**Лекція 1.**

Вступ. Основні поняття безпеки програм та даних. Інформація – сукупність даних, що використовуються для передачі повідомлення між людьми та/або машинами в комунікативних (заснованих на обміні повідомленнями) або транзакційних процесах (процесах, в яких операції передбачають, наприклад, передачу грошових коштів). На будь-яку інформацію впливають три основні властивості: конфіденційність, цілісність та доступність, відомі як "Тріада CIA" (рис. 1). Рис. 1 – Тріада CIA Конфіденційність (Confidentiality). Вся інформація повинна бути захищена відповідно до ступеня секретності її змісту, з метою обмеження доступу до неї і використання тільки тими людьми, яким вона призначена. Порушення конфіденційності бувають різних типів. При повному розкритті зловмисники отримують весь набір даних, включаючи метадані. На нижньому кінці спектру може бути незначне розкриття інформації, наприклад, внутрішнє повідомлення про помилку або подібний витік, що не має реальних наслідків. Як приклад часткового розкриття можна навести практику присвоєння послідовних номерів новим клієнтам: хитрий конкурент може реєструватися як новий клієнт і час від часу отримувати новий номер клієнта, а потім обчислювати послідовні різниці, щоб дізнатися кількість клієнтів, придбаних протягом кожного інтервалу. Будь-який витік інформації про захищені дані певною мірою є порушенням конфіденційності. Цілісність (Integrity). Вся інформація повинна зберігатися в тому ж стані, в якому вона була надана її власником, з метою захисту від неналежних змін, як навмисних, так і випадкових. Цілісність у контексті інформаційної безпеки - це просто автентичність і точність даних, захищених від несанкціонованого втручання або видалення. На додаток до захисту від несанкціонованої модифікації, точний запис походження даних - першоджерела та будь-яких санкціонованих змін - може бути важливою і надійною гарантією цілісності. Одним із класичних способів захисту від багатьох атак є збереження версій критично важливих даних і запис їхнього походження. Простіше кажучи, зберігайте хороші резервні копії. Інкрементні резервні копії можуть бути чудовим засобом захисту, оскільки вони прості та ефективні у створенні і надають серію знімків, які детально показують, які саме дані були змінені і коли. Однак потреба в цілісності виходить далеко за межі захисту даних і часто включає забезпечення цілісності компонентів, журналів сервера, вихідного коду і версій програмного забезпечення та іншої криміналістичної інформації, необхідної для визначення першоджерела втручання, коли виникають проблеми. На додаток до обмеженого адміністративного контролю доступу, надійними засобами перевірки цілісності є також захищені дайджести (подібні до контрольних сум) і цифрові підписи. Майте на увазі, що втручання може відбуватися різними способами, не обов'язково шляхом модифікації даних у сховищі. Наприклад, у веб-додатку втручання може відбуватися на стороні клієнта, на лінії зв'язку між клієнтом і сервером, шляхом обману уповноваженої сторони для внесення змін, шляхом модифікації скрипту на сторінці або багатьма іншими способами. Доступність (Availability). Вся інформація, створена або отримана особою чи установою, має бути доступною для її користувачів у той час, коли вона їм потрібна для будь-яких цілей. Атаки на доступність - сумна реальність світу, підключеного до Інтернету, і від них може бути найскладніше захиститися. У найпростіших випадках зловмисник може просто надіслати на сервер надзвичайно велике навантаження трафіку, перевантажуючи його тим, що виглядає як законне використання сервісу. Цей принцип означає, що інформація стає тимчасово недоступною; хоча дані, які втрачені назавжди, також недоступні, це, як правило, не вважається порушенням цілісності. Анонімні атаки на відмову в обслуговуванні (DoS), часто з метою отримання викупу, загрожують будь-якій інтернет-службі, створюючи складну проблему. Щоб найкраще захиститися від цих атак, розміщуйте великомасштабні сервіси з інфраструктурою, яка витримує великі навантаження, і зберігайте гнучкість для швидкого переміщення інфраструктури в разі виникнення проблем. Ніхто не знає, наскільки поширеними і дорогими є DoS-атаки, оскільки багато жертв вирішують ці проблеми в приватному порядку. Але, без сумніву, вам слід заздалегідь створити детальні плани для підготовки до таких інцидентів. Для веб-сервера неправильно сформований запит, який викликає помилку, що призводить до збою або нескінченного циклу, може зруйнувати його роботу. Інші атаки також можуть перевантажити сховище, обчислювальні або комунікаційні можливості програми, або, можливо, використовувати шаблони, які порушують ефективність кешування, і всі вони створюють серйозні проблеми. Несанкціоноване знищення програмного забезпечення, конфігурації або даних також може негативно вплинути на доступність. Інформація вважається захищеною (безпечною), якщо вона володіє трьома вищезазначеними властивостями. Комп’ютерну безпеку можна визначити як забезпечення конфіденційності, цілісності та гарантій доступності для користувачів або клієнтів інформаційних систем. Аспекти інформаційної безпеки Аутентифікація. Процес ідентифікації та формального визнання ідентичності елементів, які взаємодіють або є частиною електронної транзакції, що забезпечує доступ до інформації за допомогою засобів контролю ідентифікації цих елементів. Системи зазвичай дозволяють зареєстрованим користувачам пройти автентифікацію, довівши, що вони знають пароль, пов'язаний з їхнім обліковим записом, але автентифікація може бути набагато ширшою. Облікові дані можуть бути чимось, що довірена особа знає (пароль) або має (смарт-картка), або чимось, чим вона є (біометричні дані). Процес автентифікації перевіряє ідентифікаційні дані довірителя на основі облікових даних, які демонструють, що вони дійсно є тими, за кого себе видають. Або сервіс може використовувати більш надійну форму облікових даних, таку як цифровий підпис або виклик, який доводить, що власник володіє приватним ключем, пов'язаним з ідентифікатором, і саме так браузери автентифікують веб-сервери через HTTPS. Цифровий підпис є кращою формою автентифікації, оскільки довіритель може довести, що він знає секрет, не розголошуючи його. Взагалі кажучи, процес автентифікації повинен перевіряти облікові дані і надавати відповідь "пройдено" або "не пройдено". Уникайте вказувати на частковий успіх, оскільки це може допомогти зловмиснику підібрати облікові дані методом проб і помилок. Щоб зменшити загрозу підбору методом грубої сили, поширена стратегія полягає в тому, щоб зробити автентифікацію за своєю суттю важкою для обчислень або збільшити затримку в цьому процесі. Авторизація. Надання дозволу на доступ до інформації та функціональних можливостей програми учасникам процесу обміну інформацією (користувачеві або машині) після їх правильної ідентифікації та аутентифікації. Авторизація починається після завершення процесу автентифікації. Він передбачає надання або заборону доступу до ресурсу на основі привілеїв автентифікованого користувача. Авторизація визначає, які дії дозволено виконувати автентифікованому користувачу чи об’єкту в системі чи програмі. Пам’ятайте, що автентифікація підтверджує особу користувача, тоді як авторизація визначає та забезпечує виконання дій і ресурсів, до яких користувачеві дозволено доступ у системі чи програмі. Аудит. Процес збору доказів використання наявних ресурсів з метою ідентифікації суб'єктів, що беруть участь у процесі обміну інформацією, тобто походження, призначення та засоби передачі інформації. Забезпечує надійний реєстр того, хто, що і коли робив, який підлягає регулярній перевірці на предмет порушень і притягнення до відповідальності винних осіб. Для того, щоб організація могла проводити аудит діяльності системи, система повинна створювати надійний журнал усіх подій, які є критично важливими для підтримання безпеки. До них відносяться події автентифікації та авторизації, запуск і вимкнення системи, оновлення програмного забезпечення, адміністративні доступи тощо. Журнали аудиту також повинні бути стійкими до несанкціонованого втручання, а в ідеалі навіть складними для втручання адміністраторів, щоб вважатися повністю надійними записами. У травні 2018 року Twitter розкрив прикру помилку: вони виявили, що зміна коду ненавмисно призвела до того, що паролі для входу з'явилися у внутрішніх журналах. Малоймовірно, що це призвело до зловживань, але це, безумовно, підірвало довіру клієнтів, і цього не повинно було статися. Журнали повинні фіксувати операційні деталі, але не зберігати жодної особистої інформації, щоб мінімізувати ризик її розголошення, оскільки багато членів технічного персоналу можуть регулярно переглядати журнали. Система також повинна запобігати фальсифікації журналів з метою приховування зловмисних дій. Якщо зловмисник зможе модифікувати журнали, він просто зачистить всі сліди своєї діяльності. Автентичність. Забезпечення того, що суб'єкти (інформація, машини, користувачі), ідентифіковані в процесі комунікації як відправники або автори, є саме тими, за кого себе видають, і що повідомлення або інформація не були змінені після того, як вони були відправлені або підтверджені. Вразливості. Це слабкі місця, які можуть призвести до інциденту безпеки, що негативно впливає на один або кілька принципів інформаційної безпеки: конфіденційність, цілісність і доступність. Вразливості самі по собі не спричиняють інцидентів, оскільки вони є пасивними елементами, що потребують збудника або фактора, яким є загрози. Наприклад, веб-сервер, на якому працює застаріла версія служби Apache, може містити вразливість, яка дозволить зловмиснику здійснити атаку типу «відмова в обслуговуванні» (DoS) на веб-сайти, розміщені на цьому сервері, ставлячи під загрозу їх доступність. Фахівці з кібербезпеки в організації мають можливість усунути цю вразливість, оновивши службу Apache до останньої версії, яка не піддається DoS-атаці. Приклади вразливостей: Фізичні Будівельні конструкції, які не відповідають належній практиці або чинним нормам і правилам; відсутність вогнегасників, датчиків диму та інших засобів для гасіння пожеж у приміщеннях зі стратегічними активами або інформацією; поганий контроль доступу в місцях, що містять конфіденційну або секретну інформацію тощо. Природні Середовища з електронним обладнанням поруч з місцями, схильними до стихійних лих, таких як пожежі, повені, землетруси, шторми та інші, такі як перебої в електропостачанні, скупчення пилу, вологість і температура тощо. Апаратне забезпечення Комп'ютери схильні до впливу пилу, вологості, бруду та неналежного доступу до недостатньо захищених ресурсів, а також можуть страждати від погано сконфігурованих або неправильно налаштованих компонентів, з збоїв або коливань в електромережі чи надмірного підвищення температури навколишнього середовища. Програмне забезпечення Помилки в кодуванні, встановленні або конфігурації систем і програм можуть призвести до несанкціонованого доступу, витоку інформації, втрати даних і аудиторських слідів або до недоступності активу, коли він потрібен. Носії інформації Диски, файли, звіти та роздруківки можуть бути втрачені або пошкоджені; збої в електропостачанні можуть спричинити збої в роботі обладнання, що потенційно може пошкодити логічні доріжки даних; жорсткі диски зазвичай мають певний термін служби; електромагнітне випромінювання може впливати на багато типів магнітних носіїв. Зв'язок Телефонний зв'язок вразливий до прослуховування (несанкціонованого доступу) або проблем у фізичній чи логічній інфраструктури, які перешкоджають його встановленню. Людина Недостатня підготовка або обізнаність людей, відсутність адекватної психологічної оцінки або перевірки біографічних даних, яка б виявила приховані цілі або попередні проблеми або навіть недобросовісність чи незадоволеність працівника, серед іншого, можуть призвести до неналежного обміну конфіденційної інформації, невиконання процедур безпеки, або до помилок, упущень тощо, які ставлять інформацію під загрозу. Загроза у світі кібербезпеки – це зовнішня сила, яка може використовувати вразливість. Наприклад, хакер, який хотів би провести DoSатаку на веб-сайт і знає про вразливість Apache, створює явну загрозу кібербезпеці. Хоча багато загроз є шкідливими за своєю природою, це не обов’язково так. Наприклад, землетрус також може порушити доступність веб-сайту, пошкодивши центр обробки даних, що містить веб-сервери. Очевидно, що землетруси не мають злого наміру. У більшості випадків спеціалісти з кібербезпеки не можуть зробити багато для усунення загрози. Хакери будуть зламувати, і землетруси вражатимуть, хочемо ми цього чи ні. Ризик – це поєднання загрози та відповідної вразливості. Обидва ці фактори повинні бути присутніми, перш ніж ситуація стане загрозою для безпеки організації. Наприклад, якщо хакер атакує веб-сервер організації за допомогою DoS-атаки, але сервер було виправлено так, що він не є вразливим до цієї атаки, ризику немає, оскільки навіть якщо загроза присутня (хакер), уразливості немає . Подібним чином центр обробки даних може бути вразливим до землетрусів, оскільки стіни не побудовані для того, щоб витримувати екстремальні рухи під час землетрусу, але він може бути розташований у регіоні світу, де землетруси не відбуваються. Центр обробки даних може бути вразливим до землетрусів, але загрози землетрусу в його місці практично немає, тому ризику немає. Зв’язок між ризиками, загрозами та вразливістю є важливим, і його часто представляють таким рівнянням: Ризик = Загроза × Вразливість Це не означає бути буквальним рівнянням, у яке ви б фактично вставляли значення. Замість цього він має на меті продемонструвати той факт, що ризики існують лише тоді, коли існує як загроза, так і відповідна вразливість, якою загроза може скористатися. Якщо загроза або вразливість дорівнює нулю, ризик також дорівнює нулю. Вплив це потенційна сума збитків або втрат, які можуть бути завдані, системі або даним через успішне виконання загрози. Вплив може бути фінансовим, репутаційним, операційним або будь-яким іншим негативним наслідком в результаті порушення безпеки. Коли мова йде про ризик, варто зазначити, що його уникнення або зменшення до прийнятного рівня не завжди є найкращою стратегією. Існують випадки, коли вартість впровадження заходів безпеки для уникнення або зменшення певного ризику перевищує цінність інформації, що підлягає захисту, що робить цю дію недоцільною. У таких випадках необхідно оцінити можливість зберегти ризик, жити з ним, розділити його, укласти договір страхування. Заходи безпеки також називаються засобами контролю і можуть мати наступні характеристики: Превентивні. Заходи безпеки, спрямовані на запобігання інцидентам. Як приклади можна згадати політику безпеки, робочі інструкції та процедури, кампанії з підвищення обізнаності користувачів та лекції, специфікації безпеки, обладнання для контролю доступу, інструменти для реалізації політики безпеки (брандмауер, антивірус, відповідні конфігурації маршрутизатора та операційної системи і т.д.). Детективні. Заходи безпеки, спрямовані на виявлення умов або осіб, що спричиняють загрози, з метою запобігання використанню ними вразливостей. Деякі приклади: аналіз ризиків, системи виявлення вторгнень, оповіщення про небезпеку, камери спостереження, сигналізація тощо. Коригувальні. Дії, спрямовані на корекцію технологічної та людської структури для адаптації її до умов безпеки, встановлених установою, або спрямовані на зменшення впливу: аварійні команди, резервне відновлення, план безперервності роботи, план відновлення після аварійних ситуацій. Багаторівневий захист Поглиблений захист, також відомий як багаторівнева безпека (Defense in Depth), — це комплексний підхід до кібербезпеки, який передбачає впровадження кількох рівнів захисту для захисту активів, мереж і систем організації. Ця стратегія базується на концепції, згідно з якою жодна окрема міра безпеки не може гарантувати повний захист; тому використовується серія захисних механізмів, щоб гарантувати, що навіть якщо один шар порушено, решта шарів продовжуватимуть забезпечувати захист. Глибокий захист будується на основі інтеграції різних заходів безпеки, які можуть включати: Фізична безпека: захист об’єктів і обладнання від несанкціонованого доступу або пошкодження. Контроль доступу: керування дозволами для обмеження доступу користувачів до певних ресурсів або даних. Антивірусне програмне забезпечення: виявлення, видалення та запобігання зараженню шкідливим програмним забезпеченням. Брандмауери: фільтрація мережевого трафіку для блокування або дозволу передачі даних на основі попередньо визначених правил. Системи виявлення та запобігання вторгненням (IDPS): моніторинг та аналіз мережевого трафіку для виявлення та запобігання вторгненням і зловмисним діям. Резервне копіювання та відновлення даних: забезпечення регулярного резервного копіювання даних організації та можливість їх повного відновлення у разі втрати або випадкового видалення. Шифрування: кодування конфіденційних даних для захисту від несанкціонованого доступу або крадіжки. Впровадження цих рівнів дозволяє мінімізувати ризики порушень кібербезпеки, а в разі інциденту швидко й ефективно реагувати та відновлюватися. Zero Trust Zero Trust — це сучасна система безпеки, яка бореться з загрозами, що постійно змінюються в цифровому світі. Він підкреслює ідею «ніколи не довіряти, завжди перевіряти». Цей підхід вимагає відмовитися від традиційних моделей безпеки, заснованих на периметрі, і прийняти більш комплексний, цілісний підхід до захисту своїх даних і активів. За замовчуванням заборонити довіру: припустимо, що весь мережевий трафік, як всередині організації, так і за її межами, є потенційно шкідливим. Не довіряйте жодному користувачу, пристрою чи програмі лише тому, що вони знаходяться в межах периметра мережі. Перевіряйте кожен запит: автентифікуйте та авторизуйте всі запити (навіть ті, що надходять із мережі), перш ніж надавати доступ до будь-якого ресурсу. Переконайтеся, що кожен користувач, пристрій або програма належним чином ідентифіковані, а їхній доступ до ресурсів відповідає їхнім ролям, правам і привілеям. Застосувати найменші привілеї: обмежити користувачів, програми та пристрої мінімальним рівнем доступу, необхідним для виконання їхніх функцій. Це мінімізує ризик неавторизованого доступу та зменшує потенційну поверхню атаки. Сегментуйте мережі: ізолюйте та відокремлюйте різні частини мережі, щоб обмежити потенційний вплив злому. Якщо зловмисник отримує доступ до одного сегмента, він не зможе переміщатися по мережі та отримати доступ до інших конфіденційних даних. Перевіряйте та реєструйте весь трафік: активно відстежуйте, аналізуйте та реєструйте мережевий трафік для виявлення потенційних інцидентів безпеки та проведення судових розслідувань. Це надає цінну інформацію для команд безпеки, щоб постійно покращувати рівень безпеки та виявляти ранні ознаки зловмисної діяльності. Переваги: Зменшена площа атаки: обмеження доступу до конфіденційних ресурсів і сегментація мережі ускладнюють зловмисникам скомпрометувати системи та отримати доступ до цінних даних. Покращена видимість і моніторинг. Безперервно перевіряючи та реєструючи весь трафік, служби безпеки можуть отримати безпрецедентний рівень видимості, допомагаючи їм ефективніше виявляти потенційні загрози та атаки. Покращена відповідність і управління: впровадження моделі нульової довіри зміцнює відповідність організації та управління, забезпечуючи доступ до конфіденційних даних лише авторизованим користувачам. Адаптивність: підхід «нульової довіри» можна застосовувати до широкого діапазону середовищ і адаптувати до конкретних потреб і цілей організації в сфері безпеки. Впровадивши структуру нульової довіри, можна зміцнити свою безпеку, захищатися від внутрішніх і зовнішніх загроз і підтримувати контроль над своїми критично важливими активами. Cyber Kill Chain Cyber Kill Chain — це модель, розроблена компанією Lockheed Martin, великою аерокосмічною, військовою компанією з підтримки та безпеки, щоб зрозуміти та запобігти кібервторгненням у різні мережі та системи. Він служить основою для поділу на етапи кібератаки, що полегшує фахівцям із безпеки ідентифікацію, пом’якшення та запобігання загрозам. Концепція базується на військовій моделі, де термін «ланцюг вбивств» означає низку кроків, необхідних для успішного націлювання та боротьби з супротивником. У контексті кібербезпеки модель розбиває етапи кібератаки на сім окремих фаз: Розвідка: цей початковий етап передбачає збір розвідувальних даних про ціль, що може включати дослідження загальнодоступних баз даних, сканування мережі або методи соціальної інженерії. Вепонізація: на цій стадії зловмисник створює зброю, наприклад зловмисне програмне забезпечення, вірус або експлойт, і упаковує її з механізмом доставки, який може проникнути в систему цілі. Доставка: зловмисник вибирає та використовує спосіб доставки, щоб передати зброю до цілі. Поширені методи включають вкладення електронної пошти, шкідливі URL-адреси або інфіковані оновлення програмного забезпечення. Експлуатація: це фаза, на якій зброя активується, використовуючи вразливості в системах або програмах цілі для виконання коду зловмисника. Встановлення: після успішного використання зловмисник встановлює зловмисне програмне забезпечення в систему жертви, готуючи основу для подальших атак або викрадання даних. Командування та контроль: зловмисник встановлює канал зв’язку із зараженою системою, що дозволяє йому дистанційно контролювати зловмисне програмне забезпечення та виконувати подальші дії. Дії щодо цілей: на цьому останньому етапі зловмисник досягає своєї мети, яка може включати крадіжку конфіденційних даних, компрометацію систем або порушення роботи служб. Розуміння та аналіз Cyber Kill Chain допомагає організаціям і окремим особам застосовувати більш проактивний підхід до кібербезпеки. Розпізнаючи ознаки атаки на кожному етапі, можна застосувати відповідні контрзаходи, щоб запобігти або мінімізувати шкоду від атаки. Встановлення та конфігурація Для ефективного захисту ваших систем і даних життєво важливо розуміти, як безпечно встановлювати програмне забезпечення та налаштовувати параметри, а також оцінювати наслідки та потенційні вразливості під час процесів встановлення та налаштування. Неправильна інсталяція або конфігурація програмного забезпечення може призвести до низки ризиків безпеці, включаючи неавторизований доступ, витік даних та інші шкідливі атаки. Щоб переконатися, що ваша система захищена від цих потенційних загроз, важливо дотримуватися найкращих практик щодо встановлення та налаштування програмного забезпечення: Дослідіть програмне забезпечення: перш ніж встановлювати будь-яке програмне забезпечення чи програму, дослідіть її функції безпеки та репутацію. Перевірте наявність відомих вразливостей, останніх виправлень і загальну надійність програмного забезпечення. Використовуйте офіційні джерела: Завжди завантажуйте програмне забезпечення з надійних джерел, наприклад, з офіційного веб-сайту постачальника програмного забезпечення. Уникайте використання сторонніх посилань для завантаження, оскільки вони можуть містити шкідливий код або змінене програмне забезпечення. Перевірте цілісність файлу: перевірте цілісність завантаженого програмного забезпечення, перевіривши його криптографічний хеш, який часто надається постачальником програмного забезпечення. Це гарантує, що програмне забезпечення не було змінено або пошкоджено під час процесу завантаження. Встановіть оновлення: під час процесу встановлення переконайтеся, що встановлено всі доступні оновлення та виправлення, оскільки вони можуть містити життєво важливі виправлення безпеки. Безпечні конфігурації: після інсталяції належним чином налаштуйте програмне забезпечення, дотримуючись документації постачальника або найкращих галузевих практик. Серед інших важливих параметрів безпеки це може включати налаштування параметрів, пов’язаних із автентифікацією, шифруванням і контролем доступу. Хоча конфігурації програмного забезпечення відрізнятимуться залежно від конкретної програми чи системи, що використовується, є кілька ключових аспектів, про які слід пам’ятати: Найменші привілеї: налаштуйте облікові записи користувачів і дозволи за принципом найменших привілеїв. Обмежте доступ користувачів до мінімального рівня, необхідного для виконання їхніх завдань. Політики щодо паролів: застосовуйте надійні політики щодо паролів, включаючи вимоги до складності, мінімальну довжину пароля та термін дії пароля. Шифрування: увімкніть шифрування даних, щоб захистити конфіденційну інформацію від несанкціонованого доступу. Це може включати як шифрування зберігання, так і шифрування даних під час передавання. Брандмауери та безпека мережі: налаштуйте брандмауери та інші заходи безпеки мережі, щоб обмежити несанкціонований доступ до ваших систем. Ведення журналів і аудит: налаштуйте журналювання та аудит, щоб фіксувати відповідні події безпеки та дозволити аналіз у разі порушення безпеки або інциденту безпеки. Вимкніть непотрібні служби: вимкніть усі непотрібні служби у ваших системах. Резервне копіювання даних Резервні копії даних — це просто копії ваших цінних даних, які зберігаються в безпечному місці, окремо від основного сховища. Вони надають засоби для відновлення ваших даних у разі будь-яких інцидентів із втратою даних, наприклад випадкового видалення, збою апаратного забезпечення або кібератак, як-от програм-вимагачів. Рекомендації щодо резервного копіювання даних включають: Часте резервне копіювання за розкладом: плануйте регулярне резервне копіювання та автоматизуйте процес, щоб забезпечити узгодженість і зменшити ризик людської помилки. Кілька копій: зберігайте кілька копій своїх резервних копій, бажано на різних типах носіїв (наприклад, на зовнішніх жорстких дисках, у хмарному сховищі чи стрічках). Зовнішнє зберігання: зберігайте принаймні одну копію своїх резервних копій поза межами сайту. Це допоможе захистити від втрати даних через фізичні катастрофи або крадіжки на місці. Шифрування: зашифруйте свої резервні копії, щоб захистити конфіденційні дані від несанкціонованого доступу. Тестування та перевірка: регулярно перевіряйте свої резервні копії, щоб переконатися, що вони функціонують належним чином і їх можна відновити за потреби. Ізоляція Ізоляція є ключовим принципом кібербезпеки, який допомагає забезпечити конфіденційність, цілісність і доступність інформаційних систем і даних. Основна ідея ізоляції полягає в тому, щоб відокремити різні компоненти або процеси таким чином, що якщо один буде скомпрометовано, інші залишатимуться захищеними. Ізоляція може бути застосована на різних рівнях, включаючи апаратний, програмний і мережевий рівні. Він зазвичай використовується для захисту конфіденційних даних, критично важливих систем і для обмеження потенційної шкоди, спричиненої зловмисними діями. Апаратна ізоляція забезпечує фізичне розділення між різними компонентами або системами, тим самим запобігаючи прямому доступу або перешкодам між ними. Цього можна досягти за допомогою кількох механізмів, зокрема: Системи з повітряним проміжком: комп’ютер або мережа, які не мають прямих зв’язків із зовнішніми мережами чи системами, що гарантує, що несанкціонований доступ або витік даних практично неможливий. Апаратні модулі безпеки: виділені фізичні пристрої, які керують цифровими ключами та криптографічними операціями, гарантуючи, що конфіденційний криптографічний матеріал відокремлений від інших компонентів системи та захищений від втручання чи несанкціонованого доступу. Ізоляція програмного забезпечення прагне розділити дані та процеси в самому програмному середовищі. Серед поширених методів: Віртуалізація: створення ізольованих віртуальних машин на одному фізичному хості, що дозволяє кільком операційним системам і програмам працювати паралельно без прямого доступу до ресурсів одна одної. Контейнери: легкі віртуальні середовища, які дозволяють програмам працювати ізольовано одна від одної, спільно використовувати те саме ядро операційної системи, але маючи окремі файлові системи, бібліотеки та простори імен. Пісочниця: техніка безпеки, яка обмежує діяльність програми в обмеженому середовищі, захищаючи базову систему та інші програми від потенційної шкоди. Ізоляція мережі спрямована на розділення та контроль зв’язку між різними системами, пристроями чи мережами. Це можна реалізувати кількома засобами, наприклад: Брандмауери: пристрої або програмне забезпечення, які діють як бар’єр, фільтруючи та контролюючи трафік між мережами чи пристроями на основі попередньо визначених політик. Віртуальні локальні мережі: Логічні розділи, створені у фізичній мережі, що поділяють пристрої на окремі групи з обмеженим зв’язком між ними. Віртуальні приватні мережі (VPN): зашифровані з’єднання, які безпечно тунелюють мережевий трафік через загальнодоступний Інтернет, захищаючи його від прослуховування чи втручання та забезпечуючи конфіденційність зв’язку. Реалізація концепції ізоляції у вашій стратегії кібербезпеки може значно підвищити рівень безпеки вашої організації шляхом обмеження поверхні атаки, утримання потенційних загроз і пом’якшення впливу порушень безпеки. Система виявлення вторгнень (IDS) Система виявлення вторгнень (IDS) — це критично важливий інструмент безпеки, призначений для моніторингу й аналізу мережевого трафіку чи дій хоста на наявність будь-яких ознак зловмисної діяльності, порушень політики або спроб несанкціонованого доступу. Після виявлення загрози або аномалії IDS надсилає сповіщення адміністратору безпеки для подальшого дослідження та можливих дій. Є два види IDS: Мережева система виявлення вторгнень (NIDS): цей тип IDS розгортається на мережевих пристроях, таких як маршрутизатори, комутатори або брандмауери, для моніторингу та аналізу трафіку між хостами в мережі. Система виявлення вторгнень на основі хоста (HIDS): цей тип IDS встановлюється на окремих хостах, таких як сервери або робочі станції, для моніторингу та аналізу дій на цьому конкретному хості. Система запобігання вторгненням (IPS) Система запобігання вторгненням (IPS) — це вдосконалене рішення безпеки, тісно пов’язане з IDS. У той час як IDS в основному зосереджується на виявленні та попередженні про вторгнення, IPS робить крок далі та активно працює над запобіганням атакам. Він відстежує, аналізує та виконує попередньо налаштовані автоматичні дії на основі підозрілих дій, наприклад блокування зловмисного трафіку, скидання з’єднань або скидання шкідливих пакетів. Існує два види IPS: Мережева система запобігання вторгненням (NIPS): цей тип IPS розгортається в мережі з мережевими пристроями та ретельно відстежує мережевий трафік, що дозволяє виконувати дії в режимі реального часу. Система запобігання вторгненням на основі хосту (HIPS): цей тип IPS встановлюється на окремих хостах і активно запобігає атакам, контролюючи вхідні та вихідні дані на хості, обмежуючи доступ до ресурсів і використовуючи елементи керування на рівні програми. IDS та IPS є важливими компонентами надійної стратегії кібербезпеки. IDS зосереджується на виявленні потенційних вторгнень і сповіщенні про них, тоді як IPS робить це ще далі, активно запобігаючи та пом’якшуючи атаки. Системи на основі мережі захищають мережі, а системи на основі хостів захищають окремі хости в мережі. Для постійного захисту від нових загроз необхідно регулярно оновлювати та налаштовувати IDS/IPS. Розуміючи основи IDS та IPS, ви зможете краще оцінити свої потреби в безпеці та вжити правильних заходів для захисту своєї мережі та хостів від потенційних зловмисників. Антивірусне програмне забезпечення Антивірусне програмне забезпечення — це програма, призначена для захисту вашого комп’ютера від зловмисного програмного забезпечення, також відомого як зловмисне програмне забезпечення. Зловмисне програмне забезпечення включає, серед іншого, віруси, хробаки, програми-вимагачі, шпигунські програми та трояни. Основною функцією антивірусного програмного забезпечення є виявлення, запобігання та видалення зловмисного програмного забезпечення з вашого комп’ютера чи мережі. Основні характеристики антивірусного програмного забезпечення Сканування в режимі реального часу: антивірусні програми постійно відстежують ваш комп’ютер на наявність потенційних загроз, дозволяючи їм ідентифікувати та нейтралізувати зловмисне програмне забезпечення, перш ніж воно може завдати шкоди. Виявлення зловмисного програмного забезпечення: антивірусне програмне забезпечення використовує комбінацію виявлення на основі сигнатур і поведінкового аналізу для визначення відомого та невідомого зловмисного програмного забезпечення. Виявлення на основі сигнатур спирається на базу даних відомих сигнатур вірусів, тоді як аналіз поведінки перевіряє, як програмне забезпечення поводиться у вашій системі. Автоматичні оновлення: оскільки нове шкідливе програмне забезпечення створюється щодня, антивірусне програмне забезпечення необхідно часто оновлювати, щоб залишатися ефективним. Більшість антивірусних програм можуть автоматично оновлювати визначення вірусів (базу даних сигнатур відомих зловмисних програм) і модулі програмного забезпечення для забезпечення максимального захисту. Карантин і видалення: після виявлення зловмисного програмного забезпечення антивірусне програмне забезпечення спробує або повністю видалити загрозу, або помістити її в карантин, щоб запобігти подальшому пошкодженню вашої системи. Сканування системи: важливо виконувати регулярне сканування системи, щоб виявити та видалити зловмисне програмне забезпечення, яке могло обійти сканування в реальному часі. Більшість антивірусних програм пропонують швидкі, повні та спеціальні параметри сканування. Сканування зовнішніх пристроїв: антивірусне програмне забезпечення також може сканувати зовнішні пристрої, такі як USBнакопичувачі та компакт-диски, на наявність потенційних загроз, перш ніж вони зможуть заразити ваш комп’ютер. Захист електронної пошти. Електронна пошта є поширеним вектором поширення зловмисного програмного забезпечення. Антивірусні програми часто включають сканування електронної пошти як функцію для виявлення та запобігання загрозам, що поширюються електронною поштою. Встановлюючи та підтримуючи актуальну антивірусну програму, ви можете значно знизити ризик стати жертвою кібератак і підтримувати безпечне середовище для свого комп’ютера та особистих даних. Антизловмисне програмне забезпечення Антизловмисне програмне забезпечення (anti-malware) — це тип програмного забезпечення, призначеного для виявлення, запобігання та видалення шкідливого програмного забезпечення (зловмисного програмного забезпечення) з комп’ютерної системи чи мережі. Зловмисне програмне забезпечення може включати різні типи загроз, такі як віруси, хробаки, трояни, шпигунське програмне забезпечення, рекламне програмне забезпечення та програми-вимагачі. Програмне забезпечення для захисту від зловмисних програм відіграє вирішальну роль у підтримці безпеки та цілісності вашої системи, виявляючи та усуваючи ці загрози. Програмне забезпечення для захисту від зловмисного програмного забезпечення зазвичай використовує комбінацію методів для ідентифікації та видалення зловмисного програмного забезпечення, зокрема: Виявлення на основі сигнатур: цей метод порівнює файли у вашій системі з базою даних відомих сигнатур зловмисного програмного забезпечення, які є унікальними шаблонами або характеристиками кожного типу зловмисного програмного забезпечення. Якщо файл відповідає відомому підпису, програмне забезпечення для захисту від зловмисного програмного забезпечення поміщає його в карантин або видаляє. Евристичний аналіз. Евристичний аналіз — це більш досконала техніка, яка шукає підозрілу поведінку або раніше невідоме зловмисне програмне забезпечення. Замість того, щоб покладатися виключно на відомі сигнатури зловмисного програмного забезпечення, евристичний аналіз використовує алгоритми для виявлення нового або модифікованого зловмисного програмного забезпечення на основі характеристик або моделей поведінки відомих загроз. Захист у режимі реального часу: програмне забезпечення для захисту від зловмисного програмного забезпечення часто забезпечує захист у режимі реального часу, постійно скануючи вашу систему та відстежуючи дії, щоб ідентифікувати та зупиняти шкідливі дії, щойно вони відбуваються. Карантин і видалення файлів: якщо виявлено потенційну загрозу, програма захисту від зловмисного програмного забезпечення поміщає файл у карантин, запобігаючи подальшому пошкодженню вашої системи. Потім ви можете вирішити, чи потрібно видалити файл або відновити його, якщо він хибно позитивний. Регулярні оновлення: оскільки постійно виявляються нові типи та варіанти зловмисного програмного забезпечення, дуже важливо, щоб програмне забезпечення для захисту від зловмисного програмного забезпечення отримувало регулярні оновлення своєї бази даних сигнатур і евристичних алгоритмів. Це гарантує, що програмне забезпечення може ефективно захистити вашу систему від нових загроз. Вибираючи засіб захисту від зловмисного програмного забезпечення, враховуйте такі фактори: Сумісність: переконайтеся, що програмне забезпечення сумісне з вашою операційною системою та іншими засобами безпеки, які ви можете використовувати. Продуктивність: переконайтеся, що програмне забезпечення має мінімальний вплив на продуктивність вашої системи та не сповільнює суттєво роботу комп’ютера. Зручність використання: виберіть рішення, яке легко встановити, налаштувати та використовувати. Зручне програмне забезпечення особливо важливо для користувачів, які не розбираються в техніці. Ефективність: шукайте засіб захисту від зловмисного програмного забезпечення з високим рівнем виявлення та низьким рівнем помилкових спрацьовувань, а також комплексними можливостями захисту в режимі реального часу. Репутація: виберіть продукт для захисту від зловмисного програмного забезпечення від авторитетного постачальника з підтвердженим досвідом успішного виявлення та видалення зловмисного програмного забезпечення. Підсумовуючи, захист від зловмисного програмного забезпечення є ключовим компонентом комплексної стратегії кібербезпеки. Інвестування в комплексне рішення для захисту від зловмисного програмного забезпечення може допомогти захистити ваші комп’ютерні системи, дані та особисту інформацію від широкого спектру загроз. Регулярно оновлюйте програмне забезпечення для захисту від зловмисного програмного забезпечення та дотримуйтеся належних правил кібергігієни, щоб мінімізувати ризик зараження шкідливим програмним забезпеченням. Honeypot (приманка) Приманка — це засіб безпеки, призначений для заманювання й уловлювання потенційних кібер-зловмисників, зазвичай видаючи себе за вразливу систему чи мережу. Honeypots може бути цінним інструментом для розуміння різних тактик, які використовують зловмисники, що дозволяє фахівцям із безпеки розробляти кращі стратегії захисту від цих атак. У цьому розділі ми розглянемо різні типи приманок, їх використання та деякі важливі зауваження під час їх впровадження. Існує кілька різних типів приманок, які можна реалізувати, кожен з яких має унікальні функції та можливості. Серед поширених типів: Приманки з низьким рівнем взаємодії: ці приманки імітують обмежений набір служб або вразливостей, щоб заманити зловмисників. Вони вимагають мінімальних ресурсів і їх легше встановити, ніж інші типи приманок. Вони часто використовуються для збору основної інформації про поведінку зловмисників. Приманки високої взаємодії: ці приманки імітують повне та реалістичне середовище, у якому часто працюють певні операційні системи та служби. Вони ресурсомісткі, але забезпечують більш глибоке розуміння поведінки зловмисників і можуть бути використані для виявлення більш складних загроз. Дослідницькі приманки: ці приманки розроблено спеціально для збору детальної інформації про методи та мотиви зловмисників для подальшого аналізу. Їм часто потрібні передові знання та ресурси для підтримки, але вони надають цінну інформацію. Honeypots мають кілька застосувань у сфері кібербезпеки: Виявлення нових загроз: Honeypots може допомогти фахівцям із безпеки визначити нові методи атак, зловмисне програмне забезпечення чи інші загрози, перш ніж вони вплинуть на реальні системи. Відволікання зловмисників: представляючи, здавалося б, вразливу ціль, honeypots можуть відвернути увагу зловмисників від фактичних критичних систем, забезпечуючи таким чином додатковий рівень безпеки. Збирайте дані про атаки. Ретельно відстежуючи взаємодію з приманками, спеціалісти з безпеки можуть збирати цінну інформацію про поведінку, тактику та прийоми зловмисників, удосконалюючи стратегії кіберзахисту. Хоча honeypots можуть бути потужними інструментами в арсеналі професіонала безпеки, є кілька важливих факторів, які слід враховувати: Етика та законність: надзвичайно важливо забезпечити, щоб уся діяльність honeypot здійснювалася етично та в межах закону. У деяких юрисдикціях певні дії, пов’язані з приманками (наприклад, відлов зловмисників), можуть бути незаконними або вимагати спеціальних дозволів. Ризик компрометації: Honeypots можуть додати іншу поверхню для атаки, якою можуть скористатися зловмисники, якщо вони не будуть належним чином захищені чи обслуговуються. Якщо зловмисник визначить, що система є приманкою, він може вирішити продовжити атаку на мережу або почати більш цілеспрямовані атаки. Технічне обслуговування та ресурси: розробка та підтримка honeypots може потребувати ресурсів, вимагаючи виділених систем або віртуальних машин, досвіду системного адміністрування та постійного моніторингу. Важливо ретельно зважити переваги та ризики впровадження honeypots і переконатися, що вони використовуються відповідально та стратегічно у вашому плані кібербезпеки. Blue Team vs Red Team vs Purple Team У контексті кібербезпеки синя команда, червона команда та фіолетова команда — це терміни, які використовуються для опису різних ролей і методологій, що використовуються для забезпечення безпеки організації чи системи. Синя команда відповідає за захист інформаційних систем, мереж і критичних активів організації від загроз безпеці. Їм доручено постійний моніторинг систем, виявлення потенційних інцидентів безпеки та реагування на них, а також впровадження захисних заходів. Основні напрямки діяльності Blue Team: ▪ Розробляти та впроваджувати політики та процедури безпеки ▪ Виконувати оцінку вразливості та оцінку ризику ▪ Розгорнути інструменти та технології безпеки (наприклад, брандмауери, системи виявлення вторгнень тощо) ▪ Відстежувати журнали та аналізувати події безпеки на наявність потенційних загроз ▪ Реагувати на інциденти безпеки та розслідувати їх ▪ Проводити програми підвищення обізнаності та навчання з питань безпеки Основна мета Червоної команди — симулювати атаки в реальному світі, виявити вразливі місця та перевірити ефективність захисних стратегій Синьої команди. Вони є зовнішніми або внутрішніми членами команди, які діють як супротивники, використовуючи креативність і передові методи для перевірки захисту кібербезпеки організації. Основні напрямки діяльності Red Team: ▪ Виконують регулярне тестування на проникнення та оцінку безпеки ▪ Використовують методи соціальної інженерії, щоб використовувати людські слабкості ▪ Аналізують та використовують вразливості в системах, мережах і програмах ▪ Емулюють поширені загрози та сценарії атак ▪ Надають корисну інформацію для покращення стану безпеки організації Фіолетова команда долає розрив між синьою командою та червоною командою, допомагаючи створити середовище для більшої співпраці. Вони сприяють спілкуванню та обміну інформацією між двома командами, зрештою спрямовані на підвищення загальної ефективності програми безпеки. Основні напрямки діяльності Purple Team: ▪ Координують та планують спільні навчання синьої та червоної команд ▪ Діляться знаннями, техніками та знахідками між командами ▪ Допомагають з впровадженням визначених покращень безпеки ▪ Оцінюють і виміряють ефективність засобів контролю безпеки ▪ Розвивають культуру постійного вдосконалення та співпраці Вкладаючи зусилля в синю, червону та фіолетову команди, організації можуть досягти більш надійної та стійкої системи безпеки, здатної протистояти загрозам, що постійно змінюються, і адаптуватися до них. Легко зрозуміти, наскільки комп’ютерна безпека є важливою складовою нашого все більш цифрового стилю життя. Менш очевидно побачити, як це відбувається в повсякденному житті інженера-програміста. Насправді більшість традиційної діяльності з комп’ютерної безпеки взагалі не виконується інженерами-програмістами. Вони обробляються персоналом інформаційних технологій (ІТ), який виконує такі завдання, як реагування на інциденти (вирішення атаки, що триває), експертиза (з’ясування того, що сталося після атаки), встановлення виправлень програмного забезпечення (переконання, що все програмне забезпечення на системах захищене від відомих атак), налаштування антивірусних сканерів і брандмауерів. Що ж тоді робить інженер-програміст? Інженер-програміст повинен знати, як проектувати програмне забезпечення, щоб можна було забезпечити конфіденційність, цілісність і доступність. Це означає, що розробка та впровадження комп’ютерних систем повинні мати мінімальну кількість вразливостей, якими може скористатися зловмисник.

