся к данной тактике, помогают противнику наблюдать за окружением и ориентироваться в нем, прежде чем приступать к дальнейшим действиям.

Некоторые техники:

1. *Обнаружение аккаунтов* – попытка получения списка действующих учетных записей, имен пользователей или адресов электронной почты в системе или во взломанной среде (локальные учетные записи, аккаунты от почт, облачные учетные записи).
2. *Обнаружение окон приложений* – попытка получить список открытых окон приложений. Списки окон могут передавать информацию о том, как используется система.

Процедура: Aria-body умеет определять названия запущенных окон на взломанном хосте.

1. *Обнаружение облачной инфраструктуры* - попытка обнаружить инфраструктуру и ресурсы, доступные в среде «infrastructure-as-a-service» (IaaS).

Сюда входят ресурсы вычислительных служб, такие как экземпляры, виртуальные машины и моментальные снимки, а также ресурсы других служб, включая службы хранения данных и базы данных.

1. *Приборная панель облачного сервиса* - с помощью украденных учетных данных злоумышленник может использовать графический интерфейс приборной панели облачного сервиса для получения полезной информации из рабочей облачной среды, например о конкретных сервисах, ресурсах и функциях.

Процедура: Scattered Spider злоупотребляд инвентаризацией AWS Systems Manager для выявления целей во взломанной сети до начала бокового перемещения.

1. *Обнаружение контейнеров и ресурсов* - попытка обнаружить контейнеры и другие ресурсы, доступные в среде контейнеров.

Процедура: Hildegard использовал masscan для поиска кубелетов и kubelet API для поиска дополнительных запущенных контейнеров.

1. *Уклонение от отладчиков* - использование различных средств для обнаружения и обхода отладчиков. Отладчики обычно используются защитниками для отслеживания и/или анализа выполнения потенциальных полезных нагрузок вредоносного ПО.
2. *Обнаружение доверительных отношений в домене* - сбор информации о доверительных отношениях доменов, которая может использоваться для определения возможностей латерального перемещения в средах Windows с несколькими доменами/лесами.

Процедура: BADHATCH может использовать nltest.exe /domain\_trusts для обнаружения доверительных отношений доменов на взломанной машине.

1.11 Lateral Movement (TA0008)

Lateral Movement (латеральное перемещение, перемещение внутри сети) – состоит из техник, которые злоумышленники используют для проникновения и контроля удаленных систем в сети.

Достижение цели часто предполагает перемещение по нескольким системам и учетным записям для получения доступа. Злоумышленники могут устанавливать собственные инструменты удаленного доступа для осуществления Lateral Movement.

Некоторые техники:

1. *Эксплуатация удаленных служб* - использование удаленных сервисов для получения несанкционированного доступа к внутренним системам, находящимся в сети.

Эксплуатация уязвимости программного обеспечения происходит, когда злоумышленник использует ошибку в программе, службе или в самом программном обеспечении или ядре операционной системы для выполнения управляемого злоумышленником кода. Общей целью эксплуатации удаленных сервисов после компрометации является горизонтальное перемещение внутри сети, позволяющее получить доступ к удаленной системе.

1. *Внутренний спирфишинг* – при получении доступа к учетным записям или системам в среде злоумышленники могут осуществить внутренний спирфишинг, чтобы получить доступ к дополнительной информации или скомпрометировать других пользователей в той же организации.

Процедура: Gamaredon Group использовала модуль Outlook VBA на зараженных системах для рассылки фишинговых писем с вредоносными вложениями другим сотрудникам организации.

1. *Латеральная передача инструментов* – передача инструментов или других файлов между системами во взломанной среде.

Процедура: Во время атаки на электроэнергию в Украине в 2015 году команда Sandworm перемещала свои инструменты внутри корпоративной сети и между ИКС и корпоративной сетью.

1. *Перехват сеансов удаленных служб* - взятие под контроль уже существующих сеансов работ с удаленными службами, чтобы перемещаться в среде.

Пользователи могут использовать действительные учетные данные для входа в службу, специально предназначенную для приема удаленных подключений, например telnet, SSH и RDP.

1. *Удаленные сервисы* - использование Valid Accounts для входа в службу, принимающую удаленные соединения, например telnet, SSH и VNC, чтобы выполнять действия от имени жертвы.
2. *Средства развертывания программного обеспечения* – получение доступа к централизованным наборам программного обеспечения, установленным на предприятии, и использование их для выполнения команд и перемещения по сети. Примеры таких решений включают: SCCM, HBSS, Altiris, AWS Systems Manager, Microsoft Intune, Azure Arc и GCP Deployment Manager.

Процедура: Silence использовал RAdmin, программное обеспечение для удаленного управления рабочими станциями и банкоматами.

1. *Порча общего содержимого* - доставка полезной нагрузки на удаленные системы, добавляя содержимое в общие места хранения, такие как сетевые диски или внутренние хранилища кода.

Процедура: Cinnamon Tempest развернул вымогательскую программу из пакетного файла в сетевом ресурсе.

1.12 Collection (TA0009)

Collection (Сбор) – попытка сбора данных, которые могут представлять интерес для достижения цели злоумышленника.

Сбор состоит из методов, которые противник может использовать для сбора информации, и источников, из которых собирается информация, имеющая отношение к достижению целей противника.

К распространенным источникам данных относятся различные типы дисков, браузеры, аудио- и видеофайлы, электронная почта.

Некоторые из техник:

1. *Adversary-in-the-Middle* – попытка занять позицию между двумя или более сетевыми устройствам для поддержки последующих действий, таких как сетевой сниффинг, манипулирование передаваемыми данными или атаки повторного воспроизведения.

Сюда входят отравление ARP-кэша, подмена DHCP, создание «злого» двойника» (размещение у себя кажущиеся настоящими точек доступа Wi-Fi, чтобы обманом заставить пользователей подключиться к вредоносным сетям).

1. Архивирование собранных данных - сжатие и (или) шифрование собранных данных перед их утечкой. Сжатие данных может помочь запутать собранные данные и минимизировать объем данных, передаваемых по сети.

Шифрование может быть использовано для того, чтобы скрыть информацию от обнаружения или сделать ее менее заметной при проверке защитником.

1. Перехват аудиозаписей - использование периферийных устройств компьютера (например, микрофоны и веб-камеры) или приложения (например, службы голосовых и видеозвонков) для захвата аудиозаписей с целью прослушивания конфиденциальных разговоров для сбора информации.

Процедура: DarkComet может прослушивать разговоры жертв через микрофон системы.

1. Автоматизированный сбор - оказавшись в системе или сети, противник может использовать автоматизированные методы сбора внутренних данных. К таким методам можно отнести использование интерпретатора команд и сценариев для поиска и копирования информации, соответствующей заданным критериям, таким как тип, расположение или имя файла, через определенные промежутки времени.
2. Перехват сеанса браузера – использование уязвимостей в системе безопасности и функциональности, присущей программному обеспечению браузера, чтобы изменить содержимое, модифицировать поведение пользователя и перехватить информацию в рамках различных методов захвата сеанса браузера.

Процедура: Dridex может осуществлять атаки на браузер через веб-инъекции для кражи такой информации, как учетные данные, сертификаты и файлы cookie.

1. Данные буфера обмена - сбор данных, хранящихся в буфере обмена при копировании пользователем информации внутри или между приложениями.
2. Данные из облачных хранилищ - получение доступа к данным из облачных хранилищ.

Процедура: AADInternals может собирать файлы из OneDrive пользователя.

1.13 Command and Control (TA0011)

Command and Control (командование и управление) – попытка установки связи со взломанными системами для их дальнейшего контроля. Данная тактика включается в себя техники, которые злоумышленники используют для связи с контролируемыми ими системами в сети жертвы.

Обычно противники пытаются имитировать обычный, ожидаемый трафик, чтобы избежать обнаружения.

Некоторые из техник:

1. Протокол прикладного уровня – обмен данными с использованием протоколов прикладного уровня OSI, чтобы избежать обнаружения/сетевой фильтрации, смешиваясь с существующим трафиком.

Команды удаленной системе, а зачастую и результаты этих команд, будут встроены в трафик протокола между клиентом и сервером.

1. Связь через съемные носители - осуществление командования и управления между взломанными узлами в потенциально не связанных сетях с использованием съемных носителей для передачи команд от системы к системе (Обе системы должны быть скомпрометированы).

Процедура: APT28 использует инструмент, который перехватывает информацию с компьютеров, находящихся в воздухе, через зараженный USB и передает ее на подключенный к сети компьютер, когда USB вставляется.

1. Инъекция содержимого - получение доступа к жертвам и постоянное общение с ними для внедрения вредоносного контента в системы через сетевой трафик.

Процедура: Disco получал первоначальный доступ и исполнение путем внедрения контента в DNS, HTTP и SMB-ответы целевых узлов, которые перенаправляли их на загрузку вредоносных файлов.

1. Кодирование данных - кодирование данных с помощью стандартной или нестандартной системы кодирования данных, чтобы усложнить обнаружение содержимого командно-контрольного трафика. К распространенным схемам кодирования данных относятся ASCII, Unicode, шестнадцатеричная, Base64 и MIME.

Процедура: H1N1 обфусцирует трафик C2 с помощью измененной версии base64.

1. Обесценивание данных - завуалирование командного и управляющего трафика, чтобы затруднить его обнаружение.

Это включает в себя множество методов, таких как добавление нежелательных данных в трафик протокола, использование стеганографии или выдача себя за легитимные протоколы.

Процедура: FlawedAmmyy может затуманивать части начального рукопожатия C2.

1. Динамическое разрешение - динамическая установка соединения с командно-контрольной инфраструктурой для обхода обычных средств обнаружения и устранения последствий.

