ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЗАХИСТУ ІНФОРМАЦІЇ

КАФЕДРА ІНФОРМАЦІЙНОЇ ТА КІБЕРНЕТИЧНОЇ БЕЗПЕКИ

ПОЯСНЮВАЛЬНА ЗАПИСКА
до магістерської роботи
на тему:

«Технологія управління активами корпоративної системи на базі рішень IBM SAM»

Виконала: студентка 6 курсу, групи БСДМ-61
Спеціальності 125 Кібербезпека
Освітньо-професійної програми «Інформаційна та кібернетична безпека»
(шифр і назва спеціальності)
Кушнір І.М.
(прізвище та ініціали)
Керівник Гайдур Г.І.
(прізвище та ініціали)
Рецензент ___
(прізвище та ініціали)
Нормоконтролер Чумак Н.С.
(прізвище та ініціали)

КИЇВ – 2020

РЕФЕРАТ

Текстова частина магістерської роботи: 73 сторінки, 15 рисунків, 26 джерел.

Об'єкт дослідження - процес забезпечення безпеки активам корпоративної системи.

Предмет дослідження - технологія управління активами корпоративної системи рішень IBM SAM.

Мета роботи: дослідження технології управління активами корпоративної системи на базі рішень IBM SAM.

Методи дослідження: сукупність теоретичних та практичних методів для проведення аналізу загроз та технології захисту управління активами корпоративної системи на базі рішень IBM SAM.

В роботі проведено аналіз проблем та загроз веб-ресурсів корпоративного сегменту, методи аутентифікації а також наведений приклад конфігурування IBM SAM для організації роботи за технологією SSO. Перша частина висвітлює саме загрози, що можуть бути актуальними по відношенню до веб-ресурсів корпоративного сегменту, та від так і до активів компанії. В другій частині досліджені основні методи, етапи, протоколи аутентифікації, а також проведений аналіз SSO-технологій. В третій частині технологію управління активами на базі рішення IBM Security Access Manager. Практична частина роботи полягає у вивченні підходів налаштування інфраструктури корпоративної системи з ціллю використання SSO-технологій.

Напрямки подальших досліджень полягають у перевірці ефективності розробленої методики на практиці.

Галузь використання – інформаційна безпека організації.

ТЕХНОЛОГІЯ, SSO, АКТИВИ, КОРПОРАТИВНА ІНФОРМАЦІЙНА СИСТЕМА, IBM, ISAM, SAML, OAUTH, АУТЕНТИФІКАЦІЯ.

ЗМІСТ

ВСТУП

Актуальність дослідження. Безпека інформації, є стратегічною метою фактично будь-якої форми бізнесу. Безпечними повинні бути всі процеси виробництва, зберігання та обміну інформацією, всі технічні та програмні компоненти, які беруть участь в цих процесах.

Основними завданнями інформаційної безпеки є:

- забезпечення конфіденційності інформації;
- забезпечення цілісності та достовірності інформації;
- забезпечення доступності інформації та інформаційних ресурсів.

Градація пріоритетів серед перерахованих завдань інформаційної безпеки є питанням, що вирішуються тільки в конкретних умовах застосування, і залежить від вимог, що пред'являються безпосередньо до інформаційних систем.

В сучасних інформаційних системах зберігається величезна кількість інформації. Частина її загальнодоступна, і з нею може ознайомитися кожен охочий. Однак крім загальнодоступної інформації в базах даних може зберігатися службова, комерційна чи державна таємниця, персональні дані або просто особиста інформація користувача. Для цієї інформації необхідно виконання властивості конфіденційності. Але якщо закрита інформація (інформація обмеженого доступу) є, значить, вона повинна бути доступна тільки тим суб'єктам (користувачам), кому вона довірена. Права доступу суб'єктів до об'єктів (інформації) регламентуються сукупністю правил, які називаються правилами розмежування доступу.

Існують різні способи обробки інформації - наприклад, один користувач може працювати з усією інформацією тільки в режимі читання, а іншому дозволяється читати і змінювати об'єкти доступу, а також знищувати їх і створювати нові. Таким чином, концепція розмежування доступу передбачає таку умову: перш ніж допустити будь-кого до інформації, необхідно визначити повноваження цього

користувача, а для цього потрібно спочатку його ідентифікувати. Для ідентифікації йому видається деякий ідентифікатор, який повинен бути пред'явлений для доступу до інформації. А перевірка належності користувачеві пред'явленого їм ідентифікатора називається аутентифікацією.

Існує велика кількість способів аутентифікації, серед яких найбільш поширеним досі залишається парольний захист. Що вже само по собі може нести ряд загроз, серед яких вагоме місце буде зайнято людським фактором. В разі використання великої кількості паролів, людина, як найчастіше буде використовувати слабкі, короткі паролі, або зберігати їх в місцях що можуть бути доступними оточуючим.

Для вирішення зазначеної та інших проблем може бути застосована технологія єдиного входу (Single Sign-On, SSO), що забезпечує можливість використання одного ідентифікатора для доступу до всіх дозволеним IT-ресурсів та систем і дозволяє вирішувати завдання суворої і наскрізний аутентифікації користувачів.

Об'єкт дослідження: процес забезпечення захисту активів корпоративної інформаційної системи.

Предмет дослідження: технологія управління доступом на базі IBM SAM.