**Лекція 2.**

**Типи атак та їх відмінності.**

**Шкідливе програмне забезпечення** – це будь-яке програмне забезпечення, навмисно створене для заподіяння шкоди комп'ютерній системі, серверу, мережі або користувачеві. Це широкий термін, який охоплює різні типи шкідливого програмного забезпечення, створеного кіберзлочинцями для різних цілей.

Розглянемо основні типи шкідливих програм та їхні характеристики.

**Комп'ютерний вірус** – це тип шкідливого програмного забезпечення, який, подібно до біологічного вірусу, приєднується до хосту (наприклад, файлу або програмного забезпечення) і реплікується під час його запуску. Віруси можуть пошкоджувати, видаляти або змінювати дані, а також сповільнювати роботу системи.

**Черв'яки** – це шкідливе програмне забезпечення, що самовідтворюється і поширюється мережею без втручання людини. Вони використовують вразливості системи, споживаючи пропускну здатність, а іноді й несучи корисне навантаження для зараження цільових комп'ютерів.

**Троянський кінь** – це програмне забезпечення, яке маскується під легальну програму, але містить шкідливий код. Користувачі несвідомо завантажують і встановлюють його, надаючи зловмиснику несанкціонований доступ до комп'ютера або мережі. Трояни можуть бути використані для крадіжки даних, створення чорного ходу або запуску додаткових атак шкідливого програмного забезпечення.

**Програми-здирники** – це тип шкідливих програм, які шифрують файли своїх жертв і вимагають викуп, зазвичай у вигляді криптовалюти, за ключ для розшифровки. Якщо жертва відмовляється або не платить протягом певного часу, зашифровані дані можуть бути втрачені назавжди.

**Шпигунське ПЗ** – це тип шкідливого програмного забезпечення, призначеного для збору та передачі інформації про користувача або

організацію без їхньої згоди. Воно може перехоплювати натискання клавіш, записувати історію відвідувань і отримувати доступ до особистих даних, таких як імена користувачів і паролі.

**Рекламне ПЗ** – це підтримуване рекламою програмне забезпечення, яке автоматично відображає або завантажує рекламні матеріали, часто у вигляді спливаючих вікон, на комп'ютері користувача. Хоча рекламне ПЗ не завжди є зловмисним, воно може бути нав'язливим і відкривати двері для інших шкідливих програм.

**Руткіт** – це тип шкідливого програмного забезпечення, призначеного для приховування або маскування присутності інших шкідливих програм у комп'ютерній системі. Це дозволяє йому підтримувати постійний несанкціонований доступ до системи і може ускладнити користувачам або програмному забезпеченню виявлення та видалення заражених файлів.

**Клавіатурні шпигуни** - це тип шкідливих програм, які відстежують і записують натискання клавіш користувача, дозволяючи зловмисникам перехоплювати конфіденційну інформацію, наприклад, облікові дані для входу в систему або фінансову інформацію, введену з клавіатури.

Розуміння різних типів шкідливих програм може допомогти вам краще ідентифікувати та захиститися від різних кіберзагроз. Оскільки кіберпростір продовжує розвиватися, важливо залишатися в курсі нових шкідливих програм і володіти необхідними навичками та знаннями з безпеки.

Найкращі практики протидії загрозам.

**Поінформованість:** Ознайомте себе та свою команду з поширеними типами загроз та їхніми характеристиками.

**Запобігання:** Вживайте заходів для зменшення загроз, таких як регулярне оновлення програмного забезпечення, надійні паролі та захист кінцевих точок.

**Виявлення:** Впроваджуйте інструменти моніторингу та виявлення для виявлення загроз або підозрілої активності.

**Реагування:** Розробіть план реагування на інциденти, включаючи локалізацію, відновлення та комунікацію.

Тепер розглянемо основні типи атак.

**Підвищення привілеїв**

Атаки на підвищення привілеїв відбуваються, коли зловмисник отримує несанкціонований доступ до системи, а потім підвищує свої привілеї для виконання дій, які він не повинен був робити. Існує два основних типи ескалації привілеїв:

**Горизонтальна ескалація привілеїв:** У цьому типі атаки зловмисник отримує несанкціонований доступ до облікового запису користувача з таким же рівнем привілеїв, як і у нього самого, але може виконувати дії або отримувати доступ до даних, які належать іншому користувачеві.

**Вертикальна ескалація привілеїв:** Також відомий як "підвищення привілеїв", цей тип атаки передбачає, що зловмисник отримує несанкціонований доступ до системи, а потім підвищує свій рівень привілеїв від звичайного користувача до адміністратора, власника системи або root-користувача. Це дає зловмиснику більший контроль над системою та її ресурсами.

Щоб захистити свої системи та дані від атак на підвищення привілеїв, розгляньте можливість впровадження наступних найкращих практик:

**Принцип найменших привілеїв:** Надайте мінімально необхідний доступ і привілеї кожному обліковому запису користувача, а також регулярно переглядайте та оновлюйте дозволи доступу за необхідності.

**Регулярно оновлюйте та виправляйте програмне забезпечення:** Оновлюйте програмне забезпечення та системи найновішими патчами безпеки, щоб усунути відомі вразливості, які можуть бути використані в атаках на підвищення привілеїв.

**Впроваджуйте надійну автентифікацію та авторизацію:** Використовуйте надійні методи автентифікації (наприклад, багатофакторну автентифікацію) та забезпечте належний контроль доступу, щоб запобігти несанкціонованому доступу до конфіденційних даних або системних ресурсів.

**Проводьте аудит безпеки:** Регулярно перевіряйте наявність будь-яких неправильних конфігурацій, вразливостей або застарілого програмного забезпечення, які можуть бути використані в атаках на підвищення привілеїв.

**Відстежуйте та реєструйте системні дії:** Впровадьте системи реєстрації та моніторингу для виявлення підозрілих дій облікових записів або змін у привілеях користувачів, які можуть свідчити про атаку на підвищення привілеїв.

Розуміючи типи атак на підвищення привілеїв і дотримуючись цих найкращих практик, ви можете створити більш безпечне середовище для ваших даних і систем, а також зменшити ризик отримання необмеженого доступу неавторизованими користувачами.

**Нульовий день**

**Нульовий день** — це вразливість у програмному, апаратному чи програмно-апаратному забезпеченні, про яку невідомо сторонам, відповідальним за її усунення або виправлення. Кіберзлочинці можуть використовувати ці вразливості для отримання несанкціонованого доступу до систем, викрадення конфіденційних даних або інших зловмисних дій. Вразливості нульового дня є особливо небезпечними, оскільки їх важко виявити та запобігти їм, оскільки не існує жодних виправлень або засобів захисту від них.

Зловмисники можуть створювати експлойти нульового дня шляхом написання шкідливого коду, який використовує виявлену вразливість нульового дня. Ці експлойти можуть бути доставлені різними методами,

такими як фішингові електронні листи або випадкові завантаження зі скомпрометованих веб-сайтів.

Через невідому природу вразливостей нульового дня традиційні заходи безпеки, такі як антивірусні програми на основі сигнатур та брандмауери, можуть виявитися неефективними для їх виявлення. Однак можна зробити кілька кроків, щоб захистити себе від атак нульового дня:

**Управління виправленнями:** Регулярно оновлюйте та виправляйте все програмне, апаратне та мікропрограмне забезпечення, щоб мінімізувати точки входу для потенційних атак.

**Відстежуйте мережевий трафік:** Використовуйте інструменти мережевого моніторингу для постійного аналізу мережевого трафіку та пошуку будь-якої незвичної або підозрілої активності, яка може свідчити про спробу використання експлойтів нульового дня.

**Виявлення на основі поведінки:** Впроваджуйте рішення безпеки, які зосереджені на моніторингу поведінки програм і мережевого трафіку на предмет виявлення будь-яких ознак зловмисної діяльності, а не покладайтеся виключно на методи виявлення на основі сигнатур.

**Використовуйте аналітику загроз:** Підпишіться на канали аналітики загроз, які надають інформацію про найновіші вразливості безпеки та нові загрози, щоб бути в курсі можливих атак нульового дня.

**Впроваджуйте суворий контроль доступу:** Контролюйте доступ до критично важливих систем і даних, обмежуйте кількість привілейованих облікових записів і застосовуйте політику найменших привілеїв, де це можливо, щоб зловмисникам було складніше використовувати вразливості "нульового дня".

**Фішинг**

**Фішинг** — це спроба отримати конфіденційну інформацію, таку як облікові дані для входу в систему або дані кредитної картки, маскуючись під

надійну організацію. Зазвичай це відбувається через електронну пошту. Зловмисник часто створює електронний лист, який виглядає як лист від авторитетного джерела, такого як банк, соціальна мережа або навіть від відомого контакту. Електронний лист може містити посилання, яке перенаправляє жертву на фальшивий веб-сайт, де її просять ввести свої облікові дані або іншу конфіденційну інформацію.

Перший позов про фішинг був поданий у 2004 році проти каліфорнійського підлітка, який створив імітацію веб-сайту "America Online". За допомогою цього фальшивого веб-сайту він зміг отримати конфіденційну інформацію від користувачів і отримати доступ до даних кредитних карток, щоб зняти гроші з їхніх рахунків.

Фішинг є настільки поширеним і відомим явищем зараз, але він існував не завжди. Хоча ця практика з'явилася приблизно в 1995 році, про ці види шахрайства пересічні громадяни дізналися лише через десять років. Щоб не стати жертвою таких шахрайств, корисно мати базове уявлення про історію їх виникнення.

Фішингові шахраї використовують підроблені електронні листи та веб-сайти як приманку, щоб спонукати людей добровільно передати конфіденційну інформацію. Тож не дивно, що термін "фішинг" зазвичай використовується для опису цих хитрощів. Існує також вагома причина для використання "ph" замість "f" у написанні цього терміну. Деякі з перших хакерів були відомі як фріки. Фрікінг - це дослідження, експерименти та вивчення телекомунікаційних систем. Фріки та хакери завжди були тісно пов'язані між собою. Для того, щоб пов'язати фішингові шахрайства з цими підпільними спільнотами, використовували літеру "ph".

Вперше термін "фішинг" був використаний і зафіксований 2 січня 1996 року. Згадка про нього з'явилася в групі новин Usenet під назвою AOHell. Цілком доречно, що це було зроблено саме там; саме в America Online з'явилися перші натяки на те, що згодом стане великою кримінальною проблемою.

Багато в чому фішинг майже не змінився з часів розквіту AOL. Однак у 2001 році фішери звернули свою увагу на системи онлайн-платежів. Хоча перша атака на E-Gold у червні 2001 року не вважалася успішною, вона посіяла важливе зерно. Наприкінці 2003 року фішери зареєстрували десятки доменів, які виглядали як легальні сайти, такі як eBay та PayPal, якщо ви не були уважними. Вони використовували програми-черв'яки для розсилки підроблених електронних листів клієнтам PayPal. Ці клієнти потрапляли на підроблені сайти, де їх просили оновити дані кредитної картки та іншу ідентифікаційну інформацію.

На початку 2004 року фішери були на величезній хвилі успіху, що включала атаки на банківські сайти та їхніх клієнтів. Для отримання конфіденційної інформації від жертв використовувалися спливаючі вікна. У період з травня 2004 року по травень 2005 року близько 1,2 мільйона користувачів у США зазнали збитків від фішингу на загальну суму близько 929 мільйонів доларів. Організації втрачають від фішингу близько 2 мільярдів доларів на рік.

Фішинг офіційно визнаний повністю організованою частиною чорного ринку. У глобальному масштабі з'являється спеціалізоване програмне забезпечення, яке може обробляти фішингові платежі, що, в свою чергу, створює величезний ризик. Це програмне забезпечення впроваджується у фішингові кампанії організованими злочинними угрупованнями.

Наприкінці 2008 року з'являється біткойн та інші криптовалюти. Це робить транзакції з використанням шкідливого програмного забезпечення безпечними та анонімними, що змінює правила гри для кіберзлочинців.

У вересні 2013 року програма-здирник Cryptolocker заразила 250 000 персональних комп'ютерів, ставши першим криптографічним шкідливим програмним забезпеченням, яке розповсюджувалося шляхом завантаження зі зламаного веб-сайту та/або надсилалося жертвам у вигляді двох різних фішингових електронних листів. Перший лист містив вкладений Zip-архів, який стверджував, що це скарга клієнта, і був націлений на бізнес, а другий

містив шкідливе посилання з повідомленням про проблему з погашенням чека і був націлений на широку громадськість. Після натискання на посилання Cryptolocker шифрує та блокує файли на комп'ютері і вимагає від власника здійснити платіж в обмін на ключ для розблокування та розшифрування файлів.

Починаючи з 2017 року, фішери все частіше використовують HTTPS на своїх сайтах. Коли ви переходите за фішинговим посиланням, сайти, на які вони ведуть і які намагаються обманом змусити вас ввести облікові дані, особисту інформацію тощо, використовують веб-шифрування щонайменше у 24 відсотках випадків. Зелений висячий замок дає споживачам хибне відчуття безпеки. Насправді він говорить лише про те, що трафік між сервером і браузером користувача зашифрований і захищений від перехоплення.

Фішингова кампанія, спрямована на організації, пов'язані з Зимовою Олімпіадою 2018 року, вперше використовує інструмент PowerShell під назвою Invoke-PSImage, який дозволяє зловмисникам ховати шкідливі скрипти в пікселях безпечних на вигляд файлів зображень, а потім запускати їх безпосередньо з пам'яті. Приховування скрипту у файлі зображення не лише допомагає йому уникнути виявлення, але й виконує його безпосередньо з пам'яті - це безфайлова технологія, яка в більшості випадків не розпізнається традиційними антивірусними рішеннями. Це ще один тривожний приклад того, як атаки залишаються під радаром і еволюціонують від використання шкідливих .exe-файлів.

У 2018 році дослідники виявили нове покоління фішингових наборів, доступних у Dark Web для кіберзлочинців. Цей набір дозволяє будь-кому, хто його завантажить, легко створювати переконливі електронні листи та перенаправляти на сайти, які точно імітують елементи брендингу відомих фірм, і запускати фішингові кампанії, які збирають особисту та фінансову інформацію жертв, які нічого не підозрюють.

Фішингові кампанії з використанням подарункових карток, що розпочалися у 2018 році, продовжили розвиватися у 2019 році. Кіберзлочинці стали набагато краще створювати правдоподібні приводи (тобто "стимули"), відкрито вимагати конфіденційності, вони стали дуже жадібними - до $4000 за один запит у подарункових картках, і вони стимулюють всю схему, пропонуючи одержувачу хабар ("візьми одну для себе"), що в певному сенсі має на меті перетворити ціль на співучасника змови.

У 2019 році з'явився новий тип атак - компрометація електронної пошти постачальників, яка є різновидом атаки на ділову електронну пошту (BEC) (або CEO Fraud). Кіберзлочинці отримують доступ до електронних поштових скриньок компанії, що входить до їхнього ланцюга постачання, а потім використовують ці акаунти для націлювання на клієнтів цієї компанії. Атаки зосереджені на організаціях з глобальними ланцюгами поставок і намагаються обдурити клієнтів постачальника, щоб змусити їх оплатити фальшиві рахунки-фактури. У 2019 році компрометація електронної пошти постачальників вплинула на щонайменше 500 організацій по всьому світу.

За даними Microsoft, у 2020 році фішингові атаки розвивалися, зокрема, у таких інноваційних напрямках: Надсилання в електронних листах посилань на підроблені результати пошуку Google, які ведуть на підконтрольні зловмисникам веб-сайти зі шкідливим програмним забезпеченням; надсилання в електронних листах посилань на неіснуючі сторінки на підконтрольному зловмисникам веб-сайті, щоб відобразити спеціальну сторінку 404, яка може бути використана для імітації сторінок входу на легітимні сайти; підробка сторінок входу в Office 365 для конкретної компанії, щоб вони виглядали настільки реалістично, що користувачі могли подумати, що це справжні сторінки.

Типові риси фішингових листів.

Занадто добре, щоб бути правдою — вигідні пропозиції та привабливі заяви, що привертають увагу, покликані миттєво привернути увагу людей. Наприклад, багато хто стверджує, що ви виграли iPhone, лотерею або інший

щедрий приз. Просто не натискайте на підозрілі електронні листи. Пам'ятайте, що якщо це здається занадто добре, щоб бути правдою, то, ймовірно, так воно і є!

Відчуття терміновості — улюблена тактика кіберзлочинців — просити вас діяти швидко, тому що суперпропозиції діють лише протягом обмеженого часу. Деякі з них навіть скажуть вам, що у вас є лише кілька хвилин, щоб відповісти. Коли ви стикаєтеся з такими листами, краще просто ігнорувати їх. Іноді вони повідомляють, що ваш акаунт буде призупинено, якщо ви негайно не оновите свої персональні дані. Більшість надійних організацій дають достатньо часу, перш ніж закрити акаунт. Якщо ви сумніваєтеся, відвідайте джерело безпосередньо, а не переходьте за посиланням в електронному листі.

Гіперпосилання — посилання може бути не тим, чим здається. При наведенні на посилання ви побачите справжню URL-адресу, на яку ви потрапите, натиснувши на нього. Вона може бути зовсім іншою, або це може бути популярний веб-сайт з помилкою в написанні, тому дивіться уважно.

Вкладення — якщо ви бачите в електронному листі вкладення, якого не очікували або яке не має сенсу, не відкривайте його! Вони часто містять корисне навантаження, наприклад, програми-вимагачі або інші віруси. Єдиний тип файлів, на який завжди безпечно натискати — це файли .txt.

Незвичайний відправник — незалежно від того, чи виглядає це як повідомлення від незнайомого вам відправника, чи від того, кого ви знаєте, якщо щось здається незвичним, неочікуваним, нехарактерним або просто підозрілим, не натискайте на нього!

Хоча хакери постійно вигадують нові методи, є деякі речі, які ви можете зробити, щоб захистити себе:

Для захисту від спаму можна використовувати спам-фільтри. Як правило, фільтри оцінюють походження повідомлення, програмне забезпечення, яке використовується для надсилання повідомлення, і зовнішній вигляд повідомлення, щоб визначити, чи є воно спамом. Іноді

спам-фільтри можуть навіть блокувати електронні листи з легальних джерел, тому вони не завжди є на 100% точними. Щоб запобігти відкриттю шахрайських веб-сайтів, слід змінити налаштування браузера. Браузери зберігають список фальшивих веб-сайтів, і коли ви намагаєтеся отримати доступ до веб-сайту, адреса блокується або відображається попереджувальне повідомлення. Налаштування браузера повинні дозволяти відкривати лише надійні веб-сайти. Багато веб-сайтів вимагають від користувачів введення інформації для входу під час показу зображення користувача. Такий тип системи може бути вразливим до атак на безпеку. Один із способів забезпечити безпеку — регулярно змінювати паролі і ніколи не використовувати один і той самий пароль для кількох облікових записів.

Банки та фінансові організації використовують системи моніторингу для запобігання фішингу. Громадяни можуть повідомляти про фішинг до галузевих груп, де проти цих шахрайських веб-сайтів можуть бути вжиті правові заходи.

Якщо в електронному листі є посилання, спочатку наведіть курсор на URL-адресу. Безпечні веб-сайти з дійсним сертифікатом Secure Socket Layer (SSL) починаються з "https". Згодом всі сайти повинні будуть мати дійсний SSL.

Як правило, електронні листи, надіслані кіберзлочинцями, маскуються так, щоб здавалося, що вони надіслані компанією, послугами якої користується одержувач. Банк не буде запитувати особисту інформацію електронною поштою і не призупинить дію вашого рахунку, якщо ви не оновите свої особисті дані протягом певного періоду часу. Більшість банків та фінансових установ також зазвичай вказують номер рахунку або інші персональні дані в електронному листі, що гарантує, що він надійшов з надійного джерела.

Як захистити себе:

Будьте обережні, відкриваючи електронні листи від невідомих відправників.

Шукайте підозрілі ознаки в електронному листі, такі як погана граматика або невідповідності в брендингу.

Завжди наводьте курсор на посилання в листах, щоб перевірити справжню URL-адресу, перш ніж натискати.

Увімкніть двофакторну автентифікацію (2FA) у своїх онлайн-акаунтах.

**Вішинг**, або **голосовий фішинг**, передбачає використання зловмисниками телефонних дзвінків або голосових повідомлень, щоб переконати жертву розкрити конфіденційну інформацію, таку як банківські реквізити або паролі. Вішинг-атаки часто спираються на тактику соціальної інженерії, змушуючи жертву повірити, що вона розмовляє з законним представником компанії або авторитетною особою.

Як захистити себе:

Будьте обережні при отриманні несподіваних телефонних дзвінків, особливо з незнайомих номерів.

Перевірте особу абонента, запитуючи деталі, які може знати лише законна сторона.

Уникайте надання особистої інформації по телефону, якщо тільки ви не були ініціатором дзвінка і не довіряєте отримувачу.

У разі сумнівів покладіть слухавку і зателефонуйте на відомий, перевірений номер компанії або установи, яку представляє абонент, що телефонує.

**Смішинг**, або **SMS-фішинг**, — це використання текстових повідомлень для того, щоб обманом змусити жертву розкрити конфіденційну інформацію або завантажити шкідливе програмне забезпечення. Зловмисник може вказати скорочену URL-адресу або номер телефону, намагаючись

обманом змусити жертву перейти за посиланням або зателефонувати на цей номер.

Як захистити себе:

Будьте обережні при отриманні небажаних текстових повідомлень, особливо від невідомих відправників.

Перевірте номер телефону відправника, щоб переконатися, що він є законним або відповідає передбачуваному джерелу.

Ніколи не переходьте за підозрілими посиланнями в текстових повідомленнях.

Встановіть програмне забезпечення для мобільної безпеки, щоб захистити свій пристрій від потенційних загроз.

**Спам**

**Спам –** це будь-які небажані або нерелевантні повідомлення, що надсилаються через Інтернет, зазвичай великій кількості користувачів, з метою реклами, фішингу або розповсюдження шкідливого програмного забезпечення. Такі повідомлення зазвичай надсилаються електронною поштою. Спам може містити шкідливі вкладення або посилання, які при натисканні завантажують шкідливе програмне забезпечення або перенаправляють користувачів на скомпрометовані веб-сайти.

Спамери часто використовують автоматизовані системи для надсилання цих повідомлень великій кількості одержувачів.

Ось деякі загальні характеристики спаму:

Підозрілі адреси відправників

Загальне привітання

Незвичайні або несподівані вкладення або посилання

Наполегливі або погрозливі формулювання

Запити на отримання особистої інформації

Щоб захистити себе від спаму, ви повинні.

Налаштувати ефективні фільтри електронної пошти

Ніколи не повідомляти свою електронну адресу публічно

Не переходити за підозрілими посиланнями та вкладеннями

**Спім**, або "спам через миттєві повідомлення", схожий на спам, але відбувається через служби миттєвих повідомлень (IM), такі як Facebook Messenger, WhatsApp та інші. Основна відмінність між спамом і спімом полягає в середовищі, через яке надсилаються небажані повідомлення. Як і спам, спім може використовуватися для реклами, поширення шкідливого програмного забезпечення або проведення фішингових атак.

**Плечовий серфінг**

**Плечовий серфінг** — це різновид соціальної інженерії, коли зловмисник спостерігає за чиїмось екраном, клавіатурою або будь-яким іншим пристроєм, щоб отримати несанкціонований доступ до конфіденційної інформації. Зазвичай зловмисник таємно спостерігає за жертвою під час введення даних, прямо чи опосередковано через відображення, смартфони або інше записуюче обладнання.

Як відбувається плечовий серфінг.

**Пряме спостереження:** Зловмисник стоїть поруч із жертвою і спостерігає за її діями, такими як введення паролів, реквізитів кредитної картки або доступ до конфіденційних даних.

**Використання камер:** Зловмисник може використовувати приховану камеру або смартфон, щоб таємно записувати натискання клавіш, які потім можна проаналізувати для вилучення конфіденційної інформації.

**Перегляд відображень:** Зловмисники можуть переглядати відображення на сусідніх поверхнях, таких як вікна, блискучі предмети або навіть окуляри жертви, щоб контролювати їхню діяльність.

Запобігання плечовому серфінгу.

Щоб захистити себе від плечового серфінгу, дотримуйтесь наступних рекомендацій:

Будьте уважні до свого оточення, особливо в громадських місцях, де ризик плечового серфінгу вищий.

Використовуйте екрани конфіденційності або захисні екрани, щоб зменшити видимість вашого пристрою під різними кутами.

Якщо ви користуєтеся смартфоном або планшетом, нахиліть екран до себе і від потенційних спостерігачів.

При введенні конфіденційної інформації, такої як PIN-коди або паролі, прикривайте клавіатуру або клавіатуру тілом або рукою. Регулярно змінюйте паролі та уникайте використання паролів, які легко вгадати, або поширених паролів.

**Атака "водопою"**

**Атака "водопою"** — це цілеспрямована кібератака, під час якої зловмисник спостерігає за веб-сайтами, які часто відвідує певна група або організація, і намагається скомпрометувати ці сайти, щоб заразити бажані цілі. Ці атаки названі на честь природних відносин між хижаком і жертвою; подібно до того, як хижаки чекають біля водопою, щоб полювати на свою здобич.

У цьому типі атак зловмисник не націлюється безпосередньо на жертву, а зосереджується на веб-сайтах, які зазвичай відвідують користувачі, на яких спрямована атака.

Покроковий опис типової атаки біля водопою:

**Визначення цілі:** Зловмисник визначає конкретну організацію або групу, яку він хоче атакувати, наприклад, державну установу або корпорацію.

**Вивчення поведінки:** Зловмисник вивчає поведінку цільових користувачів в Інтернеті, спостерігаючи за тим, які веб-сайти вони часто відвідують.

**Компрометація веб-сайту:** Зловмисник використовує вразливості на одному або декількох веб-сайтах-мішенях і впроваджує на них шкідливий код. Це може відбуватися через вразливий плагін, слабкі паролі або навіть шляхом отримання доступу до хостингової платформи сайту.

**Зараження жертв:** Коли цільові користувачі відвідують скомпрометовані веб-сайти, вони несвідомо завантажують шкідливий код на свої комп'ютери, що дозволяє зловмиснику надалі експлуатувати заражені пристрої.

Для захисту від атак "водопою" важливо застосовувати найкращі практики, зокрема:

Регулярне оновлення програмного забезпечення як на серверах, так і на пристроях користувачів.

Встановлення надійних плагінів безпеки для веб-сайтів.

Впровадження надійної політики паролів та використання багатофакторної автентифікації.

Впровадження рішень для захисту мережі та кінцевих точок для виявлення та запобігання вторгненням.

**Drive-by атака**

**Drive-by атака** — це поширена загроза кібербезпеці, коли зловмисник намагається заразити комп'ютер або пристрій користувача, використовуючи вразливості у веб-браузері або його плагінах. Зазвичай користувачі несвідомо стають жертвами drive-by атак, коли відвідують шкідливий або скомпрометований веб-сайт, який, у свою чергу, автоматично виконує шкідливий код.

Як працюють drive-by атаки:

**Використання вразливостей в Інтернеті:** Зловмисники часто обирають популярні веб-сайти, які мають недоліки або вразливості в системі безпеки, що можуть бути використані для впровадження шкідливого коду.

**Шкідлива реклама:** Ще одним поширеним методом drive-by атак є реклама в Інтернеті. Кіберзлочинці використовують рекламні мережі для розповсюдження інфікованих оголошень, які після натискання запускають шкідливий код на пристрої користувача.

**Соціальна інженерія:** Зловмисники використовують тактику соціальної інженерії, щоб обманом змусити користувачів відвідувати скомпрометовані веб-сайти, які використовують вразливості браузерів.

Щоб захиститися від drive-by атак, врахуйте наступні заходи:

Оновлюйте своє програмне забезпечення: Регулярно оновлюйте веб-браузер, плагіни та операційну систему, щоб захиститися від відомих вразливостей.

Використовуйте надійне антивірусне програмне забезпечення: Використовуйте надійне антивірусне рішення зі скануванням у режимі реального часу та частими оновленнями сигнатур для виявлення та видалення шкідливого програмного забезпечення.

Увімкніть функцію click-to-play для плагінів: Налаштуйте параметри браузера так, щоб вимагати ручної активації плагінів.

Практикуйте хороші звички перегляду веб-сторінок: Уникайте відвідування підозрілих веб-сайтів, відкриття невідомих вкладень електронної пошти та переходу за неперевіреними посиланнями з джерел, яким ви не довіряєте.

Впроваджуйте веб-фільтрацію: Використовуйте фільтрацію контенту або безпечні веб-шлюзи, щоб заблокувати доступ до шкідливих веб-сайтів.

**Тайпсквотинг**

Тайпсквотинг, також відомий як викрадення URL-адрес або сквоттинг доменів, є зловмисною технікою кібератаки, яка спрямована на інтернет-користувачів, які помилково вводять неправильну адресу веб-сайту у своєму веб-браузері. Коли це відбувається, користувачі потрапляють на фальшивий веб-сайт, який дуже схожий на справжній. Зловмисники створюють ці підроблені веб-сайти, реєструючи доменні імена, схожі на цільовий веб-сайт, але з поширеними друкарськими помилками. Метою "опечаток" часто є поширення шкідливого програмного забезпечення, крадіжка особистої інформації або фінансових даних, продаж підроблених товарів або просування фішингового шахрайства.