Процедура: Gelsemium может использовать динамические доменные имена DNS в C2.

1. Зашифрованный канал - использование алгоритма шифрования (симметричная и ассиметричная криптография) для сокрытия командного и управляющего трафика, не полагаясь на защиту, присущую протоколу связи.

Процедура: Chaes использовала шифрование для своего канала C2.

1.14 Exfiltration (TA0010)

Exfiltration (Эксфильтрация) – попытка украсть данные. Включает в себя техники, которые злоумышленники могут использовать для кражи данных из сети жертвы.

Некоторые техники:

1. Автоматизированная эксфильтрация - вывод данных, такие как конфиденциальные документы, с помощью автоматизированной обработки после их сбора в процессе сбора. При дублировании трафика используется зеркалирование трафика для автоматизации утечки данных через взломанную инфраструктуру.
2. Ограничения на размер передаваемых данных - передача данных не целыми файлами, а фрагментами фиксированного размера или ограничение размеров пакетов ниже определенного порога (используется для обхода пороговых предупреждений при передаче).
3. Эксфильтрация по альтернативному протоколу - похищение данных путем передачи их по протоколу, отличному от существующего командно-контрольного канала.

Данные также могут быть отправлены в альтернативное сетевое местоположение с главного командно-контрольного сервера.

Процедуры: Chaes передавал собранные данные с зараженной машины в C2, иногда используя протокол MIME.

1. Эксфильтрация данных через физический носитель - попытка вывести данные через физический носитель, например съемный диск. Физический носитель или устройство может использоваться в качестве конечной точки эксфильтрации или для переброски между иными отключенными системами.
2. Эксфильтрация через веб-сервис – использование злоумышленниками существующего, легитимного внешнего веб-сервиса для эксфильтрации данных, а не свой основной канал управления и контроля. Могут существовать правила брандмауэра, разрешающие трафик к этим сервисам.

Процедура: AppleSeed выводил файлы через веб-сервисы.

1. Передача по расписанию - планирование передачи данных только в определенное время суток или через определенные промежутки времени. Это может быть сделано для того, чтобы совместить трафик с нормальной активностью или доступностью.
2. Передача данных в облачный аккаунт - вывод данных путем передачи их, в том числе путем обмена (синхронизации) и создания резервных копий облачных сред, на другой контролируемый облачный аккаунт в том же сервисе.

Процедура: INC Ransom использовала Megasync для утечки данных в «облако».

1.15 Impact (TA0040)

Impact (Воздействие) – попытка манипулирования пользовательскими системами и данными, прерывание их работы или уничтожение.

Данная тактика состоит из техник, которые включают уничтожение или подделку данных. Злоумышленник использует их для нарушения доступности или нарушения целостности путем манипулирования бизнес-процессами и операционными процессами.

Некоторые из техник:

1. Удаление доступа к учетным записям - нарушение доступности системных и сетевых ресурсов путем блокирования доступа к учетным записям, используемых легитимными пользователями.

Процедура: LockerGoga был замечен за изменением паролей учетных записей и выходом из системы текущих пользователей.

1. Данные, зашифрованные для воздействия - шифрование данных на целевых системах или на большом количестве систем в сети, чтобы прервать доступ к системным и сетевым ресурсам. Злоумышленники могут сделать хранящиеся данные недоступными, зашифровав файлы или данные на локальных и удаленных дисках и закрыв доступ к ключу дешифрования.

Процедура: Apostle создает новые зашифрованные версии файлов, а затем удаляет оригиналы, причем новые имена файлов состоят из случайного GUID и расширения «.lock».

1. Манипулирование данными - попыткая повлиять на бизнес-процессы, понимание организации или принятие решений путем вставления, удаления или манипулирования данными – угроза их ценности.
2. Стирание диска - стирание или повреждение необработанных дисковых данных на определенных системах или в большом количестве в сети, чтобы прервать доступ к системным и сетевым ресурсам.

Злоумышленники могут стирать произвольные части дисковых данных и/или затирать дисковые структуры, например главную загрузочную запись (MBR). Может быть предпринята попытка полного стирания всех секторов диска.

1. Отказ в обслуживании конечной точки - проведение атак типа «Denial of Service» (DoS), чтобы снизить или заблокировать доступность сервисов для пользователей.

DoS-атаки на конечных точках могут осуществляться путем исчерпания системных ресурсов, на которых размещены службы, или использования системы для создания устойчивого состояния сбоя.

Процедура: OnionDuke имеет возможность использовать модуль отказа в обслуживании.

1. Финансовые кражи - похищение денежных ресурсов у целей с помощью вымогательства, социальной инженерии, технических краж или других методов, направленных на получение собственной финансовой выгоды за счет доступности этих ресурсов для жертв.

Процедура: DarkGate может развернуть полезную нагрузку, способную перехватить учетные данные, связанные с криптовалютными кошельками.

1. Препятствие восстановлению системы - удаление встроенных данных и отключение службы, предназначенных для восстановления поврежденной системы, чтобы предотвратить восстановление.

Это может лишить доступа к имеющимся резервным копиям и возможностям восстановления.

Процедура: Akira удаляла теневые копии системных томов с помощью команд PowerShell.

1.16 Сравнение матрицы MITRE с Cyber Kill Chain

Cyber Kill Chain — это модель, разработанная Lockheed Martin, которая описывает этапы кибер-атаки от разведки до достижения цели. Она помогает в анализе угроз и построении защиты.

*Таблица 1. Этапы Cyber Kill Chain*

|  |  |  |
| --- | --- | --- |
| **Этап** | **Описание** | **Пример** |
| Разведка (Reconnaissance) | Злоумышленник выбирает цель, изучает ее и пытается выявить уязвимые места в сети цели. | Поиск в социальных сетях, сканирование портов. |
| Создание оружия (Weaponization) | Злоумышленник создает вредоносное ПО для удаленного доступа, например вирус или червь, адаптированный к одной или нескольким уязвимостям. | Создание заражённого PDF или EXE. |
| Доставка (Delivery) | Злоумышленник передает оружие цели (например, через вложения электронной почты, веб-сайты или USB-накопители). | Фишинг с вложением. |
| Эксплуатация (Exploitation) | Запускается программный код вредоносного оружия, который выполняет действия в целевой сети для использования уязвимости. | Запуск эксплойта для RCE. |
| Установка (Installation) | Вредоносное оружие устанавливает точку доступа (например, «бэкдор»), которой может воспользоваться злоумышленник. | Запуск Cobalt Strike. |
| Командование и управление (C2) | Вредоносное ПО позволяет злоумышленнику получить постоянный доступ к целевой сети по принципу «руки на клавиатуре» | DNS-туннелирование. |
| Действия по цели (Actions on Objectives) | Злоумышленник предпринимает действия для достижения своей цели, такие как утечка данных, уничтожение данных или шифрование с целью получения выкупа. | Атака ransomware. |

Применение в кибербезопасности:

1. Анализ угроз;
2. Расследование инцидентов;
3. Тестирование на проникновение.

*Таблица 2. Сравнение Cyber Kill Chain с матрицей MITRE*

|  |  |  |
| --- | --- | --- |
| **Критерий** | ***MITRE ATT&CK*** | ***Cyber Kill Chain*** |
| Форма модели | Тактико-техническая матрица | Линейная пошаговая модель |
| Порядок этапов | Тактики могут использоваться злоумышленниками в любом порядке | Линейное исполнение 7 фиксированных этапов |
| Количество этапов (техник) | 14 тактик (более 200 техник и подтехник) | 7 обобщенных этапов |
| Фокусировка | Тактики, техники, процедуры | Жизненный цикл атаки |
| Применение | Используется в качестве основы для разработки конкретных моделей и методологий угроз в частном секторе, в правительстве, а также в сообществе продуктов и услуг в области кибербезопасности. | Анализ угроз, расследование инцидентов, тестирование на проникновение. |
| Доступ | Безвозмездный | Находится в открытом доступе |
| Особенности | Применяется для глубокого анализа, учитывает реальные инциденты (процедуры), моделирование сложных атак, учитывает как внешние, так и внутренние угрозы | Простота и наглядность, небольшое количество четко сформулированных этапов, возможность пошаговой реконструкции инцидента, слабо учитывает внутренние угрозы |

# ГЛАВА 2. Изучение и описание информации с сайта OWASP

2.1 Общие сведения о проекте

OWASP (Open Web Application Security Project) — это открытый проект, направленный на улучшение безопасности программного обеспечения, предоставляющий ресурсы и инструменты для идентификации и устранения угроз безопасности веб-приложений.

2.2 Шапка

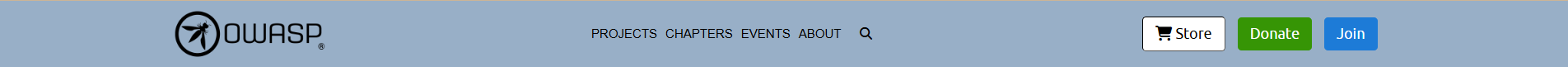


Рисунок 1. Шапка сайта OWASP

В шапке сайта находятся:

* Логотип OWASP
* Вкладка «Projects», при наведении на которую открывается список с возможность перехода ко всем проектам OWASP, разным типам проектов, а также ссылка на страницу для создания собственного проекта.
* Вкладка «CHAPTERS», содержащая список с регионами, в которых у OWASP есть отделения.
* Вкладка «EVENTS», предлагающая для просмотра последние события и возможность перехода на страницу со всеми событиями, связанными с OWASP.
* Вкладка «ABOUT», содержащая информацию о разных составляющих проекта OWASP.
* Три кнопки – Store, Donate, Join; они отвечают за переход на страницы магазина, денежной поддержки проекта, для присоединения к проекту соответственно.