Мета роботи: розробити варіант управління доступом до активів корпоративної інформаційної системи на прикладі рішення IBM SAM.

Наукові завдання:

дослідити сутність проблеми наявності вразливостей корпоративних інформаційних систем та визначення необхідності впровадження технології єдиного входу;

проаналізувати існуючі технології аутентифікації та SSO;

проаналізувати принципи реалізації технології SSO та єдиного входу до корпоративної інформаційної системи.

Методи дослідження – опрацювання літератури за даною темою, аналіз експлуатаційної документації, міжнародних стандартів.

Практичне значення одержаних результатів полягає в розробці варіанта застосування технології єдиного входу до корпоративної інформаційної системи на прикладі рішення IBM SAM.

1. СУЧАСНИЙ СТАН ПРОБЛЕМ ТА ЗАГРОЗ БЕЗПЕКИ ВЕБ-РЕСУРСІВ КОРПОРАТИВНОГО СЕГМЕНТУ

На сьогоднішній день все більше послуг і ресурсів переносяться в мережу. Частка електронної комерції зростає з кожним роком. Всі ці тенденції призводять до підвищення актуальності загроз на веб-ресурси як у кількісному, так і в якісному відношенні.

У даній главі ми розглянемо основні засоби і методи, які використовуються для збору інформації з веб-ресурсів, особливості організації та роботи даних засобів, а також методи, які дозволяють виявити і блокувати небажані джерела запитів.

1.1. Загрози та атаки на веб-ресурси

Забезпечення безпеки web-ресурсів від зовнішніх загроз є одним з важливих питань інформаційної безпеки в контексті захисту активів. На сьогоднішній день спектр технологій, що використовуються в web, стає настільки широким, що загрози, які раніше були характерні тільки для прикладного програмного забезпечення, стають актуальними і для web-ресурсів. Разом з тим, виникають нові загрози, метою і основним фокусом яких є саме web-ресурси і їх середовище функціонування.

Web-ресурс стає необхідним вузлом для забезпечення діяльності сучасних організацій. Таким чином, загрози, характерні для нього призводять до ризиків і для основних бізнес-процесів.

Захист web-ресурсів від атак, а також забезпечення безпеки активів є одним з важливих питань інформаційної безпеки.

Виділимо кілька основних напрямків атак, що призводять до порушення конфіденційності, цілісності або доступності:

- Атаки на процеси автентифікації;

- Атаки, спрямовані на клієнтів;
- Атаки на виконання коду на сервері;
- Атаки на відмову в обслуговуванні (DOS/DDOS);
- Логічні атаки;
- Спам і розсилки;
- Розголошення інформації;
- Автоматизовані атаки.

Послідовно розглянемо кожну з груп атак. Атаками називають основні техніки, які зловмисники використовують для експлуатації вразливостей веб-додатків. Важливо не плутати атаки з вразливостями, оскільки атаки описують ситуації, пов’язані з діями зловмисника, тоді як уразливості можуть бути лише слабкостями.

Гарним прикладом, що виділяє основні категорії атак на web-ресурс, є перелік OWASP Top 10, що включає десять векторів атак, що призводять до найбільших ризиків для веб-програми.

На сьогодні до десяти найпоширеніших векторів атак, відповідно до останнього стандарту OWASP Top 10, відносять :

- Ін'єкції (будь-які схеми вбудовування вмісту в командні інструкції для інтерпретації: SQL, noSQL, LDAP, OS тощо);
- Порушення автентифікації (пов'язані з некоректною реалізацією механізмів автентифікації, що призводять до компрометації паролів, сесійних токенів);
- Витік чутливих даних (недостатня захищеність важливих даних у додатку або API виникає через порушення доступу);
- Впровадження зовнішніх сутностей (характерно для неправильно сконфігурованих XML обробників і старих веб-додатків);
- Порушення контролю доступу (неправильний розрахунок матриці доступу та поява у користувачів додаткової неавторизованої функціональності);
- Помилкове налаштування (помилки налаштування операційної системи, компонентів інфраструктури);

- Міжсайтовий скриптинг (недостатня валідація даних, отриманих від користувачів);
- Небезпечна десеріалізація (помилки розбору серіалізованих даних, що призводять до виконання коду або інших вторинних атак);
- Використання вразливих компонентів (включення бібліотек, що мають відомі вразливості);
- Недостатнє логування і моніторинг (неправильно вибудувані процеси спостереження за безпекою веб-додатку).

Різноманітність векторів атак створює простір для комплексного і послідовного застосування їх варіацій з метою порушення конфіденційності, цілісності та доступності веб-додатку. Кожен з векторів може бути закритий порційно, однак, для забезпечення комплексної безпеки потрібні ешелоновані і цілісні заходи, що включають процеси від моніторингу активності автоматизованих засобів до стану генерованої користувачами інформації на веб-ресурсі.

1.1.1 Атаки на процеси автентифікації та авторизації.

До даної категорії відносяться атаки, спрямовані на компрометацію процедур перевірки користувацького ідентифікатора (ідентифікацію), підтвердження користувача (автентифікацію) і надання прав доступу (авторизацію). Існує велика кількість різних векторів атак на дані процеси, що включають як технічні прийоми, так і логічні вразливості:

- Перебір паролів та ідентифікаторів;
- Недостатня автентифікація;
- Неправильні права доступу;
- Небезпечні процедури відновлення доступу;
- Помилкова конфігурація.