Як працює «опечатка»

**Реєстрація домену:** Зловмисники реєструють доменні імена, схожі на популярні веб-сайти, але з невеликими помилками, наприклад, пропущеними або поміняними місцями символами.

**Створення підроблених веб-сайтів:** Зловмисники створюють веб-сайт, який візуально нагадує цільовий веб-сайт. Це може включати використання тих самих логотипів, зображень і макетів, що ускладнює користувачам можливість відрізнити підроблений сайт від справжнього.

**Заманювання жертв:** Нічого не підозрюючи користувачі, які роблять друкарські помилки при введенні URL-адреси, перенаправляються на фальшивий веб-сайт, де вони можуть несвідомо надати свою особисту або фінансову інформацію, завантажити шкідливе програмне забезпечення або стати жертвою фішингового шахрайства.

**Експлуатація:** Зловмисники можуть використовувати зібрану інформацію для крадіжки особистих даних, фінансового шахрайства або продавати дані. Вони також можуть використовувати заражені шкідливим програмним забезпеченням пристрої для створення бот-мереж або здійснення подальших атак на інші цілі.

Запобігання та пом'якшення наслідків

**Перевірте URL-адреси:** Завжди перевіряйте URL-адресу, яку ви вводите в браузері, щоб переконатися, що ви переходите на потрібний веб-сайт.

**Використовуйте закладки:** додавайте часто відвідувані веб-сайти до закладок, щоб не вводити URL-адресу щоразу вручну.

**Пошукові системи:** Якщо ви не впевнені у правильності URL-адреси, скористайтеся пошуковими системами, щоб знайти потрібний веб-сайт

**Використовуйте програмне забезпечення для захисту:** встановлюйте та оновлюйте сучасне програмне забезпечення для захисту на своїх пристроях, наприклад, антивірусне, антифішингове та антивірусне програмне забезпечення, щоб захиститися від потенційних загроз, пов'язаних з опечатуванням.

**Увімкніть захист браузера:** Багато веб-браузерів пропонують вбудовані функції безпеки, які допомагають виявляти та блокувати шкідливі веб-сайти. Переконайтеся, що ці функції ввімкнені та правильно налаштовані.

**Підбір паролів**

**Атаки грубої сили** — це метод спроб і помилок, який використовується зловмисниками для підбору правильних комбінацій облікових даних (імені користувача та пароля) для отримання несанкціонованого доступу до облікового запису або системи. Для цього зловмисники систематично перебирають якомога більше варіантів, поки не знайдуть правильну комбінацію. Під час атаки грубої сили зловмисник зазвичай використовує автоматизовані інструменти для генерації та тестування численних комбінацій паролів. Ця стратегія може зайняти багато часу, вимагати значних ресурсів і бути потенційно виявленою через велику кількість спроб входу за короткий проміжок часу.

Щоб зменшити ризики атаки "грубої сили", застосовуйте наступні найкращі практики:

**Надійна політика паролів:** створюйте складні та унікальні паролі, що поєднують великі та малі літери, цифри та спеціальні символи.

**Політика блокування облікових записів:** Тимчасове блокування облікових записів користувачів після певної кількості невдалих спроб входу.

**Багатофакторна автентифікація (MFA):** Впроваджуйте MFA, щоб ускладнити зловмисникам доступ до системи, навіть якщо вони отримають правильні облікові дані.

**Атаки методом розпилення паролів**

Атаки методом "розпилення паролів" використовують більш складний підхід до компрометації облікових записів. Замість того, щоб намагатися підібрати різні паролі до одного облікового запису, як у випадку атак "грубої сили", зловмисники намагаються підібрати один (часто використовуваний) пароль до кількох облікових записів. Цей метод мінімізує ризик виявлення, розподіляючи спроби на багато облікових записів і роблячи їх схожими на звичайні спроби входу користувача.

В атаці "Розпилення паролів" зловмисник зазвичай використовує список відомих імен користувачів і намагається підібрати невеликий набір часто використовуваних паролів до кожного імені користувача. Оскільки багато людей все ще використовують слабкі та поширені паролі, цей тип атаки може бути напрочуд ефективним.

Щоб захиститися від атак "розпилення паролів", дотримуйтесь цих рекомендацій:

**Навчіть користувачів вибирати паролі:** Розкажіть користувачам про важливість вибору надійних, унікальних паролів, які нелегко вгадати або знайти в словниках паролів.

**Відстежуйте незвичні шаблони входу:** Використовуйте інструменти моніторингу для виявлення незвичайних шаблонів входу, таких як численні успішні входи з певними (поширеними) паролями.

**Впроваджуйте багатофакторну автентифікацію (MFA):** Вимагайте від користувачів додаткового рівня автентифікації при вході в систему.

**DoS vs DDoS**

**DoS-атака (відмова в обслуговуванні)** — це тип кібератаки, коли зловмисник прагне зробити комп'ютер або мережевий ресурс недоступним для цільових користувачів, перевантажуючи цільову систему запитами, вона фактично стає недоступною через перевантаження сервера.

Деякі з найпоширеніших методів, що використовуються в DoS-атаках, включають:

**Наводнення** — зловмисник надсилає величезну кількість запитів до цільової системи, перевантажуючи її можливості відповідати на них і, врешті-решт, виводячи систему з ладу.

**Пінг смерті** — зловмисник надсилає великий, неправильно сформований ICMP-пакет на цільову систему, що може призвести до виходу системи з ладу.

**DDoS-атака (розподілена відмова в обслуговуванні)** схожа на DoS-атаку за своїми намірами, але для запуску атаки використовується декілька комп'ютерів або пристроїв (зазвичай скомпрометованих шкідливим програмним забезпеченням). Ці пристрої, які разом називають "ботнетом", надсилають величезну кількість запитів до цільової системи, що ще більше ускладнює нейтралізацію атаки та захист ресурсів.

Деякі поширені методи, що використовуються в DDoS-атаках, включають:

**UDP Flood** — DDoS-атака, яка надсилає численні пакети User Datagram Protocol (UDP) на цільову систему, споживаючи її ресурси і зрештою призводячи до збою.

**HTTP Flood** — DDoS-атака, яка генерує велику кількість HTTP-запитів до цільового сервера, що перевищує його пропускну здатність і призводить до уповільнення або збою в роботі.

Основні відмінності

**Масштаб:** У той час як DoS-атаки обмежені ресурсами одного зловмисника, в DDoS-атаках беруть участь кілька атакуючих пристроїв, що робить їх більш ефективними для перевантаження і виведення з ладу цільової системи.

**Пом'якшення наслідків:** DoS-атаки зазвичай можна пом'якшити за допомогою простих контрзаходів, але DDoS-атаки часто вимагають більш складних стратегій захисту через їх розподілений і скоординований характер.

**Атака "людина посередині"**

**Атака "людина посередині" (MITM)** відбувається, коли зловмисник перехоплює комунікацію між двома сторонами без їхньої згоди з метою підслуховування або маніпулювання даними, якими вони обмінюються. За допомогою цього методу зловмисники можуть викрасти конфіденційну інформацію, втрутитися в передані дані або видати себе за залучені сторони, щоб отримати несанкціонований контроль або доступ.

Деякі поширені типи MITM-атак включають:

**Підміна IP-адреси:** Зловмисник видає себе за IP-адресу іншого пристрою, щоб встановити з'єднання з жертвою.

**Підробка DNS:** Зловмисник змінює записи DNS, щоб перенаправити жертву на шкідливий веб-сайт замість потрібного.

**Підміна ARP:** Зловмисник змінює ARP-кеш жертви, щоб пов'язати її MAC-адресу (Media Access Control) з IP-адресою жертви, перенаправляючи мережевий трафік через пристрій зловмисника.

**Перехоплення SSL і TLS:** Зловмисник перехоплює і розшифровує зашифроване SSL/TLS-з'єднання між жертвою і веб-сервером, отримуючи доступ до конфіденційних даних.

Щоб зменшити ризик MITM-атак, розробники, адміністратори та користувачі повинні дотримуватися цих рекомендацій:

**Використовуйте HTTPS і шифрування:** Обов'язково шифруйте всі конфіденційні дані за допомогою захищених протоколів зв'язку, таких як HTTPS, SSL або TLS.

**Перевіряйте сертифікати:** Використовуйте центр сертифікації для перевірки цифрових сертифікатів для безпечного з'єднання.

**Захистіть DNS:** Використовуйте розширення безпеки DNS (DNSSEC) для забезпечення цілісності та автентичності записів DNS.

**Увімкніть мережеву сегрегацію:** Сегментуйте мережі та обмежуйте доступ між ними, щоб запобігти доступу зловмисників до конфіденційних даних або систем.

**Регулярно оновлюйте програмне забезпечення та мікропрограми:** Оновлюйте всі системи, програми та пристрої, щоб мінімізувати відомі вразливості.

**Evil Twin**

**Атака "злий двійник"** — це зловмисна тактика, яку використовують кіберзлочинці для обману користувачів, створюючи фальшиву бездротову точку доступу, яка імітує характеристики справжньої. Така підроблена точка доступу зазвичай має таке саме мережеве ім'я (SSID) та налаштування безпеки, як і справжня точка доступу, що ускладнює користувачам розрізнення між ними.

Зловмисник встановлює власне обладнання поблизу цільової бездротової мережі та налаштовує несанкціоновану точку доступу з тим самим SSID та налаштуваннями безпеки, що й у справжньої мережі. Нічого не підозрюючи, користувачі підключаються до несанкціонованої точки доступу, думаючи, що це легітимна мережа. Тепер зловмисник може перехоплювати, а в деяких випадках і змінювати дані користувача, що передаються мережею. Це може включати конфіденційну інформацію, таку як облікові дані для входу в систему, дані кредитної картки та особисті розмови.

Запобігання атакам злих двійників

**Використовуйте VPN:** Віртуальна приватна мережа (VPN) захищає ваші дані, шифруючи інформацію, що передається між вашим пристроєм та Інтернетом. Навіть якщо ви під'єднаєтеся до несанкціонованої точки доступу, ваші дані будуть захищені.

**Перевірте SSID:** Переконайтеся, що ви підключаєтеся до правильного SSID. Остерігайтеся мереж зі схожими назвами або тих, які не вимагають введення пароля.

**Увімкніть двофакторну автентифікацію:** Увімкніть двофакторну автентифікацію (2FA) для важливих облікових записів і служб. Це забезпечує додатковий рівень безпеки, ускладнюючи зловмисникам отримання несанкціонованого доступу.

**Оновлюйте програмне забезпечення:** Регулярно оновлюйте свої пристрої, програмне забезпечення та операційну систему, щоб захиститися від відомих вразливостей і загроз безпеці.

**Переповнення буфера**

**Переповнення буфера** — це поширений тип вразливості кібербезпеки, який виникає, коли програма записує або читає більше даних, ніж може вмістити буфер фіксованого розміру, в результаті чого дані перезаписують

інші дані в пам'яті. Переповнення може спричинити пошкодження даних і призвести до неочікуваної поведінки, наприклад, до збоїв програми або навіть до виконання шкідливого коду.

Уразливості переповнення буфера зазвичай виникають через:

**Недостатню перевірку вхідних даних:** Програма не перевіряє належним чином довжину введених даних перед тим, як записати їх у буфер.

**Помилками винесення за межі буфера:** Код використовує неправильну граничну умову, що призводить до запису одного зайвого байту за межі буферу.

**Цілочисельні переповнення:** Розмір буфера обчислюється з використанням цілочисельної змінної, яка є занадто малою для представлення необхідного розміру.

Зловмисники можуть використати уразливість переповнення буфера наступним чином:

Спричинити відмову в обслуговуванні (DoS).

Перезаписати критичні дані або керуючі структури, що призведе до непередбачуваної поведінки програми.

Виконати шкідливий код, що ставить під загрозу безпеку системи.

Для запобігання та усунення вразливостей, пов'язаних з переповненням буфера, можна застосувати наступні стратегії:

Виконуйте ретельну перевірку вхідних даних та санітарну обробку всіх вхідних даних у програмі.

Використовуйте безпечні API та бібліотеки, які перевіряють розмір даних перед тим, як копіювати їх у буфер.

Застосовуйте належні перевірки меж та використовуйте сучасні мови програмування з функціями захисту пам'яті.

Регулярно перевіряйте код на вразливості та проводьте аудит безпеки.

**Міжсайтовий скриптинг (XSS)**

**Міжсайтовий скриптинг (XSS)** — це тип вразливості кібербезпеки, який часто зустрічається у веб-додатках. Вона виникає, коли зловмисник впроваджує шкідливі скрипти на веб-сторінки, які переглядають інші користувачі. Ці скрипти можуть бути використані для крадіжки конфіденційної інформації, наприклад, облікових даних користувача або конфіденційних даних. XSS-уразливості можуть призвести до різних наслідків, таких як заволодіння акаунтами, фішингові атаки та інші зловмисні дії.

Існує три основні типи XSS-атак:

Збережені XSS-атаки: У цьому типі шкідливий скрипт зберігається на веб-сервері, як правило, через поля введення користувача, такі як коментарі або повідомлення. Коли інші користувачі відвідують уражену сторінку, їхні браузери виконують шкідливий сценарій.

Відображені XSS-атаки: У цьому випадку зловмисник надсилає шкідливу URL-адресу, що містить скрипт, нічого не підозрюючим користувачам. Коли вони переходять за посиланням, їхні браузери запускають шкідливий скрипт, який може викрасти конфіденційну інформацію або виконати несанкціоновані дії.

XSS-атаки на основі DOM-файлів: У цих випадках зловмисник маніпулює об'єктною моделлю документа (DOM) веб-сторінки в браузері користувача, що призводить до виконання шкідливого сценарію. Цей метод не передбачає прямої взаємодії з веб-сервером.

Щоб захистити свої веб-додатки від XSS-атак, розгляньте можливість впровадження наступних найкращих практик:

**Перевірка вхідних даних:** Перевіряйте та дезінфікуйте вхідні дані користувача, щоб переконатися, що вони містять лише прийнятні дані. Відкидайте будь-які дані, що містять шкідливі коди або неочікувані символи.

**Кодування вихідних даних:** Правильно кодуйте вихідні дані програми, щоб спеціальні символи відображалися так, щоб запобігти виконанню скриптів.

**Регулярне тестування безпеки:** Регулярно проводьте аудит безпеки та тести на проникнення, щоб виявити та виправити будь-які вразливості у ваших веб-додатках.

**SQL-ін'єкція**

**SQL-ін'єкція** — це тип кібератаки, спрямований на веб-додатки та бази даних. Цей метод використовує вразливості в коді програми, впроваджуючи шкідливі SQL-запити і використовуючи їх для отримання несанкціонованого доступу або маніпулювання даними в базі даних. Зловмисники можуть використовувати цю техніку для отримання, модифікації, видалення або навіть додавання даних до бази даних без належного дозволу.

SQL-ін'єкція працює шляхом виявлення полів введення у веб-додатку, таких як текстові поля або параметри URL-адреси, і перевірки того, чи є ці поля вразливими для впровадження SQL-коду. Коли зловмисник виявляє вразливе поле введення, він вводить SQL-код, щоб маніпулювати основним SQL-запитом або виконати додаткові запити до бази даних.

Наприклад, розглянемо веб-додаток, який дозволяє користувачам увійти в систему, надавши ім'я користувача та пароль. Для автентифікації користувача програма може використовувати наступний SQL-запит:

SELECT \* FROM users WHERE username = '$username' AND password = '$password'

У цьому випадку $username і $password замінюються значеннями, наданими користувачем. Якщо зловмисник введе наступне значення для поля імені користувача, він зможе маніпулювати запитом, щоб обійти перевірку пароля:

' OR 1=1 --

Результуючий запит буде виглядати наступним чином:

SELECT \* FROM users WHERE username = '' OR 1=1 -- ' AND password = '$password'

Оскільки 1=1 завжди істинно, запит повертає результат, і зловмисник отримує несанкціонований доступ.

Щоб захистити свої веб-додатки від атак SQL Injection, ви повинні:

**Використовувати параметризовані запити та підготовлені оператори:** Ці методи відокремлюють введення користувача від SQL-запиту, що ускладнює зловмиснику впровадження шкідливого коду. Більшість сучасних фреймворків веб-розробки та бібліотек баз даних підтримують параметризовані запити та підготовлені оператори.

**Перевіряйте дані, введені користувачем:** Завжди перевіряйте і дезінфікуйте дані, введені користувачем, перш ніж включати їх в SQL-запит. Використовуйте строгі типи даних і перевіряйте введені дані на відповідність попередньо визначеним шаблонам або діапазонам значень.

**Обмеження прав доступу до бази даних:** Обмежте привілеї облікових записів баз даних, що використовуються вашими веб-додатками. Це обмежить потенційну шкоду, якщо зловмиснику вдасться виконати атаку SQL-ін'єкції.

**Підробка міжсайтових запитів (CSRF)**

**Підробка міжсайтових запитів (CSRF)** — це тип атаки, який використовує довіру браузера користувача до веб-додатку. Вона змушує браузер користувача виконувати небажані дії у веб-додатку, в якому користувач наразі автентифікований.

Користувач входить у вразливий веб-додаток. Веб-додаток повертає браузеру користувача файл cookie, який вказує на те, що користувач пройшов аутентифікацію. Зловмисник створює шкідливе посилання або вбудовує шкідливий HTML/JavaScript-код на інший веб-сайт.

Користувач, все ще автентифікований у веб-додатку, переходить на веб-сайт зловмисника, де запускається шкідливий код. Код зловмисника надсилає запит до цільового веб-застосунку, використовуючи аутентифікований користувачем файл cookie. Вразливий веб-додаток виконує шкідливу дію так, ніби запит надійшов від користувача.

CSRF-атаки можуть призвести до виконання несанкціонованих дій від імені користувача, часто без його відома. Наслідки можуть включати несанкціоновані:

Модифікації даних

підвищення привілеїв

захоплення облікових записів

Кілька методів, які допоможуть запобігти CSRF-атакам:

**Використовуйте токени CSRF:** Використовуйте унікальний, непередбачуваний токен у кожному чутливому запиті (наприклад, при заповненні форм), щоб переконатися, що запит походить з одного домену.

**Подвійне відправлення файлів cookie:** Створювати унікальний маркер для кожного сеансу та включати його як приховане значення у форми, а потім перевіряти його за допомогою відповідного сеансового файлу cookie.

**Односайтові файли cookie:** Використовуйте атрибут SameSite у файлах cookie, щоб вказати браузеру надсилати файл cookie лише тоді, коли запит надходить з того самого домену.

**Обмеження CORS:** Обмежте спільне використання ресурсів різного походження (Cross-Origin Resource Sharing, CORS) довіреними доменами, щоб запобігти несанкціонованому спілкуванню між сайтами різного походження.

**Обхід каталогів**

**Обхід каталогів**, також відомий як обхід шляху, — це тип кібератаки, який дозволяє зловмиснику отримати доступ до файлів і каталогів з обмеженим доступом на сервері, зазвичай з метою отримання конфіденційної інформації. Ця вразливість виникає, коли вхідні дані користувача не перевіряються належним чином, і зловмисник може маніпулювати ними, щоб обійти структуру каталогів на сервері.

В атаці обходу каталогу зловмисник намагається використати поле введення (наприклад, форму завантаження файлу або зображення, параметри URL-адреси тощо), яке приймає шлях до файлу в якості вхідних даних. Вводячи спеціально створені дані, зловмисник може маніпулювати сервером, щоб отримати доступ до несанкціонованих файлів і каталогів.

Наприклад, розглянемо веб-додаток, який дозволяє користувачам переглядати вміст певного файлу, вказавши шлях до нього через параметр URL, наприклад https://www.example.com/file.php?path=/user/documents/report.pdf У цьому випадку зловмисник може маніпулювати параметром шляху для обходу каталогів сервера, наприклад, так https://www.example.com/file.php?path=../../../../etc/passwd Якщо сервер не перевіряє належним чином вхідні дані, він може розкрити вміст файлу /etc/passwd, який містить конфіденційну інформацію про користувачів системи.

Існує декілька методів запобігання атакам на обхід каталогу:

**Перевірка вхідних даних:** Переконайтеся, що дані, які вводяться користувачем, суворо перевіряються та очищаються. Наприклад, можна перевірити наявність спеціальних символів (наприклад, "...", "/", "") і заборонити їх використання, якщо вони будуть знайдені.

**Контроль доступу:** Впровадьте належні механізми контролю доступу, щоб запобігти несанкціонованому доступу до файлів і каталогів. Наприклад, використовуйте білий список, щоб визначити, до яких файлів і каталогів користувачеві дозволено доступ.

**Найменші привілеї:** Практикуйте принцип найменших привілеїв, гарантуючи, що програма працює лише з необхідними для її роботи дозволами. Це може мінімізувати потенційний вплив атаки обходу каталогів.

**Лекція 3.**

**Принципи захисту програм від несанкціонованого доступу.**

**Загальні поняття**

Під **несанкціонованим доступом (НСД) – unauthorized access –** розумітимемо нелегальні дії щодо використання, зміни та знищення виконуваних модулів.

**Системою захисту від несанкціонованого доступу (system of protection against unauthorized access)** називають комплекс програмних засобів, що забезпечують ускладнення або заборону нелегального розповсюдження, використання і/або зміну програмних продуктів.

**Надійність системи захисту (reliability of protection)** – це здатність протистояти спробам проникнення в алгоритм її роботи і обходу механізмів захисту.

Під **зламом програми (breaking program)** матимемо на увазі порушення функціональності об'єктів захисту програмного забезпечення (адже їх може бути декілька).

**Дослідження роботи програмних продуктів**

Для аналізу об'єктів захисту використовуються два основні види засобів: **налагоджувачі**, які дозволяють контролювати виконання програм і таким чином прослідкувати весь алгоритм на практиці і **дизасемблери**, що дозволяють одержувати асемблерний лістинг програми для його подальшого статичного вивчення. І тому, якою би мовою ми не писали, які б способи захисту не використовували, все, зрештою, перетвориться на команди процесора.

Перед тим, як розглянути способи захисту програм від НСД, необхідно з’ясувати, яким же чином здійснюється дослідження роботи програмних продуктів.

Всі засоби дослідження роботи програмних продуктів, у тому числі і захищених, можна розбити на чотири класи:

1. Статичні засоби, які оперують початковим кодом програми як даними і будують її алгоритм без виконання. Ці засоби є більш універсальними в тому значенні, що теоретично можуть одержати алгоритм усієї програми, у тому числі і тих блоків, які ніколи не отримують управління.

2. Динамічні засоби, які вивчають програму, інтерпретуючи її в реальному або віртуальному обчислювальному середовищі. Ці засоби можуть будувати алгоритм програми тільки на підставі конкретної її траси, одержаної при певних вхідних даних. Тому задача отримання повного алгоритму програми в цьому випадку еквівалентна побудові вичерпного набору текстів для підтвердження правильності програми, що практично неможливо, і взагалі при динамічному дослідженні можна говорити тільки про побудову деякої частини алгоритму.

3. Синтаксичні методи. До цієї групи відносяться методи, що ґрунтуються тільки на результатах лексичного, синтаксичного і семантичного аналізу програми.

4. Статистичні методи. Статистичні методи використовують інформацію, зібрану в результаті значної кількості запусків програми на великій кількості наборів вхідних даних.

Розглянемо конкретні інструменти, що використовуються зламниками для несанкціонованого дослідження захищених програм.

Налагоджувачі і дизасемблери. Традиційно обидва ці типи інструментів використовуються в парі, оскільки дизасемблер видає лише "чистий код". Але сучасні дизасемблери здатні також розпізнати виклики стандартних функцій, виділити локальні змінні в процедурах і надати інший

подібний сервіс. Користуючись дизасемблером, можна лише здогадуватися про те, які дані одержує та або інша функція як параметри і що вони 8 означають. Щоб з'ясувати це, найчастіше потрібне вивчення якщо не всієї програми, то досить значної її частини.

Налагоджувачі виконують принципово інші функції, вони дозволяють аналізувати код в процесі його роботи, відстежувати і змінювати стан регістрів і стека, правити код "на льоту" – загалом, спостерігати за роботою програми і навіть активно в неї втручатися. Зворотним боком цього є "неінтелектуальність" багатьох налагоджувачів – їх природжені здібності до аналізу коду рідко виходять за межі визначення напряму переходу.

Утиліти для розпакування та дампінга процесів. Дизасемблювати запаковану або зашифровану програму напряму неможливо. Але якщо дуже хочеться одержати хоч якийсь лістинг, можна спробувати витягнути з пам'яті комп'ютера знімок (дамп) програми у момент її роботи. Цей дамп вже можна більш менш успішно дизасемблювати. Більш того, на основі дампу можна відтворити виконуваний файл програми, причому цей файл успішно завантажуватиметься, запускатиметься і працюватиме.

Патчери, які дозволяють створити невеликий виконуваний файл, що автоматично вносить зміни в оригінальний файл програми або в код цієї програми безпосередньо в пам'яті.

API-шпигуни і інші утиліти моніторингу. Дуже часто буває необхідно знати, які саме дії виконує та або інша програма, звідки читає і куди записує дані, які стандартні функції і з якими параметрами вона викликає. Одержати ці відомості якраз і допомагають утиліти моніторингу. Такі утиліти діляться на дві великі групи:

- ті, які відстежують сам факт виникнення яких-небудь подій;

- ті, які дозволяють виявити один або декілька специфічних типів змін, що відбулися в системі за певний проміжок часу.

До першої групи відносяться всілякі API-шпигуни, що перехоплюють виклики системних функцій (деякі API-шпигуни вміють перехоплювати не

тільки системні функції), перехоплювачі системних повідомлень і багато інших. Ці утиліти, як правило, надають докладну інформацію про відстежувані події.

Друга група представлена програмами, що створюють знімки реєстру, жорсткого диска, системних файлів і т. п. Ці програми дозволяють порівняти стан комп'ютера до і після якоїсь події, побудувати список відмінностей між станами і на основі цього списку робити певні висновки.

Вбудовування захисних механізмів у захищувані програми можна виконати такими основними способами:

- вставкою перевірочного механізму в початковий код на етапі розробки і налагодження програмного продукту – вбудований захист;

- вставкою фрагмента перевірочного коду у виконуваний файл – навісний захист;

- перетворенням виконуваного файлу до невиконуваного вигляду (шифрування, архівація з невідомим параметром і т. д.) і застосуванням для завантаження не засобів операційного середовища, а деякої програми, в тілі якої і здійснюються необхідні перевірки;

- комбінуванням вказаних методів.

До процесу і результату вбудовування захисних механізмів можна пред'явити такі вимоги:

- висока трудомісткість виявлення захисного фрагмента при статичному дослідженні (особливо актуальна при вбудовуванні в початковий код програмного продукту);

- висока трудомісткість виявлення захисного фрагмента при динамічному дослідженні (налагоджуванні і трасуванні за зовнішніми подіями);

- висока трудомісткість обходу або редагування захисного фрагменту.

Для захисту програмного забезпечення від дослідження необхідно застосовувати методи захисту від дослідження файлу з виконуваним кодом програми, що зберігається на зовнішньому носії, а також використати методи

захисту виконуваного коду, завантажуваного в оперативну пам'ять для виконання цієї програми.

У першому випадку захист може бути заснований, наприклад, на шифруванні конфіденційної частини програми, а в другому – на блокуванні доступу до виконуваного коду програми в оперативній пам'яті з боку налагоджувачів.

Крім того, перед завершенням роботи захищуваної програми повинен онулюватися весь її код в оперативній пам'яті. Це запобігатиме можливості несанкціонованого копіювання з оперативної пам'яті дешифрованого виконуваного коду після виконання програми, що захищається. Таким чином, програма, що захищається від несанкціонованого дослідження, повинна складатися з основних трьох компонент.

1. Ініціалізація, яка повинна забезпечувати виконання таких функцій:

- збереження параметрів операційного середовища функціонування (векторів переривань, вмісту регістрів процесора і т. д.);

- заборону всіх внутрішніх і зовнішніх переривань, обробка яких не може бути запротокольована в захищуваній програмі;

- завантаження в оперативну пам'ять і дешифрування (якщо захист відбувається за допомогою шифрування) коду конфіденційної (контрольної, секретної) частини програми;

- передачу управління конфіденційній частини програми.

2. Конфіденційна (секретна) частина програми, призначена для виконання основних цільових функцій і повинна бути захищена, наприклад, шифруванням для попередження внесення в неї програмної закладки.

3. Деініціалізатор (деструктор), який після виконання конфіденційної частини програми повинен виконати такі дії:

- онулення конфіденційного коду програми в оперативній пам'яті;

- відновлення параметрів операційної системи (векторів переривань, вмісту регістрів процесора і т. д.), які були встановлені до заборони неконтрольованих переривань;

- виконання операцій, які неможливо було виконати при забороні неконтрольованих переривань;

- звільнення всіх незадіяних ресурсів комп'ютера і завершення роботи програми.

Для більшої надійності ініціалізація може бути частково зашифрованою і у міру виконання дешифрувати саму себе. Дешифруватися у міру виконання може і конфіденційна частина програми. Таке дешифрування називається динамічним дешифруванням виконуваного коду. У цьому випадку чергові ділянки програм перед безпосереднім виконанням розшифровуються, а після виконання відразу знищуються.

Для підвищення ефективності захисту програм від дослідження необхідним є внесення в програму додаткових функцій безпеки, направлених на захист від трасування. До таких функцій можна віднести:

- періодичний підрахунок контрольної суми області оперативної пам'яті, зайнятої початковим захищуваним кодом;

- порівняння поточної контрольної суми із заздалегідь сформованою еталонною і вживання необхідних заходів у разі неспівпадіння;

- перевірку кількості оперативної пам'яті, яку займає захищувана програма, порівняння з об'ємом, до якого програма адаптована, і вживання необхідних заходів у разі невідповідності;

- контроль часу виконання окремих ділянок програми;

- блокування клавіатури на час відпрацювання особливо критичних алгоритмів.

Захист програмного забезпечення від несанкціонованої декомпіляції є важливою складовою безпеки програм та захисту від розголошення конфіденційної інформації. Декомпіляція — це процес аналізу вихідного коду програми для отримання доступу до її логіки, алгоритмів та інших деталей. Кілька стратегій та методів для захисту від декомпіляції:

Шифрування виконуваного коду. Використовуйте спеціальні інструменти та технології, що дозволяють шифрувати виконуваний код

програми. Це робить декомпіляцію набагато складнішою, оскільки аналіз відкритого тексту буде неможливим.

Обфускація коду. Обфускація - це процес перетворення вихідного коду програми в менш зрозумілу і набагато складнішу версію, зберігаючи при цьому її функціональність. Це робить процес декомпіляції менш ефективним, оскільки структура та ім'я змінних стають важкозрозумілими.

Підписка та верифікація цифрових підписів. Підписуйте вашу програму цифровим підписом та використовуйте механізми верифікації підпису для перевірки її цілісності та автентичності перед виконанням.

Контроль доступу до ресурсів. Обмежуйте доступ до ресурсів та функцій програми за допомогою прав доступу. Це допоможе уникнути витоку конфіденційних даних чи атак, пов'язаних з некоректним використанням ресурсів.

Перевірка цілісності та антивірусна захист. Додайте в програму механізми перевірки цілісності, які будуть виявляти будь-які зміни в виконуваному коді. Використовуйте антивірусне програмне забезпечення для виявлення можливих загроз.

**Хуки**

В операційній системі Microsoft Windows хуком називається механізм перехоплення особливою функцією подій (таких, наприклад, як повідомлення від маніпулятора миші або клавіатури) до того, як вони дійдуть до програми. Ця функція може потім реагувати на події, а в деяких випадках змінювати або відміняти їх. З цього випливає, що підключення хуків призводить до збільшення обсягу роботи системи, тобто робота системи сповільнюється. Отже, хуки повинні встановлюватися в самих крайніх випадках і видалятися відразу ж після того, як потреба в них відпала.

Якщо в системі виникає повідомлення, асоційоване з якимось хуком, то Windows передає його на обробку першому хуку, потім другому й т.д. І лише

після того, як всі хуки відпрацюють, повідомлення надійде за своїм прямим призначенням. Функції, які одержують повідомлення про події, називаються фільтруючими функціями і відрізняються за типами перехоплених ними подій. Наприклад, це може бути фільтруюча функція для перехоплення всіх подій миші або клавіатури. Щоб Windows змогла викликати функцію фільтр, ця функція повинна бути встановлена, тобто, прикріплена до хука (наприклад, до клавіатурного хука). Прикріплення однієї або декількох фільтруючих функцій до будь-якого хука називається встановленням хука. Якщо до одного хука прикріплено декілька фільтруючих функцій, Windows реалізує чергу функцій, причому функція, прикріплена останньою, виявляється на початку черги, а найперша функція – у її кінці. Якщо до хука прикріплена одна або більше функцій-фільтрів і відбувається подія, що приводить до спрацьовування хука, Windows викликає першу функцію з черги функцій-фільтрів. Ця дія називається викликом хука.

Для встановлення й доступу до фільтруючих функцій Windows програм використовують функції, описані в заголовному файлі "winuser.h": SetWindowsHookEx(), CallNextHookEx() і UnhookWindowsHookEx().

Таким чином, хуки у Windows — це механізм, який дозволяє програмам і системі взаємодіяти одна з одною та реагувати на різні події і подійні винятки. Вони використовуються для різних цілей і мають різні функції:

Моніторинг і перехоплення подій: Хуки можуть бути використані для відслідковування і перехоплення різних подій, таких як натискання клавіш, миші, зміни вікон та інші події в системі. Це корисно для розробників програм, які вимагають додаткового управління інтерфейсом користувача або для створення системних утиліт, які моніторять та аналізують дії користувача.

Модифікація і обробка подій: Хуки дозволяють програмам модифікувати або обробляти події до того, як вони досягнуть свого

призначення. Наприклад, ви можете створити хук для перехоплення клавіші "Enter" і змінювати її функціональність.

Спільний доступ до ресурсів: Хуки дозволяють більшій кількості програм спільно використовувати ресурси системи. Наприклад, додатки для запису кліпборда можуть використовувати хуки, щоб отримувати доступ до тексту, який копіюється та вставляється користувачем.

Захист від шкідливих програм: Хуки також використовуються для захисту від шкідливих програм та вірусів, які можуть спробувати зламати або модифікувати системні функції. Антивіруси та антишпигуни можуть встановлювати хуки для моніторингу та блокування підозрілих дій.

Важливо відзначити, що невірно налаштовані або зловживані хуки можуть призвести до проблем в роботі системи та програм, тому вони повинні використовуватися обережно і тільки відповідно до конкретних завдань.

**Списки контролю доступу**

Список контролю доступу (ACL) містить правила, які надають або забороняють доступ до певних цифрових середовищ. Існує два типи ACL:

- Файлові ACL ━ фільтрують доступ до файлів та/або каталогів. Файлові ACL вказують операційним системам, які користувачі можуть отримати доступ до системи та які привілеї їм дозволено.

- Мережеві ACL ━ фільтрують доступ до мережі. Мережеві ACL вказують маршрутизаторам і комутаторам, який тип трафіку може отримати доступ до мережі, і яка активність дозволена.

ACL файлової системи - це таблиця, яка інформує операційну систему комп'ютера про привілеї доступу користувача до системних об'єктів, зокрема до окремого файлу або каталогу файлів. Кожен об'єкт має властивість безпеки, яка пов'язує його зі списком контролю доступу. У списку є запис для кожного користувача з правами доступу до системи.

Типові привілеї включають право на читання окремого файлу (або всіх файлів) у каталозі, виконання файлу або запис до файлу чи файлів. Операційні системи, які використовують ACL, включають, наприклад, Microsoft Windows та системи на базі UNIX.

Коли користувач запитує об'єкт у моделі безпеки на основі ACL, операційна система вивчає ACL на предмет наявності відповідного запису і визначає, чи дозволена запитувана операція.

Список керування доступом містить елементи, які називаються Access-Control Entries (ACE). У ОС Windows кожен елемент ACE містить наступну інформацію:

 ідентифікатор безпеки суб’єкта, якому дозволено або заборонено доступ до об'єкту, що охороняється;

 маску доступу;

 прапор, який визначає тип елемента ACE;

 прапори, які визначають властивості спадкування даного елемента ACE;

 прапори, які керують аудитом доступу до охоронюваного об'єкту.