2.3 Главная страница

Вверху главной страницы присутствует краткая информация о проекте, его достоинства и особенности.

Далее идет информация о лицах, которые могут принять участие в проектах от OWASP и общая информация о руководстве над этими проектами.

Ниже приведена диаграмма жизненного цикла разработки программного обеспечения с отображением проектов OWASP.

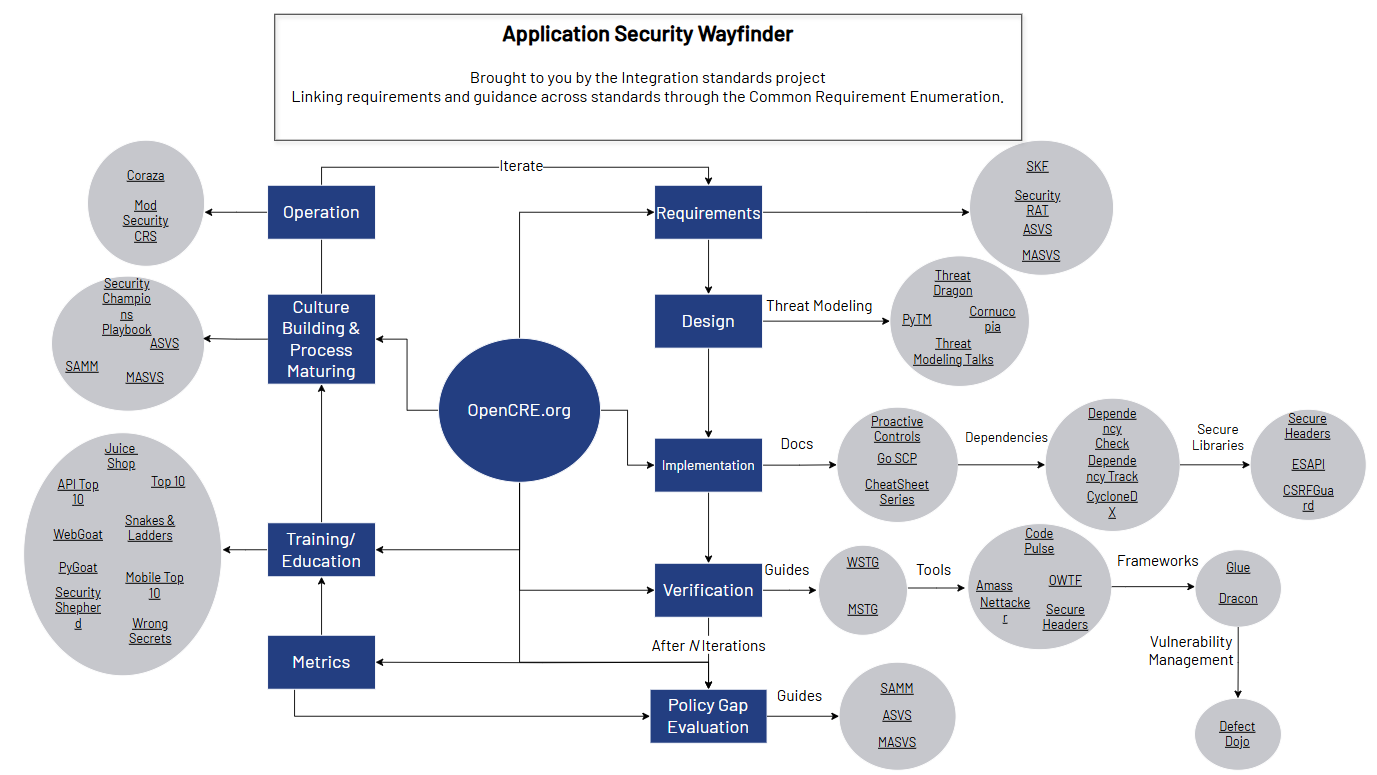


Рисунок 2. Отображение проектов OWASP на диаграмме жизненного цикла разработки ПО

Перечисление категорий, в которые организованы проекты OWASP:

* флагманские проекты;
* производственные проекты;
* другие проекты.

Далее идет список проектов, относящихся к категориям «флагманские» и «производственные».

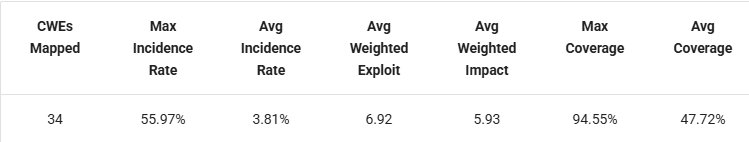
Далее рассмотрим различные проекты OWASP.

2.4 OWASP Top Ten

OWASP Top 10 — это стандартный документ по повышению осведомленности для разработчиков и безопасности веб-приложений. Он представляет собой широкий консенсус относительно наиболее критических рисков безопасности для веб-приложений. Использование OWASP Top 10, возможно, является самым эффективным первым шагом к изменению культуры разработки программного обеспечения в организациях.

Топ 10 основных рисков безопасности веб-приложений за 2021 год.

**2.4.1 A01:2021 – Broken Access Control**



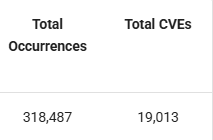


Рисунок 3. Факторы «Неисправного контроля доступа»

Контроль доступа обеспечивает соблюдение политики таким образом, что пользователи не могут действовать вне своих предполагаемых разрешений. Сбои обычно приводят к несанкционированному раскрытию информации, изменению или уничтожению всех данных или выполнению бизнес-функции вне рамок пользователя.

Общие уязвимости контроля доступа включают:

* Нарушение принципа наименьших привилегий или отказа по умолчанию, когда доступ должен быть предоставлен только определенным возможностям, ролям или пользователям, но доступен любому.
* Обход проверок контроля доступа путем изменения URL-адреса (подделка параметров или принудительный просмотр), внутреннего состояния приложения или HTML-страницы либо путем использования инструмента атаки, изменяющего запросы API.
* Разрешение просматривать или редактировать чужую учетную запись, предоставляя ее уникальный идентификатор (небезопасные прямые ссылки на объекты)
* Доступ к API с отсутствующими элементами управления доступом для POST, PUT и DELETE.
* Повышение привилегий. Действия в качестве пользователя без входа в систему или действия в качестве администратора при входе в систему как пользователь.
* Манипулирование метаданными, например, воспроизведение или подделка токена контроля доступа JSON Web Token (JWT), а также манипулирование файлами cookie или скрытыми полями с целью повышения привилегий или злоупотребления аннулированием JWT.
* Неправильная конфигурация CORS допускает доступ к API из неавторизованных/ненадежных источников.
* Принудительный просмотр аутентифицированных страниц как неаутентифицированный пользователь или привилегированных страниц как обычный пользователь.

Контроль доступа эффективен только в доверенном коде на стороне сервера или API без сервера, где злоумышленник не может изменить проверку контроля доступа или метаданные.

Для предотвращения:

* За исключением общедоступных ресурсов, запрещайте их по умолчанию.
* Реализуйте механизмы контроля доступа один раз и используйте их повторно во всем приложении, в том числе минимизируйте использование кросс-оригинального обмена ресурсами (CORS).
* Типовые средства управления доступом должны обеспечивать право собственности на запись, а не допускать, что пользователь может создавать, читать, обновлять или удалять любую запись.
* Уникальные требования к бизнес-ограничениям приложения должны обеспечиваться моделями доменов.
* Отключите листинг каталогов веб-сервера и убедитесь, что метаданные файлов (например, .git) и файлы резервных копий не присутствуют в корнях веб-серверов.
* Регистрируйте сбои в управлении доступом, предупреждайте администраторов при необходимости (например, при повторных сбоях).
* Ограничьте доступ к API и контроллеру, чтобы минимизировать вред от автоматизированных средств атаки.
* Государственные идентификаторы сессий должны аннулироваться на сервере после выхода из системы. Токены JWT без состояния должны быть недолговечными, чтобы окно возможностей для злоумышленника было минимальным. Для более долгоживущих JWT настоятельно рекомендуется следовать стандартам OAuth для отзыва доступа.

Примеры сценариев атак:

* Злоумышленник просто принудительно переходит на целевые URL-адреса. Для доступа к странице администратора требуются права администратора.

Если неаутентифицированный пользователь может получить доступ к любой из страниц, это недостаток. Если неадминистратор может получить доступ к странице администратора, это недостаток.

* Приложение использует непроверенные данные в SQL-запрос, который обращается к информации об учетной записи.

Злоумышленник просто изменяет параметр 'acct' в браузере, чтобы отправить любой номер счета. При отсутствии правильной проверки злоумышленник может получить доступ к учетной записи любого пользователя.

**2.4.2 A02:2021 – Cryptographic Failures**

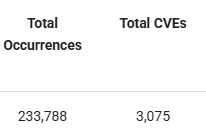
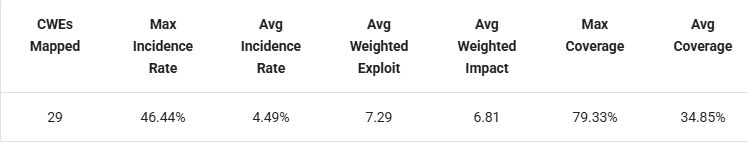


Рисунок 4. Факторы «Криптографических сбоев»

Сбои, связанные с криптографическим шифрованием (или его отсутствием) часто приводят к раскрытию конфиденциальных данных. К числу распространенных слабых мест (Common Weakness Enumerations, CWE) относятся CWE-259: Использование жестко закодированного пароля, CWE-327: Сломанный или рискованный криптоалгоритм и CWE-331: Недостаточная энтропия.