Найпростішим методом є підбір паролів - автоматизований процес з метою вгадати якийсь секретний параметр, використовується процесами автентифікації та

авторизації. Також до компрометації системи і неавторизованого доступу до критичних ресурсів може призвести порушений механізм аутентифікації. Часто виникають вразливості сторонніх елементів процесу автентифікації, таких як система відновлення пароля, перевірка даних користувача, механізм генерації та перевірки сесій користувачів.

Окремо варто виділити проблеми з процесом авторизації та вразливості розмежування доступу:

- Небезпечні ідентифікатори сесій;
- Недостатнє обмеження доступу до значущих даних;
- Недостатній тайм аут сесій і проблеми доступу.

Ці проблеми характерні для різних механізмів автентифікації. Існують відмінності при використанні інших механізмів (токени, біометрія тощо), однак основні принципи зазвичай залишаються незмінними.

1.1.2 Атаки на клієнтів

До даної категорії відносяться атаки, засновані на спробах введення в оману користувачів веб-ресурсу (наприклад, соціальна інженерія та фішинг, а також підміна вмісту веб-сторінки на рівні клієнта через експлуатацію різного роду технічних і логічних вразливостей:

- Підміна вмісту сторінки і даних;
- Крос сайт скриптинг;
- Атаки на з’єднання (Man in the middle);
- Атаки на обман і отруєння кешу.

До даної категорії відносяться атаки, засновані на спробах обману користувачів і маніпуляції з правами доступу. Яскравим прикладом є атака на обман кешу, коли комбінація атаки на користувача (перехід за посиланням) збігається з неправильним налаштуванням кешування і прав доступу до статичних ресурсів, що призводить до несанкціонованого доступу до даних користувача.

1.1.3 Виконання коду на веб-сервері

Атаки на виконання коду є типовими прикладами злому сайту за допомогою експлуатації вразливостей. Зловмисник отримує можливість за певних умов виконати довільний код у рамках веб-додатка або сервера. Найбільш поширеною причиною даного класу помилок є переповнення буфера в пам'яті процесу, що надає можливість записати спеціально сформовану послідовність у виконувану область даних. Іншим популярним класом атак є підходи, засновані на експлуатації функцій форматування рядків. Суть даних атак у тому, щоб використовувати помилкову або недостатню фільтрацію вхідних параметрів для підміни аргументів, що призводить до виконання команд операційної системи на веб-сервері. У разі, якщо інформацію, отриману від клієнта, належним чином не верифікувати, зловмисник отримує можливість виконання коду. Подібним чином функціонують SQL і PHP Injection та інші підходи:

- Атаки на переповнення буфера;
- Атаки на форматну сходинку;
- Ін'єкції команд (SQL, LDAP, noSQL);
- Ін'єкції при виконанні команд операційної системи;
- Впровадження зовнішніх сутностей (XXE).

1.1.4 Логічні атаки

Атаки, що відносяться до даного класу, спрямовані на несподіване використання можливостей веб-додатків та зміна логіки його роботи. Недостатнє покриття коду тестами та обробка не всіх виняткових ситуацій призводять до того, що зловмисник може змінити процес функціонування додатку в своїх інтересах. Прикладами цього класу атак є:

- процес відновлення паролів;
- електронні транзакції;
- реєстрація користувачів;

- інші процеси з великою кількістю гілок коду.

Зміна логіки може призводити не тільки до обходу процедур автентифікації, а й до відмови в обслуговуванні та отримання додаткової цінної інформації про систему.

Логічні атаки найбільш близькі автоматизованим атакам, оскільки використовуються схожі маніпуляції легітимним функціоналом веб-ресурсу з метою отримання додаткових можливостей або ж переваги над іншими користувачами системи:

- Зловживання функціональністю;
- Відмова в обслуговуванні;
- Недостатня протидія автоматизації;
- Недостатня перевірка даних;
- Відсутність моніторингу та логування;

Логічні атаки небезпечні за рахунок того, що автоматизовані засоби виявлення та запобігання вторгнень, а також різні прикладні фільтри та інші засоби захисту не завжди можуть виявити спроби експлуатації таких вразливостей за рахунок використання атакуючими засобами легітимного функціоналу веб-ресурсу. Захист полягає в правильному вибудовуванні процесів розробки:

- забезпечення захищеної розробки;
- покриття коду тестами;
- інтеграційного тестування;
- надлишкової перевірки даних;
- протидії автоматизованим засобам;
- грамотне налаштування моніторингу ресурсів і консистентності даних.

1.1.5 Атаки на відмову в обслуговуванні

До даної категорії відносять методи розподілених атак на відмову в обслуговування як на мережевому рівні, так і на рівні додатків. Атаки даної

категорії в більшості своїй засновані на вичерпанні ресурсів веб-додатків для порушення коректної роботи звичайних користувачів.

Для вичерпання ресурсів застосовуються різні техніки як мережевого характеру, так і логічні, що функціонують безпосередньо на рівні веб-додатку:

- Рекурсія;
- Помилки обробки завантажених файлів;
- Помилки десеріалізації;
- Впровадження зовнішніх сутностей;
- Вичерпання ресурсів каналу доступу;
- Зловживання функціоналом;
- Залежність потоків даних і блокування.