Маска доступу являє собою 32-бітове поле, в якому кожен біт специфікує певне право доступу. Кожне право доступу являє собою набір операцій, які суб'єкту дозволяється або забороняється виконувати над об'єктом, що охороняється.

Елементи списку можуть бути двох типів:

 ACCESS\_ALLOWED\_ACE - елемент, що дозволяє доступ;

 ACCESS\_DENIED\_ACE - елемент, що забороняє доступ.

Під час доступу суб'єкта до об'єкта система розрізняє випадки відсутності списку DACL і наявність порожнього списку DACL, тобто списку DACL, який не містить елементів. У першому випадку доступ до об'єкта відкритий для всіх суб'єктів, а в другому - доступ до об'єкта закритий для всіх суб'єктів.

Як для створення нових списків управління доступом, так і для модифікації наявних списків управління доступом призначена функція

SetEntriesInAcl. Пам'ять під новий список управління доступом резервує система. Тому після використання нового списку керування доступом цю пам'ять потрібно звільнити шляхом виклику функції LocalFree.

Важливе зауваження щодо порядку елементів у новому списку управління доступом, який створює функція SetEntriesInAcl. Елементи типу ACCESS\_DENIED\_ACE, тобто такі, що забороняють доступ до об'єкта, включаються на початок нового списку управління доступом. Елементи типу ACCESS\_ALLOWED\_ACE, тобто такі, що забороняють доступ до об'єкта, включають у новий список керування доступом після всіх елементів, що забороняють доступ, але перед усіма елементами, що дозволяють доступ, зі старого списку керування доступом до об'єкту, що охороняється.

Додавання елементів АСЕ в розмежувальні списки ACL здійснюється за допомогою двох функцій - AddAccessAllowedAce і AddAccessDenieddAce.

На рис. 1 показано список DACL, який забороняє доступ одному користувачеві і надає доступ до двох груп. Члени групи A отримують права доступу на читання, запис і виконання, накопичуючи права, дозволені групі A, і права, дозволені всім. Винятком є Ендрю, який не має доступу до ACE, якому заборонено доступ, незважаючи на те, що він є членом групи "Всі".

Рисунок 1 – DACL

Якщо список управління доступом (DACL), що належить дескриптору безпеки об'єкта, має значення NULL, створюється список DACL зі значенням NULL. DACL зі значенням NULL надає повний доступ будь-якому користувачеві, який його запитує; звичайна перевірка безпеки не виконується щодо об'єкта. Не слід плутати NULL DACL з порожнім DACL. Порожній DACL - це правильно виділений та ініціалізований DACL, який не містить записів управління доступом (ACEs). Порожній DACL не надає доступу до об'єкта, якому він призначений.

Тепер розглянемо принципи проектування безпечних програм.

**Принцип найменших привілеїв**

Принцип найменших привілеїв - це принцип проектування безпеки, який вимагає, щоб користувачам було надано мінімум дозволів, необхідних для виконання їхніх завдань. Тому цей принцип також іноді називають принципом найменших повноважень. Його часто називають одним з найважливіших принципів побудови системи безпеки. Він допомагає зменшити ризик несанкціонованого доступу до конфіденційних даних і систем. Надаючи користувачам лише ті дозволи, які необхідні для виконання їхніх завдань, зменшується ймовірність того, що вони зможуть отримати доступ до даних або систем, до яких вони не повинні мати доступу.

Принцип найменших привілеїв передбачає регулярний перегляд привілеїв доступу користувачів, щоб переконатися, що вони мають мінімальний доступ до конфіденційних даних і критично важливих систем.

Наприклад, обліковий запис користувача, створений виключно для створення резервних копій, не потребує встановлення програмного забезпечення: отже, він має права лише на запуск резервного копіювання та пов'язаних із ним програм. Будь-які інші привілеї, такі як встановлення нового програмного забезпечення, блокуються. Цей принцип застосовується і

до користувача персонального комп'ютера, який зазвичай виконує роботу в звичайному обліковому записі користувача, а привілейований, захищений паролем обліковий запис відкриває лише тоді, коли цього абсолютно вимагає ситуація.

Переваги принципу полягають у наступному:

— Інтелектуальна безпека. Коли код обмежений в обсязі змін, які він може вносити в систему, легше протестувати його можливі дії та взаємодію з іншими програмами, орієнтованими на безпеку. На практиці, наприклад, програми, що працюють з обмеженими правами, не матимуть доступу до виконання операцій, які можуть вивести з ладу машину або негативно вплинути на інші програми, що працюють у тій самій системі.

— Краща безпека системи. Коли код обмежений у загальносистемних діях, які він може виконувати, вразливості в одній програмі не можуть бути використані для експлуатації решти комп'ютера.

— Простота розгортання. Загалом, чим менше привілеїв вимагає програма, тим легше її розгортати в більшому середовищі. Зазвичай це є наслідком перших двох переваг: програми, які встановлюють драйвери пристроїв або потребують підвищених привілеїв безпеки, зазвичай мають додаткові кроки для розгортання. Наприклад, у Windows рішення без драйверів пристроїв можна запустити безпосередньо без інсталяції, тоді як драйвери пристроїв потрібно встановлювати окремо за допомогою служби інсталятора Windows, щоб надати драйверу підвищені привілеї.

Декілька прикладів та сценаріїв, де принцип найменших привілеїв застосовується для забезпечення безпеки програм та систем:

Операційні системи. ОС надає користувачам різні ролі та рівні доступу. Наприклад, адміністратор має найвищий рівень привілеїв, тоді як звичайні користувачі мають обмежені права. Це допомагає запобігти несанкціонованому доступу до критичних системних ресурсів.

Бази даних. В базах даних ролі і доступність прав на рівні таблиць та об'єктів дозволяють обмежити доступ до конфіденційних даних. Наприклад,

різні користувачі можуть мати різні рівні доступу до таблиць (читання, запис, видалення) в залежності від їх ролей.

Веб-застосунки. У веб-застосунках принцип може бути застосований для обмеження доступу до адміністративних панелей. Адміністратори мають доступ до всіх функцій, тоді як звичайні користувачі мають обмежені права доступу.

Офісні програми. У текстових редакторах та архіваторах принцип може бути використаний для обмеження доступу до операцій з файлами. Наприклад, звичайний користувач може читати та редагувати файли, але не може видаляти або перейменовувати їх.

Серверні додатки. У серверних додатках принцип може бути використаний для обмеження доступу до ресурсів на основі ролей та прав доступу. Наприклад, веб-сервер може обмежувати доступ до окремих URL або API-точок залежно від ролі користувача.

Операції з файлами. При роботі з файловою системою принцип може бути використаний для обмеження прав доступу до файлів та папок. Наприклад, в операційній системі Linux, права доступу до файлів і каталогів визначаються на рівні власника, групи та інших користувачів.

Виконання принципу найменших привілеїв в програмному коді вимагає від розробників ретельно розроблювати програму, забезпечуючи, що доступ до ресурсів та функціоналу надається лише тим користувачам чи частинам програми, які дійсно потребують цього доступу.

Декілька кроків та практичних порад щодо виконання принципу найменших привілеїв в програмному коді:

Визначення ролей та прав доступу. Спочатку визначте різні ролі або категорії користувачів в вашій програмі. Розгляньте, які завдання вони виконують та які ресурси їм потрібні. Наприклад, адміністратори, модератори, звичайні користувачі.

Обмеження доступу на рівні ролей. Розробіть механізм, який обмежує доступ до функціоналу та ресурсів на основі ролей. Використовуйте умовні

перевірки та умовні дозволи для забезпечення доступу лише для конкретних ролей.

Мінімізація прав доступу. Надавайте користувачам лише ті права доступу, які їм дійсно потрібні для виконання їх завдань. Наприклад, якщо користувачеві потрібно лише читати дані з бази даних, не надавайте йому права на редагування чи видалення записів.

Використання авторизації. Забезпечте механізм авторизації, який перевіряє, чи має конкретний користувач право виконувати певну операцію. Використовуйте авторизацію для обмеження доступу до окремих функцій.

Аудит та журналювання. Ведіть журнал дій користувачів та реакцію системи на їхні дії. Це допоможе виявити намагання несанкціонованого доступу та вчасно реагувати на них.

Наприклад: якщо ви розробляєте систему, яка зберігає конфіденційну фінансову інформацію про клієнтів, рекомендується обмежити коло осіб, які можуть отримати доступ до цієї інформації. Людині, яка відповідає на телефонні дзвінки та призначає зустрічі, ймовірно, не потрібен доступ до всієї конфіденційної інформації. З іншого боку, менеджер по роботі з клієнтами, ймовірно, потребує доступу до цієї інформації. Ключовим моментом є те, що менеджер по роботі з клієнтами не має доступу до інформації з акаунтів, якими він не керує.

Переконавшись, що акаунти мають лише ті привілеї, які необхідні для виконання їхньої роботи, ви гарантуєте, що якщо зловмисник скомпрометує акаунт, ви мінімізуєте інформацію, до якої він зможе отримати доступ. Це обмежує шкоду від атаки.

**Принцип розподілу обов'язків** випливає з принципу найменших привілеїв. Ідея розподілу обов'язків полягає в тому, що жодна роль не повинна мати занадто багато повноважень. Це відрізняється від концепції найменших привілеїв. Якщо перша зосереджується на тому, щоб люди мали лише ті привілеї, які їм потрібні для виконання своєї роботи, то тут ідеться про те, щоб їхня робота не була надто великою. Коли хтось виконує занадто

велику роботу, ми знову повертаємося до того, що йому потрібно багато дозволів, щоб виконувати цю роботу. Крім того, коли на когось покладено багато обов'язків, це означає, що він схильний до прийняття неправильних рішень.

У нашій гіпотетичній фінансовій системі ми б не хотіли, щоб людина, яка відповідає за продажі, також могла затверджувати знижки. Ця людина була б зацікавлена у знижці на програмне забезпечення і могла б приймати неправильні рішення щодо знижок, щоб збільшити кількість продажів. Натомість, хтось інший, наприклад, менеджер, повинен затверджувати знижку до завершення розпродажу.

**Мінімізація атакованої площини**

Мінімізація атакованої площини (Attack Surface Reduction) - це стратегія забезпечення безпеки програм та систем, яка спрямована на зменшення кількості точок, через які можуть відбутися атаки на систему. Головною метою цієї стратегії є зниження можливостей для потенційних загроз і зменшення вразливостей системи.

Деякі ключові аспекти мінімізації атакованої площини та практичні поради щодо її виконання:

Зменшення функціоналу. Видалення або вимкнення непотрібних функцій та сервісів, які не використовуються. Зменшення обсягу функціоналу до мінімуму, необхідного для роботи програми.

Заборона зайвого доступу. Вимкнення чи обмеження віддаленого доступу до системи або сервісів, якщо це необхідно. Обмеження доступу до системних ресурсів, таких як файли, каталоги та порти, лише для необхідних користувачів і процесів.

Валідація та фільтрація вводу. Перевірка введеного користувачем вводу на наявність шкідливого коду або некоректних даних. Використання

механізмів фільтрації, таких як санітарізація та екранування, для очищення введеного вводу перед обробкою.

Моніторинг та журналювання. Постійний моніторинг діяльності системи та журналювання подій для виявлення незвичайних або підозрілих активностей. Автоматичне сповіщення про інциденти та загрози.

Постійне оновлення. Постійно оновлюйте програми, бібліотеки та системний програмний засіб, щоб виправити виявлені вразливості.

Відсікання зовнішньої взаємодії. Зменшуйте кількість точок взаємодії зі зовнішніми системами, якщо це можливо. Вимикайте або блокуйте непотрібні мережеві порти та служби.

Відсікання та ізоляція процесів. Використовуйте ізольовані контейнери або віртуальні машини для виділення та ізоляції різних компонентів системи. Зменшення можливості розповсюдження атаки з одного компонента на інший.

**Принцип відкритого проектування**

Принцип відкритого проектування говорить, що безпека вашої системи не повинна залежати від секретності вашої реалізації. Це особливо важливий принцип для таких концепцій безпеки, як криптографічні реалізації. Добре розроблені криптографічні реалізації публікуються у відкритому доступі.

Те ж саме має бути і з будь-яким програмним забезпеченням. Наприклад, система, яка не дає безпечного збою, як раніше, може покладатися на ідею, що "ніхто ніколи не здогадається про це". Хоча малоймовірно, що зловмисник може зробити висновок, що баг надає доступ до системи, це не є неможливим. Більше того, якщо він коли-небудь отримає доступ до вашого вихідного коду, він швидко це з'ясує. Натомість, дотримуйтесь принципів безпечного проектування, щоб забезпечити безпеку системи, незалежно від того, чи отримає хтось зловмисний доступ до вашого коду.

**Безпека через невідомість** схожа на принцип відкритого дизайну. Уявіть собі програмне забезпечення, яке має жорстко закодовану секретну комбінацію імені користувача та пароля. Після автентифікації цей обліковий запис має повний доступ до всіх облікових записів у системі. Безпека цієї системи ґрунтується на тому, що облікові дані цього облікового запису залишаються в таємниці. З часом все більше користувачів отримуватимуть доступ до цього облікового запису. Деякі з них звільняться з компанії, але не забудуть пароль. На цьому етапі безпека вашого додатку залежить від доброї волі цих співробітників.

Це правда, що безпека кожного додатку базується на секретах. Паролі є важливою частиною більшості схем автентифікації. З цим нічого не поробиш. Однак, кращий дизайн системи - це той, де облікового запису з повним доступом взагалі не існує. Якщо вам потрібно його створити, не варто жорстко кодувати облікові дані. Замість цього зробіть його звичайним обліковим записом. Коли хтось, хто має доступ до облікового запису, звільняється з компанії, змініть пароль.

**Розповсюдження програмного забезпечення**

Комерційні програми зазвичай захищають від несанкціонованого тиражування. Наявність доступу тільки до носія інформації з дистрибутивом (набором інсталяційних файлів) програмного продукту не повинна давати можливості встановити працездатну копію програми. Тобто даних дистрибутива, який можна скопіювати або непомітно взяти на декілька днів, не повинно вистачати для створення працездатної копії програми. Подібні обмеження можуть бути реалізовані різними способами. Наприклад, дуже багато комерційних програм при інсталяції вимагають ввести серійний номер, надрукований на коробці або вказаний в одному з документів, що додаються до програмного продукту.

Також часто виникає потреба обмежити число користувачів, що одночасно працюють з програмою. Тобто людина, яка придбала ліцензію на одне робоче місце, не повинна мати можливості створити 2 робочих місця, що функціонують одночасно. Це досягається за рахунок використання апаратних ключів, менеджерів ліцензій і процедури активації.

Для комерційних програмних продуктів, обмежених за часом або числом запусків, необхідно правильно реалізувати зберігання лічильників, щоб зловмисник не зміг примусити працювати програму, просто перевівши годинник або видаливши файл, в який записується кількість запусків програми або число оброблених файлів.

При захисті ПЗ від несанкціонованого копіювання використовуються методи, що дозволяю реалізовувати в ПЗ функції прив’язки процесу виконання коду програми до ПК на яких дана програма виконується. Інстальована програма для захисту від копіювання при кожному запуску повинна виконувати наступні дії:

- аналізувати програмно-апаратне середовище ПК, на якому вона запущена та формувати на основі цього аналізу характеристики свого середовища виконання;

- перевіряти ідентичність середовища виконання шляхом порівняння поточних характеристик з еталонними, що зберігаються на жорсткому диску ПК;

- блокувати свій запуск у випадку неспівпадання поточних характеристик з еталонними.

До основних методів захисту від копіювання можна віднести:

- криптографічний метод. Для цього методу інсталятор програми повинен виконувати:

1. аналіз програмно-апаратного середовище ПК, на якому він інстальована програма та формувати на основі цього аналізу еталонні характеристики середовища виконання програми;

2. проводити запис криптографічних перетворень еталонних характеристик програмно-апаратного середовище ПК на жорсткий диск.

- метод прив’язки до ідентифікатора. Зміст даного методу заключається в тому, що на жорсткому диску при інсталяції захищеної від копіювання програми формується унікальний ідентифікатор, який звіряється при кожному запуску програми. При відсутності чи неспівпадінні цього ідентифікатора програма блокує своє виконання.

- метод маніпуляції з кодом програми. Є два способи реалізації даного методу:

1. включення в тіло програми пустих модулів, на які імітується передача керування;

2. зміна початку захищеної програми таким чином, щоб стандартний дизасемблер не зміг її правильно дизасемблювати.

Під час вибору криптографічної системи найважливішою проблемою є керування ключами. Будь-яка, навіть найнадійніша, криптосистема базується на використанні ключів. Ефективність криптографічного захисту інформації в комп'ютерних системах і мережах визначається стійкістю алгоритмів криптографічних перетворень, які в ній використовуються, й надійністю протоколів керування ключами. Управління ключами – інформаційний процес, що включає в себе три елементи: генерація ключів; накопичення ключів; розподіл ключів. Правильне розв’язання названих задач має величезне значення, адже в більшості випадків зловмиснику легше атакувати саме ключову підсистему, а не алгоритм криптографічного захисту. Використання стійкого алгоритму шифрування є необхідною, але далеко не достатньою умовою побудови надійної системи криптографічного захисту інформації. Ключі, які використовуються в процесі інформаційного обміну, потребують однакового захисту на всіх стадіях життєвого циклу. Крім вибору потрібної для конкретної апаратної системи криптографічної системи, ще є одна важлива проблема – політика управління ключами.

Безпека будь-якого криптографічного алгоритму визначається використовуваним криптографічним ключем. Цей ключ повинен бути досить довгим і мати випадкові значення бітів.

Під накопиченням ключів розуміють організацію їх збереження, обміну, видалення. Цим питанням необхідно приділяти особливу увагу, секретні ключі ніколи не повинні записуватися в явному виді на носії, який може бути прочитаним, скопійованим.

Іншим напрямом захисту програм є використання підходу SaaS, тобто надання функціоналу цих програм (всього або частини), як сервісу. При цьому код програми розміщений і виконується на сервері, доступному в глобальній мережі. Доступ до нього здійснюється за принципом тонкого клієнта.

Код виконується на «довіреній» стороні, звідки не може бути скопійований.

Однак, і тут виникає низка проблем, пов'язаних з безпекою:

• стійкість такого захисту залежить, насамперед, від захищеності серверів, на яких він виконується (йдеться про Інтернет-безпеку)

• важливе забезпечення конфіденційності запитів, аутентифікації користувачів, цілісності ресурсу (можливість «гарячого» резервування), і доступності рішення в цілому

Виникають також питання довіри до сервісу (зокрема правові), оскільки йому фактично «у відкритому вигляді» передаються як саме ПЗ, так і дані, які воно обробляє (наприклад, персональні дані користувачів).

**Управління цифровими правами**

DRM, або Digital Rights Management (Управління цифровими правами), є набором технологій та методів, призначених для контролю, обмеження та захисту прав на використання цифрового контенту, такого як аудіо, відео, текст, програмне забезпечення і багато іншого. DRM використовується для забезпечення ліцензованого та легального використання цифрових матеріалів, а також для запобігання несанкціонованому копіюванню, поширенню та використанню цього контенту.

Основні компоненти DRM включають:

1. Шифрування: DRM може використовувати сильне шифрування для захисту цифрового контенту під час зберігання та передачі. Це робить неможливим несанкціонований доступ до контенту без відповідного розшифрування.

2. Аутентифікація та авторизація: DRM може вимагати, щоб користувач авторизувався або аутентифікувався перед отриманням доступу до цифрового контенту. Це дозволяє власникам контенту контролювати, хто і коли може використовувати контент.

3. Ліцензування: DRM використовує систему ліцензування для визначення обмежень щодо використання контенту. Ліцензія може включати в себе правила щодо кількості пристроїв, на яких можна відтворювати контент, терміни дії ліцензії та інші параметри.

4. Контроль доступу: DRM може обмежувати доступ до певних функцій або можливостей, наприклад, копіювання, друк або передачу контенту.

5. Системи управління правами (Rights Management Systems, RMS): Це спеціальні сервери та програмні засоби, що дозволяють власникам контенту керувати правами на цей контент. Вони слідкують за використанням ліцензованого контенту та забезпечують виконання правил ліцензій.

**Ліцензування**

**Генерація ліцензійних ключів**

Генерація ліцензійних ключів є важливим кроком у забезпеченні безпеки програм або сервісів. Нижче наведено деякі кращі практики для генерації ліцензійних ключів:

Використовуйте криптографічні бібліотеки. Використовуйте надійні криптографічні бібліотеки та функції для генерації ключів. Не спробуйте створювати власні алгоритми генерації ключів, оскільки це може призвести до вразливостей.

Використовуйте достатньо довгі ключі. Ключі повинні бути достатньо довгими і складними для підбору методом перебору (брутфорсу). Зазвичай рекомендується використовувати ключі довжиною не менше 128 біт або 256 біт для вищого рівня безпеки.

Використовуйте генератори випадкових чисел. Важливо використовувати надійні генератори випадкових чисел для створення ключів. Непередбачувість вихідних даних важлива для безпеки ключів.

Використовуйте криптографічні солі. Для підвищення безпеки генерації ключів можна використовувати криптографічні солі. Сіль - це випадковий рядок, який додається до вхідних даних перед генерацією ключа. Це робить ключ більш випадковим та унікальним.

Періодично змінюйте ключі. Якщо це можливо, розгляньте можливість періодичної зміни ключів, особливо для довгострокових ліцензій. Це допоможе запобігти накопиченню потенційно вразливих ключів з часом.

Зберігайте ключі в безпечному місці. Ключі повинні зберігатися в безпечному місці, і доступ до них повинен бути обмежений. Важливо уникати зберігання ключів у відкритому тексті чи у відкритому репозиторії.

Шифруйте ключі в спеціальному сховищі. Розгляньте можливість шифрування ключів перед зберіганням у сховищі. Це забезпечить їх додатковий рівень безпеки.

Відстеження і аудит ключів. Ведіть журнал подій щодо створення та використання ключів, щоб виявити потенційні аномалії та вразливості.

Резервні копії ключів. Робіть резервні копії ключів та зберігайте їх в безпечному місці. Це допоможе уникнути втрати ключів у випадку непередбачених обставин.

Використовуйте сучасні криптографічні алгоритми. Переконайтеся, що ви використовуєте сучасні криптографічні алгоритми, які вважаються безпечними на сьогоднішній день. Старі алгоритми можуть бути вразливими до атак.

**Зберігання ліцензійних ключів**

Зберігання ліцензійних ключів - це критично важливий аспект управління безпекою і ліцензуванням програм або сервісів. Нижче наведено кілька рекомендацій щодо місця зберігання ліцензійних ключів:

Спеціальне сховище ключів (Hardware Security Module, HSM). HSM - це фізичний апаратний пристрій, спеціально розроблений для зберігання і управління криптографічними ключами. Вони надійні та захищені від фізичних та програмних атак. Використання HSM може забезпечити найвищий рівень безпеки для ключів.

Централізована система керування ключами (Key Management System, KMS). Розгляньте можливість використання KMS для централізованого управління всіма ключами вашої організації. KMS дозволяють зберігати ключі в безпеці та забезпечувати автоматизований процес управління ними.

Хмарне сховище ключів (Cloud Key Vault). Деякі хмарні платформи пропонують хмарні сховища ключів, які забезпечують безпеку та доступ до

ключів з будь-якої точки світу. Вони можуть бути зручними для проектів, які використовують хмарні ресурси.

Використання багаторазових ключів. В ряді випадків, зазвичай у веб-продуктах, ключі можуть бути збережені на серверах, але їх важливо шифрувати та захищати від несанкціонованого доступу. Важливо використовувати криптографічні методи для зберігання цих ключів.

Шифрування ключів. Перед зберіганням ключів на будь-якому носії, вони повинні бути шифровані. Це допоможе захистити ключі у разі фізичного доступу до носія.

Двофакторна аутентифікація. Для доступу до ключів важливо вимагати двофакторну аутентифікацію, таку як пароль та фізичний токен. Це додатково ускладнить доступ до ключів.

Резервні копії ключів. Завжди робіть резервні копії ключів, і зберігайте їх в безпечному місці. Резервні копії можуть бути важливі у випадку втрати або пошкодження основних ключів.

Аудит і журналювання доступу. Ведіть журнали доступу до ключів, щоб відслідковувати, хто і коли мав доступ до них. Це допоможе виявити несанкціонований доступ.

**Розповсюдження ключів ліцензій**

Розповсюдження ключів ліцензій - це процес, який дозволяє користувачам отримати доступ до вашого програмного забезпечення або послуг після оплати та реєстрації. Цей процес може варіюватися в залежності від вашої бізнес-моделі та вимог. Декілька кроків для видавання ключів ліцензій:

Створіть систему реєстрації. Розробіть механізм реєстрації користувачів на вашому веб-сайті або в програмі. Це може включати в себе створення облікового запису, введення особистих даних та оплату.

Оплата та перевірка. Після реєстрації користувачі мають здійснити оплату за ліцензію. Використовуйте платіжні системи для приймання оплати, а також перевіряйте платіжні дані користувача для валідації оплати.

Видача ліцензійного ключа. Після успішної оплати та перевірки надайте користувачеві ліцензійний ключ. Цей ключ може бути надісланий електронною поштою або показаний на веб-сторінці після оплати.

Активація ключа. Забезпечте можливість користувачам ввести отриманий ключ ліцензії в програмі для активації. Переконайтеся, що програма може валідувати та активувати ключ.

Підтвердження активації. Після успішної активації надішліть користувачеві підтвердження, що ліцензія активована і вони можуть використовувати програму.

Ведіть журнал реєстрації та активацій. Реєструйте всі реєстрації та активації, щоб ви могли відслідковувати використання ліцензій та контролювати їхню валідність.

**Перевірка ліцензії**

Перевірка ліцензії в програмі - це важливий крок для забезпечення легального використання програмного забезпечення та запобігання несанкціонованому використанню. Кілька кроків, які можна виконати для реалізації перевірки ліцензії в програмі:

Створіть систему ліцензій. Спочатку створіть систему ліцензій, яка визначає правила використання вашої програми. Ця система повинна включати в себе тип ліцензії, обмеження щодо кількості користувачів або комп'ютерів, термін дії ліцензії і інші важливі деталі.

Створіть ключі ліцензії. Для кожної ліцензії створіть унікальний ключ або серійний номер. Ключ ліцензії може бути створений у вигляді рядка або файлу.

Зберігайте ключі ліцензії. Захищайте ключі ліцензії від несанкціонованого доступу. Їх зберігання може бути шифрованим або захищеним паролем. Найкраще місце для зберігання ключів - це окрема база даних або файл, недоступний ззовні.

Реалізуйте перевірку ліцензії в програмі. Додайте код до вашої програми, який перевіряє, чи існує і чи є дійсною ліцензія. Це може бути реалізовано шляхом перевірки ключа ліцензії, введеного користувачем під час установки або активації програми.

Обробляйте несправний ключ або ліцензію. Якщо ключ ліцензії недійсний або прострочений, ваша програма повинна відобразити повідомлення про помилку і, можливо, обмежити функціонал програми.

Захищайте ключі від перехоплення. Ключі ліцензії не повинні передаватися у відкритому вигляді через мережу. Використовуйте шифрування для передачі ключів, якщо це необхідно.

Періодично перевіряйте ліцензії. Якщо ваші ліцензії мають обмежену дійсність, створіть механізм автоматичної перевірки ліцензій через певні інтервали часу (наприклад, щоденно або щотижнево).

Запишіть події. Запишіть в журнал подій всі перевірки ліцензій та їх результати. Це може допомогти вам виявити спроби обхіду системи ліцензування.

Захищайте саму програму. Переконайтеся, що вашу програму важко модифікувати чи зламати. Використовуйте заходи безпеки, такі як обфускація коду, щоб ускладнити аналіз і модифікацію.

Перевірка апаратних параметрів. У деяких випадках ключ може бути пов'язаний з конкретною апаратною або програмною конфігурацією. Перевіряйте, чи відповідають апаратні параметри параметрам, збереженим у ключі.

Захист від перебору (Brute Force Protection). Для захисту від атак перебору паролів або ключів, рекомендується встановити обмеження на

кількість спроб введення ключа та використовувати захист від автоматичних атак.

**Реагування на інциденти**

Реагування на виявлені загрози та інциденти є критично важливою частиною забезпечення безпеки програм та систем. Ефективна стратегія реагування може зменшити можливий збиток і відновити стабільність системи. Кроки, які ви можете вжити при реагуванні на загрози та інциденти:

Визначення природи загрози чи інциденту. По-перше, визначте, що сталося. Це може бути атака, вразливість, помилка або інший тип інциденту. Важливо розуміти природу проблеми, щоб правильно реагувати.

Зупинення загрози. Якщо можливо, негайно зупиніть або обмежте здійснюючий загрозу процес чи активність. Наприклад, вимкніть атакований сервер або відключіть заражений комп'ютер від мережі.

Ізоляція та відновлення. Якщо інцидент вплинув на окремий компонент системи, ізолюйте його від решти системи для запобігання подальшому розповсюдженню інциденту. Після цього почніть процес відновлення цього компонента.

Відновлення безпеки. Після припинення загрози або виправлення інциденту відновіть безпеку системи. Змініть паролі, закрийте вразливості, відновіть зіпсовані дані тощо.

Збір і аналіз даних. Зберіть всю необхідну інформацію про інцидент: журнали подій, відомості про атаку чи помилку, дані про системи та мережу. Аналізуйте ці дані для встановлення причин та знаходження можливих вразливостей.

Виявлення вразливостей та усунення причин. Визначте, як атака стала можливою та чому вразливість не була виявлена раніше. Після цього вживайте заходів для усунення причин інциденту та запобігання подібним проблемам у майбутньому.

**Лекція 4.**

**Введення в етичний хакінг.**

Хакерство або хакінг - це процес пошуку слабких місць у комп'ютерних або приватних мережах, щоб використати їх і отримати доступ до системи. Під терміном "система" мається на увазі будь-який веб-сайт, база даних, комп'ютерна мережу - будь-що з чіткою структурою та специфічними механізмами захисту.

Етичний хакінг - це законна форма злому, за допомогою якого можна знайти помилки в чужих системах і звернути на них увагу розробників.

Етичний злом - це метод підвищення безпеки мережі. У цьому методі хакери виправляють вразливості та слабкі місця комп'ютера або мережі. Це санкціонована спроба отримати несанкціонований доступ до комп’ютерної системи, програми або даних. Виконуючи етичний хак, ви дублюєте ті самі стратегії, які використовували зловмисники, що допомагає виявити вразливі місця безпеки, які можна усунути, перш ніж вони будуть використані ззовні. Будь-яка система, процес, веб-сайт або пристрій можуть бути зламані, тому етичним хакерам дуже важливо розуміти, як може статися така атака та можливі наслідки.

Інструменти для злому - це комп'ютерні програми та скрипти, які допомагають знаходити та використовувати слабкі місця в комп'ютерних системах, веб-додатках, серверах та мережах. На ринку є багато різновидів таких інструментів. Деякі з них мають відкритий вихідний код, тоді як інші є комерційними рішеннями.

Найчастіше хакерів поділяють на два типи: білих і чорних капелюхів (назва походить з вестернів, де ковбої-вороги традиційно носять капелюхи цих кольорів).

Якщо особа виявляє вразливість і незаконно використовує її та/або розповідає іншим, як здійснювати цю діяльність, вона вважається "чорним

капелюхом". Якщо людина виявляє вразливість і використовує її з дозволу, вона вважається "білим капелюхом".

Також є поняття «сірого капелюха». Якщо інша людина виявляє вразливість, не використовує її незаконно і не розповідає іншим, як це зробити, ця людина вважається "сірим капелюхом". Сірі хакери знаходяться десь посередині між хорошими та поганими хлопцями. На відміну від чорних хакерів, вони намагаються порушувати стандарти і принципи, але без наміру завдати шкоди або отримати фінансову вигоду. Їхні дії, як правило, здійснюються для загального блага. Наприклад, вони можуть використати вразливість, щоб підвищити обізнаність про її існування, але, на відміну від білих хакерів, вони роблять це публічно. Це попереджає зловмисників про існування вразливості.

Етичний хакер, також званий "білим капелюхом", - це експерт з інформаційної безпеки, який проникає в комп'ютерну систему, мережу, застосунок або інший обчислювальний ресурс від імені власників з їхнього дозволу.

Організації звертаються до етичних хакерів для виявлення потенційних вразливостей безпеки, якими можуть скористатися зловмисні хакери.

Метою етичного хакінгу є оцінка безпеки та виявлення вразливостей у цільових системах, мережах або системній інфраструктурі. Цей процес

передбачає пошук і подальшу спробу використання вразливостей для визначення можливості несанкціонованого доступу або інших шкідливих дій.

Вам потрібно розуміти тактику вашого супротивника. Ви не можете належним чином захистити себе від загроз, яких не розумієте.

Чому військові в усьому світі вивчають тактику, інструменти, стратегії, технології і так далі своїх ворогів? Тому що чим більше ви знаєте про те, що замишляє ваш ворог, тим краще ви розумієте, які механізми захисту вам потрібно запровадити, щоб захистити себе.

У більшості випадків набір інструментів, які використовують зловмисники - це той самий набір інструментів, який використовують фахівці з безпеки. Насправді, книги, курси, статті, веб-сайти та семінари про хакерство можна було б з повним правом перейменувати на "навчання професійному інструментарію безпеки". Етичні хакери проходять через ті ж процеси і процедури, що й неетичні, тому цілком логічно, що вони використовують той самий базовий набір інструментів.

Етичний хакер повинен знати, що використовують погані хлопці, знати нові експлойти, які з'являються в підпіллі, і постійно оновлювати свої навички та базу знань.

Як хакерські інструменти використовуються для добра, а не для зла? Як можна гарантувати, що всі співробітники створюють складні паролі, які відповідають політиці паролів компанії? Ви можете встановити конфігурацію операційної системи, щоб переконатися, що паролі мають певну довжину, містять великі та малі літери, числові значення та зберігають історію паролів. Але ці конфігурації не можуть перевірити наявність слів у словнику або розрахувати, наскільки захищеними є паролі від атак грубого підбору.

Таким чином, ви можете використовувати хакерський інструмент для проведення словникових атак та атак грубої сили на окремі паролі, щоб фактично перевірити їхню надійність.

Так само ви повинні переконатися, що конфігурації брандмауерів і маршрутизаторів дійсно забезпечують рівень захисту, який вимагає компанія.

Ви можете прочитати посібники, внести зміни в конфігурацію, впровадити ACL. Або ж ви можете впровадити конфігурації, а потім запустити тести з цими налаштуваннями, щоб перевірити, чи не пропускають вони зловмисний трафік. Ці тести часто вимагають використання хакерських інструментів. Ці інструменти здійснюють різні типи атак, які дозволяють побачити, як пристрої периметру реагуватимуть за певних обставин. Нічому не слід довіряти, поки це не перевірено.

Важливо знати, як відбуваються різні типи атак, щоб їх можна було правильно розпізнати і зупинити.

Інструменти можуть використовуватися в добрих (захисних) цілях або в поганих (наступальних) цілях. Хороші і погані хлопці використовують один і той самий набір інструментів; різниця полягає в їхніх намірах при роботі з цими утилітами. Фахівець з безпеки повинен розуміти, як використовувати ці інструменти і як здійснюються атаки, якщо він збирається бути корисним своїм клієнтам і галузі в цілому.

Як тільки мережеві адміністратори, інженери та фахівці з безпеки зрозуміють, як працюють зловмисники, вони можуть імітувати їхню діяльність, щоб провести корисний тест на проникнення. Але навіщо комусь імітувати атаку? Тому що це єдиний спосіб по-справжньому перевірити рівень безпеки середовища - ви повинні знати, як воно відреагує на реальну атаку.

Більшість недоліків у безпеці виникають на ранній стадії процесу розробки додатків. Хороші та погані додатки відрізняються за шістьма ключовими факторами:

1. Автентифікація та авторизація. Найкращі додатки гарантують, що етапи автентифікації та авторизації є повними і не можуть бути обійдені.

2. Недовіра до вводу користувача. Користувачі повинні розглядатися як "ворожі агенти", оскільки дані перевіряються на стороні сервера, а рядки позбавляються тегів, щоб запобігти переповненню буфера.

3. Наскрізне шифрування сеансів. Шифруватися повинні всі сеанси, а не лише частини активності, які містять конфіденційну інформацію. Крім того, безпечні програми повинні мати короткі таймаути, які вимагають від користувачів повторної автентифікації після періодів неактивності.