Прежде всего, необходимо определить потребности в защите данных, находящихся в пути и в состоянии покоя. Например, пароли, номера кредитных карт, медицинские записи, личная информация и коммерческие тайны требуют дополнительной защиты так как в большинстве случаев эти данные подпадают под действие законов.

Для всех таких данных:

* Проверьте на передачу данных в виде открытого текста. Это касается таких протоколов, как HTTP, SMTP, FTP, а также использующих обновления TLS, например STARTTLS. Внешний интернет-трафик опасен. Проверьте весь внутренний трафик.
* Проверьте: не используются ли старые или слабые криптографические алгоритмы или протоколы по умолчанию или в старом коде.
* Проверьте на использование криптографических ключей по умолчанию, генерацию или повторное использование слабых криптографических ключей, отсутствие надлежащего управления ключами или их ротация. Проверяются ли криптографические ключи в репозиториях исходного кода?
* Не применяется ли шифрование, например, отсутствуют ли директивы безопасности или заголовки HTTP-заголовков (браузера)?
* Проверяется ли должным образом полученный сертификат сервера и цепочка доверия?
* Игнорируются ли векторы инициализации, используются ли они повторно или не генерируются достаточно безопасно для криптографического режима работы? Используется ли небезопасный режим работы, например ECB? Используется ли шифрование, когда более подходящим является аутентифицированное шифрование?
* Используются ли пароли в качестве криптографических ключей при отсутствии функции выведения ключа на основе пароля?
* Используется ли для криптографических целей случайность, которая не была разработана для удовлетворения криптографических требований? Даже если выбрана правильная функция, должна ли она быть засеяна разработчиком, а если нет, то не переписал ли разработчик встроенную в нее функцию сильного засева с засевом, который не обладает достаточной энтропией/непредсказуемостью?
* Используются ли устаревшие хэш-функции, такие как MD5 или SHA1, или некриптографические хэш-функции используются, когда необходимы криптографические хэш-функции?
* Используются ли устаревшие методы криптографической подшивки, такие как PKCS number 1 v1.5?
* Можно ли использовать криптографические сообщения об ошибках или информацию о побочных каналах, например, в виде атак на оракула подшивки?

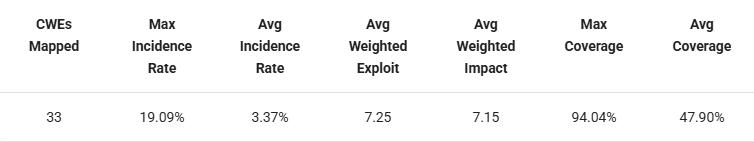
Для предотвращения сбоев, связанных с криптографическим шифрованием, необходимо, как минимум, выполнить следующий набор действий:

* Классифицируйте данные, обрабатываемые, хранимые или передаваемые приложением. Определите, какие данные являются конфиденциальными в соответствии с законами о конфиденциальности, нормативными требованиями или потребностями бизнеса.
* Не храните конфиденциальные данные без необходимости. Выбрасывайте их как можно скорее или используйте токенизацию или даже усечение, соответствующее требованиям PCI DSS. Данные, которые не хранятся, невозможно украсть.
* Обязательно шифруйте все конфиденциальные данные в состоянии покоя.
* Убедитесь в наличии современных и надежных стандартных алгоритмов, протоколов и ключей; используйте правильное управление ключами.
* Шифруйте все данные при передаче с помощью безопасных протоколов, таких как TLS с шифрами forward secrecy (FS), приоритет шифров на сервере и безопасные параметры. Обеспечьте шифрование с помощью таких директив, как HTTP Strict Transport Security (HSTS).
* Отключите кэширование для ответов, содержащих конфиденциальные данные.
* Применяйте необходимые средства контроля безопасности в соответствии с классификацией данных.
* Не используйте устаревшие протоколы, такие как FTP и SMTP, для передачи конфиденциальных данных.
* Храните пароли с помощью сильных адаптивных и соленых функций хеширования с рабочим фактором (фактором задержки), таких как Argon2, scrypt, bcrypt или PBKDF2.
* Векторы инициализации должны быть выбраны в соответствии с режимом работы. Для многих режимов это означает использование CSPRNG (криптографически защищенного генератора псевдослучайных чисел). Для режимов, требующих использования nonce, вектор инициализации (IV) не нуждается в CSPRNG. В любом случае IV никогда не должен использоваться дважды для фиксированного ключа.
* Всегда используйте аутентифицированное шифрование, а не просто шифрование.
* Ключи должны генерироваться криптографически случайным образом и храниться в памяти в виде массивов байтов. Если используется пароль, то он должен быть преобразован в ключ с помощью соответствующей функции получения ключа на основе пароля.
* Убедитесь, что криптографическая случайность используется там, где это необходимо, и что она не была заложена предсказуемым образом или с низкой энтропией. Большинство современных API не требуют от разработчика засевать CSPRNG для обеспечения безопасности.
* Избегайте устаревших криптографических функций и схем набивки, таких как MD5, SHA1, PKCS number 1 v1.5.
* Самостоятельно проверяйте эффективность конфигурации и настроек.

Примерные сценарии атак:

* Приложение шифрует номера кредитных карт в базе данных с помощью автоматического шифрования базы данных. Однако при извлечении эти данные автоматически расшифровываются, что позволяет с помощью SQL-инъекции получить номера кредитных карт открытым текстом.
* База данных паролей использует несоленые или простые хэши для хранения паролей каждого пользователя. Дефект загрузки файла позволяет злоумышленнику получить базу данных паролей. Все несоленые хэши могут быть раскрыты с помощью радужной таблицы предварительно вычисленных хэшей. Хэши, сгенерированные простыми или быстрыми хэш-функциями, могут быть взломаны графическими процессорами, даже если они были солеными.

**2.4.3 A03:2021 – Injection**



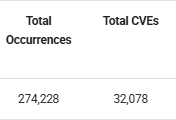


Рисунок 5. Факторы «Инъекции»

Те или иные виды инъекции. Среди распространенных слабых мест можно отметить CWE-79: Межсайтовый скриптинг, CWE-89: SQL Injection, и CWE-73: Внешний контроль имени или пути файла.

Приложение уязвимо для атак, когда:

* Данные, предоставляемые пользователем, не проверяются, не фильтруются и не дезинфицируются приложением.
* Динамические запросы или непараметризованные вызовы без контекстно-зависимого экранирования используются непосредственно в интерпретаторе.
* Вражеские данные используются в параметрах поиска объектно-реляционного отображения (ORM) для извлечения дополнительных конфиденциальных записей.
* Вражеские данные используются напрямую или конкатенируются. SQL или команда содержит структуру и вредоносные данные в динамических запросах, командах или хранимых процедурах.
* Среди наиболее распространенных инъекций - SQL, NoSQL, команды ОС, объектно-реляционное отображение (ORM), LDAP, а также инъекции языка выражений (EL) или библиотеки навигации по объектам (OGNL). Концепция идентична для всех интерпретаторов. Анализ исходного кода - лучший метод выявления уязвимости приложений к инъекциям. Настоятельно рекомендуется автоматическое тестирование всех параметров, заголовков, URL, cookies, JSON, SOAP и XML данных. Организации могут включить инструменты статического (SAST), динамического (DAST) и интерактивного (IAST) тестирования безопасности приложений в конвейер CI/CD, чтобы выявлять внедренные ошибки инъекций до развертывания на производстве.

Для предотвращения инъекций необходимо хранить данные отдельно от команд и запросов:

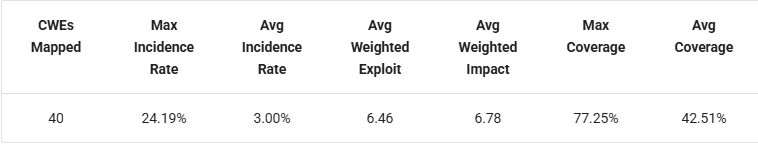
* Предпочтительным вариантом является использование безопасного API, который полностью исключает использование интерпретатора, предоставляет параметризованный интерфейс или переходит на средства объектно-реляционного отображения (ORM).
* Используйте положительную проверку ввода на стороне сервера. Это не полная защита, поскольку многие приложения требуют специальных символов, например текстовые области или API для мобильных приложений.
* Для любых остаточных динамических запросов экранируйте специальные символы, используя специальный синтаксис экранирования для данного интерпретатора.
* Примечание: структуры SQL, такие как имена таблиц, столбцов и т. д., не могут быть экранированы, поэтому имена структур, введенные пользователем, опасны. Это распространенная проблема в программах для написания отчетов.
* Используйте LIMIT и другие элементы управления SQL в запросах, чтобы предотвратить массовое раскрытие записей в случае SQL-инъекции.

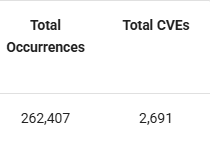
Примерные сценарии атак:

* Приложение использует недоверенные данные при построении следующего уязвимого SQL-запрос.
* Аналогично, слепое доверие приложения к фреймворкам может привести к тому, что запросы все равно будут уязвимы (например, Hibernate Query Language (HQL)).

В обоих случаях злоумышленник изменяет значение параметра 'id' в браузере, чтобы отправить.

**2.4.4 A04:2021 – Insecure Design**

****

  
Рисунок 6. Факторы «Небезопасного дизайна»

Данная категория посвящена рискам, связанным с недостатками дизайна и архитектуры, и призывает к более активному использованию моделирования угроз, безопасных паттернов проектирования и эталонных архитектур.

К числу заметных перечислений общих слабых мест (Common Weakness Enumerations, CWE) относится CWE-209: Генерация сообщения об ошибке, содержащего конфиденциальную информацию, CWE-256: незащищенное хранение учетных данных, CWE-501: нарушение границ доверия и CWE-522: недостаточно защищенные учетные данные.