Основними ресурсами, вичерпання яких викликає зловмисник, є:

- оперативна пам’ять;
- процесорні потужності;
- дисковий простір.

Для захисту від даного класу атак застосовують підходи подібні до попередньої категорії:

- навантажувальне тестування;
- моніторинг;
- резервування;
- оперативне збільшення ресурсів.

1.1.6 Спам

Веб-спам на сьогоднішній день становить серйозну загрозу для веб-ресурсів. До даної категорії відноситься як традиційний поштовий спам, так і пошуковий спам, а також шкідливі посилання. Веб-спам безпосередньо пов'язаний з шахрайськими схемами, ботнетами, фішингом і фармінгом. Також у сукупності з даними методами широко використовуються методи соціальної інженерії, які змушують користувачів звертати увагу на різні проекти і послуги.

Приклади спаму також тісно пов'язані з автоматизованими загрозами, оскільки реалізація масштабних спам атак на веб-ресурс (особливо побудовані за принципом генерованого користувачами контенту) вимагає автоматизованої активності, генерації постів і тестів, роботизованого заповнення форм та інших спеціалізованих технік.

1.1.7 Розголошення інформації

Атаки на розголошення інформації пов'язані з отриманням додаткової інформації про веб-ресурс. Зловмисник може використовувати знання про версії програмного забезпечення, веб-сервера, наявності відповідних оновлень для проведення супутніх видів атак. Крім цього, сама інформація може містити комерційну таємницю і персональні дані. Найчастіше веб-ресурс надає користувачам більше інформації, ніж їм потрібно, цим і користуються зловмисники. Будь-яка, навіть незначна, інформація про систему може призвести до її повної компрометації за певних умов. Збір інформації сьогодні виконується автоматизованими засобами, що дозволяє зловмисникам масово збирати всі дані з різних ресурсів. Такі системи використовуються в ході конкурентної розвідки для пошуку додаткових відомостей, які можуть бути використані бізнес конкурентами для створення власних ефективних систем агрегації та порівняння запозиченого контенту.

Існують кілька основних напрямків, що призводять до розголошення надлишкової інформації:

- некоректне індексування;
- ідентифікація версій програм, бібліотек та елементів інфраструктури;
- витік інформації;
- маніпуляції шляхами доступу до ресурсів (мається на увазі прямий та відносний шлях);
- передбачуване розташування ресурсів.

До цих категорій можна віднести також універсальні проблеми, такі як некоректне налаштування систем, помилки розмежування доступу і відсутність коректно налаштованих систем моніторингу та логування дій користувачів.

1.1.8 Автоматизовані атаки

Окремою категорією атак, пов'язаних з усіма попередніми категоріями, є атаки, що реалізують автоматизовані загрози. Вони включають питання автоматизації дій на веб-ресурсі. Причини і цілі даної автоматизації включають широкий діапазон активностей: від спроб здійснення атак на відмову в обслуговуванні та розсилки спаму, до реалізації логічних загроз, маніпуляції логікою додатку та збору інформації з веб-ресурсу.

Спеціалізовані засоби, що реалізують ці атаки, називаються веб-роботами. Вони несуть суттєві загрози веб-ресурсам, що містять унікальний контент, що представляє комерційну цінність. Зазвичай, діяльність таких ресурсів пов'язана з електронною комерцією:

- сайти-каталоги;
- сайти-аукціони;
- активні сайти з продажу певних товарів;
- сайти з конкурсами, голосуваннями та розіграшем цінних призів;
- видавництва та блоги, що публікують контент;
- великі фінансові майданчики;
- будь-які веб-ресурси компаній-конкурентів для проведення збору інформації.

На прикінці 2019 року, частка трафіку що належить веб-роботам на веб-ресурсах, склала близько 38%. Серед усього трафіку не менше 20% виходить від веб-роботів зловмисників, причому близько 74% веб-роботів представляють з себе складні автоматизовані системи. Решта веб-робіт відносяться до аматорських категорій і не містять механізмів протидії виявленню.

Найбільшу частку серед індустрій, які цікавлять веб-роботів, займає фінансова (42,2%), однак, примітно, що серед професійних веб-роботів з трафіку лідирують сайти з продажу певних товарів, близько третини.

Найбільш схильними до автоматизованих атак галузями є:

- електронна комерція;
- ігри;
- фінанси;
- медицина;
- авіаперевезення;
- подорожі;
- продаж квитків;

Близько 54% всього роботизованого трафіку виходить із США. Багато веб-роботів працюють з хмарних серверів хостингу, лідером за обсягом вихідного роботизованого трафіку є Amazon близько 20%. З'являються різні засоби автоматизації збору інформації, наприклад, плагіни для веб-браузерів. Поведінка веб-роботів стає все більш і більш агресивною, вони використовують велике число IP адрес для приховування своєї діяльності. Для збору інформації сьогодні використовуються ботнети з комп'ютерів, серверів і навіть смартфонів.

Варто також зазначити, що автоматизовані атаки можуть мати як активний (явну взаємодію з веб-ресурсом), так і пасивний (використання тільки легітимного функціоналу) характер, що безпосереднім чином впливає на профіль поведінки веб-робота і стратегії його виявлення.