4. Безпечна обробка даних. Безпечні додатки також забезпечують безпеку даних, коли система знаходиться в неактивному стані. Наприклад, паролі повинні залишатися зашифрованими під час зберігання в базах даних, а також має бути впроваджена безпечна сегрегація даних. Неправильна реалізація компонентів криптографії зазвичай відкриває багато можливостей для несанкціонованого доступу до конфіденційних даних.

5. Усунення неправильних конфігурацій, бекдорів та налаштувань за замовчуванням. Поширеною, але небезпечною практикою для багатьох постачальників програмного забезпечення є постачання програмного забезпечення з бекдорами, утилітами та адміністративними функціями, які допомагають адміністратору, що отримує продукт, вивчити та впровадити його. Проблема полягає в тому, що ці вдосконалення зазвичай містять серйозні недоліки в безпеці. Ці елементи завжди повинні бути вимкнені і вимагати, щоб замовник їх увімкнув, а всі бекдори повинні бути належним чином вилучені з вихідного коду.

6. Забезпечення якості безпеки. Безпека повинна бути основною дисципліною при розробці продукту, на етапах специфікації та розробки, а також на етапах тестування.

Хакінг важко проводити без спеціальних інструментів. Зламати щось слабо захищене можливо, особливо якщо в вашому розпорядженні міцна конфігурація ПК. Навіть найпростіший метод хакінгу - брутфорс - потребує потужної графічної карти, якщо ви хочете провести операцію достатньо швидко.

Занурюючись у світ кібербезпеки, важливо бути знайомим з поширеними інструментами злому, які використовують кіберзлочинці. Ці інструменти допомагають хакерам використовувати вразливості в системах і

мережах, але вони також можуть бути етично використані фахівцями з безпеки для перевірки власних мереж і систем на наявність вразливостей.

Нижче наведено короткий огляд деяких поширених хакерських інструментів:

**Nmap (Network Mapper).** Nmap - це популярний мережевий сканер з відкритим вихідним кодом, який використовується як фахівцями з кібербезпеки, так і хакерами для виявлення хостів і сервісів у мережі. Він допомагає ідентифікувати хости, відкриті порти, запущені служби, типи ОС і багато інших деталей. Він особливо корисний для інвентаризації мережі та аудиту безпеки.

**Wireshark.** Wireshark - ще один інструмент з відкритим вихідним кодом, який використовується для аналізу мережі та усунення несправностей. Він дозволяє користувачеві перехоплювати і аналізувати трафік, який передається через мережу. Це допомагає виявити будь-яку підозрілу активність, таку як спілкування зі шкідливим програмним забезпеченням або спроби несанкціонованого доступу.

Комп'ютери спілкуються за допомогою мереж. Ці мережі можуть бути в локальній мережі LAN або виходити в інтернет. Мережеві сніфери - це програми, які перехоплюють низькорівневі дані пакетів, що передаються мережею. Зловмисник може проаналізувати ці дані, щоб виявити цінну інформацію, таку як ідентифікатори користувачів та паролі.

Комп'ютери спілкуються, транслюючи повідомлення в мережі за допомогою IP-адрес. Після того, як повідомлення надіслано в мережу, комп'ютер-одержувач з відповідною IP-адресою відповідає на нього своєю MAC-адресою.

Мережевий сніффінг - це процес перехоплення пакетів даних, що надсилаються мережею. Це може бути зроблено за допомогою спеціалізованого програмного забезпечення або апаратного обладнання.

Сніфінг можна використовувати для;

- Захоплення конфіденційних даних, таких як облікові дані для входу в систему.

- Підслуховування повідомлень чату

- Перехоплення файлів, переданих через мережу

**Metasploit.** Metasploit - це потужний фреймворк для тестування на проникнення, який охоплює широкий спектр експлойтів та вразливостей. Завдяки набору інструментів, що налаштовується та розширюється, Metasploit особливо корисний для моделювання реальних кібератак і допомагає визначити, де ваша система є найбільш вразливою.

**John the Ripper.** John the Ripper - це відомий інструмент для злому паролів, який можна використовувати для виявлення слабких паролів і тестування безпеки паролів. Він підтримує різні алгоритми шифрування, а також може використовуватися для визначення хешів.

**Burp Suite.** Burp Suite - це інструмент для тестування безпеки веб-додатків, який в основному використовується для пошуку вразливостей у веб-додатках. Він включає в себе інструменти для перехоплення і модифікації запитів, автоматизації тестів, сканування і багато іншого.

**Aircrack-ng.** Aircrack-ng - це набір інструментів для забезпечення безпеки Wi-Fi. До нього входять інструменти для перехоплення та аналізу мережевих пакетів, зламу паролів Wi-Fi та тестування загальної безпеки бездротових мереж.

**Kali Linux.** Kali Linux - це дистрибутив Linux, спеціально створений для тестування на проникнення та аудиту безпеки. Він постачається з попередньо встановленим широким спектром інструментів для злому і зазвичай використовується етичними хакерами та фахівцями з безпеки.

Kali Linux має понад 600 попередньо встановлених програм для тестування на проникнення. Кожна програма має свою унікальну гнучкість та можливості використання.

Kali Linux є дійсно унікальною операційною системою, оскільки це одна з небагатьох платформ, яку відкрито використовують як хороші, так і

погані хлопці. Адміністратори безпеки та хакери Black Hat широко використовують цю операційну систему. Перші - для виявлення та запобігання порушенням безпеки, а другі - для виявлення та можливого використання порушень безпеки. Кількість інструментів, налаштованих і попередньо встановлених в операційній системі, робить Kali Linux швейцарським армійським ножем в інструментарії будь-якого фахівця з безпеки.

Професіонали, які використовують Kali Linux 1. Адміністратори безпеки - Адміністратори безпеки відповідають за захист інформації та даних своєї установи. Вони використовують Kali Linux для перегляду своїх середовищ і забезпечення відсутності вразливостей, які легко виявити. 2. Мережеві адміністратори - мережеві адміністратори відповідають за підтримку ефективної та безпечної мережі. Вони використовують Kali Linux для аудиту своєї мережі. Наприклад, Kali Linux має можливість виявляти несанкціоновані точки доступу. 3. Мережеві архітектори - Мережеві архітектори відповідають за проектування безпечних мережевих середовищ. Вони використовують Kali Linux для аудиту своїх початкових проектів і гарантують, що нічого не було пропущено або неправильно налаштовано.

4. Pen Testers - тестувальники, використовують Kali Linux для аудиту середовищ і виконують розвідку корпоративних середовищ, які вони найняли для перевірки.

5. Керівники служби інформаційної безпеки використовують Kali Linux для внутрішнього аудиту свого середовища і виявлення нових додатків або складних конфігурацій, які були встановлені.

6. Інженери-криміналісти - Kali Linux має "Режим судової експертизи", який дозволяє інженерам-криміналістам виконувати пошук та відновлення даних у деяких випадках.

7. Хакери в білих капелюхах - Хакери в білих капелюхах, подібно до тестувальників, використовують Kali Linux для аудиту та виявлення вразливостей, які можуть бути присутніми в середовищі.

8. Black Hat Hackers - Хакери Black Hat Hackers використовують Kali Linux для виявлення та використання вразливостей. Kali Linux також має численні додатки для соціальних інженерів, які можуть бути використані хакерами Black Hat для компрометації організації або приватної особи.

10. Комп'ютерний ентузіаст - Комп'ютерний ентузіаст є досить загальним терміном, але будь-хто, хто зацікавлений дізнатися більше про мережу або комп'ютери в цілому, може використовувати Kali Linux, щоб дізнатися більше про інформаційні технології, мережу та поширені вразливості.

Майте на увазі, що хоча ці інструменти зазвичай використовуються хакерами, вони також можуть бути етично застосовані фахівцями з безпеки для розуміння та усунення вразливостей у власних системах. Головне - використовувати їх відповідально і завжди питати дозволу, перш ніж тестувати будь-яку мережу або систему, яка вам не належить.

Найпопулярніший різновид етичного хакінгу - програми пошуку помилок за винагороду Bug Bounty. Їм уже понад двадцять років. Ці сервіси дають змогу компаніям своєчасно виявити й усунути баги у своїх продуктах, перш ніж про них дізнаються зловмисники.

Зазвичай це відбувається так: компанія оголошує конкурс на пошук вразливостей у своїх системах і суму винагороди. Після цього інформаційну інфраструктуру (пристрій/програму/додаток) компанії починають промацувати з усіх боків численні експерти (пентестери). У деяких випадках корпорації оголошують про початок закритої програми. У такому разі організатор сам обирає потенційних учасників і розсилає їм запрошення та умови участі.

Є дві великі платформи, які допомагають зустрітися дослідникам вразливостей і компаніям, які хочуть перевірити свої сервіси: HackerOne і

Bugcrowd. Вони, по суті, агрегують усі програми IT-компаній, а зареєстровані учасники сервісів можуть обрати те, що їм цікаво. Зараз обидві платформи об'єднують тисячі фахівців з інформаційної безпеки з різних країн. До речі, навіть державні структури використовують подібні сервіси. Наприклад, Пентагон вибрав HackerOne для запуску своєї програми Hack the Pentagon.

У конкурсах bug bounty фігурують дуже пристойні суми. Компанія HackerOne опублікувала звіт Hacker-Powered Security, з якого випливає, що в 2017 році середня винагорода за знайдений баг становила понад $1 900. Загалом за останні 4 роки білим хакерам виплатили понад $17 млн за 50 тис. знайдених помилок.

Взагалі подякувати за модель bug bounty варто компанії Netscape Communications Corporation. Їхній сервіс Netscape Bugs Bounty, запущений у середині 90-х, давав змогу за винагороду шукати недоліки в браузері Netscape Navigator. Компанія одна з перших здогадалася, що кращими за своїх розробників можуть бути тільки тисячі інших IT-фахівців, здатних за гроші відшукати проблемні місця. Ідея програми мала такий успіх, що її модель уже дуже скоро перейняли відомі IT-корпорації.

Бувають і разові особливо великі виплати. Так торік Microsoft оголосила про запуск програми bug bounty для Windows з максимальною премією в розмірі $250 000. Гроші обіцяли за вразливості в гіпервізорі і ядрі Microsoft Hyper-V, що дають змогу віддалено виконувати код. Трохи раніше Facebook виплатила фахівцеві з інформаційної безпеки $40 000 за одну знайдену критичну вразливість.

Google свого часу перерахувала експертам понад $6 млн, а Facebook за п'ять років існування її bug bounty заплатила добропорядним хакерам $5 млн.

Компаніям вигідніше платити за знайдені помилки, ніж розбиратися з наслідками, до яких можуть призвести експлойти та вразливості.

Етичні хакери можуть допомогти організаціям різними способами, включно з такими:

1. Пошук вразливостей. Етичні хакери допомагають компаніям визначити, які з їхніх заходів ІТ-безпеки є ефективними, які потребують оновлення, а які містять вразливі місця, що можуть бути використані. Коли етичні хакери закінчують оцінку систем організації, вони повідомляють керівникам компанії про вразливі місця, які можуть включати відсутність достатнього шифрування паролів, небезпечні додатки або відкриті системи з непропатченим програмним забезпеченням. Організації можуть використовувати дані цих тестів для ухвалення обґрунтованих рішень про те, де і як поліпшити свій захист для запобігання кібератак.

2. Демонстрація методів, які використовують кіберзлочинці. Такі демонстрації показують керівникам компаній методи злому, які зловмисники можуть використовувати для атаки на їхні системи і створення хаосу в їхньому бізнесі. Компанії, що володіють глибокими знаннями про методи, які зловмисники використовують для проникнення в їхні системи, мають більше можливостей для запобігання таким вторгненням.

3. Допомога в підготовці до кібератаки. Кібератаки можуть скалічити або знищити бізнес. Етичні хакери розуміють, як діють суб'єкти загроз, і знають, як ці погані суб'єкти використовуватимуть нову інформацію та методи для атак на системи. Фахівці з безпеки, які працюють з етичними хакерами, краще підготовлені до майбутніх атак, оскільки вони можуть краще реагувати на мінливу природу онлайн-загроз, що постійно змінюється.

Етичний хакер працює виключно за програмами Bug Bounty або після укладення контракту з компанією. Але навіть опублікована Bug Bounty не означає, що можна ламати і шукати дірки де попало. В описі програми власники прописують, які саме вразливості розглядатимуться.

Наприклад, Uber дає дуже докладне пояснення, що входить до їхньої програми Bug Bounty, а що - ні.

Компанія хоче знайти вразливості в системах доступу і зберігання даних, можливостей фішингу, оплати і рахунків, несанкціонованих дій з боку користувача і співробітників компанії. Але в програму не входять загальні

баги застосунку, звіти про шахрайство, баги в роботі з соцмережами та email-розсилкою.

При цьому є загальні правила, які порушувати не можна. Наприклад, під час виявлення вразливостей у базах даних користувачів не можна намагатися завантажити будь-які особисті дані. Навіть за участі в програмі це може бути розцінено як порушення закону. Тому що тут порушуються права саме користувачів, до яких Bug bounty не має жодного стосунку.

Популярність баг-баунті програм для бізнесу та державних установ пояснюється рядом конкурентних переваг:

• Економія ресурсів. Баг-баунті програми на потребують додаткового працевлаштування співробітників, а етичні хакери отримують фінансову винагороду лише за конкретні результати своєї роботи.

• Гнучкість. Замовник може змінювати умови програми залежно від поставлених цілей. Також замовник сам визначає, які компоненти продукту потребують незалежного тестування. Таким чином, компанії не потрібно замовляти тестування усього продукту, а достатньо обмежитись лише тими його компонентами, які потребують особливої уваги.

• Довіра ринку. Організовуючи баг-баунті, компанія демонструє ринку свою готовність до подальшого вдосконалення продукту, незважаючи на можливі додаткові фінансові витрати. У цьому випадку продукти, які пройшли баг-баунті, організовані на відомих незалежних платформах, отримують високий рівень довіри ринку.

• Залучення талановитих спеціалістів. Під час проведення бау баунті програми компанія може виявити кваліфікованих спеціалістів, зацікавлених у подальшій роботі над покращенням безпеки продукту. У цьому випадку, баг-баунті слугує ще одним каналом залучення талановитих спеціалістів до компанії, не створюючи при цьому значних додаткових витрат для HR-відділу компанії.

У 2022 році криптокомпанія Aurora виплатила етичному хакеру $6 млн винагороди за виявлення критичної вразливості, використавши яку кіберзлочинці могли завдати проєкту шкоди на $330 млн.

У 2022 році, лише за 7 днів функціонування нової баг-баунті програми, Pentagon отримав 648 звітів про вразливості своїх систем, в результаті чого було виявлено та усунуто близько 350 вразливостей, а заробіток білих хакерів склав близько $100 000.

У 2019 році електронна система публічних закупівель Prozorro у співпраці із платформою HackenProof організувала марафон з пошуку багів, які потенційно могли б робити її вразливою до кібератак.

Таким чином Prozorro стало першим українським підприємством, яке самостійно ініціювали співпрацю із білими хакерами, а Україна, на той час, лише дев’ятою країною у світі, яка публічно організувала баг-баунті програму у державному секторі.

Марафон з пошуку вразливостей у Prozorro тривав понад 7 годин в тестовому середовищі у закритому форматі. Участь у даному марафоні взяли 12 білих хакерів, які підготували 16 унікальних звітів про вразливості у Prozorro, які, однак, не загрожували безпеці системи.

У 2020 році Міністерство цифрової трансформації України у співпраці з баг-баунті платформою BugCrowd та Агентством з міжнародного розвитку США (USAID) організувало перевірку безпечності застосунку «ДІЯ» в рамках баг-баунті програми. Під час тестування було підверджено високий рівень захищеності продукту.

Тривалий час сфера етичного хакінгу в Україні функціонувала у стані законодавчого вакууму.

Лише у березні 2022 року Верховна Рада України підтримала внесення змін до Кримінального Кодексу України, за якими втручання в роботу інформаційних, електронних комунікаційних, інформаційно-комунікаційних систем, електронних комунікаційних не вважається несанкціонованим за умови, що це втручання відбувається в межах програми баг-баунті.

16 травня 2023 року Кабінет Міністрів України дозволив етичним хакерам легально перевіряти системи на наявність вразливостей, давши таким чином можливість компаніям посилити свій рівень захисту перед реальними цифровими викликами в рамках законодавчого поля.

В цілому, всі платформи працють за приблизно однаковими принципами. Клієнтами можуть бути як великі державні та приватні компанії та установи, так і невеликі проєкти, зацікавлені у виявленні вразливостей у своїх системах. В переважній більшості, вони підходять як для професійних білих хакерів, так і для тих, хто лише відкриває для себе світ етичного хакінгу.

Платформа, на якій організовуються баг-баунті програми, розміщує детальний опис кожної програми із зазначенням переліку вразливостей для пріоритетного пошуку. Чим вищий рівень загрози від вразливості, тим більшу винагороду отримає білий хакер.

Виявивши вразливість у продукті, користувач надсилає (завантажує) доповідь, яка, своєю чергою, опрацьовується командою платформи. У доповіді користувач описує виявлену вразливість та зазначає варіанти її виправлення.

Команда аналізує доповідь на предмет повторення вразливості, а також важливості, адже доповіді можуть і не містити цінної інформації та охоплювати, скажімо, суто «косметичні» правки.

Якщо інформація в доповіді є актуальною та використовується замовником для усунення вразливості, тоді користувач отримує зазначену винагороду.

Багато етичних хакерів дотримуються того самого процесу, що й зловмисники, який включає п’ять етапів:

**Першим етапом** етичного хакерства є розвідка, яка є етапом збору інформації. Ця підготовка передбачає збір якомога більшої кількості інформації перед початком атаки. Тип зібраних даних може містити паролі, важливі відомості про співробітників та інші важливі дані. Хакер може

збирати ці дані за допомогою кількох інструментів, і це допомагає визначити, які атаки мають найкращі шанси на успіх і які системи організації найбільш вразливі. Існує два типи розвідки або відстеження слідів.

Пасивна розвідка передбачає отримання інформації про об'єкт без безпосередньої взаємодії з ним, наприклад, пошук публічної інформації.

Активна розвідка передбачає взаємодію з об'єктом, наприклад, звернення до служби технічної підтримки для отримання конфіденційної інформації.

Розвідка не лише технічна. Це також важлива зброя конкурентної розвідки. Знання деяких фінансових аспектів цілі може означати, що атака буде успішною.

**Другий етап — с**канування, під час якого хакери визначають різні способи отримання інформації про ціль. Ця інформація часто містить облікові записи користувачів, IP-адреси та облікові дані, які забезпечують швидкий доступ до мережі.

Зібравши достатню кількість інформації про ціль, зловмисник повинен просканувати її, щоб виявити корисну інформацію про систему і використати цю інформацію для наступної фази (фази отримання доступу). Під час цього процесу зловмисник буде шукати різні типи інформації, і для цього він буде використовувати різні типи сканування.

Сканування портів - це процес надсилання пакетів на ціль з метою дізнатися більше про неї у зв'язку з відомими номерами портів. Існує дві категорії сканування портів: TCP-сканування і UDP-сканування. Для сканування портів рекомендується використовувати Nmap, який є сканером портів з відкритим вихідним кодом і інструментом для дослідження мережі.

Сканування мережі описує процес знаходження всіх активних хостів у мережі. Сканування діапазону IP-адрес є різновидом мережевого сканування. Основною технікою для виявлення живих хостів є пінг-опитування. Він просто надсилає ехо-запити ICMP на декілька хостів з певного діапазону IP-адрес. Hping2 - це простий мережевий сканер командного рядка для

протоколу TCP/IP. Сканування вразливостей. Під час цього підетапу зловмисник намагається виявити слабкі місця в об'єкті атаки. Основна мета цього типу сканування - знайти потенційний спосіб експлуатації системи. Існує безліч інструментів для сканування вразливостей, таких як Nessus, Nexpose та багато інших сканерів.

**Третя фаза** полягає в отриманні доступу до цільових систем, програм або мереж. Цей доступ досягається за допомогою різних інструментів і методів, що дозволяє використовувати систему шляхом завантаження шкідливого програмного забезпечення, викрадення конфіденційних даних, отримання доступу, запитів на викуп тощо. Етичні хакери часто звертаються до брандмауерів, щоб захистити точки входу та мережеву інфраструктуру.

На цьому етапі зловмисник вже має все необхідне для запуску атаки, включаючи діапазон IP-адрес, ідентифіковані системи, сервіси, списки користувачів, вразливості системи безпеки та потоки даних. Тепер зловмиснику потрібно лише обійти засоби контролю безпеки, щоб отримати доступ до системи, використовуючи кілька методів, таких як злам паролів, соціальна інженерія або підвищення привілеїв, а також отримання інших дозволів користувача. Здебільшого метою хакерської атаки є не лише отримання інформації шляхом несанкціонованого доступу, але й збереження цього доступу. Щодня зловмисники вигадують нові способи збереження доступу. Найвідоміший з них - приховування файлів від власника системи та користувачів, щоб їх не викрили.

**Четверта фаза** — збереження доступу після того, як хакер отримав доступ до системи. На цьому етапі хакер постійно використовує систему за допомогою DDoS-атак і викрадення бази даних. Потім хакер зберігає доступ до тих пір, поки зловмисна діяльність не буде здійснена без відома організації.

**На останньому етапі** хакери очищають свої сліди та приховують усі сліди несанкціонованого доступу. Хакер повинен підтримувати своє

з'єднання в системі, не залишаючи підказок, які можуть привести до його ідентифікації або реакції з боку організації. Під час цього етапу папки, програми та програмне забезпечення зазвичай видаляються. Дуже важливо, щоб після отримання доступу і зловживання мережею зловмисник замітав сліди, щоб його не відстежили і не спіймали. Для цього зловмисник видаляє всі види журналів і шкідливе програмне забезпечення, пов'язане з атакою. На цьому етапі зловмисник відключає аудит, очищає та маніпулює журналами.

Це п’ять типових кроків, які виконують етичні хакери, намагаючись виявити будь-які вразливості, які можуть надати доступ зловмисникам.

Етичні хакери=фрілансери, які виконують роботу за винагороду в рамках програм bug bounty, або просто з цікавості, можуть допомогти знайти вразливості. Кожен може займатись етичним хакінгом.

Методи та тактики білих хакерів наступні:

- Соціальна інженерія. Ця тактика спрямована на обман і маніпуляцію людиною. За допомогою цієї тактики користувачів змушують ділитися своїми паролями до облікових записів, а також надавати конфіденційні дані про свої банківські рахунки.

- Тестування на проникнення. Цей метод дозволяє знайти вразливості у системі безпеки за допомогою тестування та дозволяє хакеру виправити їх так, як він бажає.

- Розвідка та дослідження. Цей метод допомагає хакеру знайти слабкі місця або вразливості в фізичній частині, будь то в IT-підприємстві. Цей метод не передбачає зламу або вторгнення в систему. Навпаки, він дозволяє обійти систему безпеки, щоб отримати те, що потрібно.

- Програмування. Білі хакери створюють пастки для кіберзлочинців, щоб отримати від них інформацію про них або відволікти їх.

- Використання різноманітних цифрових та фізичних інструментів. У цьому випадку хакери оснащені апаратним забезпеченням, яке допомагає їм встановлювати шкідливі програми та боти, які дають їм доступ до серверів або мереж.

Сьогоднішній небезпечний кіберландшафт вимагає від усіх підприємств позиціонувати себе на крок попереду кіберзлочинців, щоб забезпечити власну безпеку. Це завжди починається з виявлення ваших слабких місць, розуміння того, як ваша компанія може бути скомпрометована, і впровадження найбільш підходящих методів запобігання та виявлення, які допоможуть вам досягти кіберстійкості. Але спочатку ви повинні зрозуміти, з якими векторами атак ви можете зіткнутися, які можуть підірвати ваш бізнес.

Вектори атак (або вектори загроз) - це шляхи, якими кіберзловмисники проникають в ІТ-інфраструктуру. По суті, вектор атаки - це процес або маршрут, який використовує зловмисник, щоб досягти мети, або, іншими словами, заходи, яких вживає зловмисник для проведення атаки.

Як правило, вектори атаки є навмисними загрозами, оскільки вони вимагають певного планування та аналізу.

Ці вектори атаки можуть використовувати різні суб'єкти - від засмучених колишніх співробітників до зловмисних хакерів, груп кібершпигунства, конкурентів тощо. Незалежно від того, хто це може бути, вони можуть мати на меті або підірвати ваш бізнес, або викрасти ваші технології, конфіденційну інформацію, або вимагати гроші у ваших співробітників. У будь-якому випадку, вони зроблять все можливе, щоб успішно використати вектори атаки та отримати доступ до ваших систем.

Поверхня атаки - це загальна можлива кількість векторів атаки, які можуть бути використані зловмисником для отримання доступу або викрадення даних з вашої мережі або кінцевих точок.

Найпоширеніші вектори атак в кібербезпеці:

1. Внутрішні загрози. Інсайдерська загроза - один з найпоширеніших векторів атак. Проте не всі види інсайдерських загроз є зловмисними, оскільки наївні працівники іноді можуть ненавмисно розголошувати внутрішні дані. Однак зловмисники, які працюють в компанії, можуть навмисно розголошувати конфіденційну інформацію або встановлювати

шкідливе програмне забезпечення, керуючись різними мотивами та заради власної вигоди.

Останні статистичні дані про інсайдерські загрози свідчать про тривожні проблеми, які повинні бути розглянуті та вирішені всіма організаціями. Наприклад, за останні два роки кількість внутрішніх загроз зросла на 47%, а 70% організацій стали свідками більш частих внутрішніх атак.

2. Фішингові електронні листи. Фішинг - це лише один з багатьох різновидів соціальної інженерії. Це тактика маніпуляцій, яку застосовує зловмисник, щоб змусити співробітників натиснути на підозрілі посилання, відкрити інфіковані шкідливим програмним забезпеченням вкладення або розкрити свої облікові дані для входу в систему.

3. Атаки на ланцюги постачання. Бізнес-партнери також можуть стати основними векторами атак у сфері кібербезпеки. Деякі з найбільших інцидентів безпеки та витоків даних були спричинені постачальниками. Атаки на ланцюги постачання є поширеним способом, за допомогою якого зловмисники націлюються на клієнтів постачальника. Саме тому великі та малі організації разом зі своїми діловими партнерами повинні розвивати культуру, в якій відбувається обмін найкращими практиками кібербезпеки та демонструється взаємна прозорість.

4. Слабкі або скомпрометовані облікові дані для входу. Якщо облікові дані для входу ваших співробітників занадто слабкі або скомпрометовані, це може стати для зловмисника надійним способом отримати несанкціонований доступ до ваших ІТ-систем.

Імена користувачів та паролі є найпопулярнішою формою автентифікації, якою можна легко зловживати за допомогою фішингу, витоків даних та шкідливих програм, що викрадають облікові дані, надаючи зловмисникам вільний доступ до облікових записів ваших працівників.

5. Атаки грубої сили. Практика, за допомогою якої зловмисники надсилають кілька паролів з метою їх остаточного вгадування, також є

серйозним вектором атаки. Після пандемії нового коронавірусу дані Heimdal Security показали, що кількість атак грубого перебору зросла в геометричній прогресії. Після того, як більшість співробітників почали працювати з дому.

6. Невиправлені вразливості. Невиправлені вразливості у ваших системах можуть призвести до появи експлойтів і дозволити зловмисникам пройти крізь них. Найвідоміших на сьогоднішній день атак вірусів-здирників (таких як WannaCry та NotPetya) можна було б уникнути, якби системи були вчасно виправлені.

Програми-вимагачі продовжують залишатися дуже прибутковим бізнесом для кіберзлочинців. З огляду на величезні прибутки, не дивно, що програми-здирники навіть перетворилися на "бізнес-модель" - Ransomware as a Service ("Вимагачі як послуга"). Це дозволяє йому стати легкодоступним навіть для людей з досить слабкими технічними навичками, але з бажанням нажитися на вразливих користувачах.

7. Міжсайтовий скриптинг. Міжсайтовий скриптинг (XSS) - це тип вразливості комп'ютерної безпеки, який зазвичай зустрічається у веб-додатках. XSS дозволяє зловмисникам впроваджувати шкідливий код на веб-сторінки, які переглядають інші користувачі. Коли користувач переглядає веб-сторінку, будь-який шкідливий код, введений зловмисником, виконується браузером, що призводить до потенційного розкриття конфіденційної інформації або виконання небажаного коду.

8. Атаки типу "людина посередині. Атака "людина посередині" - це тип кібератаки, коли зловмисник втручається в розмову між двома жертвами з метою підслуховування або втручання в комунікацію. Зловмисник перехоплює повідомлення між жертвами і змінює їх, перш ніж повторно відправити іншій жертві, створюючи враження, що повідомлення надійшло від початкового відправника. Цей тип атаки може бути використаний для викрадення конфіденційної інформації, такої як облікові дані для входу в систему, фінансова інформація або комерційна таємниця. Атаки типу "людина посередині" також можуть бути використані для впровадження

шкідливого коду на веб-сайти або програмне забезпечення, який потім може бути використаний для зараження комп'ютерів жертв.

9. Отруєння DNS. Отруєння DNS, також відоме як підміна DNS, - це тип кібератаки, коли зловмисник пошкоджує систему доменних імен (DNS), щоб вказати доменне ім'я на неправильну IP-адресу. Це може перенаправляти користувачів на шкідливий веб-сайт або сервер, де вони можуть бути інфіковані шкідливим програмним забезпеченням або піддані фішингу з метою отримання конфіденційної інформації.

10. Шкідливі програми / трояни. Існує багато шкідливих програм, які можуть заразити ваш пристрій і викрасти вашу інформацію. Деякі з них маскуються під легітимні програми, тоді як інші є відверто підробленими.

Є кілька ролей, пов’язаних з етичним хакерством, обсяг яких може змінюватися залежно від вашої сфери знань і організації/сектору, в якому ви працюєте. Деякі з найпопулярніших професій етичних хакерів включають:

• Тестер проникнення. Пентестери виконують авторизовані тести на комп’ютерах і мережах, щоб виявити слабкі місця. Тестери проникнення часто спеціалізуються на одному конкретному типі системи.

• Розслідувач комп'ютерних злочинів. Ця роль більше зосереджується на тому, що відбувається після витоку даних. Вони розслідують широкий спектр злочинів, від хакерства до інших видів незаконної діяльності.

• Аналітик безпеки даних. Аналітик даних або кібербезпеки зазвичай є внутрішньою посадою, яка зосереджена на виявленні потенційних слабких місць в ІТ-системі. Потім вони запровадять заходи для запобігання зламам, такі як створення брандмауерів і шифрування.

• Адміністратор мережі. Ця роль також зосереджена на підтримці комп’ютерних мереж і вирішенні будь-яких проблем, з якими вони стикаються. Окрім встановлення та налаштування мереж, вони допомагають виявляти та вирішувати будь-які проблеми, що виникають у системах.

**Лекція 5.**

**Проведення розвідки в Інтернеті. OSINT.**

Розвідка відкритих джерел (OSINT) є важливою частиною розвідки кіберзагроз. Це збір та аналіз загальнодоступної інформації з різних джерел для виявлення потенційних загроз інформаційній безпеці.

Використовуючи OSINT, команди безпеки можуть:

▪ виявляти та відстежувати супротивників, які націлені на їхню організацію;

▪ отримувати знання про останні стратегії та тенденції атак;

▪ оцінити ефективність існуючих заходів безпеки;

▪ розробляти проактивні стратегії захисту для мінімізації потенційних загроз.

Існує багато джерел даних OSINT, які можуть бути цінними для розвідки загроз. Деякі з основних джерел включають:

**Загальнодоступні веб-сайти та блоги.** Дослідники безпеки, хакери та суб'єкти загроз часто діляться інформацією про свої знахідки, інструменти та методи у своїх блогах та на веб-сайтах.

**Платформи соціальних мереж.** Платформи соціальних мереж, такі як Twitter, Reddit і LinkedIn, пропонують багато інформації про діяльність суб'єктів загроз і можуть бути цінним ресурсом для розвідки загроз.

**Матеріали конференцій з питань безпеки.** Багато галузевих конференцій і семінарів публікують свої дослідницькі роботи, відеозаписи та презентації в Інтернеті, що дозволяє вам отримати цінну інформацію від експертів у цій галузі.

**Онлайн-форуми та чати.** Хакерські форуми, онлайн-чати та дошки оголошень часто містять дискусії, пов'язані з останніми вразливостями, експлойтами та методами атак.

**Pastebin та GitHub.** Ці платформи пропонують фрагменти коду та репозиторії, які можуть містити робочі хакерські інструменти або експлойти для перевірки концепції, що робить їх цінними джерелами OSINT.

Збір та аналіз OSINT для розвідки загроз може здатися складним завданням, але, дотримуючись цих найкращих практик, ви можете ефективно включити його у свої стратегії кіберзахисту:

**Встановіть чіткі цілі та завдання.** Визначте, чого ви хочете досягти, збираючи OSINT, і як це сприяє ініціативам з розвідки загроз у вашій організації.

**Створіть методологію.** Розробіть структурований підхід і процес для пошуку, збору та аналізу OSINT-даних.

**Фільтруйте дані.** Оскільки обсяг даних, доступних з джерел OSINT, може бути величезним, важливо ефективно фільтрувати зібрані дані. Визначайте пріоритетність інформації, яка має відношення до вашого організаційного контексту і конкретних вимог до розвідданих.

**Підтримуйте актуальність знань.** Регулярно переглядайте нові версії OSINT та будьте в курсі новітніх тактик, методів і процедур, що використовуються зловмисниками.

**Співпрацюйте та діліться інформацією з колегами.** Спільнота фахівців з безпеки відома своєю співпрацею та обміном знаннями. Спілкуйтеся з іншими фахівцями з безпеки, щоб скористатися їхніми знаннями та досвідом.

OSINT є важливим аспектом розвідки загроз, який допомагає організаціям виявляти і зменшувати потенційні загрози безпеці. Ефективно збираючи та аналізуючи OSINT, ви зможете краще зрозуміти ландшафт загроз, що постійно змінюється, і розробити більш ефективні стратегії для захисту вашої організації.

Перший етап злому будь-якої інформаційної системи починається зі збору максимальної кількості інформації про ціль. Практично ніколи не вдається зібрати всю інформацію з одного-єдиного джерела. Дані доводиться

збирати з безлічі різних місць, щоб згодом отримати повну картину інформаційної системи організації.

Зазвичай, маючи лише назву організації, починають збір таких даних:

▪ домени;

▪ мережеві адреси або мережеві блоки;

▪ місцезнаходження;

▪ контактна інформація;

▪ новини про злиття або придбання;

▪ вакансії;

▪ посилання на пов'язані з організацією веб-сервіси;

▪ різні документи;

▪ структура організації.

Це тільки приблизний список, і продовжувати його можна досить довго. Наприклад, переглянувши вакансії підприємства, можна дізнатися, які інформаційні системи використовуються всередині організації. А проаналізувавши HTML-код домашньої сторінки, можна знайти посилання на внутрішні ресурси. Від того, як буде проведено збір інформації, залежить напрямок, а також тип і успішність атаки. Більша частина процесу збору інформації не потребує спеціальних знань, достатньо вміння користуватися пошуковими системами. Найчастіше вони індексують навіть ту інформацію, яку намагалися приховати від зовнішнього світу.

Для прискорення і полегшення процесу пошуку та збору інформації можна використовувати оператори пошуку. Без них відшукати потрібну інформацію буде не просто складно, але практично неможливо. Наприклад, за запитом “Український державний університет науки і технологій” Google видає близько 330 000 результатів. За запитом site:ust.edu.ua - 15 000, а після уточнення site:ust.edu.ua filetype:xls - лише 6. Таким чином, ми відфільтрували тільки те, що нам було цікаво.

Оператори:

▪ site обмежить виведення результатів запиту інформацією з одного сайту (приклад використання – site:ust.edu.ua);

▪ filetype використовується для пошуку файлів певного типу (приклад використання - filetype:doc);

▪ inurl шукає заданий текст тільки в url сайту;

▪ оператор intitle шукає інформацію, виходячи із заголовка документа.