Небезопасная(-ый) конструкция (дизайн) - это широкая категория, включающая различные недостатки, выраженные как «отсутствие или неэффективность конструкции контроля». Одним из факторов, способствующих появлению небезопасного дизайна, является отсутствие профилирования бизнес-рисков, присущих разрабатываемому программному обеспечению или системе, и, следовательно, неспособность определить, какой уровень безопасности необходим.

Набор советов, которые могут пригодиться при создании дизайна:

* Соберите и согласуйте с бизнесом бизнес-требования к приложению, включая требования к защите, касающиеся конфиденциальности, целостности, доступности и аутентичности всех активов данных и ожидаемой бизнес-логики.
* Примите во внимание, насколько открыто будет ваше приложение и нужно ли вам разделение арендаторов (в дополнение к контролю доступа).
* Составьте технические требования, включая функциональные и нефункциональные требования безопасности.
* Спланируйте и согласуйте бюджет на все работы по проектированию, созданию, тестированию и эксплуатации, включая мероприятия по обеспечению безопасности.

Безопасное программное обеспечение требует безопасного жизненного цикла разработки, определенного шаблона безопасного проектирования, методологии «проложенной дороги», библиотеки защищенных компонентов, инструментария и моделирования угроз. Для достижения этих требований необходимо обращаться к специалистам по безопасности не в начале проекта по разработке ПО, а на протяжении всего проекта и его сопровождения.

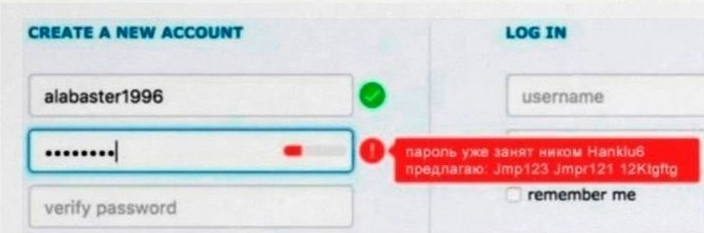


Рисунок 7. Пример генерации сообщения об ошибке, содержащего конфиденциальную информацию

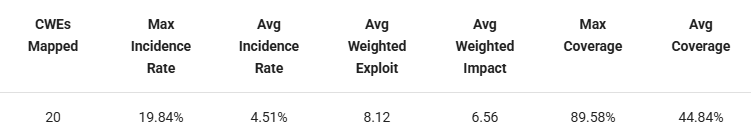
Способы предотвращения:

* Создайте и используйте жизненный цикл безопасной разработки с привлечением специалистов AppSec, которые помогут оценить и спроектировать средства контроля безопасности и конфиденциальности.
* Создайте и используйте библиотеку шаблонов безопасного проектирования или готовых к использованию компонентов.
* Используйте моделирование угроз для критически важной аутентификации, контроля доступа, бизнес-логики и ключевых потоков.
* Интегрируйте формулировки безопасности и элементы управления в пользовательские истории
* Интегрируйте проверки правдоподобности на каждом уровне приложения (от фронтенда до бэкенда).
* Напишите модульные и интеграционные тесты, чтобы убедиться, что все критические потоки устойчивы к модели угроз. Составьте сценарии использования и сценарии неправильного использования для каждого уровня приложения.
* Разделите уровни ярусов на системном и сетевом уровнях в зависимости от потребностей в воздействии и защите.
* Надежное разделение арендаторов на всех уровнях.
* Ограничьте потребление ресурсов пользователем или сервисом

Примерные сценарии атак:

* Рабочий процесс восстановления учетных данных может включать «вопросы и ответы», что запрещено стандартами. Вопросам и ответам нельзя доверять как доказательству личности, поскольку ответы могут быть известны более чем одному человеку, поэтому они запрещены.
* Сеть кинотеатров предоставляет скидки при групповом бронировании и требует внести предоплату не более чем за пятнадцать человек. Злоумышленники могли бы смоделировать этот поток угроз и проверить, смогут ли они забронировать шестьсот мест и все кинотеатры сразу за несколько запросов, что привело бы к огромной потере дохода.

**2.4.5 A05:2021 – Security Misconfiguration**



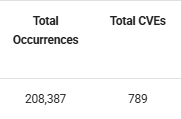


Рисунок 8. Факторы «Неправильной конфигурации системы безопасности»

Приложение может быть уязвимым, если в нем:

* Отсутствует соответствующее усиление безопасности в любой части стека приложений или неправильно настроены разрешения на облачные сервисы.
* Включены или установлены ненужные функции (например, ненужные порты, службы, страницы, учетные записи или привилегии).
* Учетные записи по умолчанию и их пароли по-прежнему включены и не изменены.
* При обработке ошибок пользователям показываются трассировки стека или другие чрезмерно информативные сообщения об ошибках.
* В обновленных системах новейшие функции безопасности отключены или не настроены безопасно.
* Настройки безопасности в серверах приложений, прикладных фреймворках (например, Struts, Spring, ASP.NET), библиотеках, базах данных и т. д. не установлены на безопасные значения.
* Сервер не отправляет заголовки или директивы безопасности, либо они не установлены в безопасные значения.
* Программное обеспечение устарело или уязвимо.
* Без согласованного, повторяющегося процесса настройки безопасности приложений системы подвергаются повышенному риску.

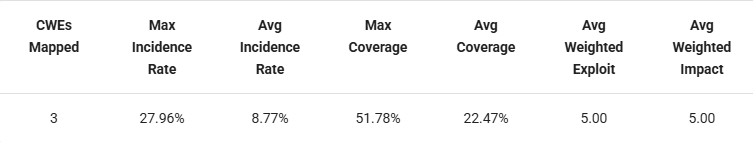
Для предотвращения возникновений неправильной конфигурации системы безопасности необходимо внедрить процессы безопасной установки, в том числе:

* Повторяющийся процесс усиления позволяет быстро и легко развернуть другую среду, которая должным образом заблокирована. Среды разработки, QA и производства должны быть настроены одинаково, при этом в каждой среде должны использоваться разные учетные данные. Этот процесс должен быть автоматизирован, чтобы минимизировать усилия, необходимые для создания новой безопасной среды.
* Минимальная платформа без лишних функций, компонентов, документации и образцов. Удалите или не устанавливайте неиспользуемые функции и фреймворки.
* Задача по проверке и обновлению конфигураций, соответствующих всем заметкам по безопасности, обновлениям и патчам, как часть процесса управления патчами (см. A06:2021-Уязвимые и устаревшие компоненты). Проверьте разрешения на облачные хранилища (например, разрешения на ведро S3).
* Сегментированная архитектура приложений обеспечивает эффективное и безопасное разделение компонентов или арендаторов с помощью сегментации, контейнеризации или групп безопасности облака (ACL).
* Отправка клиентам директив безопасности, например, заголовков безопасности.
* Автоматизированный процесс проверки эффективности конфигураций и настроек во всех средах.

Примерные сценарии атак:

* Сервер приложений поставляется с образцами приложений, не удаленными с рабочего сервера. Эти примеры приложений имеют известные недостатки безопасности, которые злоумышленники используют для компрометации сервера. Предположим, одно из этих приложений - консоль администратора, и учетные записи по умолчанию не были изменены. В этом случае злоумышленник войдет в систему с паролем по умолчанию и возьмет управление на себя.
* На сервере не отключен просмотр каталогов. Злоумышленник обнаруживает, что может просто перечислить каталоги. Злоумышленник находит и скачивает скомпилированные классы Java, которые декомпилирует и проводит обратное проектирование, чтобы просмотреть код. Затем злоумышленник обнаруживает серьезный недостаток в управлении доступом в приложении.
* Конфигурация сервера приложений позволяет возвращать пользователям подробные сообщения об ошибках, например, трассировку стека. Это потенциально может раскрыть конфиденциальную информацию или скрытые недостатки, такие как версии компонентов, которые, как известно, являются уязвимыми.

**2.4.6 A06:2021 – Vulnerable and Outdated Components**

****

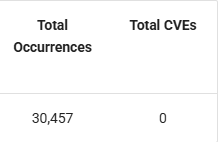
****

Рисунок 9. Факторы «Уязвимых и устаревших компонентов»

Большая вероятность существования уязвимостей:

* При незнании версии всех используемых компонентов (как клиентских, так и серверных). Это касается как компонентов, которые используются напрямую, так и вложенных зависимостей.
* При уязвимом ПО, если оно не поддерживается или устарело.
* При отсутствии регулярных сканирований на наличие уязвимостей и подписи на бюллетени безопасности, касающейся используемых компонентов.
* При отсутствии исправлений и обновлений базовой платформы, фреймворков и зависимостей своевременно и с учетом рисков.
* При отсутствии проверки совместимости обновленных, модернизированных или исправленных библиотек разработчиками программного обеспечения.
* Если вы не защищаете конфигурации компонентов.