Автоматизовані атаки можуть бути супутніми для інших різновидів атак, оскільки, багато з них вимагають автоматизації дій. Так, для експлуатації вразливостей або ін'єкцій необхідно здійснити спочатку пошук векторів реалізації даних загроз, а потім безпосередньо застосувати атаку, яка часто складається з великої кількості послідовних запитів. Тому, методи виявлення і протидії автоматизованому збору інформації з веб-ресурсу часто можуть бути використані і для виявлення інших загроз веб-ресурсу.

1.2 Автоматизовані загрози

Відкритий проект забезпечення безпеки веб-програм OWASP є спільнотою компаній і дослідників, які створюють різні інструменти, методології, довідники та іншу документацію, що спрощує і систематизує процеси захисту інформації. Дана спільнота є одним з найавторитетніших і найвідоміших у галузі захисту веб-ресурсів від широкого спектру загроз. Методології, випущені цим проектом, наприклад, OWASP Top 10 Project, цитуються і використовуються майже всюди.

Одним з проектів, що відкрився лише в 2015 році і отримав статус лабораторії в 2017 є "Automated Threats to Web Applications". Цей проект цікавий тим, що систематизує загрози, пов'язані з активністю автоматизованих засобів на веб-ресурсі. Складається онтологія загроз, схеми ідентифікації та вжиття заходів щодо захисту. При цьому не розробляються системи виявлення та протидії веб-роботам. Але, веб-роботи, класифікуються за типами загроз і цілей, які вони реалізують.

Сьогодні загрози автоматизації визнаються помітними і небезпечними, що веде до необхідності розуміння які конкретно цілі переслідують зловмисники і які категорії коштів дозволяють з ними боротися. Чітка класифікація та системна методологія є виключно необхідною в даній ситуації.

Веб-роботи можуть впливати на веб-ресурс наступним чином:

- збір інформації з метою подальшого використання (наприклад, перепродажу, агрегації або навіть публікації на своєму власному веб-ресурсі);
- маніпуляція функціоналом веб-додатку з метою отримання певної вигоди або переваги (репутаційної, фінансової чи іншої);
- вилучення фінансового прибутку за рахунок автоматизації дій на сайті;
- створення додаткового навантаження на веб-ресурс;
- негативний вплив на статистику, псування маркетингових показників;
- маніпуляція громадською думкою.

Можна розкрити дані категорії і виділити ще багато різних негативних факторів, які викликають веб-роботи. Однак, охопити всі категорії не представляється можливим в рамках одного продукту або дослідження. Деякі

загрози можуть бути виявлені або нейтралізовані схожим чином, для інших загроз можливе формування спеціалізованих методів тощо.

1.3. Автоматизований збір інформації з веб-ресурсів

Сьогодні існує безліч способів організації збору інформації з веб-ресурсів. Не існує єдиного терміну для визначення даного процесу. У науковій літературі прийнято використовувати такі терміни: краулінг, парсинг, автоматизований збір інформації, скрепінг і веб-майнінг.

Краулінг (веб-майнінг) - це процес вилучення доступної інформації з веб-ресурсів. Цей процес полягає в автоматичному виявленні ресурсів, документів та інформації, а також застосуванні різних методів аналізу даних для обробки такої інформації.

Багато систем збору інформації часто використовують скомпрометовані або автоматично реєстровані акаунти та облікові записи для доступу до інформації, закритої від неавторизованого доступу. Відкрита інформація збирається автоматичними веб-роботами. Такий веб-робот відвідує веб-ресурс і витягує з нього всю наявну інформацію та метаінформацію (HTML-розмітку, дані, тощо), а також при необхідності завантажує статичні та динамічні ресурси (скрипти, зображення, документи).

Збір інформації може виконуватися як в реальному часі, так і мати періодичний характер. Збір інформації також може виконуватися не тільки з метою збору самого контенту веб-ресурсу, а й для отримання інших відомостей про структуру та принципи його функціонування. Тому даний процес використовується для реалізації практично всіх різновидів автоматизованих загроз.

Ця інформація потім передається підсистемам розпізнавання, які записуються формальною мовою, прийнятою в системі. Правила розпізнавання можуть бути сформовані як вручну програмістом, так і автоматизованим чином на основі спеціальної навчальної вибірки. Система застосовує цей набір правил до кожного набору даних, що надходить від підсистеми збору інформації. У її основі лежить

перебір веб-ресурсів і сторінок на них за певним алгоритмом, який зазвичай заснований на прямому переборі сторінок і витягуванні з них посилань для переходу на наступні сторінки, таким чином формується процедура обходу графа зв'язків веб-сторінок між собою. Системи збору інформації вирішують величезну кількість завдань для оптимізації процесу парсингу сторінок і зниження витрат, пов'язаних з організацією масового і постійного збору інформації. Сюди входять: збір даних, класифікація, використання сторонніх ресурсів для пошуку, моніторинг, отримання репрезентативних вибірок, обхід графа зв'язків веб-сторінок, масштабування, розподіл навантаження тощо.

Виділяють три основні категорії збору інформації в веб, розглянемо їх у контексті автоматизованого вилучення інформації веб-роботом:

- витяг змісту - застосування методів вилучення знань з текстових, графічних та інших матеріалів, що розміщуються на веб-ресурсі;

- вилучення структури - вивчення інформації про взаємозв'язки сторінок і документів веб-ресурсу;

- аналіз використання веб-ресурсу - вивчення даних про взаємодію користувачів веб-ресурсу з ним.