Щоб знайти інформацію, яку організація колись публікувала в Інтернеті, а потім видалила (через допущення помилки або втрату актуальності цієї інформації) можна скористатися сервісом archive.org. Це так званий архів Інтернету, який збирає копії веб-сторінок, графічні матеріали і програмне забезпечення. Архів забезпечує довгострокове архівування зібраного матеріалу та безкоштовний доступ до своїх баз даних для широкого загалу.

Для отримання списку закритих для індексації пошуковими системами директорій, необхідно відкрити файл "robots.txt".

Для аналізу сайту на предмет попередніх версій необхідно скористатися сервісом "waybackmachine.org".

OSINT - це найпоширеніший метод або техніка збору інформації про цільовий домен або співробітника організації з відкритих джерел або загальнодоступних даних. Здебільшого зловмисники використовують цю техніку в атаках соціальної інженерії, фішингу тощо. Але з іншого боку, ми можемо використовувати цю техніку OSINT, розуміючи сферу застосування і знайомлячись з цільовим доменом.

Google Dorking - це метод, який використовується хакерами та дослідниками безпеки для пошуку конфіденційної інформації на веб-сайтах за допомогою пошукової системи Google. Він також відомий як Google Hacking або Google Dorking.

Google Dorking передбачає використання розширених пошукових операцій у Google для пошуку певних ключових слів, типів файлів або параметрів веб-сайтів. Ці оператори можна комбінувати для створення більш

потужних пошукових запитів, які можуть виявити інформацію, яка була б важкодоступною в інший спосіб.

Хоча Google Dorking можна використовувати в законних цілях, наприклад, для дослідження вразливостей безпеки веб-сайтів, хакери зловмисно використовують цю техніку для пошуку конфіденційної інформації, такої як імена користувачів, паролі та іншу потенційну інформацію. Як наслідок, власникам веб-сайтів важливо захищати свої сайти та уникати розміщення конфіденційної інформації в загальнодоступних каталогах.

Загалом, Google Dorking - це потужна технологія, яка може бути використана як в добрих, так і в поганих цілях. Власники веб-сайтів та інтернет-користувачі повинні усвідомлювати його потенційні ризики і вживати заходів для захисту від будь-яких потенційних порушень безпеки.

Як власник/розробник, ви захочете, щоб ваш сайт був захищеним від гугл-доркінгу. Ви можете зробити це, дотримуючись наведених нижче рекомендацій:

- Використовуйте файл Robots.txt: Ви можете заборонити пошуковим системам індексувати певні веб-сторінки або каталоги на вашому сайті за допомогою файлу robots.txt. Таким чином, ви можете перешкодити зловмисникам знайти слабкі місця на вашому сайті. Існує багато модифікацій, які можна внести до robots.txt.

- Вимкнути індексування каталогів: Веб-сервери часто дозволяють сканування каталогів за замовчуванням, що дозволяє будь-кому бачити вміст каталогів. Ви можете запобігти цьому, вимкнувши індексацію каталогів у налаштуваннях вашого веб-сервера.

- Використовуйте брандмауер: Ви можете використовувати WAF (брандмауер веб-додатків) для підвищення безпеки вашого веб-сайту. Це забезпечить вам додатковий рівень безпеки.

- Використовуйте контроль доступу: Ви можете використовувати автентифікацію або MFA (багатофакторну автентифікацію) на сторінках,

якщо не хочете, щоб хтось мав доступ. Це дозволить запобігти несанкціонованому доступу до сайту.

Збір інформації означає збір різних видів інформації про ціль. По суті, це перший крок або початковий етап етичного хакінгу, на якому тестувальники проникнення або хакери (як чорні, так і білі) намагаються зібрати всю інформацію про ціль, щоб використати її для злому. Щоб отримати більш релевантні результати, ми повинні зібрати більше інформації про ціль, щоб збільшити ймовірність успішної атаки.

Відбиток - це цифровий слід вашої активності, який ви залишаєте в Інтернеті. Він схожий на слід, який ви залишаєте на піску на пляжі. Ці сліди можуть бути нешкідливими, наприклад, обліковий запис електронної пошти, про який ви забули, або ж вони можуть видати дуже конфіденційну інформацію через історію переглядів на вашому робочому комп'ютері. Сліди також включають інформацію про те, які соціальні мережі та інші веб-сайти відвідують люди, який контент вони переглядають і як довго, хто їхні друзі у Facebook і коли вони востаннє були онлайн; всі ці дані доступні за допомогою одного кліку в Google або в ряді спеціалізованих пошукових систем.

Існує багато способів, як ваш цифровий слід може бути зібраний і використаний проти вас. Це може зробити третя сторона або навіть ви самі. У більшості випадків використання пошукових систем, таких як Google є найпростішим способом дізнатися інформацію про вас або будь-кого іншого, хто має адресу електронної пошти.

Багато зусиль докладається для того, щоб приховати всю інформацію про вас в Інтернеті, але якщо вона вже потрапила туди, то її вже не повернути назад. Вона може стати надбанням громадськості не одразу, але якщо хтось захоче щось дізнатися, то, швидше за все, з часом знайде те, що шукає.

Якщо ви ще не знайшли ідеальний інструмент OSINT (або його немає в цьому списку), OSINT Framework підкаже вам правильний напрямок.

OSINT Framework - це не конкретне програмне забезпечення, а набір інструментів, які значно спрощують завдання OSINT.

За допомогою цієї платформи ви можете переглядати різні інструменти OSINT, які фільтруються за категоріями.

Безумовно, найсерйозніший тип файлів - це той, який несе в собі облікові дані користувачів або ж усієї компанії. Зазвичай це відбувається двома способами. У першому, сервер налаштований неправильно і виставляє свої адміністративні логи або журнали у відкритому доступі в Інтернеті. Коли паролі змінюються або користувач не може увійти в систему, ці архіви можуть витекти разом з обліковими даними.

Другий варіант відбувається тоді, коли конфігураційні файли, що містять ту саму інформацію (логіни, паролі, найменування баз даних тощо), стають загальнодоступними. Ці файли мають бути обов'язково приховані від будь-якого публічного доступу, оскільки в них часто залишають важливу інформацію. Будь-яка з цих помилок може призвести до того, що зловмисник знайде ці лазівки й отримає всю потрібну інформацію.

У повсякденному використанні пошукові системи, такі як Google приймають пошуковий запит або рядок пошукових запитів і повертають відповідні результати. Також ці ж системи запрограмовані приймати більш просунуті та складні оператори, які значно звужують ці умови пошуку. Оператор - це ключове слово або фраза, що несе особливе значення для пошукової системи. Ось приклади часто використовуваних операторів: "inurl", "intext", "site", "feed", "language". За кожним оператором слідує двокрапка, за якою слідує відповідна ключова фраза або фрази.

Ці оператори дають змогу виконувати пошук більш конкретної інформації, наприклад: певні рядки тексту всередині сторінок веб-сайту або файли, розміщені за конкретною URL-адресою. Крім іншого, Google Dorking може також знаходити приховані сторінки для входу в систему, повідомлення про помилки, що видають інформацію про доступні вразливості та файли загального доступу. Здебільшого причина полягає в тому, що адміністратор веб-сайту міг просто забути виключити з відкритого доступу.

Найбільш практичним і водночас цікавим сервісом Google, є можливість пошуку видалених або архівних сторінок. Це можна зробити за допомогою оператора "cache:". Оператор працює таким чином, що показує збережену (віддалену) версію веб-сторінки, що зберігається в кеші Google.

cache:www.youtube.com

Після здійснення вищевказаного запиту в Google, надається доступ до попередньої або застарілої версії веб-сторінки Youtube. Команда дозволяє викликати повну версію сторінки, текстову версію або саме джерело сторінки. Також вказується точний час індексації, зробленої Google. Сторінка відображається у вигляді графічного файлу, хоч і пошук по самій сторінці здійснюється точно так само як у звичайній сторінці HTML (поєднання клавіш CTRL + F). Результати виконання команди "cache:" залежать від того, як часто веб-сторінка індексувалася роботом Google.

Пошук інформації про користувачів використовується за допомогою розширених операторів, які роблять результати пошуку точними і докладними. Оператор "@" використовується для пошуку індексації користувачів у соціальних мережах: Twitter, Facebook, Instagram.

Здебільшого всі вразливості та загрози, пов'язані з "Dorking", генеруються через неуважність або недбалість користувачів різних програм, серверів або інших веб-пристроїв. Тому правила самозахисту та захисту даних не спричиняють жодних труднощів чи ускладнень.

Для того щоб ретельно підійти до запобігання індексаціям з боку будь-яких пошукових систем, варто звернути увагу на два основні файли конфігурацій будь-якого мережевого ресурсу: ".htaccess" і "robots.txt". Перший - захищає позначені шляхи і директорії за допомогою паролів. Другий - виключає директорії з індексування пошукових систем.

Рекомендується налаштовувати файл robots.txt, який відповідає за індексацію сторінок будь-якого веб ресурсу. Він служить провідником для будь-якої пошукової системи, яка посилається на певні адреси сторінок. І

перш ніж перейти безпосередньо до шуканого джерела, robots.txt буде або блокувати подібні запити, або пропускати їх.

Сам файл розташований у кореневому каталозі будь-якої веб платформи, запущеної в Інтернеті. Конфігурація здійснюється всього лише зміною двох головних параметрів: "User-agent" і "Disallow". Перший виділяє і зазначає або всі, або якісь певні пошукові системи. Тоді як другий зазначає, що саме потрібно заблокувати (файли, каталоги, файли з певними розширеннями тощо).

Google Dorking є невід'ємною частиною процесу збору конфіденційної інформації та процесу її аналізу. Його по праву можна вважати одним із найбільш кореневих і головних інструментів OSINT. Оператори Google Dorking допомагають як у тестуванні свого власного сервера, так і в пошуку всієї можливої інформації про потенційну жертву. Це дійсно дуже яскравий приклад коректного використання пошукових механізмів з метою розвідки конкретної інформації.

Сам термін “Google Dorks” вперше з’явився у 2002 році, коли cybersecurity-експерт Johny Long зібрав пошукові оператори й запити Google (Google Dork Queries) для пошуку вразливостей та сенситивної інформації в інтернеті. Цей список він назвав “Google Dorks”, що походить від англійсьского сленгового слова “dork” — дурник, оказія, опус, курйоз. У 2004 році цей список переріс у цілий проєкт — Google Hacking Database (GHDB).

Для ефективного використання Google Dorks, та й Google загалом, потрібно знати деякі моменти, що стосуються роботи самого гугла:

- регістр не має значення. Не важливо, в якому регістрі, верхньому чи нижньому, пишеш запит, для гугла це одне і теж. Це означає що, наприклад, назви міст або прізвища, можна писати з маленької літери, нічого не зміниться;

- пробіл між словами. Для гугла це логічне "І". Тобто він буде шукати не тільки фразу цілком, а й кожне слово окремо, що, відповідно, сильно збільшує кількість знайдених результатів, і відповідно час, який витратиш, поки перелопатиш їх усі. Для того щоб гугл шукав точний збіг, потрібно

пошуковий запит брати в лапки. І використовувати це варто завжди коли є така можливість, по-перше для отримання більш точних результатів, по-друге для економії часу;

- Google вміє відмінювати слова. Відповідно немає жодного сенсу робити це вручну, крім випадків, коли шукаєш точний збіг, тобто використовуєш лапки;

- Google сортує результати за релевантністю, тобто так, як йому здається найактуальнішим. І думка гугла не завжди збігається з нашими потребами. Звісно, що точніше і коректніше ми зробимо запит, то вищою в результатах буде потрібна нам відповідь. Але, при цьому, завжди є сенс вивчити всі запропоновані нам результати;

- Google шукає тією мовою, якою зроблено запит. У нього, звичайно, виходить коректно перекладати, наприклад назви, але розраховувати на це особливо не варто. Тому найбільш правильний підхід використовувати в запиті ту мову, якою нам потрібен результат;

- на результат впливає історія попередніх запитів. А тому, найпростіший варіант це використовувати "режим інкогніто";

- на результати впливає геопозиція. Це означає, що якщо шукаєш якусь інформацію, наприклад щодо Франції, то буде зовсім не зайвим, якщо твій IP буде французьким;

- найголовніший пункт. Дорки можна і потрібно комбінувати. Тому що поєднання правильних дорків багаторазово зменшує кількість непотрібних результатів і сильно прискорює знаходження результатів потрібних.

Інструменти для OSINT можна знайти на:

https://gridinsoft.ua/osint

https://osintframework.com

Приклади запитів можна знайти на:

https://www.tutorialspoint.com/google\_hacking\_tests.htm

<https://www.exploit-db.com/google-hacking-database>

**Лекція 6.**

**Тестування на проникнення**

Тестування на проникнення (Penetration testing) – це тестування безпеки, в якому експерти імітують реальні атаки, щоб визначити методи обходу засобів захисту програми, системи або мережі. Воно часто включає в себе запуск реальних атак на реальні системи і дані з використанням інструментів і методів, які зазвичай застосовуються зловмисниками.

Тест на проникнення або пентест є одним із основних методів, що використовуються для виявлення областей системи, вразливих до вторгнень і компрометації. Сам процес тестування включає навмисні атаки на систему, які можуть виявити її найслабкіші місця, а також прогалини в безпеці. Тобто група пентестерів діє як справжні хакери та використовує уразливості системи для отримання адміністративного доступу або доступу до конкретної інформації.

Як правило, метою тесту на проникнення є виявлення максимальної кількості реальних вразливостей, щоб швидко їх закрити і перевірити пильність співробітників компанії. Тому для реалізації такого об’ємного завдання використовується комплекс різноманітних засобів, підходів і методів.

Тестування на проникнення має важливе значення, оскільки:

 ідентифікує середовище моделювання, тобто, як зловмисник може атакувати систему через атаку білого капелюха;

 допомагає знайти слабкі зони, де зловмисник може атакувати, щоб отримати доступ до даних і можливостей комп'ютера;

 підтримує запобігання атаці чорного капелюха і захищає вихідні дані;

 оцінює масштаби нападу на потенційний бізнес;

 надає докази того, чому важливо збільшити інвестиції в аспект безпеки технології.

Тестування на проникнення також може бути корисним для визначення:

1. Наскільки добре система толерантна до реальних шаблонів атак.

2. Ймовірний рівень складності, необхідний зловмиснику для успішної компрометації системи.

3. Додаткові контрзаходи, які можуть зменшити загрози для системи.

4. Здатність захисників виявляти атаки та реагувати на них належним чином.

Чим пентест відрізняється від аудиту кібербезпеки?

Насамперед тим, що пентест відтворює справжню кібер-атаку, обходить кіберзахист, проникає в систему та здійснює злам – експлуатує баги, помилки, вразливості цільової системи. Пентест може поставити під загрозу цілісність і доступність системи, адже являє собою санкціонований злам з застосуванням класичних хакерських прийомів та сценаріїв: фаззінг/брутфорс/експлуатація/підвищення привілеїв і т.д.

Аудит кібербезпеки (Vulnerability Scanning & Assessment) — це процес збору, обробки та оцінки вразливостей. Аудит не передбачає хакінгу, а тому майже не впливає на систему і не вносить змін в її роботу чи будову. Аудит лише аналізує, упорядковує та структурує накопичені дані, формує експертні висновки і рекомендації. Методики Аудиту застосовуються пентестерами на різних етапах, однак сам Аудит подається як окрема послуга і є прерогативою більше Аудиторів, ніж Пентестерів. Хоча, на практиці обидві навички сильно пов’язані між собою і хороший пентестер, на нашу думку, повинен бути одночасно й хорошим аудитором.

**Тестування на проникнення та етичний хакінг**

**Тестування на проникнення** є специфічним терміном і зосереджується лише на виявленні вразливостей, ризиків і цільового середовища з метою забезпечення та прийняття контролю над системою.

Іншими словами, тестування на проникнення відповідає оборонним системам відповідної організації, що складаються з усіх комп’ютерних систем та її інфраструктури.

**Етичний хакінг** є загальним терміном, який охоплює всі методи злому і інші пов’язані з ними методи комп’ютерної атаки. Таким чином, разом з виявленням недоліків і вразливостей безпеки, а також забезпеченням безпеки цільової системи, вона виходить за межі злому системи, але з дозволу на захист безпеки для майбутніх цілей. Отже, ми можемо сказати, що це узагальнюючий термін і тестування на проникнення є однією з частин етичного хакінгу.

Основні відмінності між тестуванням на проникнення та етичним хакінгом, наведені у таблиці 1.

|  |  |
| --- | --- |
| Таблиця 1 — Відмінності між тестуванням на проникнення та етичним хакінгом **Тестування на проникнення**  **(Penetration Testing)** | **Етичний хакінг**  **(Ethical Hacking)** |
| Вузький термін зосереджується на тестуванні на проникнення лише для забезпечення безпеки системи. | Комплексний термін і тестування на проникнення є однією з його частин. |
| Тестер повинен мати всебічне знання, тільки про конкретну область, для якої він проводить тестування. | Етичний хакер повинен мати всебічні знання з програмного та апаратного забезпечення. |
| Тестер не обов'язково повинен бути хорошим автором доповіді. | Етичний хакер повинен бути експертом з написання звітів. |
| Будь-який тестер з деякими знаннями тестування на проникнення може виконати тест на проникнення. | Повинен бути фахівцем-експертом у цьому предметі, який має обов'язкову сертифікацію етичних хакерів, щоб бути ефективним. |
| Діловодство менше в порівнянні з | Потрібні детальні паперові роботи, |

**Лекція 7.**

**Oснови криптографії**

Криптографія - це метод захисту комунікації шляхом перетворення відкритого тексту в нерозбірливий шифрований текст. Вона включає в себе різні алгоритми і протоколи для забезпечення конфіденційності, цілісності, автентифікації та неспростування даних.

Слово "криптографія" походить від двох грецьких слів "крипто" - прихований і "графен" - писати.

Коріння криптографії можна знайти в римській та єгипетській цивілізаціях. Нижче наведені деякі з найдавніших видів криптографії:

1. Криптографія ієрогліфів: Найдавніше відоме використання криптографії може бути датоване 1900 роком до н.е. за часів Старого Єгипетського царства у вигляді нестандартних ієрогліфів.

Ієрогліфи були таємною формою комунікації, яку єгиптяни використовували для спілкування один з одним.

Цей таємний текст був відомий лише писарям царів, які передавали повідомлення від їхнього імені.

2. Шифр Цезаря: Стародавні греки були добре відомі завдяки використанню шифрів. Шифр Цезаря або Шифр Зсуву є одним з найперших і найпростіших відомих криптографічних методів. Він є різновидом шифру підстановки, в якому кожен символ у слові замінюється на фіксовану кількість позицій. Наприклад, при зсуві на 3, A замінюється на D, B на E і так далі.

3. Шифр Віженера: У 16 столітті Віженера розробив шифр, в якому ключ шифрування повторюється кілька разів, охоплюючи все повідомлення, а потім текст шифру генерується шляхом додавання символу повідомлення до символу ключа за модулем 26. Цей підхід також вразливий до атак, коли секретність повідомлення залежить від секретності ключа шифрування.

4. Обертова машина Геберна: На початку 19 століття Геберн розробив обертову машину Геберна. У цій машині використовується один ротор, де секретний ключ вбудований в обертовий диск, а ключ має вбудовану таблицю підстановок. Кожне натискання клавіші на клавіатурі призводило до виведення зашифрованого тексту. Цей код зламується за допомогою частот літер.

5. Машина "Енігма": Криптографія відіграла життєво важливу роль у перемозі союзних військ під час Першої та Другої світових воєн. У Другій світовій війні широко використовувалися електромеханічні шифрувальні машини. Історія перемоги союзників над німцями завдяки зламу всесвітньо відомої машини "Енігма" добре відома. Як і всі роторні машини, "Енігма" - це комбінація електромеханічних підсистем. Вона складалася приблизно з трьох-п'яти роторів. Щоразу, коли натискалася клавіша, один або кілька роторів оберталися на шпинделі, і, відповідно, ключ перешифровувався на щось інше.

6. Стандарт шифрування даних (DES) На початку 1970-х років компанія IBM зрозуміла, що її клієнти вимагають певного методу шифрування для захисту даних. Вони створили криптографічну групу на чолі з Хорстом-Фейстелем. Ця група розробила шифр під назвою "Люцифер". У 1973 році Національне бюро стандартів (NBS), яке зараз відоме як Національний інститут стандартів і технологій (NIST), висунуло пропозицію щодо блочного шифру. Зрештою, Люцифер був прийнятий і названий Стандартом шифрування даних (DES). У 1997 році DES був зламаний за допомогою атаки повного перебору. Але згодом його використання було припинено, оскільки було виявлено, що він є незахищеним, особливо проти атак грубої сили, через відносно невеликий розмір ключа.

7. Стандарт попереднього шифрування (AES). У 1997 році NIST знову запропонував новий блоковий шифр. Шифр Rijndael був прийнятий і перейменований в Advanced Encryption Standard (AES). У 2001 році DES був замінений на Advanced Encryption Standard або AES.

8. Національний інститут стандартів і технологій (NIST) протягом 2007-2012 провів конкурс на нову криптографічну хеш-функцію, призначену для заміни SHA-1 і SHA-2 (Алгоритм безпечного хешування). Сімейство хеш-функцій SHA-2 спочатку було розроблено Агентством національної безпеки США (АНБ) і є стандартом для хешування приватних даних. Зараз його використовують криптовалюти та ЦРУ. Організатори опублікували деякі критерії, на яких ґрунтувався вибір фіналістів:

- Безпека

Здатність протистояти атакам зловмисників

- Продуктивність і вартість

Обчислювальна ефективність алгоритму і вимоги до оперативної пам'яті для програмних реалізацій, а також кількість елементів для апаратних реалізацій

- Гнучкість і простота дизайну

Гнучкість в ефективній роботі на найрізноманітніших платформах, гнучкість у використанні паралелізму або розширень ISA для досягнення більш високої продуктивності

До фінального туру потрапили всього 5 алгоритмів:

- BLAKE

- Grøstl

- JH

- Keccak

- Skein

Переможцем і новим SHA-3 став алгоритм Keccak.

Мета криптографії - захистити конфіденційну інформацію шляхом її кодування таким чином, щоб її могли зрозуміти лише уповноважені особи.

Нижче наведено деякі особливості криптографії:

Конфіденційність: Криптографія зберігає вашу конфіденційну інформацію прихованою від хакерів, перетворюючи її в нечитабельну форму.

Цілісність: Криптографія гарантує, що ваші дані залишаються цілими і незмінними під час передачі або зберігання.

Аутентифікація: Криптографія допомагає перевірити особу відправника та підтверджує походження повідомлення.

Відмова від відповідальності: Криптографія не дозволяє відправнику заперечувати свою причетність до повідомлення або транзакції.

Управління ключами: Криптографія безпечно управляє ключами, які використовуються для шифрування та дешифрування.

Масштабованість: Криптографія може працювати з різними рівнями обсягу та складності даних, від окремих повідомлень до великих баз даних.

Інтероперабельність: Криптографія забезпечує безпечний зв'язок між різними системами та платформами.

Адаптивність: Криптографія постійно розвивається, щоб випереджати загрози безпеці та технологічний прогрес.

Процес перетворення відкритого тексту на зашифрований називається шифруванням. Процес шифрування складається з алгоритму та ключа. Ключ - це значення, незалежне від відкритого тексту.

Безпека звичайного шифрування залежить від двох основних факторів: Алгоритм шифрування та Секретність ключа.

Припустимо, ви хочете відправити секретне повідомлення своєму другові. Ви можете написати повідомлення на аркуші паперу і запечатати його в конверт. Однак, якщо хтось перехопить конверт, він зможе відкрити його і прочитати повідомлення. Натомість можна зашифрувати повідомлення за допомогою криптографічного алгоритму. Це перетворило б повідомлення на зашифрований текст, який не може бути прочитаний сторонніми особами.

Потім ви можете надіслати зашифрований текст своєму другові, який зможе розшифрувати його, використовуючи той самий криптографічний алгоритм і ключ. Безпека криптографічної системи залежить від надійності криптографічного алгоритму та секретності ключів. Якщо криптографічний алгоритм слабкий, то можна зламати шифрування і прочитати відкритий текст. Якщо ключі не зберігаються в таємниці, то вони можуть бути скомпрометовані, що дозволить неавторизованим особам розшифрувати зашифрований текст.

Криптографія знаходить широке застосування в різних сферах для забезпечення безпеки даних і захисту конфіденційної інформації.

Безпечний зв'язок: Криптографія уможливлює безпечні канали зв'язку, такі як зашифровані додатки для обміну повідомленнями та віртуальні приватні мережі (VPN), для захисту розмов і даних, що передаються через Інтернет.

Електронна комерція та онлайн-транзакції: Криптографія має вирішальне значення для захисту транзакцій електронної комерції, онлайн-банкінгу та цифрових платіжних систем. Вона захищає конфіденційну фінансову інформацію, таку як дані кредитних карток і персональні ідентифікаційні номери (PIN-коди).

Зберігання паролів: Безпечне зберігання паролів має важливе значення для запобігання несанкціонованому доступу до облікових записів користувачів. Криптографічні методи, такі як хешування та соління, допомагають захистити паролі від легкої компрометації у випадку витоку даних. Коли користувач входить в систему, його пароль хешується і порівнюється з хешем, який зберігався раніше. Паролі хешуються і шифруються перед зберіганням. За допомогою цього методу паролі шифруються таким чином, що навіть якщо хакер отримає доступ до бази даних паролів, він не зможе їх прочитати.

Цифрові валюти: Для захисту транзакцій і запобігання шахрайству цифрові валюти, такі як біткойн, також використовують криптографію. Для захисту транзакцій використовуються складні алгоритми і криптографічні ключі, що робить майже неможливим втручання або підробку транзакцій.

Безпечний перегляд веб-сторінок: Безпека перегляду веб-сторінок забезпечується використанням криптографії, яка захищає користувачів від підслуховування та атак "зловмисника посередині". Криптографія з відкритим ключем використовується в протоколах Secure Sockets Layer (SSL) і Transport Layer Security (TLS) для шифрування даних, що передаються між веб-сервером і клієнтом, створюючи безпечний канал зв'язку.

Електронні підписи: Електронні підписи слугують цифровим еквівалентом власноручного підпису і використовуються для підписання

документів. Цифрові підписи створюються за допомогою криптографії і можуть бути перевірені за допомогою криптографії з відкритим ключем.

Загальний принцип роботи електронних підписів доволі простий. У нас є дані. Ми рахуємо їхній хеш і додаємо до хеша додаткову інформацію - наприклад, час підписання. Потім шифруємо всю отриману інформацію за допомогою приватного ключа. Для уніфікації електронних підписів запроваджено певні стандарти, які визначають порядок пакування підписуваних повідомлень. Але при цьому не визначають необхідні криптографічні алгоритми. Навіщо взагалі тут потрібні стандарти? Уявімо, що для криптографії ми беремо алгоритм sha-256. І який-небудь інший алгоритм для підпису - скажімо, 512-, а не 256-бітний. Після шифрування ми відправляємо підписаний документ одержувачу на перевірку і підпис. Для нього наше повідомлення виглядатиме як хаотичний набір байтів. Щоб перевірити наш підпис, йому потрібно провести низку операцій: самостійно порахувати хеш документа, розшифрувати наш підпис відкритим ключем і порівняти отримані дані. Для цього потрібно знати, які алгоритми використовували ми. Але інформацію про це не можна додати в сам підпис; ці дані потрібно додати поруч. Для опису, як їх додавати, і потрібні стандарти.

Наскрізне шифрування: Наскрізне шифрування використовується для захисту двосторонніх комунікацій, таких як відео-розмови, миттєві повідомлення та електронна пошта. Навіть якщо повідомлення зашифровано, це гарантує, що його зможуть прочитати лише ті, кому воно призначене. Наскрізне шифрування широко використовується в таких комунікаційних додатках, як WhatsApp і Signal, і забезпечує високий рівень безпеки та конфіденційності для користувачів.

Управління цифровими правами (DRM): Системи управління цифровими правами (DRM) використовують криптографію для забезпечення захисту авторських прав і запобігання несанкціонованому копіюванню або

розповсюдженню цифрового контенту, такого як електронні книги, музика і фільми.

Існує кілька видів криптографії, кожен з яких має свої унікальні особливості та сфери застосування. Деякі з найпоширеніших видів криптографії включають

1. Криптографія з симетричним ключем: Цей тип криптографії передбачає використання одного ключа для шифрування і розшифрування даних. І відправник, і одержувач використовують один і той же ключ, який повинен зберігатися в таємниці для забезпечення безпеки комунікації. Системи з симетричним ключем працюють швидше і простіше, але проблема полягає в тому, що відправник і одержувач повинні якось обмінюватися ключами в безпечний спосіб. Найпопулярнішими системами криптографії з симетричним ключем є Data Encryption System (DES) і Advanced Encryption System (AES).

2. Криптографія з асиметричним ключем: Криптографія з асиметричним ключем, також відома як криптографія з відкритим ключем, використовує пару ключів - відкритий і закритий - для шифрування і розшифрування даних. Відкритий ключ доступний будь-кому, тоді як закритий ключ зберігається в таємниці власником. Також називається шифруванням з відкритим ключем. Найпопулярнішим алгоритмом криптографії з асиметричним ключем є алгоритм RSA. Завдяки великій довжині ключа, RSA популярний і широко використовується в Інтернеті. Він використовується браузерами для створення безпечних з'єднань у

незахищених мережах і є частиною багатьох протоколів безпеки, таких як SSH, OpenPGP, S/MIME і SSL/TLS. Шифрування з відкритим ключем є важливим, оскільки неможливо визначити ключ розшифрування, знаючи лише криптографічний алгоритм і ключ шифрування. Завдяки криптосистемі з відкритим ключем, відкриті ключі можуть вільно використовуватися, надаючи користувачам простий і зручний метод шифрування контенту і перевірки цифрових підписів, а закриті ключі можуть зберігатися в таємниці, гарантуючи, що тільки власники закритих ключів можуть розшифровувати контент і створювати цифрові підписи.

Відкриті ключі кожного користувача знаходяться в реєстрі відкритих ключів. Якщо B хоче відправити конфіденційне повідомлення C, то B шифрує повідомлення, використовуючи відкритий ключ C. Коли C отримує повідомлення від B, він може розшифрувати його, використовуючи свій власний приватний ключ. Жоден інший одержувач, окрім С, не зможе розшифрувати повідомлення, тому що тільки С знає приватний ключ С.

Асиметричне шифрування має кілька переваг перед симетричним шифруванням, яке використовує один і той же ключ для шифрування і розшифрування. Однією з головних переваг є те, що воно усуває необхідність обміну секретними ключами, що може бути складним процесом, особливо при спілкуванні з декількома сторонами. Крім того, асиметричне шифрування дозволяє створювати цифрові підписи, які можна використовувати для перевірки автентичності даних. Асиметричне шифрування широко використовується в різних додатках, включаючи безпечне спілкування в Інтернеті, цифрові підписи і безпечну передачу даних.

Загалом, асиметричне шифрування пропонує кілька унікальних функцій, які роблять його важливим інструментом для безпечного спілкування та обміну даними. Однак воно також має деякі обмеження, наприклад, меншу швидкість обробки, що слід враховувати при виборі методу шифрування для конкретної програми.

3. Хеш-функції: Хеш-функція - це математичний алгоритм, який перетворює дані будь-якого розміру у вихідні дані фіксованого розміру. Хеш-функції часто використовуються для перевірки цілісності даних і гарантування того, що вони не були підроблені. У цьому алгоритмі не використовується ніяких ключів. Багато операційних систем використовують хеш-функції для шифрування паролів. Коли ми вводимо дані в цю функцію, вона видає нерегулярне значення. Таке нерегулярне значення називається "хеш-значення". Хеш-значення - це просто числа, але часто вони записуються у шістнадцятковій системі числення.

Якщо вхідні дані однакові, дві хеш-функції завжди дають однаковий результат. Навіть якщо вхідні дані відрізняються на один біт, вихідні значення дуже сильно зміняться. Навіть якщо вхідні дані дуже сильно відрізняються, існує дуже мала ймовірність того, що отримані хеш-значення будуть ідентичними. Якщо вони збігаються, це називається "хеш-колізія". Перетворення хеш-кодів до їх початкового значення є неможливим завданням. У цьому полягає основна відмінність шифрування за допомогою

хеш-функції. Хеш-функції розроблені як односторонні функції, що означає, що легко обчислити хеш-значення для заданих вхідних даних, але складно обчислити вхідні дані для заданого хеш-значення. Ця властивість робить хеш-функції корисними для перевірки цілісності даних, оскільки будь-які зміни в даних призведуть до зміни хеш-значення. Хеш-функції створюють результат фіксованого розміру, незалежно від розміру вхідних даних. Ця властивість робить хеш-функції корисними для зберігання і передачі даних, оскільки хеш-значення можна зберігати або передавати ефективніше, ніж оригінальні дані. Хеш-функції повинні бути стійкими до колізій, тобто важко знайти два різні вхідні дані, які дають однакове хеш-значення. Ця властивість гарантує, що зловмисники не зможуть створити фальшиве повідомлення з таким самим хеш-значенням, що й у справжнього повідомлення. Хеш-функції є незворотними, що означає, що процес генерації хеш-значення важко або неможливо повернути назад, щоб відновити початкові вхідні дані. Ця властивість робить хеш-функції корисними для зберігання паролів або іншої конфіденційної інформації, оскільки за хеш-значенням неможливо відновити початкові дані.

Деякі хеш-функції мають відомі недоліки, наприклад, хеш-функція MD5. Важливо вибрати хеш-функцію, яка є безпечною.

Хоча криптографія є потужним інструментом для захисту інформації, вона також пов'язана з низкою проблем, зокрема:

Управління ключами: Криптографія покладається на використання ключів, якими необхідно ретельно керувати, щоб підтримувати безпеку комунікації.

Квантові обчислення: Розвиток квантових обчислень становить потенційну загрозу для сучасних криптографічних алгоритмів, які можуть стати вразливими до атак.

Людські помилки: Криптографія сильна настільки, наскільки сильна її найслабша ланка, і людська помилка може легко поставити під загрозу безпеку комунікації.

Цифровий підпис - це електронний підпис, який гарантує, що повідомлення було надіслано конкретним відправником. При виконанні цифрових транзакцій автентичність і цілісність повинні бути гарантовані, інакше дані можуть бути змінені або хтось може діяти так, ніби він є відправником, і очікувати відповіді. Для створення цифрового підпису алгоритми підпису, такі як програми електронної пошти, створюють односторонній хеш електронних даних, які потрібно підписати. Потім алгоритм підпису шифрує хеш-значення за допомогою приватного ключа (ключа підпису). Цей зашифрований хеш разом з іншою інформацією, такою як алгоритм хешування, є цифровим підписом. Цей цифровий підпис додається до даних і надсилається верифікатору. Причина шифрування хешу, а не всього повідомлення або документа, полягає в тому, що хеш-функція перетворює будь-який довільний ввід у набагато коротше значення фіксованої довжини. Це економить час, оскільки тепер замість того, щоб підписувати довге повідомлення, потрібно підписувати коротше хеш-значення, до того ж хешування відбувається набагато швидше, ніж підписання.

Верифікатор отримує цифровий підпис разом з даними. Потім він використовує алгоритм верифікації для обробки цифрового підпису та відкритого ключа (ключа верифікації) і генерує деяке значення. Він також застосовує ту ж саму хеш-функцію до отриманих даних і генерує хеш-значення. Потім порівнюється хеш-значення і результат роботи алгоритму верифікації. Якщо вони збігаються, то цифровий підпис є дійсним, в іншому випадку — недійсним.

Цифровий сертифікат - це сертифікат, виданий центром сертифікації (ЦС) для перевірки особи власника сертифіката. ЦС видає зашифрований цифровий сертифікат, що містить відкритий ключ заявника та різноманітну іншу ідентифікаційну інформацію. Цифровий сертифікат використовується для прив'язки відкритого ключа до конкретної фізичної або юридичної особи.

Цифровий сертифікат містить:

▪ ім'я власника сертифіката - серійний номер, який використовується для унікальної ідентифікації сертифіката, фізичної або юридичної особи, ідентифікованої сертифікатом;

▪ дату закінчення терміну дії;

▪ цифровий підпис органу, що видав сертифікат.