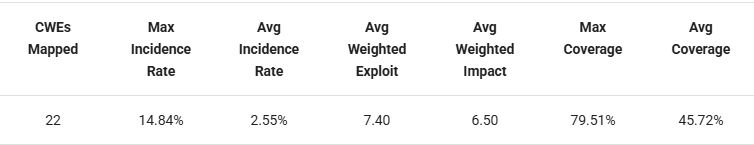
Для предотвращения необходимо существование процесса управления исправлениями, чтобы обеспечивать:

* Удаление неиспользуемых зависимостей, ненужных функций, компонентов, файлов и документации.
* Постоянную инвентаризацию версий клиентских и серверных компонентов (например, фреймворков, библиотек) и их зависимостей с помощью таких инструментов, как versions, OWASP Dependency Check, retire.js и т. д. Постоянный мониторинг таких источников, как Common Vulnerability and Exposures (CVE) и National Vulnerability Database (NVD), на предмет наличия уязвимостей в компонентах. Используйте инструменты анализа состава программного обеспечения для автоматизации этого процесса. Подпишитесь на уведомления по электронной почте об уязвимостях безопасности, связанных с используемыми компонентами.
* Получение компонентов только из официальных источников по защищенным ссылкам. Предпочитайте подписанные пакеты, чтобы снизить вероятность включения модифицированного вредоносного компонента (см. A08:2021 - Нарушения целостности программного обеспечения и данных).
* Слежку за библиотеками и компонентами, которые не поддерживаются или не создают исправлений безопасности для старых версий. Если установка патча невозможна, рассмотрите возможность развертывания виртуального патча для мониторинга, обнаружения или защиты от обнаруженной проблемы.
* Постоянный план мониторинга, устранения и применения обновлений или изменений конфигурации на протяжении всего срока службы приложения или портфеля.

Примерные сценарии атак:

* CVE-2017-5638, уязвимость удаленного выполнения кода в Struts 2, позволяющая выполнить произвольный код на сервере.
* Существуют автоматизированные инструменты, помогающие злоумышленникам находить непропатченные или неправильно сконфигурированные системы. Например, поисковая система Shodan IoT поможет вам найти устройства, которые все еще страдают от уязвимости Heartbleed, исправленной в апреле 2014 года.

**2.4.7 A07:2021 – Vulnerable and Outdated Components**

****

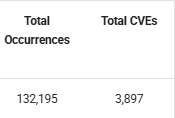
****

Рисунок 10. Факторы «Сбоев идентификации и аутентификации»

Сбои идентификации и аутентификации включают в себя перечисления общих слабых мест (Common Weakness Enumerations, CWE), связанных с нарушениями идентификации и аутентификации. Среди них можно отметить CWE-297: Неправильная проверка сертификата с несоответствием хоста, CWE-287: Неправильная аутентификация, и CWE-384: Фиксация сеанса.

Слабые места в аутентификации могут быть обнаружены, если приложение:

* Допускает автоматические атаки, такие как набивка учетных данных, когда у злоумышленника есть список действительных имен и паролей.
* Разрешает грубую силу или другие автоматические атаки.
* Разрешает использовать стандартные, слабые или хорошо известные пароли, такие как «Password1» или «admin/admin».
* Использует слабые или неэффективные процессы восстановления учетных данных и забытых паролей, такие как «ответы на основе знаний», которые невозможно сделать безопасными.
* Используются хранилища данных с открытым текстом, зашифрованными или слабо хэшированными паролями (см. A02:2021 - Криптографические сбои).
* Отсутствует или неэффективна многофакторная аутентификация.
* Раскрывает идентификатор сеанса в URL.
* Повторное использование идентификатора сеанса после успешного входа в систему.
* Неправильно аннулирует идентификаторы сеансов. Пользовательские сессии или маркеры аутентификации (в основном маркеры единого входа (SSO)) не аннулируются должным образом при выходе из системы или в период бездействия.

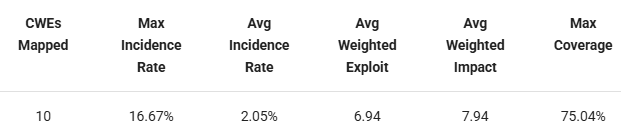
Способы предотвращения сбоев данной категории:

* По возможности внедряйте многофакторную аутентификацию для предотвращения атак с автоматическим набиванием учетных данных, грубой силой и повторным использованием украденных учетных данных.
* Не поставляйте и не развертывайте системы с учетными данными по умолчанию, особенно для пользователей-администраторов.
* Внедряйте проверку слабых паролей, например, проверяйте новые или измененные пароли по списку 10 000 худших паролей.
* Согласуйте длину, сложность и ротацию паролей с рекомендациями Национального института стандартов и технологий (NIST) 800-63b в разделе 5.1.1 для запоминаемых секретов или другими современными политиками паролей, основанными на фактах.
* Обеспечьте защиту путей регистрации, восстановления учетных данных и API от атак перечисления учетных записей, используя одинаковые сообщения для всех результатов.
* Ограничьте или все чаще задерживайте неудачные попытки входа в систему, но будьте осторожны, чтобы не создать сценарий отказа в обслуживании. Регистрируйте все неудачи и предупреждайте администраторов при обнаружении атак набивания учетных данных, грубой силы или других атак.
* Используйте встроенный в сервер безопасный менеджер сеансов, который генерирует новый случайный идентификатор сеанса с высокой энтропией после входа в систему. Идентификатор сессии не должен содержаться в URL, должен надежно храниться и аннулироваться после выхода из системы, простоя и абсолютного таймаута.

Примерные сценарии атак:

* Набивка учетных данных - использование списков известных паролей - является распространенной атакой. Предположим, что в приложении не реализована автоматическая защита от угроз или вброса учетных данных. В этом случае приложение можно использовать в качестве оракула паролей, чтобы определить, являются ли учетные данные действительными.
* Большинство атак на аутентификацию происходит из-за постоянного использования паролей в качестве единственного фактора. Когда-то считавшиеся лучшими практиками требования к ротации и сложности паролей побуждали пользователей использовать слабые пароли повторно.
* Неправильно установлены тайм-ауты сеансов приложений. Пользователь использует общедоступный компьютер для доступа к приложению. Вместо того чтобы выбрать «выход», пользователь просто закрывает вкладку браузера и уходит. Злоумышленник использует тот же браузер час спустя, а пользователь все еще аутентифицирован.

**2.4.8 A08:2021 – Software and Data Integrity Failures**



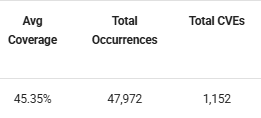


Рисунок 11. Факторы «Сбоев в работе программного обеспечения и целостности данных»

Данная категория посвящена принятию предположений, связанных с обновлениями программного обеспечения, критическими данными и конвейерами CI/CD без проверки целостности. Нарушения целостности программного обеспечения и данных связаны с кодом и инфраструктурой, которые не защищают от нарушений целостности.

Среди перечислений общих слабых мест (CWE) можно отметить CWE-829: включение функциональности из недоверенной сферы управления, CWE-494: Загрузка кода без проверки целостности, и CWE-502: Десериализация недоверенных данных.

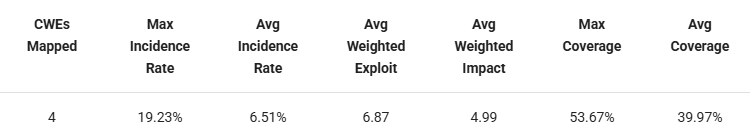
Способы предотвращения сбоев:

* Используйте цифровые подписи или аналогичные механизмы для проверки того, что программное обеспечение или данные получены из предполагаемого источника и не были изменены.
* Убедитесь, что библиотеки и зависимости, такие как npm или Maven, используют надежные репозитории. Если у вас более высокий профиль риска, рассмотрите возможность размещения внутреннего репозитория с известной репутацией, прошедшего проверку.
* Убедитесь, что инструмент безопасности цепочки поставок программного обеспечения, такой как OWASP Dependency Check или OWASP CycloneDX, используется для проверки того, что компоненты не содержат известных уязвимостей.
* Убедитесь, что существует процесс проверки изменений кода и конфигурации, чтобы свести к минимуму вероятность внедрения вредоносного кода или конфигурации в ваш программный конвейер.
* Убедитесь, что ваш конвейер CI/CD имеет надлежащую сегрегацию, конфигурацию и контроль доступа для обеспечения целостности кода, проходящего через процессы сборки и развертывания.
* Убедитесь, что неподписанные или незашифрованные сериализованные данные не отправляются ненадежным клиентам без какой-либо проверки целостности или цифровой подписи для обнаружения фальсификации или воспроизведения сериализованных данных.

Примерные сценарии атак:

* Обновление без подписи: Многие домашние маршрутизаторы, телевизионные приставки, прошивки устройств и другие устройства не проверяют обновления с помощью подписанной прошивки. Неподписанное микропрограммное обеспечение становится все большей мишенью для злоумышленников, и ожидается, что ситуация будет только ухудшаться.
* Вредоносное обновление SolarWinds: Известно, что государственные структуры атакуют механизмы обновления, а недавней известной атакой стала атака на SolarWinds Orion. Компания-разработчик программного обеспечения имела безопасные процессы сборки и целостности обновлений. Тем не менее, их удалось нарушить, и в течение нескольких месяцев компания распространяла вредоносное обновление, направленное на более чем 18 000 организаций, из которых пострадали около 100. Это одно из самых масштабных и значительных нарушений такого рода в истории.

**2.4.9 A09:2021 – Security Logging and Monitoring Failures**

****

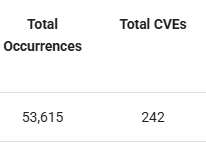
****

Рисунок 12.Факторы «Журналирования и мониторинга сбоев в системе безопасности»

Эта категория призвана помочь обнаружить, эскалировать и реагировать на активные нарушения. Без ведения журнала и мониторинга невозможно обнаружить брешь.

Недостаточное протоколирование, обнаружение, мониторинг и активное реагирование происходят в любое время:

* События, подлежащие аудиту, такие как вход в систему, неудачный вход в систему и дорогостоящие транзакции, не регистрируются.
* Предупреждения и ошибки не генерируют никаких, неадекватных или неясных сообщений в журнале.
* Журналы приложений и API не отслеживаются на предмет подозрительной активности.
* Журналы хранятся только локально.
* Соответствующие пороги оповещения и процессы эскалации реакции не установлены или неэффективны.
* Тестирование на проникновение и сканирование с помощью инструментов динамического тестирования безопасности приложений (DAST) (например, OWASP ZAP) не вызывают оповещений.
* Приложение не может обнаружить, повысить или предупредить об активных атаках в реальном или близком к реальному времени.