Веб-роботи відрізняються за цілями збору інформації, однак, для виконання свого завдання, вони складають максимальну картину про функціонування веб-ресурсу. Особливо це проявляється для масових веб-роботів, які діють нецільовим чином і змушені застосовувати однакові методи вилучення та обробки інформації для різних веб-ресурсів, що відрізняються структурою подачі інформації і навіть мовою.

Інформація, що витягується веб-роботами, має різний вигляд для різних засобів. Так, існують веб-роботи, які збирають простий текст, витягуючи його з HTML розмітки, інші роботи збирають всі зображення з веб-ресурсу (наприклад, для подальшого надання можливості зворотного пошуку, або для несанкціонованого запозичення). Просунуті веб-роботи вміють витягувати неструктуровані дані з таблиць і навіть зображень із застосуванням технологій розпізнавання.

Для забезпечення можливості адміністратору ресурсу закрити від індексування та автоматизованого збору інформації весь веб-ресурс або деякі його частини, було розроблено угоду, згідно з якою, веб-роботи запитують правила доступу зі спеціального файлу "robots.txt", розташованого в кореневій директорії веб-сайту. Цей файл складається з набору записів, розташованих на окремих рядках. Кожен запис повинен відповідати формату *<field >:<optional_space><value><optional_space>*. У HTTP протоколі, на якому заснована комунікація в світі веб, існує заголовок "User-Agent", який браузери заповнюють своєю назвою і версією. Для кожного User-Agent адміністратор веб-ресурсу може налаштувати різні доступні і заборонені директорії для відвідування. Ці правила мають рекомендаційний характер, та через це не обов'язково виконуються.

Процес збору інформації веб-роботами входить в різні класифікації загроз і напрямків атак. Так, база CWE (Common Weakness Enumeration), яка включає стандартизований список вразливих місць програмного забезпечення, містить запис № 799 (Improper Control of Interaction Frequency), що відповідає відсутності контролю лімітів використання веб-ресурсу з боку веб-додатку. Сюди входять як звичайні ліміти на трафік або кількість підключень, так і логічні ліміти на взаємодії (спроби веб-робота отримати перевагу над іншими користувачами, витягти дані автоматизованим чином, здійснити перебір паролів та інші автоматизовані атаки).

Автоматизований збір інформації також розглядається в декількох категоріях переліку CAPEC (Common Attack Pattern Enumeration and Classification), що містить класифікацію відомих доменів і механізмів атак.

Перелік класифікації загроз WASC (Web Application Security Consortium) також містить відсилання до автоматизованого збору інформації з веб-ресурсів.

Варто також зазначити, що існують й інші вразливості, вектори атак і загрози безпосередньо або опосередковано пов'язані з автоматизованим збором інформації з веб-ресурсів. Вивчення цієї проблеми і запобігання автоматизованого збору інформації з веб-ресурсу впливає і на протидія іншим автоматизованим загрозам.

Ознаками збору інформації з веб-ресурсу є:

- Аномальна активність;
- Збільшення трафіку та відвідуваності;
- Проблеми з навантаженнями та продуктивністю;
- Поява контенту з веб-ресурсу на інших сайтах;
- Поява нових конкурентів та агрегаторів інформації;

1.4. Класифікація веб-роботів

Можна розділити існуючих веб-роботів на три основні категорії за поведінковим критерієм:

1. Відомі роботи з етичною поведінкою, що дотримуються всіх правил і побажання адміністраторів ресурсів і не приховують своєї присутності на сайті;

2. Відомі роботи з неетичною поведінкою, які ігнорують правила поведінки, але не приховують факту свого відвідування веб-ресурсу, наприклад, шляхом заповнення поля User-Agent;

3. Роботи, що приховують свою поведінку і присутність на веб-ресурсі.

Веб-роботи, що належать до третьої категорії представляють основний інтерес і загрозу. Таких роботів необхідно виявляти і блокувати їх активність. Існує безліч способів реалізації таких роботів залежно від вимог, які ставить перед ними розробник, а також ресурсів, якими він володіє.

Веб-роботи можна також розділити на декілька груп залежно від їхніх функцій та можливостей:

1. Роботи пошукових систем, що індексують текстовий зміст ресурсу і мультимедіа контент;

2. Роботи, які перевіряють веб-сайт на предмет працездатності в конкретний момент часу;

3. API запити від мобільних додатків або інших, підключених до сайту зовнішніх засобів;

4. Роботи, які виконують пасивний негласний збір інформації, розташованої на веб-ресурсі;

5. Роботи, що здійснюють активну взаємодію з сайтом з різними цілями автоматизації своїх дій: розсилки спаму, тестування на предмет вразливостей, конкурентної розвідки.

1.4.1 Структура запиту

Сьогодні в існує величезна кількість різних технологій та протоколів взаємодії з веб-ресурсом. Однак найбільше переважання має HTTP (S) прокол.