Цифровий сертифікат також надсилається разом з цифровим підписом і повідомленням. Цифровий підпис використовується для перевірки автентичності, цілісності, неспростування, тобто він гарантує, що повідомлення надіслано відомим користувачем і не було змінено, в той час як цифровий сертифікат використовується для перевірки особи користувача, можливо, відправника або одержувача. Таким чином, цифровий підпис і сертифікат - це різні речі, але обидва використовуються для забезпечення безпеки. Більшість веб-сайтів використовують цифрові сертифікати для підвищення довіри своїх користувачів.

Класичні атаки. Їх можна розділити на:

Математичний аналіз: Це тип атаки, який використовує структурні недоліки певного алгоритму.

Атаки грубої сили: Зловмисник використовує атаку грубої сили (BFA), щоб спробувати всі потенційні ключі, щоб знайти ключ. Якщо ключ довгий, атака займе багато часу на виконання. Атака перебором виконує алгоритм шифрування для всіх можливих варіантів ключів, поки не буде знайдено збіг. Алгоритм шифрування розглядається як чорний ящик.

Аналітичні атаки - це атаки, які зосереджені на зламі криптосистеми шляхом аналізу внутрішньої структури алгоритму шифрування.

Атаки соціальної інженерії: Це те, що залежить від людського фактору. Обманом змусити когось розкрити свої паролі зловмиснику або дозволити доступ до зони з обмеженим доступом підпадає під цю атаку. Люди повинні бути обережними, розкриваючи свої паролі будь-якій третій стороні, якій не довіряють.

Атаки на реалізацію: Атаки на реалізацію, такі як аналіз побічних каналів, можуть бути використані для отримання секретного ключа. Вони актуальні в тих випадках, коли зловмисник може отримати фізичний доступ до криптосистеми.

Пара відкритих і закритих ключів - це набір з двох унікальних ключів, які використовуються для симетричного і асиметричного шифрування і дешифрування даних. У криптографічній системі, яка використовує пару відкритих і закритих ключів, існує приватний і паралельний відкритий ключ. Закритий ключ залишається конфіденційним і використовується для розшифрування повідомлень або зашифрованих даних за допомогою паралельного відкритого ключа. З іншого боку, люди можуть вільно ділитися відкритим ключем і використовувати його для шифрування повідомлень або даних, призначених для власника закритого ключа.

Важливо переконатися, що для шифрування повідомлення або документа використовується правильний відкритий ключ, оскільки тільки паралельний закритий ключ зможе розшифрувати повідомлення. Так само важливо зберігати приватний ключ у безпеці та захищати його від несанкціонованого доступу, оскільки будь-хто, хто має доступ до нього, зможе розшифрувати конфіденційну інформацію, передану через зашифроване з'єднання.

Відправник генерує пару відкритого і закритого ключів, використовуючи різні форми алгоритмів шифрування, такі як RSA, DSA і ECC (криптографія еліптичних кривих). Люди тримають приватний ключ у таємниці, а публічний ключ надають адресату. Щоб відправити зашифроване повідомлення, відправник використовує відкритий ключ одержувача для шифрування повідомлення. Відправник надсилає зашифроване повідомлення одержувачу. За допомогою приватного ключа одержувач може розшифрувати повідомлення, оскільки тільки той, хто володіє приватним ключем, може

розшифрувати повідомлення, оскільки це єдиний ключ, який може змінити процес шифрування, виконаний за допомогою відкритого ключа.

Так працює комбінація шифрування з відкритим і закритим ключами.

Як створити пару публічних/приватних ключів?

Пари відкритих/закритих ключів - це криптографічний інструмент, який дозволяє шифрувати та розшифровувати дані. Відкритим ключем можна ділитися з будь-ким без побоювань, але закритий ключ ніколи не слід передавати без захисту.

OpenSSL - це криптографічна програмна бібліотека або інструментарій, який робить спілкування через комп'ютерні мережі більш безпечним. Програма OpenSSL - це інструмент командного рядка для використання різних криптографічних функцій криптобібліотеки OpenSSL з оболонки. Зазвичай використовується для протоколів Transport Layer Security (TSL) або Secure Socket Layer (SSL). OpenSSL ліцензовано за ліцензією у стилі apache, що означає, що за певних простих ліцензійних умов можна використовувати інструментарій для комерційних або некомерційних цілей.

Запустіть .exe-файл і встановіть OpenSSL в систему.

Відкрийте командний рядок (cmd) і перенаправте шлях до папки bin.

Кроки для створення пари публічного та приватного ключів

Крок 1: Створення приватного ключа

Введіть команду openssl, натисніть клавішу Enter, а потім скористайтеся наступною командою для створення приватного ключа:

genrsa -out myprivatekey.pem

Ця команда генерує приватний ключ довжиною 512 біт і зберігає його у файлі з розширенням .pem.

Крок 2: Створення відкритого ключа

Введіть команду openssl, натисніть enter, а потім скористайтеся наступною командою для створення відкритого ключа:

rsa -in myprivatekey.pem -pubout -out mypublickey.pem

Ця команда створить відкритий ключ для закритого ключа, створеного на попередньому кроці.

Серед хеш-функцій виділяють слабкі, за якими можна підібрати прообраз, і сильні, де прообраз підібрати не можна. До перших належать, наприклад, md5, sha1, до других - sha2 (sha256, sha384, sha512) і sha3.

Число 256 у назві алгоритму означає, що на виході ми отримаємо рядок фіксованої довжини 256 біт незалежно від того, які дані надійдуть на вхід.

Також виділяють повільні хеш-функції, такі як, наприклад, scrypt. Вони використовуються для шифрування паролів користувачів та інших секретів. Повільні хеш-функції реалізовані так, що їхнє виконання неможливо розпаралелити або оптимізувати. Вони повинні виконуватися в одному потоці, і навіть на потужних конфігураціях розрахунок хеша пароля з scrypt займає 500-1000 мс. Така повільність сильно ускладнює атаку перебором, і для неї потрібен великий обсяг пам'яті з випадковим доступом. Тому для паролів живих користувачів, де простір перебору порівняно невеликий, має сенс використовувати максимально повільні сильні функції, щоб ускладнити зловмисникам роботу - той же scrypt. А для баз даних оптимальнішим вибором буде, наприклад, sha3.

MD5 (Message Digest 5): Створює 128-бітове хеш-значення. Більше не вважається безпечним через вразливість до атак на зіткнення.

SHA-1 (Secure Hash Algorithm 1): Створює 160-бітне хеш-значення. Як і MD5, більше не вважається безпечним через вразливість до атак на зіткнення і поступово знімається з використання.

SHA-256 і SHA-512: частина сімейства SHA-2, SHA-256 генерує 256-бітове хеш-значення, в той час як SHA-512 генерує 512-бітове хеш-значення. Обидва алгоритми широко поширені і вважаються безпечними.

MD5 - це алгоритм криптографічної хеш-функції, який приймає на вхід повідомлення будь-якої довжини і перетворює його на повідомлення фіксованої довжини в 16 байт. MD5 був розроблений в 1991 році Рональдом Рівестом. MD5 вважається небезпечним алгоритмом. Тому зараз рекомендується використовувати SHA256 замість MD5. MD5 не є ні симетричним, ні асиметричним алгоритмом.

Ключі використовуються для керування роботою шифру, щоб тільки правильний ключ міг перетворити зашифрований текст у зашифрований Довжина ключа - це кількість бітів ключа, які використовуються для шифрування повідомлення. Сама по собі довжина не є показником того, наскільки захищеним є зашифрований текст. Для безпечних шифрів, чим довший ключ, тим надійніше шифрування. Ключ повинен бути достатньо великим, щоб атака грубої сили була неможливою.

Часто веб-сайт потребує форми входу для автентифікації своїх користувачів. Зазвичай на таких сайтах багато різних користувачів, і їхні дані для входу потрібно зберігати в якійсь базі даних. Однак зберігати паролі у вигляді відкритого тексту в базі даних неймовірно небезпечно.

Наприклад, уявімо, що хакеру вдалося отримати доступ до бази даних, яка зберігається на веб-сайті. З цієї таблиці стає очевидним, який пароль у кожного користувача. Крім того, хакер отримав доступ до облікового запису адміністратора.

Замість цього веб-сайт вирішує зберігати паролі у вигляді хешів. Розглядаючи цей сценарій, уявімо, що хакеру вдається отримати доступ до нової бази даних веб-сайту. Тепер хакер бачить лише хеш пароля і теоретично не може увійти до жодного з цих акаунтів. Без справжнього пароля хеш (знову ж таки, теоретично) марний для хакера. Але як тепер веб-сайт зможе автентифікувати своїх користувачів? Замість того, щоб звіряти введений користувачем пароль з базою даних, сайт буде перевіряти хеш пароля, який він ввів. Якщо цей хеш збігається з хешем в базі даних, користувач автентифікований!

І, хоча ви завжди можете перейти від пароля до хешу, (теоретично) неймовірно складно повернутися від хешу назад до пароля.

Слово "теоретично" зараз часто використовується, і ось чому.

Щоб зламати хеш, потрібно взяти великий список слів або словник і хешувати кожне слово. Потім ви порівнюєте хеш кожного слова в словнику з хешем, який намагаєтеся зламати. Як тільки ви знайшли збіг, ви знайшли своє слово! Ось чому не рекомендується використовувати загальновживані слова в якості пароля.

Оскільки хакери не здаються лише тому, що паролі захешовані, вони пропустять цей хеш через хеш-крекер (який перевіряє хеші, що найчастіше використовуються), наприклад, такий. Це проста справа - ввести хеш "5f4dcc3b5aa765d61d8327deb882cf99", розгадати reCaptcha і отримати результат

Тепер хакер знає, що пароль pharrell157 - це просто "пароль", який можна підібрати за пару секунд за допомогою безкоштовного онлайн-інструменту.

Це одна з головних причин, чому хеш MD5 не є безпечним. Оскільки це несолоний хеш (на відміну від BCrypt), з одних і тих же даних кожного разу виходить один і той же хеш. Тобто, хеш MD5 "password" завжди буде "5f4dcc3b5aa765d61d8327deb882cf99.

На додаток до простого злому хешу (до якого вразливий кожен хеш), MD5 є неймовірно небезпечним з іншої, більш важливої причини.

У світі існує нескінченна кількість можливих комбінацій будь-якої кількості бітів. Отже, існує нескінченна кількість можливих даних, які можна хешувати. Зверніть увагу на визначення хешу, наведене вище, яке стверджує, що хеш завжди має фіксовану довжину. Наприклад, хеш MD5 завжди має довжину 128 біт (зазвичай це 16 шістнадцяткових байт). Таким чином, існує 2^128 можливих хешів MD5. Хоча це надзвичайно велике число, воно, безумовно, скінченне... хоча кількість можливих паролів, які можна хешувати, нескінченна. Це означає, що нескінченна кількість різних паролів може мати однаковий хеш. Це також означає, що якщо хакер отримує доступ до хешів MD5 паролів, йому не обов'язково потрібно знайти сам пароль, а щось інше, що має такий самий хеш. Завдяки останнім технологічним інноваціям, знайти колізії в хешах MD5 стало майже тривіальним завданням.

На щастя, зараз, коли MD5 не забезпечує того рівня безпеки, на який ми могли б сподіватися, було створено багато нових хешів, які можуть це зробити. Наприклад, SHA-256 (також відомий як SHA-2) є більш безпечним, оскільки має довжину 256 біт замість 128. Зараз більшість веб-сайтів використовують солоні хеші, такі як BCrypt, які можуть створювати різні хеші з одного і того ж пароля, якщо є різна сіль.

Сіль - це випадковий рядок, який додається до відкритого пароля перед його повним хешуванням. Потім сіль можна зберігати (у відкритому або зашифрованому вигляді) разом із хешованим паролем у базі даних. Навіть якщо зловмисник отримає хешований пароль разом з сіллю, існуюча таблиця стане марною, оскільки йому доведеться генерувати нову таблицю з додаванням солі (за умови, що він знає, в яку позицію в чистому паролі додавати сіль). Основне призначення солі - зробити хешований результат пароля унікальним, навіть якщо два користувачі використовують один і той самий пароль. Оскільки солі зазвичай генеруються випадковим чином для

кожного користувача, ймовірність того, що у двох користувачів буде однакова сіль, мінімальна.

Однією з атак на алгоритми хешування коли всі вихідні значення попередньо обчислюються і створюється таблиця. Зловмисник може взяти хеш, знайти його в таблиці і зіставити з вихідними даними. Використання унікального значення солі для кожних даних, що хешуються, захищає від таких таблиць, оскільки для кожного значення солі потрібно було б створювати таблицю, що значно збільшило б час, який витрачає зловмисник. Сіль не є секретом і може бути збережена або надіслана разом з хешом.

У будь-якій криптографічній системі захист ключа є найважливішим аспектом. Розкриття симетричного або закритого ключа означає, що зашифровані дані більше не є приватними. Суворо контролюйте, хто має доступ до введення або перегляду ключів, а також те, як ключі використовуються в додатках. Будь-який код, що реалізує криптографічні процеси та алгоритми, повинен бути переглянутий і перевірений. Переконайтеся, що довжина ключа становить не менше 128 біт.

Ніколи не приймайте в огляді коду алгоритм, створений програмістом для хешування. Завжди використовуйте криптографічні функції, які надаються мовою, фреймворком або загальними (перевіреними) криптографічними бібліотеками. Ці функції добре перевірені і добре протестовані досвідченими криптографами.

Що стосується безпеки, то ніщо не є безпечним назавжди. Особливо це стосується криптографічних функцій хешування. Деякі алгоритми хешування, такі як хеші Windows LanMan, вважаються повністю зламаними. Інші, такі як MD5, які в минулому вважалися безпечними для хешування паролів, мають відомі проблеми, такі як атаки на зіткнення (зверніть увагу, що атаки на зіткнення не впливають на хеші паролів). Рецензент коду повинен розуміти слабкі сторони застарілих функцій хешування, а також сучасні найкращі практики вибору криптографічних алгоритмів.

Найпоширенішою програмною проблемою при хешуванні є:

1. Невикористання значення солі

2. Використовується занадто коротке значення солі

3. Однакове значення солі використовується в декількох хешах

Призначення солі полягає в тому, щоб ускладнити зловмиснику виконання заздалегідь розрахованої атаки на хешування. Візьмемо, наприклад, SHA512 хеш "password", будь-який зловмисник, який використовує таблицю, знайде хеш-значення, що відповідає "password". Беручи до уваги, що на обчислення таблиці для значень до 8 або 10 символів може піти кілька днів або тижнів, зусилля по створенню такої таблиці варті того, якщо програма не використовує ніяких солей. Тепер візьмемо сценарій, коли програма додає сіль "WindowCleaner" до всіх введених паролів. Тепер хеш "password" стає хешем "passwordWindowCleaner". Це навряд чи потрапить до хеш-таблиці зловмисника, однак зловмисник може витратити наступні 10 днів (наприклад) на обчислення нової хеш-таблиці з "WindowCleaner" в кінці кожного рядка з 8-10 символів, і він знову зможе розшифрувати нашу хешовану базу паролів. На останньому кроці програма може створити випадкову сіль для кожного запису і зберегти цю сіль у базі даних разом із хешованим паролем.

Тепер для користувача1 випадкова сіль 'a0w8hsdfas8ls587uas87', що означає, що пароль для хешування 'passworda0w8hsdfas8ls587uas87', а для користувача2 випадкова сіль '8ash87123klnf9d8dq3w', що означає, що пароль для хешування 'password8ash87123klnf9d8dq3w', і так для всіх користувачів. Тепер зловмиснику знадобиться таблиця для кожного пароля користувача, який він хоче розшифрувати - якщо раніше на розшифрування всіх паролів БД з використанням однієї і тієї ж солі йшло 10 днів, то тепер потрібно 10 днів, щоб створити таблицю для пароля користувача1, ще 10 днів для пароля користувача2 і т.д. Якщо було 100 000 користувачів, то тепер 100 000 х 10 днів = 1 000 000 днів або 2738 років, щоб створити таблицю для всіх користувачів. Як бачимо, сіль не обов'язково повинна бути секретною,

оскільки саме той факт, що для кожного користувача використовуються унікальні солі, уповільнює зловмисника.

Один із способів згенерувати значення солі - використовувати генератор псевдовипадкових чисел.

Зверніть увагу, що значення солі не обов'язково має бути криптографічно безпечною випадковою величиною. Найкраще використовувати криптографічну функцію для створення солі, створювати значення солі для кожного хеш-значення і мінімальне значення 128 біт (16 символів). Біти не є дорогими, тому не заощаджуйте кілька бітів, думаючи, що ви виграєте в продуктивності, замість цього використовуйте 256-бітове значення солі.

Провідні криптографи галузі рекомендують не використовувати MD5 і SHA-1 для будь-яких додатків.

Публікація United State FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION (FIPS) визначає сім криптографічних алгоритмів хешування - SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 і SHA-512/256 схвалені для використання на федеральному рівні. Рецензент коду повинен враховувати цей стандарт, оскільки FIPS також широко використовується в індустрії інформаційних технологій.

Рецензент коду повинен підняти червоний прапорець, якщо використовуються MD5 і SHA-1, і провести оцінку ризику, щоб зрозуміти, чому ці функції використовуються замість інших, більш придатних хеш-функцій.

Посолювання запобігає використанню заздалегідь обчислених таблиць: Зловмисники часто використовують заздалегідь обчислені таблиці, такі як таблиці веселки або таблиці пошуку, щоб ефективно зламати хеші паролів. Введення унікальних солей робить ці таблиці неефективними, оскільки вони не враховують варіації хешу пароля, що виникають в результаті додавання солі. Захищає від атак за словником: Оскільки солі створюють унікальні хеші для ідентичних паролів, зловмисники більше не можуть покладатися на

прості атаки по словнику для одночасного злому декількох хешів. Замість цього вони повинні намагатися зламати кожен посолений хеш окремо, що вимагає значно більше часу і ресурсів.

Реалізуючи посол, пам'ятайте про наступні кроки:

Генерація унікальної солі: Коли користувач створює або оновлює свій пароль, за допомогою криптографічно захищеного генератора випадкових чисел генерується унікальна сіль.

Об'єднання солі та пароля: згенерована сіль потім поєднується з паролем користувача за допомогою конкатенації або іншого подібного методу.

Хешування солі та пароля: засолений пароль хешується за допомогою безпечного алгоритму хешування, створюючи унікальний результат хешування.

Зберігання солі та хешованого пароля: сіль та хешований пароль надійно зберігаються в базі даних разом з інформацією про обліковий запис користувача. Сіль необхідна для перевірки пароля під час майбутніх спроб автентифікації.

Використовуйте безпечний генератор випадкових чисел: Використання криптографічно захищеного генератора випадкових чисел мінімізує ймовірність повторення шаблону та підвищує надійність солей.

PKI - це абревіатура "Public key infrastructure", що означає інфраструктуру відкритих ключів, технології, що лежить в основі цифрових сертифікатів. Цифровий сертифікат виконує ту саму мету, що й водійські права або паспорт - це документ, що засвідчує вашу особу та надає певні норми. Цифровий сертифікат дає змогу його власнику шифрувати, підписувати та автентифікувати. Відповідно, PKI - це технологія, яка дає змогу шифрувати дані, підписувати документи в цифровій формі та автентифікувати себе за допомогою сертифікатів.

За останні роки інфраструктура приватних ключів PKI (Public Key Infrastructure) непомітно оточила нас з усіх боків:

Більшість сайтів у мережі Інтернет використовують HTTPS протокол. Для його працездатності необхідно отримувати сертифікати з засвідчувальних центрів (Certificate Authority).

Компанії організовують доступ до своєї IT інфраструктури та інформаційних ресурсів за допомогою ключів і сертифікатів, які співробітники отримують зі спеціальних систем.

Для надсилання документів у державні та комерційні структури потрібен цифровий підпис, який реалізується тими самими механізмами.

Найвідоміша у світі реалізація PKI - HTTPS протокол у мережі Інтернет.

Системи PKI побудовані на криптографії, яка використовує пару ключів: публічний (public key) і приватний (private key). Глобальна мета систем PKI - зв'язати користувачів і відповідні публічні ключі. Користувачами можуть бути як реальні люди, так і IP адреси пристроїв, доменні імена комп'ютерів тощо. Публічний і приватний ключі криптографічно пов'язані між собою. Публічний ключ можна і потрібно поширювати відкрито. Приватний ключ власник має зберігати в секреті. Якщо власник приватного ключа зашифрує дані, то кожен зможе їх розшифрувати за допомогою публічного ключа.

Знаючи публічний ключ будь-хто може зашифрувати дані, які розшифрує тільки власник приватного ключа.

Спрощена схема PKI для HTTPS

CA (Certificate Authority)

Головним елементом систем PKI є засвідчувальний центр (УЦ) - це організація, яка видає цифрові сертифікати користувачам. Із засвідчувальним центром можна взаємодіяти як у ручному режимі, так і організовувати автоматичне отримання сертифікатів.

Кожен браузер і операційна система зберігає заздалегідь визначений список довірчих засвідчувальних центрів.

Сертифікати видають у форматі X.509. Такий сертифікат містить структуру даних, яка складається з ідентифікаційних даних (Identity), додаткових розширень (Extensions) і публічного ключа (Public Key) його власника. Сертифікат підписується кореневим або проміжним CA і його підпис пов'язує Identity власника з його публічним ключем.

Якщо трафік HTTP шифрується і передається за допомогою протоколу TLS, то він називається HTTPS. При цьому шифруються як самі дані, так і HTTP заголовки.

Важливим елементом технології PKI є центр сертифікації (CA). CA - це організація, яка видає цифрові сертифікати.

Для того щоб криптографія з відкритим ключем була ефективною, користувачі мають бути впевнені, що інші сторони, з якими вони спілкуються, "безпечні", тобто їхні особистості та ключі дійсні та заслуговують на довіру. Для забезпечення такої гарантії всі користувачі PKI повинні мати зареєстровані особисті дані. Ці дані зберігаються в стандартному форматі сертифіката цифрового відкритого ключа X.509. Центри сертифікації (CA) представляють людей, процеси та інструменти для створення цифрових сертифікатів, які безпечно пов'язують імена користувачів з їхніми відкритими ключами. Під час створення сертифікатів CA виступають як довірені особи в PKI. Поки користувачі довіряють CA і його бізнес-політикам видачі та управління сертифікатами, вони можуть довіряти сертифікатам, виданим CA.

Доти, доки підпис CA на сертифікаті можна перевірити, сертифікат цілий. Оскільки цілісність сертифіката можна визначити за допомогою перевірки підпису CA, сертифікати безпечні і можуть поширюватися повністю відкрито (наприклад, через загальнодоступні системи каталогів).

Інфраструктура відкритих ключів PKI є найважливішою частиною стратегічної основи ІТ. PKI відіграє важливу роль, оскільки технологія, що базується на сертифікатах, допомагає організаціям встановлювати надійний підпис, шифрування та ідентичність між людьми, системами та речами.

У міру того як бізнес-моделі, що розвиваються, стають дедалі залежнішими від електронних транзакцій і цифрових документів, а також у міру під'єднання до корпоративних мереж більшої кількості пристроїв через Інтернет, роль інфраструктури публічних ключів більше не обмежується ізольованими системами, як-от захищена електронна пошта, смарт-картки для фізичного доступу або зашифрований веб-трафік. Зараз від PKI вимагається підтримка більшої кількості додатків, користувачів і пристроїв у складних екосистемах. Національні та галузеві норми забезпечення безпеки

даних стають жорсткішими, тому основні операційні системи та бізнес-додатки стають дедалі залежнішими від наявності в організації PKI, що забезпечує довіру.

Коли браузер робить запит до Вашого улюбленого веб-сайту, цей запит повинен пройти через безліч різноманітних мереж, будь-яка з яких може бути потенційно використана для прослуховування або для втручання у встановлене з'єднання.

З вашого власного комп'ютера на інші комп'ютери вашої локальної мережі, через роутери і світчі, через вашого провайдера і через безліч інших проміжних провайдерів - величезна кількість організацій ретранслює ваші дані. Якщо зловмисник опиниться хоча б в одній із них - у нього є можливість подивитися, які дані передаються.

Як правило, запити передаються за допомогою звичайного HTTP, в якому і запит клієнта, і відповідь сервера передаються у відкритому вигляді. І є безліч вагомих аргументів, чому HTTP не використовує шифрування за замовчуванням:

- Для цього потрібно більше обчислювальних потужностей

- Передається більше даних

- Не можна використовувати кешування

Але в деяких випадках, коли каналом зв'язку передається винятково важлива інформація (як-от паролі або дані кредитних карток), необхідно забезпечити додаткові заходи, що запобігають прослуховуванню таких з'єднань.

TLS - спадкоємець SSL - це такий протокол, який найчастіше застосовують для забезпечення безпечного HTTP-з'єднання (так званого HTTPS). TLS розташований на рівень нижче від протоколу HTTP у моделі OSI. Пояснюючи на пальцях, це означає, що в процесі виконання запиту

спершу відбуваються всі "речі", пов'язані з TLS-з'єднанням, і вже потім, все що пов'язано з HTTP-з'єднанням.

TLS - гібридна криптографічна система. Це означає, що вона використовує кілька криптографічних підходів.

Одним із найпоширеніших підходів є алгоритм обміну ключами Діффі - Геллмана (DH). Цей алгоритм дає змогу клієнту і серверу домовитися щодо спільного секретного ключа, без необхідності передавання секретного ключа по з'єднанню. Таким чином, зловмисники, які прослуховують канал, не зможуть визначити секретний ключ, навіть якщо вони будуть перехоплювати всі пакети даних без винятку.

Щойно відбувся обмін ключами за DH-алгоритмом, отриманий секретний ключ можна використовувати для шифрування подальшого з'єднання в межах цієї сесії, використовуючи набагато простіше симетричне шифрування.

Математичні функції, що лежать в основі цього алгоритму, мають важливу відмітну особливість - вони відносно просто обчислюються в прямому напрямку, але практично не обчислюються у зворотному.

Алгоритм Діффі-Геллмана дає змогу двом сторонам отримати закритий секретний ключ. Але звідки обидві сторони можуть упевнені, що розмовляють справді одна з одною? Ми ще не говорили про автентифікацію. Що якщо я зателефоную своєму приятелеві, ми здійснимо DH-обмін ключами, але раптом виявиться, що мій дзвінок був перехоплений і насправді я спілкувався з кимось іншим?! Я, як і раніше, зможу безпечно спілкуватися з цією людиною - ніхто більше не зможе нас прослуховувати - але це буде зовсім не той, з ким я думаю, що спілкуюся. Це не дуже безпечно!

Для розв'язання проблеми автентифікації нам потрібна Інфраструктура відкритих ключів, яка дає змогу бути впевненим, що суб'єкти є тими, за кого себе видають. Ця інфраструктура створена для створення, управління,

розповсюдження та відкликання цифрових сертифікатів. Сертифікати - це ті дратівливі штуки, за які потрібно платити, щоб сайт працював за HTTPS.

У найгрубішому наближенні, цифровий сертифікат - це файл, що використовує електронний цифровий підпис, який пов'язує відкритий (публічний) ключ комп'ютера з його приналежністю. Цифровий підпис на сертифікаті означає, що хтось засвідчує той факт, що цей відкритий ключ належить певній особі або організації.

По суті, сертифікати пов'язують доменні імена з певним публічним ключем. Це запобігає можливості того, що зловмисник надасть свій публічний ключ, видаючи себе за сервер, до якого звертається клієнт.

У прикладі з телефоном, наведеному вище, хакер може спробувати пред'явити мені свій публічний ключ, видаючи себе за мого друга, але підпис на його сертифікаті не належатиме тому, кому я довіряю.

Щоб сертифікату довіряв будь-який веб-браузер, він має бути підписаний акредитованим засвідчувальним центром (центром сертифікації, Certificate Authority, CA). CA - це компанії, які виконують ручну перевірку того, що особа, яка намагається отримати сертифікат, задовольняє таким двом умовам:

1. є реально існуючою;

2. має доступ до домену, сертифікат для якого вона намагається отримати.

Щойно CA засвідчується в тому, що заявник - реальний і він реально контролює домен, CA підписує сертифікат для цього сайту, по суті, встановлюючи штамп підтвердження на тому факті, що публічний ключ сайту дійсно належить йому і йому можна довіряти.

У ваш браузер уже від самого початку попередньо завантажено список акредитованих CA. Якщо сервер повертає сертифікат, не підписаний акредитованим CA, то з'явиться велике червоне попередження. В іншому разі, кожен міг би підписувати фіктивні сертифікати.

На додаток до звичайних X.509 сертифікатів, існують Extended validation сертифікати, що забезпечують більш високий рівень довіри. Видаючи такий сертифікат, CA здійснює ще більше перевірок щодо особи, яка отримує сертифікат (зазвичай використовуючи паспортні дані або рахунки).

Під час отримання такого сертифіката браузер відображає в адресному рядку зелену плашку, на додаток до звичайної іконки із замочком.

**Лекція 8.**

**Захист операційної системи**

Зміцнення - це процес захисту певної системи, мережі або програми шляхом зменшення поверхні атаки, посилення заходів безпеки та мінімізації потенційних вразливостей. Основна мета захисту - зменшити ризик, пов'язаний з кіберзагрозами, і захистити систему від несанкціонованого доступу або атак.

Зміцнення операційної системи (OS hardening) - це процес посилення налаштувань безпеки вашої операційної системи для запобігання несанкціонованому доступу, витоку даних та іншим зловмисним діям. Цей крок є важливим для підвищення рівня безпеки вашого пристрою або мережі та мінімізації потенційних кібер-ризиків.

Впровадивши у свої практики кібербезпеки функцію зміцнення операційної системи, ви зможете значно знизити ризики, пов'язані з кіберзагрозами, і захистити цінні активи вашого бізнесу.

У сучасному світі кіберзагроз і вразливостей, що постійно розвиваються, стандартних конфігурацій безпеки, що надаються операційними системами, часто недостатньо. Зміцнення ОС необхідне для того, щоб:

1. Заборонити несанкціонований доступ. Обмежити потенційні точки входу для зловмисників.

2. Закрити прогалини в безпеці. Зменшити ризики використання вразливостей у вашій системі.

3. Запобігти витоку даних. Захистіть конфіденційні дані від кіберзлочинців.

Ключові принципи зміцнення ОС, які можуть допомогти посилити безпеку вашої операційної системи:

1. Найменше привілеїв. Обмежте права та дозволи користувачів, надаючи лише мінімальний доступ, необхідний для виконання основних завдань. Впроваджуйте суворий контроль доступу та розподіл обов'язків.

Обмежуючи доступ і дії, які може виконувати користувач або додаток, ми зменшуємо ризик несанкціонованих дій, проникнення або експлуатації системи.

2. Вимкніть або видаліть непотрібні служби. Непотрібне програмне забезпечення, програми та служби можуть створювати вразливості. Вимикайте їх або видаляйте, коли вони не потрібні.

3. Керування виправленнями. Оновлюйте свою систему та програми найновішими патчами та оновленнями безпеки. Управління виправленнями передбачає, що все програмне забезпечення, операційні системи та додатки мають бути оновлені найновішими виправленнями та оновленнями безпеки. Це гарантує усунення потенційних вразливостей, знижуючи ризик їх використання кіберзлочинцями.

4. Регулярний моніторинг. Впроваджуйте механізми моніторингу для швидкого виявлення та реагування на потенційні загрози. Виконуйте регулярний аудит безпеки ваших систем, мереж і додатків, щоб виявити потенційні прогалини в системі безпеки. Аудит може включати системні журнали, виявлення вторгнень та оцінку вразливостей. Важливо переглядати та виправляти будь-які виявлені недоліки, щоб постійно підтримувати надійну систему безпеки.

5. Автентифікація та безпека паролів. Використовуйте надійні, унікальні паролі та багатофакторну автентифікацію (MFA) для додаткового захисту.

6. Використовуйте кілька рівнів заходів безпеки, щоб запобігти виникненню єдиної точки відмови в системі. Глибинний захист передбачає використання декількох рішень безпеки, таких як брандмауери, системи виявлення вторгнень (IDS), антивірусне програмне забезпечення та політики безпеки для забезпечення цілісного підходу до безпеки.

7. Шифруйте будь-які конфіденційні дані як під час зберігання, так і під час передачі. Шифрування захищає дані, гарантуючи, що навіть якщо вони потраплять до чужих рук, вони залишаться нечитабельними та непридатними для використання.

8. Блокування портів є важливою практикою для посилення безпеки вашої мережі та пристроїв. Воно передбачає обмеження, фільтрацію або повну заборону доступу до певних мережевих портів, щоб мінімізувати вразливість до потенційних кіберзагроз. Обмежуючи доступ до певних портів, ви можете ефективно захистити свої системи від несанкціонованого доступу та зменшити ймовірність порушень безпеки.

Чому блокування портів важливе?

Зменшення поверхні атаки: Кожен відкритий порт є потенційною точкою входу для зловмисників. Блокуючи невикористовувані або непотрібні порти, ви зменшуєте поверхню атаки на вашу мережу.

Захист конфіденційних даних: Обмеження доступу до певних портів може допомогти захистити конфіденційні дані, гарантуючи, що лише авторизовані користувачі матимуть доступ до певних мережевих служб.

Відповідність нормативним вимогам: Різні нормативні акти, такі як PCI DSS, HIPAA та GDPR, вимагають від організацій мати безпечну інфраструктуру захисту даних, яка включає в себе контроль доступу до вашої мережі.

Проаналізуйте свою мережу, щоб визначити, які порти повинні залишатися відкритими для ключових служб і функцій, а які можна безпечно заблокувати.

Розробіть політику, яка визначатиме, які порти слід заблокувати і чому, а також обґрунтуйте, чому дозволено доступ до певних портів.

Налаштуйте брандмауер на пристроях та мережевій інфраструктурі так, щоб він блокував порти, які вважаються відповідними до вашої політики.

Протестуйте свою конфігурацію, щоб переконатися, що доступні лише необхідні порти, а заблоковані порти дійсно заблоковані.

9. Групова політика - це функція в операційних системах Windows, яка дозволяє адміністраторам визначати та керувати конфігураціями, налаштуваннями та політиками безпеки для різних аспектів користувачів і пристроїв у мережі. Ця функція допомагає створювати та підтримувати

узгоджене та безпечне середовище, що має вирішальне значення для організацій будь-якого розміру.

Групова політика працює завдяки підтримці ієрархії об'єктів групової політики (GPO), які містять безліч параметрів політики. Об'єкти GPO можуть бути пов'язані з різними рівнями структури Active Directory (AD), такими як рівні домену, сайту та організаційного підрозділу (OU). Прив'язавши GPO до певних рівнів, ви можете створити середовище, в якому різні налаштування застосовуються до різних груп користувачів і комп'ютерів, залежно від їхнього розташування в структурі AD.

Коли користувач входить у систему або комп'ютер запускається, відповідні об'єкти GPO зі структури AD оцінюються для визначення остаточних параметрів політики. Об'єкти GPO обробляються у певному порядку - локальні, сайти, домени та OU, причому останні мають найвищий пріоритет. Такий порядок гарантує, що ви можете мати базовий набір політик на рівні домену, а більш специфічні політики застосовуються на рівні OU, якщо це необхідно.

Нижче наведено кілька типових сценаріїв, в яких групова політика може використовуватися для застосування політик і параметрів безпеки: Політика паролів: Ви можете використовувати групову політику для визначення мінімальної довжини пароля, вимог до складності, історії паролів і максимального віку пароля для всіх користувачів у домені.

Політики блокування облікових записів: За допомогою групової політики ви можете вказати умови, за яких облікові записи користувачів буде заблоковано, наприклад, після певної кількості невдалих спроб входу. Це допомагає запобігти атакам грубого підбору. Розгортання програмного забезпечення: Розгортайте та керуйте встановленням програмних пакетів та оновлень безпеки у всій мережі. Переконайтеся, що на всіх пристроях встановлені найновіші та найбезпечніші версії програмного забезпечення. Захист пристроїв: Застосовуйте конфігурації для примусового шифрування,

налаштування брандмауера та інших параметрів пристроїв, пов'язаних із безпекою, щоб захистити мережу та конфіденційні дані вашої організації. Призначення прав користувачів: Керування різними правами користувачів, такими як можливість локального або віддаленого входу, доступу до цього комп'ютера з мережі або вимкнення системи. Групи з обмеженим доступом: Керування членством у групах, включаючи групи локальних адміністраторів, щоб гарантувати, що тільки авторизовані користувачі мають підвищені привілеї на цільових пристроях.