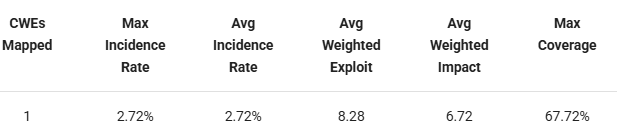
Для предотвращения инцидентов из данной категории разработчики должны внедрить некоторые или все следующие средства контроля, в зависимости от степени риска приложения:

* Убедитесь, что все сбои при входе в систему, контроле доступа и проверке ввода на стороне сервера могут быть зарегистрированы с достаточным пользовательским контекстом для выявления подозрительных или вредоносных учетных записей и сохранены в течение времени, достаточного для отложенного криминалистического анализа.
* Убедитесь, что журналы генерируются в формате, который легко воспринимается решениями по управлению журналами.
* Убедитесь, что данные журнала правильно закодированы, чтобы предотвратить инъекции или атаки на системы регистрации и мониторинга.
* Убедитесь, что высокоценные транзакции имеют аудиторский след с контролем целостности для предотвращения подделки или удаления, например, таблицы базы данных только с приложением или другие подобные системы.
* Команды DevSecOps должны организовать эффективный мониторинг и оповещение, чтобы подозрительная деятельность обнаруживалась и быстро реагировалась.
* Разработайте или утвердите план реагирования на инциденты и восстановления, например, по стандарту Национального института стандартов и технологий (NIST) 800-61r2 или более позднему.

Примерные сценарии атак:

* Оператор веб-сайта поставщика услуг детского медицинского страхования не смог обнаружить взлом из-за отсутствия мониторинга и регистрации. Внешняя сторона сообщила поставщику медицинских услуг, что злоумышленник получил доступ и изменил тысячи конфиденциальных медицинских записей более чем 3,5 миллиона детей. Проверка, проведенная после инцидента, показала, что разработчики веб-сайта не устранили существенные уязвимости.
* В крупной индийской авиакомпании произошла утечка данных, в результате которой пострадали личные данные миллионов пассажиров за более чем десять лет, включая паспортные данные и данные кредитных карт. Утечка данных произошла у стороннего провайдера облачного хостинга, который через некоторое время уведомил авиакомпанию о нарушении.

**2.4.10 A10:2021 – Server-Side Request Forgery (SSRF)**



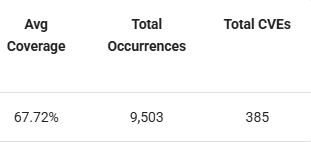


Рисунок 13. Факторы «Подделки запросов со стороны сервера (SSRF)»

Дефект SSRF возникает, когда веб-приложение получает удаленный ресурс без проверки URL, предоставленного пользователем. Это позволяет злоумышленнику заставить приложение отправить поддельный запрос в неожиданное место назначения, даже если оно защищено брандмауэром, VPN или другим типом списка контроля доступа к сети (ACL).

Поскольку современные веб-приложения предоставляют конечным пользователям удобные функции, получение URL-адреса становится распространенным сценарием.

Разработчики могут предотвратить SSRF, внедрив некоторые или все следующие средства глубокой защиты:

* Сетевой уровень: разделите функции удаленного доступа к ресурсам на отдельные сети, чтобы уменьшить влияние SSRF,а также применяйте политики брандмауэра «запрет по умолчанию» или правила управления сетевым доступом, чтобы блокировать весь трафик интрасети, кроме самого необходимого.
* На уровне приложений: проводите дезинфекция и проверку всех входных данных, предоставляемых клиентом, обеспечьте соблюдение схемы URL, порта и назначения с помощью положительного разрешительного списка, не отправляйте необработанные ответы клиентам, отключите HTTP-перенаправления, следите за согласованностью URL, чтобы избежать таких атак, как перепривязка DNS.
* Дополнительные меры: не развертывайте другие службы, имеющие отношение к безопасности, на фронтальных системах; контролируйте локальный трафик на этих системах; для фронтендов с выделенными и управляемыми группами пользователей используйте сетевое шифрование (например, VPN) на независимых системах, чтобы учесть очень высокие требования к защите.

Примеры сценариев атак:

* Сканирование портов внутренних серверов - Если сетевая архитектура не сегментирована, злоумышленники могут составить карту внутренних сетей и определить, открыты или закрыты порты на внутренних серверах, по результатам соединений или времени, затраченному на подключение или отклонение соединений с полезной нагрузкой SSRF.
* раскрытие важных данных - злоумышленники могут получить доступ к локальным файлам или внутренним службам, чтобы получить важную информацию.

2.5 OWASP Application Security Verification Standard (ASVS)

Проект стандарта проверки безопасности приложений OWASP (ASVS) обеспечивает основу для тестирования технических средств контроля безопасности веб-приложений, а также предоставляет разработчикам список требований для безопасной разработки.

Основная цель *проекта OWASP Application Security Verification Standard (ASVS)* — нормализовать диапазон охвата и уровень строгости, доступные на рынке, когда речь идет о выполнении проверки безопасности веб-приложений с использованием коммерчески работоспособного открытого стандарта. Стандарт обеспечивает основу для тестирования технических средств безопасности приложений, а также любых технических средств безопасности в среде, которые используются для защиты от уязвимостей, таких как межсайтовый скриптинг (XSS) и SQL-инъекция. Этот стандарт может использоваться для установления уровня уверенности в безопасности веб-приложений. Требования были разработаны с учетом следующих целей:

*Использование в качестве метрики* — предоставление разработчикам и владельцам приложений критерия, с помощью которого можно оценить степень доверия, которое можно оказать их веб-приложениям.

*Использовать в качестве руководства* — предоставить разработчикам средств контроля безопасности рекомендации относительно того, что следует встраивать в средства контроля безопасности, чтобы удовлетворить требования безопасности приложений, а также

*Использование при закупках* — предоставление основы для указания требований проверки безопасности приложений в контрактах.

2.6 OWASP SAMM

Software Assurance Maturity Model или Модель зрелости обеспечения безопасности программного обеспечения предоставляет эффективный и измеримый способ анализа и улучшения жизненного цикла безопасной разработки. SAMM поддерживает полный жизненный цикл программного обеспечения и не зависит от технологий и процессов.

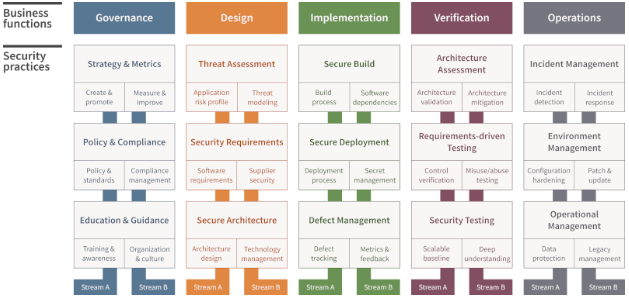


Рисунок 14. Модель SAMM

2.7 OWASP Web Security Testing Guide

OWASP Web Security Testing Guide или Руководство OWASP по тестированию веб-безопасности представляет собой комплексное руководство по тестированию безопасности веб-приложений и веб-сервисов.

2.8 The OWASP Gen AI Security Project

The OWASP Gen AI Security Project или Проект безопасности Gen AI OWASP помогает организациям и специалистам ориентироваться в быстро меняющемся ландшафте генеративного ИИ. Благодаря практическим ресурсам, стратегиям риска и глобальному сотрудничеству он обеспечивает уверенные инновации, одновременно снижая угрозы в LLM, агентах ИИ и других генеративных технологиях ИИ, способствуя доверию, соблюдению требований и устойчивости.

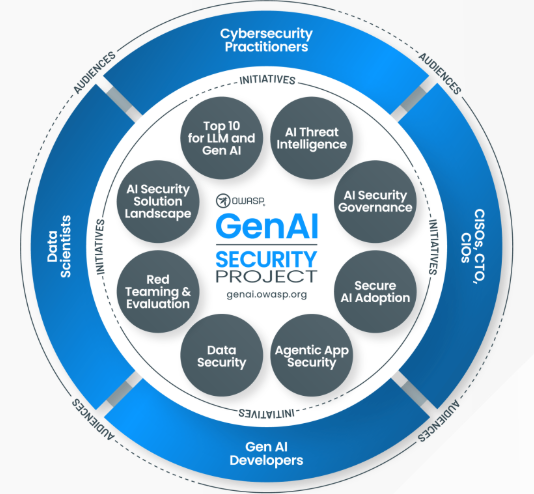


Рисунок 15. The OWASP Gen AI Security Project

2.9 About the OWASP Foundation

The Open Worldwide Application Security Project (OWASP) — некоммерческий фонд, который работает над повышением безопасности программного обеспечения. Его программирование включает:

* Проекты с открытым исходным кодом, возглавляемые сообществом, включая код, документацию и стандарты
* Более 250 местных отделений по всему миру
* Десятки тысяч членов
* Ведущие образовательные и обучающие конференции в отрасли

Будучи крупнейшей в мире некоммерческой организацией, занимающейся безопасностью программного обеспечения, OWASP:

* Поддерживает создание эффективных проектов ;
* Развивает и поддерживает сообщества посредством мероприятий и встреч отделений по всему миру; а также
* Предоставляет образовательные публикации и ресурсы

чтобы дать возможность разработчикам писать лучшее программное обеспечение, а специалистам по безопасности — сделать программное обеспечение во всем мире более безопасным.