HTTP - це текстовий протокол, що містить тіло запиту і заголовки. У разі, коли система виявлення та протидії має доступ до трафіку веб-сервера, можна отримувати додаткові параметри TCP/IP стеку і мати доступ до повного вмісту HTTP запиту. У загальному випадку, такого доступу немає, але є можливість отримувати вибіркові заголовки і тіло запиту для визначення ким є автор цього запиту, також існує доступ до адреси джерела (під цим зазвичай мається на увазі IP адреса клієнта, який встановив з'єднання з сервером).

- Запити також можна розділяти за логічною складовою:
- Вхідний HTTP запит на отримання коду сторінки;
- Запит вкладених ресурсів (зображення, favicon, файли стилів, файли javascript);
- AJAX запит;
- Запити в рамках WebSocket з’єднання.

Облік такої смислової складової дозволяє краще зрозуміти специфіку запиту і позбавляє від помилок і внесених спотворень у статистику. Існуючи дослідження не роблять такого обліку і здебільшого поділяють HTTP запити тільки на запити до сторінок, або до статичних вкладених ресурсів. Деякі роботи виділяють ще файлові протоколи (FTP) і доступ до лістингу директорії веб-сервером, але ми не будемо розглядати такі виняткові ситуації. Сьогодні популярним є створення динамічних веб-ресурсів, що активно використовують AJAX, WebSocket і розділяють фронтенд і бекенд на окремі складові.

У сучасному ресурсі найчастіше використовується фронтенд мовою JavaScript, який виконується у користувача в браузері подібно десктоп програмі, а з сервером взаємодіє через HTTP (AJAX) запити і вебсокети. Технологія бекенду на сайті часто є саморобною, або відповідає стилю REST або GraphQL. Не виключено, що з'являться і нові концепції, однак, принципи, що лежать в їх основі, ймовірно будуть дозволяти застосовувати схожі підходи до виявлення та протидії автоматизованим загрозам.

1.4.2 Призначення веб-роботів

Можна виділити такі різновиди веб-роботів за призначенням:

Роботи індексації - основна категорія для веб-роботів загального призначення;

Експериментальні роботи - категорія веб-роботів, розроблених для експериментальних і специфічних цілей;

Роботи верифікації - категорія засобів перевірки цілісності веб-сторінок і доступності сервера в момент відвідування роботом, використовуються для моніторингу стану ресурсу;

Збірники RSS - категорія роботів, які отримують інформацію за допомогою RSS;

Парсери статичного контенту - це категорія веб-роботів, які збирають тільки певний контент з веб-сторінки, виключаючи інші вкладені ресурси, наприклад, тільки файли-зображення, музику, відео або PDF-документи;

Роботи аналітики - це категорія, що включає засоби збору аналітичної інформації про веб-ресурс, зазвичай вони підпорядковуються чітким правилам поведінки і служать цілям архівації вмісту та ранжування сайтів за різними параметрами;

HTML парсери - категорія веб-роботів, які збирають тільки HTML контент, ігноруючи вкладені документи, часто використовуються зловмисниками через швидкість роботи та економію трафіку, дозволяють витягувати текстовий контент, який можна потім використовувати для подальшої відкладеної обробки;

Анонімайзери - засоби що проксують запити окремих користувачів, або прискорюють доступ до сайту за рахунок попередньої перевірки або завантаження елементів;

Інші - категорія, що включає недокументовані і невідомі веб-роботи, про поведінку яких немає достовірної інформації.

1.5. Проблеми автоматизованого збору інформації

Завдання автоматизації роботи з веб-ресурсами включає в себе безліч дисциплін, починаючи від машинного навчання і закінчуючи обробкою природної мови. При створенні професійних веб-роботів доводиться враховувати велику кількість різних факторів, що обмежує створення універсальних систем. Різні рішення можуть бути прийнятними і неприйнятними для різних веб-ресурсів. Основні проблеми, які стоять перед розробниками таких систем, можна згрупувати наступним чином:

- Щоб налаштувати систему автоматизації взаємодії з кожним конкретним веб-ресурсом потрібна участь людини в процедурі налаштування та налагодження системи. Особливо це стосується ресурсів зі складною структурою. Існують технології адаптивного аналізу сторінок і вилучення слабоструктурованої інформації, але вони працюють з обмеженнями і вимагають більш значних фінансових вкладень на етапі реалізації та налагодження алгоритмів.
- Складні веб-роботи повинні вміти обробляти великі обсяги даних за відносно короткий час. Особливо це стосується збору інформації, яка часто оновлюється, наприклад, пов'язаної з бізнесом або аналітикою, або виконання великого число взаємопов’язаних дій. Тому необхідно враховувати дані фактори при розробці системи. Якщо веб-роботи працюють занадто повільно і не можуть зібрати інформацію достатньої міри актуальності, доводиться збільшувати інтенсивність їх роботи і

масштабувати архітектуру, що призводить до додаткових витрат і підвищення навантаження на аналізований веб-ресурс.

- Веб-ресурси часто піддаються змінам структури та форми подачі контенту, тому потрібно постійно оновлювати систему, підлаштовуючи її під зміни. При одночасній роботі з кількома ресурсами завдання стає трудомістким. До того ж, на даному етапі часто потрібна участь людини.

При розробці або впровадженні системи протидії автоматизованим загрозам дуже важливо розуміти які складнощі стоять перед розробниками веб-роботів і використовувати їх у процесі захисту веб-ресурсу. Найчастіше захиститися від дій веб-робота не представляється можливим, але збільшення витрат, які несуть власники систем, може змусити їх відмовитися від здійснення атак.