10. Списки контролю доступу (ACL) є важливою частиною інфраструктури безпеки, допомагаючи керувати правами доступу до ресурсів і підтримувати безпеку між користувачами, групами і системами.

Списки контролю доступу - це набори правил, які визначають, який користувач, група або система має доступ до певних ресурсів, і який тип доступу вони мають (наприклад, читання або запис). ACL діють як бар'єр для запобігання несанкціонованому доступу до конфіденційних даних і систем; це може допомогти зберегти конфіденційність, цілісність і доступність критично важливих ресурсів.

Існує два основних типи ACL: Дискреційні та обов'язкові.

Дискреційні списки контролю доступу (DACL) дозволяють власнику ресурсу визначати, хто може отримати доступ до ресурсу і який рівень доступу вони можуть мати. Наприклад, користувач або група користувачів може мати права на читання певного файлу, тоді як інша група може мати повний контроль над файлом.

Обов'язкові списки контролю доступу (MACL) покладаються на заздалегідь визначені мітки безпеки або класифікації для забезпечення контролю доступу. У цьому випадку ресурсам присвоюються мітки безпеки, а користувачам або системам надаються дозволи на доступ. Доступ надається лише тоді, коли рівень дозволу користувача відповідає мітці ресурсу.

Кілька найкращих практик, яких ви можете дотримуватися при впровадженні та адмініструванні списків контролю доступу:

Визначте чіткі політики доступу: Встановіть чіткі правила та настанови щодо доступу до ресурсів, наприклад, хто може отримати доступ до певних ресурсів і який тип доступу вони можуть мати.

Використовуйте контроль доступу на основі ролей (RBAC): Призначайте дозволи ролям, а не окремим користувачам. Це допоможе спростити процес управління ACL.

Регулярні аудити та огляди: Періодично переглядайте та оновлюйте ACL, щоб переконатися, що дозволи доступу відповідають бізнес-вимогам та політикам безпеки.

Застосовуйте принцип найменших привілеїв: Надавайте користувачам мінімум привілеїв, необхідних для виконання їхніх завдань.

Підтримуйте процес управління змінами: Документуйте всі зміни до списків доступу, включаючи дату зміни, причину зміни та особу, відповідальну за виконання зміни.

Пам'ятайте, що добре впроваджена і підтримувана система ACL може значно знизити ризики, пов'язані з несанкціонованим доступом до критично важливих ресурсів.

11. "Воронка" (sinkhole) - це механізм безпеки, який використовується в кібербезпеці для перенаправлення та ізоляції шкідливого трафіку, в першу чергу спрямований на захист мереж від розподілених атак на відмову в обслуговуванні (DDoS) і бот-мереж. Основним принципом роботи "чорних дір" є створення "чорної діри", куди спрямовується і контролюється шкідливий трафік, що дозволяє іншим мережевим операціям працювати безперешкодно.

Як працюють воронки:

1. Перенаправлення мережі. Коли зловмисник намагається атакувати мережу, він часто покладається на кілька джерел трафіку або запитів. Sinkholes працюють, перенаправляючи цей вхідний шкідливий трафік на окремий, ізольований сервер або IP-адресу, відомий як сервер sinkhole.

2. Аналіз трафіку: Після перенаправлення шкідливого трафіку "воронка" дає можливість фахівцям з кібербезпеки проаналізувати вхідні дані. Цей аналіз може допомогти визначити характер атаки і, можливо, відстежити її походження.

3. Запобігання та пом'якшення наслідків: Перенаправляючи шкідливий трафік від початкової цілі, "воронки" запобігають або мінімізують наслідки DDoS-атак або діяльності бот-мережі в мережі. Крім того, інформація, може допомогти в розробці нових заходів безпеки для запобігання майбутнім атакам.

У кібербезпеці використовуються в основному два типи "воронок": пасивні та активні.

У пасивній воронці сервер налаштований на пасивне перехоплення і реєстрацію будь-якого шкідливого трафіку, спрямованого на нього. Це дозволяє аналізувати шаблони атак, корисне навантаження даних та іншу корисну інформацію, не вживаючи жодних прямих дій.

З іншого боку, активна "воронка" йде на крок далі, не тільки перехоплюючи і реєструючи шкідливий трафік, але й реагуючи на джерело, потенційно порушуючи роботу зловмисника.

Sinkholes - цінні інструменти в арсеналі кібербезпеки, які допомагають запобігати та пом'якшувати наслідки DDoS-атак і бот-мереж. Ізолюючи шкідливий трафік, вони не лише мінімізують вплив атак на мережі, але й надають цінну інформацію про моделі атак, що сприяє розробці більш надійних заходів кібербезпеки.

Щоб ефективно захистити свої кінцеві точки, розгляньте можливість впровадження наступних заходів:

Антивірус і захист від шкідливих програм: Переконайтеся, що на кожному кінцевому пристрої встановлено сучасне антивірусне програмне забезпечення. Це допоможе виявити та видалити шкідливі файли, не даючи їм завдати шкоди вашій мережі.

Керування виправленнями: Будьте в курсі останніх виправлень безпеки для ваших операційних систем і сторонніх додатків. Регулярне оновлення програмного забезпечення допоможе захиститися від вразливостей, якими можуть скористатися кіберзлочинці.

Керування пристроями: Впровадьте централізоване рішення для управління пристроями, яке дозволить адміністраторам контролювати, керувати та захищати кінцеві точки. Це включає в себе застосування політик безпеки, відстеження інвентаризації пристроїв та віддалене очищення втрачених або викрадених пристроїв.

Контроль доступу: Обмежте доступ до конфіденційних даних, впровадивши сувору політику контролю доступу. Надавайте необхідні дозволи лише тим, хто їх потребує, і використовуйте методи автентифікації, такі як багатофакторна автентифікація (MFA), для перевірки особи користувачів. Шифрування: Шифруйте конфіденційні дані, що зберігаються на кінцевих пристроях, щоб запобігти несанкціонованому доступу до них у разі крадіжки або втрати пристрою.

Брандмауер та захист від вторгнень: Розгортання брандмауерів і систем запобігання вторгненням для блокування зовнішніх загроз і оповіщення адміністраторів про потенційні атаки.

**Захист операційних систем**

macOS, операційна система Apple для комп'ютерів Macintosh, має репутацію надійної системи безпеки. Apple розробила macOS з кількома вбудованими функціями для захисту конфіденційності та даних користувачів:

Gatekeeper: Гарантує, що завантажені програми походять з надійних джерел.

FileVault: шифрування всього диска для захисту даних.

XProtect: Антивірусний інструмент, який сканує щойно встановлені програми на наявність шкідливого програмного забезпечення.

Linux - це операційна система з відкритим вихідним кодом, яка вважається більш безпечною, ніж її комерційні аналоги. Linux використовує багатокористувацьке середовище, пом'якшуючи вплив потенційних загроз шляхом розділення інформації та привілеїв користувачів. Інші важливі особливості включають:

Репозиторії програмного забезпечення: Офіційні репозиторії програмного забезпечення, що підтримуються дистрибутивами Linux, забезпечують надійні джерела для встановлення програмного забезпечення.

SELinux (Security-Enhanced Linux): Архітектура безпеки, яка дозволяє адміністраторам контролювати доступ до системи.

Оновлення системи: Регулярні оновлення, що пропонуються дистрибутивами, містять важливі виправлення безпеки.

Хоча дистрибутиви Linux є меншою мішенню для кіберзлочинців, життєво важливо дотримуватися найкращих практик безпеки, таких як постійне оновлення системи та використання інструментів безпеки, таких як антивірусне програмне забезпечення та брандмауери.

Пам'ятайте, що безпека вашої операційної системи залежить від своєчасного оновлення, правильної конфігурації та використання відповідних інструментів безпеки. Будьте пильними та поінформованими, щоб забезпечити захист вашої системи від кіберзагроз, які постійно змінюються.

Прискорення цифрової трансформації та поширення віддаленої та гібридної роботи відкриває нові можливості для організацій, громад та окремих осіб. Разом з тим, з'являються нові загрози та ризики.

**Захист операційної системи Windows**

Організації по всьому світу переходять на модель безпеки Zero Trust, в основі якої лежить передумова, що жодна людина чи пристрій не може отримати доступ, доки не буде доведено безпеку та цілісність. Windows 11 побудована на принципах нульової довіри, щоб забезпечити гібридну продуктивність і нові можливості в будь-якому місці без шкоди для безпеки. Windows 11 підвищує базовий рівень безпеки, висуваючи нові вимоги до вдосконаленого захисту апаратного та програмного забезпечення, що поширюється від мікросхем до хмарних сховищ.

Як Windows 11 забезпечує захист за принципом нульової довіри ?

Модель безпеки Zero Trust надає потрібним людям потрібний доступ у потрібний час. Безпека Zero Trust ґрунтується на трьох принципах: Зниження ризику шляхом явної перевірки таких даних, як ідентичність користувача, місцезнаходження та стан пристрою, для кожного без винятку запиту на доступ. Після перевірки надавати людям і пристроям доступ лише до необхідних ресурсів на необхідний проміжок часу.

Щоб захистити особисту та ділову інформацію, Windows 11 має кілька рівнів захисту додатків, які захищають важливі дані та цілісність коду. Ізоляція та контроль додатків, цілісність коду, засоби контролю конфіденційності та принцип найменших привілеїв дають змогу розробникам вбудовувати засоби безпеки та конфіденційності з нуля. Ця інтегрована система безпеки захищає від зломів і шкідливого програмного забезпечення, допомагає зберегти конфіденційність даних і надає ІТ-адміністраторам необхідні засоби контролю.

У Windows 11 Microsoft Defender Application Guard використовує технологію віртуалізації Hyper-V для ізоляції ненадійних веб-сайтів і файлів Microsoft Office у контейнерах, відокремлених від основної операційної системи та корпоративних даних, які не мають до них доступу. Для захисту конфіденційності Windows 11 також надає більше можливостей контролю над тим, які програми та функції можуть збирати та використовувати такі

дані, як місцезнаходження пристрою або доступ до таких ресурсів, як камера та мікрофон.

Паролі вже давно є важливою частиною цифрової безпеки, а також однією з головних цілей кіберзлочинців. Windows 11 забезпечує потужний захист від крадіжки облікових даних за допомогою апаратного захисту на рівні мікросхем. Облікові дані захищені рівнями апаратного та програмного захисту, що ускладнює зловмисникам викрадення облікових даних з пристрою. Користувачі можуть швидко входити в систему за допомогою обличчя, відбитка пальця або PIN-коду для захисту без пароля.

Windows містить багато функцій безпеки, які допомагають захистити користувачів від шкідливого програмного забезпечення та атак. Однак довіряти компонентам безпеки Windows можна лише тоді, коли платформа завантажується належним чином і не була підроблена. Windows покладається на безпечне завантаження Unified Extensible Firmware Interface (UEFI), антивірусне програмне забезпечення раннього запуску (ELAM), динамічний корінь довіри для вимірювання (DRTM), довірене завантаження та інші низькорівневі апаратні та мікропрограмні засоби безпеки.

Коли ви вмикаєте комп'ютер, доки не запуститься антивірус, Windows використовує відповідну конфігурацію апаратного забезпечення, щоб забезпечити вашу безпеку. Виміряне та надійне завантаження, реалізоване завантажувачами та BIOS, перевіряє та криптографічно записує кожен крок завантаження ланцюжком. Ці події прив'язані до співпроцесора безпеки (TPM), який діє як корінь довіри. Віддалена атестація - це механізм, за допомогою якого ці події зчитуються і перевіряються сервісом для надання достовірного, об'єктивного і стійкого до несанкціонованого втручання звіту. Віддалена атестація - це довірений аудитор завантаження вашої системи, що дозволяє певним суб'єктам довіряти пристрою.

Безпечне завантаження та Довірене завантаження допомагають запобігти завантаженню шкідливого програмного забезпечення та пошкоджених компонентів під час запуску пристрою.

Безпечне завантаження починається із захисту початкового завантаження, а потім процес підхоплює Trusted Boot. Разом Secure Boot і Trusted Boot допомагають забезпечити безпечне та надійне завантаження системи.

Вимірюване завантаження вимірює всі важливі параметри коду та конфігурації під час завантаження Windows. Сюди входять: мікропрограма, диспетчер завантаження, гіпервізор, ядро, захищене ядро та операційна система. Measured Boot зберігає результати вимірювань в TPM на машині і робить їх доступними в журналі, який можна перевірити віддалено для перевірки стану завантаження клієнта.

Функція вимірюваного завантаження надає антивірусному програмному забезпеченню достовірний (стійкий до підміни та фальсифікації) журнал усіх компонентів завантаження, які запускалися до нього. Антивірусне програмне забезпечення може використовувати цей журнал, щоб визначити, чи компоненти, які працювали до нього, заслуговують на довіру, чи вони заражені шкідливим програмним забезпеченням. Антивірусне програмне забезпечення на локальному комп'ютері може надіслати журнал на віддалений сервер для оцінки. Віддалений сервер може ініціювати дії з виправлення, взаємодіючи з програмним забезпеченням на клієнті або за допомогою позамережевих механізмів, залежно від ситуації.

Процес атестації стану пристроїв Windows підтримує парадигму нульової довіри, яка зміщує фокус зі статичних, мережевих периметрів на користувачів, активи та ресурси. Процес атестації підтверджує, що пристрій, мікропрограма і процес завантаження знаходяться в належному стані і не були підроблені, перш ніж вони зможуть отримати доступ до корпоративних ресурсів. Визначення здійснюється за допомогою даних, що зберігаються в TPM, який забезпечує безпечний корінь довіри.

Протягом багатьох років Windows зробила значні кроки для покращення своєї безпеки. Деякі з функцій безпеки включають:

Захисник Windows (Windows Defender): Вбудоване антивірусне програмне забезпечення, яке забезпечує захист від шкідливих програм, програм-вимагачів та інших загроз у режимі реального часу. Антивірус Microsoft Defender - це рішення для захисту, що входить до складу всіх версій Windows. З моменту завантаження Windows Microsoft Defender Antivirus постійно відстежує шкідливе програмне забезпечення, віруси та загрози безпеці. Оновлення завантажуються автоматично, щоб забезпечити безпеку вашого пристрою та захистити його від загроз. Microsoft Defender Antivirus включає антивірусний захист у режимі реального часу, на основі поведінки та евристичний захист. Microsoft Defender SmartScreen захищає від фішингу, шкідливих веб-сайтів і програм та завантаження потенційно шкідливих файлів. Для посилення захисту від фішингу SmartScreen також сповіщає користувачів, коли вони вводять свої облікові дані в потенційно небезпечному місці.

Microsoft Defender SmartScreen захищає від фішингових або шкідливих веб-сайтів і програм, а також від завантаження потенційно шкідливих файлів.

Microsoft Defender SmartScreen визначає, чи є сайт потенційно шкідливим, за допомогою:

Аналізуючи відвідані веб-сторінки та шукаючи ознаки підозрілої поведінки. Якщо Microsoft Defender SmartScreen визначає, що сторінка є підозрілою, він показує попереджувальну сторінку з застереженням про обережність.

Перевірка відвіданих сайтів за динамічним списком повідомлень про фішингові сайти та сайти зі шкідливим програмним забезпеченням. Якщо він знаходить збіг, Microsoft Defender SmartScreen показує попередження, щоб повідомити користувачеві, що сайт може бути зловмисним.

Microsoft Defender SmartScreen визначає, чи є завантажена програма або інсталятор програми потенційно шкідливою, за допомогою:

Перевіряючи завантажені файли за списком сайтів і програм, про які повідомлялося про зловмисне програмне забезпечення та які вважаються

небезпечними. Якщо він знаходить збіг, Microsoft Defender SmartScreen показує попередження, щоб повідомити користувачеві, що сайт може бути зловмисним.

Перевірка завантажених файлів у списку файлів, які добре відомі та часто завантажуються. Якщо файлу немає в цьому списку, Microsoft Defender SmartScreen показує попередження, закликаючи до обережності.

Переваги Microsoft Defender SmartScreen

Захист від фішингу та шкідливого програмного забезпечення: Microsoft Defender SmartScreen допомагає захистити користувачів від сайтів, які, як повідомляється, проводять фішингові атаки або намагаються поширювати шкідливе програмне забезпечення. Він також може допомогти захистити від оманливих рекламних оголошень, шахрайських сайтів і атак зловмисників. Drive-by атаки - це веб-атаки, які, як правило, починаються на довірених сайтах і націлені на вразливості безпеки в загальновживаному програмному забезпеченні. Оскільки drive-by атаки можуть відбуватися, навіть якщо користувач нічого не вибирає і не завантажує на сторінці, небезпека часто залишається непоміченою.

Захист URL-адрес і програм на основі репутації: Microsoft Defender SmartScreen оцінює URL-адреси веб-сайтів, щоб визначити, чи відомо, що вони поширюють або містять небезпечний вміст. Він також перевіряє репутацію програм, перевіряє завантажені програми та цифровий підпис, який використовується для підписання файлів. Якщо URL-адреса, файл, програма або сертифікат мають встановлену репутацію, користувачі не бачать жодних попереджень. Якщо репутація відсутня, елемент позначається як більш ризикований, і користувач отримує попередження.

Інтеграція з операційною системою: Microsoft Defender SmartScreen інтегрований в операційну систему Windows 10. Він перевіряє будь-які файли, які намагаються завантажити та запустити програми (включно зі сторонніми браузерами та поштовими клієнтами).

Покращена евристика та діагностичні дані: Microsoft Defender SmartScreen постійно навчається і намагається залишатися в курсі подій, щоб допомогти вам захиститися від потенційно шкідливих сайтів і файлів.

Wi-Fi Protected Access (WPA) - це програма сертифікації безпеки, призначена для захисту бездротових мереж. WPA3 є останньою версією сертифікації і забезпечує більш безпечний і надійний метод підключення в порівнянні з WPA2 і більш старими протоколами безпеки.

Брандмауер Windows: Ця функція допомагає захистити ваш пристрій від несанкціонованого доступу або вторгнення, блокуючи потенційно шкідливі мережеві з'єднання. Брандмауер Windows з розширеним захистом забезпечує двосторонню фільтрацію мережевого трафіку на основі хоста, блокуючи несанкціонований трафік, що надходить на локальний пристрій або виходить з нього, на основі типів мереж, до яких підключено пристрій. Брандмауер Windows зменшує поверхню атаки на пристрій за допомогою правил, що обмежують або дозволяють трафік за багатьма властивостями, такими як IP-адреси, порти або програмні шляхи. Зменшення поверхні атаки пристрою підвищує керованість і знижує ймовірність успішної атаки.

Контроль облікових записів користувачів (UAC): UAC допомагає запобігти несанкціонованим змінам системних налаштувань, запитуючи у користувачів дозвіл адміністратора під час внесення змін до системи.

Windows Update: Регулярні оновлення гарантують, що у вашій системі будуть встановлені найновіші патчі безпеки, виправлення помилок і поліпшення функцій.

BitLocker: Функція шифрування дисків, доступна в деяких версіях Windows, BitLocker допомагає захистити ваші дані, забезпечуючи шифрування жорсткого диска або зовнішніх пристроїв зберігання даних. BitLocker Drive Encryption - це функція захисту даних, яка інтегрується з операційною системою та усуває загрози крадіжки або витоку даних із загублених, викрадених або неналежним чином виведених з експлуатації комп'ютерів. BitLocker використовує алгоритм AES в режимі роботи XTS або

CBC з довжиною ключа 128 або 256 біт для шифрування даних на томі. Для збереження вмісту ключів відновлення можна використовувати хмарне сховище Microsoft OneDrive або Azure.

Зашифровані жорсткі диски - це клас жорстких дисків, які самошифруються на апаратному рівні та забезпечують повне апаратне шифрування диска, залишаючись при цьому прозорими для користувача пристрою. Ці диски поєднують в собі переваги безпеки та керування, що надаються BitLocker Drive Encryption, з потужністю самошифрування дисків.

Перекладаючи криптографічні операції на апаратне забезпечення, зашифровані жорсткі диски підвищують продуктивність BitLocker і зменшують навантаження на процесор та енергоспоживання.

Bitlocker - це функція шифрування дисків Windows, призначена для захисту даних шляхом шифрування цілих томів.

BitLocker усуває загрози крадіжки даних або їх витоку з загублених, викрадених або неналежним чином виведених з експлуатації пристроїв.

BitLocker забезпечує максимальний захист при використанні з модулем довіреної платформи (TPM). TPM - це апаратний компонент, встановлений на багатьох пристроях, який працює з BitLocker, щоб допомогти захистити дані користувача та гарантувати, що комп'ютер не був зламаний, коли система не в режимі офлайн.

На пристроях, які не мають TPM, BitLocker можна використовувати для шифрування диска з операційною системою Windows. Однак ця реалізація вимагає, щоб користувач вставив ключ запуску USB, щоб запустити пристрій або вийти зі сплячого режиму. Для захисту тома операційної системи на комп'ютері без TPM можна використовувати пароль тома операційної системи. Обидва варіанти не забезпечують перевірку цілісності системи перед запуском, яку пропонує BitLocker з TPM.

На додаток до TPM, BitLocker пропонує можливість заблокувати звичайний процес запуску, поки користувач не надасть персональний ідентифікаційний номер (PIN) або не вставить знімний пристрій (наприклад,

USB-флешку), що містить ключ запуску. Ці додаткові заходи безпеки забезпечують багатофакторну автентифікацію та гарантують, що комп'ютер не ввімкнеться або не вийде зі сплячого режиму, доки не буде введено правильний PIN-код або ключ запуску.

Windows має кілька рівнів захисту для програм і даних на комп'ютері. Антивірус Windows Defender використовує хмарну систему виявлення в режимі реального часу, щоб виявляти та поміщати на карантин програми, про які відомо, що вони є зловмисними. Windows Defender SmartScreen попереджає користувачів, перш ніж дозволити їм запустити ненадійну програму, навіть якщо вона визнана шкідливою. Перш ніж програма зможе змінити системні налаштування, користувачеві доведеться надати їй адміністративні привілеї за допомогою Керування обліковими записами користувачів.

Ці компоненти - лише деякі з способів, за допомогою яких Windows захищає вас від шкідливого програмного забезпечення. Однак ці засоби захисту діють лише після запуску Windows. Сучасні шкідливі програми, зокрема буткіти, здатні запускатися до запуску Windows, повністю оминаючи захист ОС і залишаючись невидимими.

Запуск Windows 10 або Windows 11 на комп'ютері з підтримкою уніфікованого розширюваного інтерфейсу мікропрограми (UEFI) гарантує, що надійне завантаження захистить ваш комп'ютер від шкідливого програмного забезпечення з моменту його ввімкнення. Цей захист триває доти, доки його не перебере на себе ваше антивірусне програмне забезпечення. Якщо зловмиснику все ж вдасться заразити ваш комп'ютер, він не зможе залишитися непоміченим. Trusted Boot може перевірити цілісність системи до вашої інфраструктури таким чином, що шкідливе програмне забезпечення не зможе її замаскувати. Навіть для комп'ютерів без UEFI Windows пропонує покращений захист під час запуску порівняно з попередніми версіями Windows.

Руткіти - це складний і небезпечний тип шкідливого програмного забезпечення. Вони працюють у режимі ядра, використовуючи ті ж привілеї, що й ОС. Оскільки руткіти мають ті ж права, що й ОС, і запускаються раніше за неї, вони можуть повністю приховувати себе та інші програми. Часто руткіти є частиною цілого набору шкідливих програм, які можуть обходити локальні логіни, записувати паролі та натискання клавіш, передавати приватні файли та перехоплювати криптографічні дані.

Windows підтримує чотири функції, які допомагають запобігти завантаженню руткітів і буткітів під час запуску:

- Безпечне завантаження. Комп'ютери з мікропрограмою UEFI і модулем довіреної платформи (TPM) можна налаштувати на завантаження лише перевірених завантажувачів ОС.

- Довірене завантаження. Windows перевіряє цілісність кожного компонента процесу запуску перед завантаженням.

- Антивірус раннього запуску (ELAM). ELAM перевіряє всі драйвери перед завантаженням і запобігає завантаженню несанкціонованих драйверів.

- Вимірювання завантаження. Вбудоване програмне забезпечення ПК записує процес завантаження, і Windows може надсилати його на довірений сервер, який може об'єктивно оцінити стан комп'ютера.

На рисунку 1 показано процес запуску Windows.

Безпечне завантаження. Коли комп'ютер запускається, він спочатку знаходить завантажувач операційної системи. Комп'ютери без захищеного завантаження запускають будь-який завантажувач, який є на жорсткому диску комп'ютера. Комп'ютер не може визначити, чи це надійна ОС, чи руткіт. Під час запуску ПК з UEFI комп'ютер спочатку перевіряє наявність цифрового підпису мікропрограми, що зменшує ризик руткітів мікропрограми. Якщо увімкнено безпечне завантаження, прошивка перевіряє цифровий підпис завантажувача, щоб переконатися, що його не було змінено. Якщо завантажувач не пошкоджено, мікропрограма запускає його лише у випадку, якщо виконується одна з наступних умов:

- Завантажувач було підписано за допомогою довіреного сертифіката. Для комп'ютерів, сертифікованих для Windows, довіреним є сертифікат Microsoft.

- Користувач вручну схвалив цифровий підпис завантажувача. Ця дія дозволяє користувачеві завантажувати операційні системи інших виробників.

У вас є три варіанти запуску операційних систем не від Microsoft:

- Використовувати ОС із сертифікованим завантажувачем. Оскільки всі комп'ютери, сертифіковані для Windows, повинні довіряти сертифікату Microsoft, корпорація Майкрософт пропонує послугу з аналізу та підписання будь-якого завантажувача не від Microsoft, щоб йому довіряли всі комп'ютери, сертифіковані для Windows. Насправді, завантажувач з відкритим кодом, здатний завантажувати Linux, вже доступний.

- Налаштуйте UEFI так, щоб він довіряв вашому кастомному завантажувачу. Всі ПК, сертифіковані для Windows, дозволяють довіряти несертифікованому завантажувачу шляхом додавання підпису до бази даних UEFI, що дозволяє запускати будь-яку ОС, включаючи саморобні операційні системи. - Вимкніть безпечне завантаження. На всіх комп'ютерах, сертифікованих для Windows, ви можете вимкнути безпечне завантаження, щоб запустити будь-яке програмне забезпечення. Однак це не допоможе вам захиститися від буткітів. Щоб запобігти зловживанню цими параметрами шкідливими програмами, користувач повинен вручну налаштувати прошивку UEFI на довіру несертифікованому завантажувачу або вимкнути безпечне завантаження. Програмне забезпечення не може змінити налаштування безпечного завантаження.

Довірене завантаження

Довірене завантаження починається там, де закінчується безпечне завантаження. Перед завантаженням завантажувач перевіряє цифровий підпис ядра Windows. Ядро Windows, у свою чергу, перевіряє всі інші

компоненти процесу запуску Windows, включаючи завантажувальні драйвери, файли запуску та ELAM. Якщо файл було змінено, завантажувач виявляє проблему і відмовляється завантажувати пошкоджений компонент. Часто Windows може автоматично відремонтувати пошкоджений компонент, відновивши цілісність Windows і дозволивши комп'ютеру нормально запуститися.

Захист від раннього запуску шкідливого програмного забезпечення

Оскільки безпечне завантаження захистило завантажувач, а надійне завантаження - ядро Windows, наступною можливістю для запуску шкідливого програмного забезпечення є зараження завантажувального драйвера не від Microsoft. Традиційні програми для захисту від шкідливих програм запускаються лише після завантаження завантажувальних драйверів, що дає можливість руткіту, замаскованому під драйвер, працювати.

Early Launch Anti-Malware (ELAM) може завантажити антивірусний драйвер Microsoft або іншого виробника перед усіма іншими завантажувальними драйверами та програмами, таким чином продовжуючи ланцюжок довіри, встановлений безпечним завантаженням і довіреним завантаженням. Оскільки ОС ще не стартувала, а Windows потрібно завантажити якомога швидше, перед ELAM стоїть просте завдання: перевірити кожен завантажувальний драйвер і визначити, чи є він у списку довірених драйверів. Якщо йому не довіряють, Windows не завантажить його.

Драйвер ELAM не є повнофункціональним рішенням для захисту від зловмисного програмного забезпечення; він завантажується пізніше в процесі завантаження. Захисник Windows (входить до складу Windows) підтримує ELAM, як і деякі інші програми для захисту від зловмисного програмного забезпечення інших виробників.

Виважене завантаження. Якщо комп'ютер було заражено руткітом, ви повинні знати про це. Корпоративні антивірусні програми можуть повідомляти ІТ-відділ про зараження шкідливим програмним забезпеченням, але це не працює з руткітами, які приховують свою присутність. Іншими

словами, ви не можете довіряти клієнту, що він скаже вам, чи він здоровий. Як наслідок, комп'ютери, заражені руткітами, виглядають здоровими, навіть якщо на них працює антивірусне ПЗ. Заражені комп'ютери продовжують підключатися до корпоративної мережі, надаючи руткіту доступ до величезних обсягів конфіденційних даних і потенційно дозволяючи руткіту поширитися по всій внутрішній мережі. Measured Boot працює з TPM і програмним забезпеченням не від Microsoft в Windows. Він дозволяє довіреному серверу в мережі перевіряти цілісність процесу запуску Windows. Measured Boot використовує наступний процес:

1. Прошивка UEFI ПК зберігає в TPM хеш прошивки, завантажувача, завантажувальних драйверів і всього, що завантажується перед програмою захисту від шкідливого програмного забезпечення.

2. Наприкінці процесу запуску Windows запускає віддалений клієнт атестації не від Microsoft. Довірений сервер атестації надсилає клієнту унікальний ключ.

3. TPM використовує унікальний ключ для цифрового підпису журналу, записаного UEFI.

4. Клієнт надсилає журнал на сервер, можливо, разом з іншою інформацією про безпеку.

Залежно від реалізації та конфігурації, сервер тепер може визначити, чи здоровий клієнт. Він може надати клієнту доступ або до обмеженої карантинної мережі, або до повної мережі.

На рисунку 2 показано процес вимірюваного завантаження та віддаленої атестації.

Безпечне завантаження, довірене завантаження та вимірюване завантаження створюють архітектуру, яка є принципово стійкою до буткітів та руткітів. У Windows ці функції здатні усунути зловмисне програмне забезпечення на рівні ядра з вашої мережі. З Windows ви можете довіряти цілісності своєї ОС.

Захищеність операційної системи Windows відповідає наступним вимогам:

1. Обов'язкова ідентифікація й автентифікація всіх користувачів операційної системи.

2. Розмежувальний контроль доступу — надання користувачам можливості захисту даних, що їм належать.

3. Системний аудит — здатність системи вести докладний аудит усіх дій, що виконують користувачі й сама операційна система.

4. Захист об'єктів від повторного використання — здатність системи запобігти доступу користувача до ресурсів, з якими до цього працював інший користувач (наприклад, забезпечення неможливості повторного використання звільненої пам'яті або читання інформації з файлів, що були вилучені).

Автентифікація та ідентифікація відбуваються за допомогою процеса WinLogon, що перевіряє істинність користувача за допомогою інших модулів підсистеми авторизації (LSA, MSV) та запускає в разі успіху процес UserInit.exe. Останній виконується з повноваженнями даного користувача і створює для останнього робоче середовище – підключає відповідний користувачеві ключ реєстру, настройки з user profile, та запускає програмну оболонку (explorer.exe). Архітектура підсистеми авторизації досить гнучка і дозволяє використовувати будь-які способи перевірки істинності. Проте в стандартній конфігурації використовується лише проста парольна автентифікація. Образи паролів (їх хеш значення) зберігаються в спеціальному розділі реєстру.

За допомогою утіліти User Manager Windows забезпечує широкі можливості по керуванню обліковими записами користувачів. Так, для кожного користувача може бути задано ряд атрибутів, таких як належність до груп, місцезнаходження user profile, робочі години, повноваження на доступ по комутованих лініях і т.д. Крім того, може бути задана політика керування обліковими записами, що регламентує: 1) мінімальний та максимальний терміни життя паролю; 2) мінімальну довжину паролю; 3) унікальність паролю як вимога не належати до заданої кількості востаннє використаних; 4) кількість невдалих спроб автентифікації, після яких обліковий запис блокується, та відрізок часу, протягом якого вони мають відбутися; 5) тривалість блокування облікового запису. Також ця утіліта дозволяє призначати користувачам привілеї (права на всю систему, а не на конкретний об’єкт, наприклад входити в систему локально або змінювати системний час) та задавати політику аудиту. ОС Windows реалізує дискреційну модель розмежування доступу. Керування доступом здійснюється в Windows за допомогою спеціального модулю, що

носить назву reference monitor та реалізується викликом функції SeAccessCheck ядра ОС при будь-якій спробі суб’єкта отримати доступ. При цьому використовуються дві структури даних – маркер доступу суб’єкта, що є носієм його повноважень, та дескриптор захисту об’єкта, що містить ідентифікатори власника об’єкта та його первинної групи, список контролю доступу (ACL) та список аудиту (SACL). Матриця доступу в даній ОС, таким чином, зберігається у вигляді множини списків контролю доступу об’єктів.

**Висновки**

Windows 11 - це найбезпечніша версія Windows, в якій реалізовано комплексні заходи безпеки, покликані захистити пристрої, особисті дані та інформацію. Ці заходи включають вбудоване вдосконалене шифрування та захист даних, надійний захист мережевої системи та інтелектуальні засоби захисту від вірусів і загроз, що постійно еволюціонують. Безпека системи Надійне завантаження (Безпечне завантаження + Вимірюване завантаження) Windows 11 вимагає, щоб усі ПК використовували функцію безпечного завантаження Unified Extensible Firmware Interface (UEFI). Під час запуску пристрою з Windows 11 функції безпечного завантаження та довіреного завантаження працюють разом, щоб запобігти завантаженню шкідливого програмного забезпечення та пошкоджених компонентів. Безпечне завантаження забезпечує початковий захист, а потім процес продовжує надійне завантаження. Безпечне завантаження забезпечує безпечний і надійний шлях від уніфікованого розширюваного інтерфейсу мікропрограми (UEFI) через послідовність довіреного завантаження ядра Windows. Атаки зловмисного програмного забезпечення на послідовність завантаження Windows блокуються за допомогою рукостискань із застосуванням підписів протягом усієї послідовності завантаження між UEFI, завантажувачем, ядром і середовищами додатків. Щоб зменшити ризик руткітів, комп'ютер перевіряє наявність цифрового підпису на мікропрограмі перед початком процесу завантаження.

Потім Secure Boot перевіряє цифровий підпис завантажувача ОС, а також весь код, який виконується до запуску операційної системи, щоб переконатися, що підпис і код є безкомпромісними і довіряють політиці безпечного завантаження.

Надійне завантаження підхоплює процес, який починається з безпечного завантаження. Перед завантаженням завантажувач Windows перевіряє цифровий підпис ядра Windows. Ядро Windows, у свою чергу, перевіряє всі інші компоненти процесу запуску Windows, включаючи завантажувальні драйвери, файли запуску та драйвер раннього запуску антивірусного продукту (ELAM). Якщо будь-який з цих файлів було пошкоджено, завантажувач виявить проблему і відмовиться завантажувати пошкоджений компонент. Часто Windows може автоматично відремонтувати пошкоджений компонент, відновивши цілісність Windows і дозволивши комп'ютеру нормально запуститися. Несанкціоноване втручання або атаки шкідливого програмного забезпечення на послідовність завантаження Windows блокуються за допомогою рукостискань між UEFI, завантажувачем, ядром і середовищами додатків.

Щоб підвищити безпеку пристроїв з Windows, користувачі повинні:

Переконатися, що ОС Windows і все встановлене програмне забезпечення є актуальними.

Регулярно оновлювати та запускати антивірусне програмне забезпечення.

Увімкнути вбудований брандмауер Windows для захисту пристрою від несанкціонованого доступу.

Використовуйте надійні та унікальні паролі для облікових записів користувачів і вмикайте двофакторну автентифікацію, де це можливо.

Регулярно створюйте резервні копії важливих даних на зовнішньому накопичувачі або в захищеному хмарному сервісі, щоб уникнути їх втрати.

Дотримуючись цих порад з безпеки та залишаючись в курсі потенційних загроз, користувачі Windows можуть захистити свої пристрої та дані від різноманітних кібератак.