2.10 OWASP CRS

OWASP CRS - это набор общих правил обнаружения атак для использования с ModSecurity или совместимыми брандмауэрами веб-приложений. Он направлен на защиту веб-приложений от широкого спектра атак с минимальным количеством ложных срабатываний. CRS обеспечивает защиту от многих распространенных категорий атак, включая SQL-инъекцию, межсайтовый скриптинг, включение локальных файлов и т. д.

OWASP CRS можно использовать бесплатно. Он лицензирован в соответствии с Apache Software License версии 2 (ASLv2).

2.11 OWASP Mobile Application Security

Флагманский проект OWASP Mobile Application Security (MAS) предоставляет стандарт безопасности для мобильных приложений (OWASP MASVS) и комплексное руководство по тестированию (OWASP MASTG), которое охватывает процессы, методы и инструменты, используемые во время тестирования безопасности мобильных приложений, а также исчерпывающий набор тестовых случаев, позволяющий тестировщикам предоставлять последовательные и полные результаты.

Контрольный список безопасности мобильных приложений OWASP содержит ссылки на тестовые случаи MASTG для каждого элемента управления MASVS.

* Оценка безопасности/тесты на проникновение : убедитесь, что вы покрываете как минимум стандартную поверхность атаки, и начните исследование.
* Соответствие стандартам : включает версии MASVS и MASTG, а также идентификаторы коммитов.
* Изучайте и практикуйте свои навыки обеспечения безопасности мобильных устройств.
* Награды за обнаружение ошибок : шаг за шагом изучите поверхность мобильных атак.

# ГЛАВА 3. Разбор инцидента на основании отчета аналитического центра

* 1. Выбор инцидента и сбор сведений

В качестве источника инцидентов используется «краткий обзор основных инцидентов промышленной кибербезопасности за первый квартал 2024 года», предоставляемый в открытом доступе Kaspersky ICS CERT.

Выбранный инцидент - Кибератака на BerlinerLuft .

Согласно этому же отчету немецкая компания BerlinerLuft, занимающаяся разработкой и производством заводского оборудования, 16 марта стала жертвой кибератаки. Согласно сообщению на сайте компании, она стала недоступна для связи по телефону и электронной почте. Команда компании приложила все усилия, чтобы как можно быстрее восстановить доступность. Компания предупредила, что в связи с экстренной ситуацией в ее ИТ-инфраструктуре возможны кратковременные ограничения при проведении деловых операций, а также сбои в технологических процессах и задержки поставок.

Вот сообщение на официальном сайте компании о данном инциденте:

«Сразу после атаки наши ИТ-системы были отключены от Интернета, чтобы предотвратить дальнейшее распространение. В настоящее время мы полностью перепроектировали всю нашу ИТ-инфраструктуру в специально защищенной зоне. Все наши сотрудники занимаются сбросом настроек и полной переустановкой устройств.

Наши электронные письма отправляются с использованием усиленных мер безопасности через Microsoft 365. Кроме того, все входящие и исходящие электронные письма многократно сканируются антивирусными сканерами через текущий почтовый шлюз. Поэтому мы можем заверить вас, что безопасность наших общих каналов связи, особенно трафика электронной почты, теперь полностью восстановлена.

В этой связи мы хотели бы вежливо попросить вас обновить наши адреса электронной почты в ваших белых списках, чтобы вы могли продолжать получать от нас важную информацию и сообщения и не пропустить ни одного сообщения.

Более того, наша ERP-система теперь снова работает без ограничений во всех компаниях. Это позволяет нам снова принимать заказы и эффективно продолжать наши обычные процессы.

Мы будем рады ответить на любые дополнительные вопросы, предоставить информацию и предложения по улучшению безопасности нашей общей коммуникации.»

Компания уведомила об инциденте правоохранительные органы и государственные органы по защите персональных данных. 27 марта компания выпустила новое сообщение, где говорилось, что производство компонентов воздуховодов, клапанов жалюзи и звукоизоляционных перегородок на немецком и польском заводах возобновлено, а также восстановлена работа электронной почты и телефонной связи.

* 1. Определение тактик и техник

Определим тактики и их техники из матрицы MITRE ATT&CK, которые могли бы использовать злоумышленники в данном инциденте.

* Тактика Reconnaissance с использованием техник Gather Victim Identity Information: Email Addresses для сбора адресов электронной почты, Gather Victim Org Information: Business Relationships для поиска информации о поставщиках услуг.
* Тактика Initial Access с использованием техник Phishing: Spearphishing Attachment для попытки внедрения вредоносного ПО, Supply Chain Compromise с целью компрометации данных (номеров телефонов) или системы (например, системы телефонной связи), Phishing: Spearphishing Link для получения доступа к почте или внедрения вредоносного ПО.
* Тактика Execution с использованием техник User Execution: Malicious Link для получения доступа к данным системы (телефонным номерам или аккаунтам электронной почты), Scheduled Task/Job использование планирования задач для выполнения программ при запуске системы или по расписанию для сохранения.
* Тактика Persistence с использованием техник Boot or Logon Autostart Execution добавление автозагрузки вредоносного ПО для сохранения доступа после перезапуска устройства или системы.
* Тактика Discovery с использованием техник File and Directory Discovery для поиска интересующих файлов и каталогов (например, списки с номерами телефонов компании), Remote System Discovery для сканирования сети с целью поиска других систем.
* Тактика Lateral Movement с использованием техник Remote Services с целью использования удаленных сервисов и протоколов для распространения в сети, Lateral Tool Transfer для передачи вредоносных файлов между системами в о взломанной среде (копирование и пересылка файлов для быстрого заражения системы).
* Тактика Impact с использованием техник Data Encrypted for Impact для шифрования данных в систем с целью закрытия данных доступа (шифрование данных системы, чтобы ограничить доступ компании к телефонным спискам и др.), Endpoint Denial of Service для DDoS-атаки на сервисы компании, Data Manipulation (порча данных для нарушения работы предприятия)
* Тактика Credential Access с использованием техник Brute Force для подбора пароля к аккаунтам, Credentials from Password Stores: Password Managers для получения паролей.
* Тактика Collection с использованием техник Adversary-in-the-Middle, чтобы заставить устройство взаимодействовать через контролируемую злоумышленником систему для дальнейшего сбора информации или выполнении дополнительных действий (например, отключение устройства от системы).
* Тактика Resource Development с использованием техник Compromise Accounts: Email Accounts для дальнейшего проведения фишинга, Obtain Capabilities: Malware для получения образца вредоносного ПО.
* Тактика Command and Control с использованием техник Application Layer Protocol для маскировки действий в сети, Data Encoding для уменьшения вероятности обнаружения в сети.
  1. Один из возможных вариантов действий злоумышленников.

С помощью открытых источников злоумышленники нашли публичные адреса электронных почт, номера телефонов и другую информацию. Далее с помощью фишинга и (или) брутфорса они проникли в систему, куда загружают вредоносное ПО. Происходил этап закрепления в системе. Сбор информации о системе. После этого злоумышленники изучали сеть, доступ к которой они получили для поиска других систем. Следующих шагом могло быть копирование вредоносных файлов и их распространение в других системах. В конечном итоге злоумышленники модифицировали файлы для приостановки работы компании путем блокировки доступа к системам или совершали DDoS-атаку по заранее ослабленным узлам.

* 1. Принятые компанией меры
* Предупреждение клиентов
* Отключение ИТ-систем от Интернета для предотвращения дальнейшего распространения
* Перепроектирование ИТ-инфраструктуры в специально защищенной зоне
* Привлечение сотрудников к сбросу настроек и полной переустановке систем на устройствах
* Отправка электронных писем с использованием усиленных мер безопасности через Microsoft 365
* Сканирование всех входящих и исходящих писем антивирусными сканерами через текущий почтовый шлюз
* Выдача рекомендаций

# ВЫВОД

В ходе учебной практики удалось ознакомиться с тактико-технической матрицей MITRE ATT&CK, а также сравнить ее с линейной пошаговой моделью кибер-атак Cyber Kill Chain. Была проанализирована информация с сайта проекта OWASP, рассмотрены как документ, содержащий 10 наиболее критических рисков безопасности веб-приложений, так и другие стандарты, проекты и методики находящиеся в открытом доступе для повышения осведомленности в сфере кибербезопасности.

В ходе работы был проанализирован инцидент из отчета аналитического центра Kaspersky ICS CERT. Были выявлены вероятно использованные тактики и техники в соответствии с матрицей MITRE ATT&CK, рассмотрены действия, принятые компанией после инцидента, а также составлен примерный набор действий злоумышленников, приведший к инциденту.

В результате выполнения задачи мне удалось получить новые знания и практические навыки в области ИБ, а также познакомиться с открытыми источниками, содержащими полезный теоретический материал.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. ATT&CK Matrix for Enterprise // MITRE ATT&CK URL: https://attack.mitre.org/ (дата обращения: 07.04.2025).
2. OWASP // OWASP URL: https://owasp.org/ (дата обращения: 07.04.2025).
3. Cyber Kill Chain // LOCKHEED MARTIN URL: https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html (дата обращения: 07.04.2025).
4. DeepL // DeepL URL: https://www.deepl.com/ (дата обращения: 07.04.2025).
5. THE MODEL // SAMM URL: https://owaspsamm.org/model/ (дата обращения: 07.04.2025).
6. WSTG - Stable // OWASP URL: https://owasp.org/www-project-web-security-testing-guide/stable/ (дата обращения: 07.04.2025).
7. «Краткий обзор основных инцидентов промышленной кибербезопасности за первый квартал 2024 года» // Kaspersky ICS CERT URL: https://ics-cert.kaspersky.ru/publications/reports/2024/06/03/q1-2024-a-brief-overview-of-the-main-incidents-in-industrial-cybersecurity/ (дата обращения: 07.04.2025).