1.6. Класифікація методів виявлення веб-роботів

Питання класифікації методів виявлення веб-роботів активно розробляються різними командами. Існує широкий набір методів виявлення та протидії, що ґрунтуються на різних принципах. До того ж варто відзначити, що дана тематика тісно пов'язана з питаннями забезпечення безпеки веб-ресурсів, захисту активів, протидії автоматизованим загрозам, виявлення та запобігання вторгнень в локальну мережу з боку мережі Інтернет.

Виявлення веб-роботів насамперед залежить від можливості взаємодії з веб-ресурсом і даних про користувацькі запити, якими можуть оперувати методи виявлення.

За застосовуваними техніками можна класифікувати методи виявлення на чотири основні категорії:

- Синтаксичний аналіз логів - техніка, яка для виявлення веб-роботів використовує найпростіші механізми обробки логів веб-сервера. У цю категорію входять механізми пошуку за ключовими словами, шаблонами, регулярними виразами. Аналізуються HTTP запити, шляхи до файлів,

джерела запитів та їх статистика відвідувань. На жаль, ці методи вимагають складання вичерпного словника і самі по собі є не дуже результативними.

- Аналіз шаблонів трафіку - це набір технік, що полягають у пошуку загальних характеристик роботизованого трафіку. До цієї категорії належать статистичні метрики користувальницьких і роботизованих запитів. Сюди відносяться вже не тільки синтаксичні особливості запитів, а аналіз сесій користувача і пошук в них шаблонів поведінки, характерних для веб-роботів.
- Аналітичні методи - підходи, що включають більш просунуті статистичні механізми виявлення відмінностей трафіку, що включають методи машинного навчання та ймовірнісні моделі. Становлять найбільший інтерес за рахунок можливості виявлення невідомих раніше погроз і аномалій поведінки користувачів.
- Програмні пастки - до даної категорії відносяться різноманітні програмні методи виявлення веб-роботів, пастки на основі JavaScript тестів, невидимих посилань, файлів-маркерів та інших трюків, що дозволяють відрізнити веб-робота від легітимного користувача. До цих методів також належать різноманітні варіації CAPTCHA тестів.

За активністю дій методи можна розділити на дві категорії:

- Активні - застосовувані в режимі реального часу і уточнюючі результат у міру появи нових запитів і сесій;
- Відкладені - методи, що виконуються над безліччю закритих сесій для складання чорних списків джерел трафіку або оновлення списку правил і шаблонів для блокування.

Окремо можна розділити методи за природою даних, з якими відбувається взаємодія:

Часові - методи, що враховують тимчасові характеристики запитів, розподілу відвідувань різних джерел за часом і частотою;

Поведінкові - методи, що враховують переміщення по веб-ресурсу і виконання тих чи інших дій;

Синтаксичні - методи, що взаємодіють зі структурою запитів і даних, що передаються.

Сучасні просунуті методи використовують як правило комбінацію даних підходів і об'єднують результати прогнозів по кожній з категорій для отримання остаточного рішення.

1.7. Проблеми виявлення автоматизованого збору інформації

У процесі виявлення автоматизованого збору інформації з веб-ресурсу виникають певні складності не тільки на рівні виявлення шаблонів поведінки, характерних для користувачів і веб-роботів, але і на технічному рівні вилучення даних. Одним з основних завдань є правильна ідентифікація сесій.

1.7.1 Ідентифікація сесій

Сесія це "набір запитів, об'єднаних однією метою" або "серія успішних запитів від одного джерела за фіксований проміжок часу".

Протоколи, на яких ґрунтується сучасний веб, влаштовані таким чином, що запити до веб-ресурсів мають слабку зв'язаність між собою. Можна отримати будь-яку сторінку та інформацію без повторного виконання однакових кроків. Наприклад, відкрити сторінку будь-якого товару можна, знаючи її адресу (URL), однак, немає ніякої необхідності перед цим отримувати список всіх товарів. Користувачі теж переходять на сторінки у випадкових вузлах веб-ресурсу через використання переходів за посиланнями з інших сайтів і пошукових систем, або використання закладок у браузері, а також прямих посилань.

Навіть якщо розглядати більш сесійні протоколи, такі як websockets, або сайти з просунутим API інтерфейсом, де потрібно підписувати запити, всіляко такі ресурси оптимізовані під отримання незалежних точок входу.

У результаті питання ідентифікації сеансу користувача стає важливим завданням. Не можна просто розділити користувачів за IP адресами і рахувати запити з окремої адреси за проміжок часу єдиною сесією (правда в деяких випадках це якраз допустимо і призводить до оптимальних результатів).

Кожна сесія в результаті повинна бути класифікована як легітимна або роботизована. Однак, залежно від сценарію захисту веб-ресурсу, такі класифікації можуть бути розширені (легітимні веб-роботи, автоматизовані засоби перевірки працездатності веб-ресурсу, веб-роботи SEO агентств та інші).

Завдання ідентифікації сесій вирішується, як правило, на основі співвідношення IP адреси і незмінних параметрів HTTP запиту, наприклад, версії браузера користувача.

Існують рішення даного завдання шляхом віднесення кожного запиту до однієї з чотирьох груп (рис. 1.1).

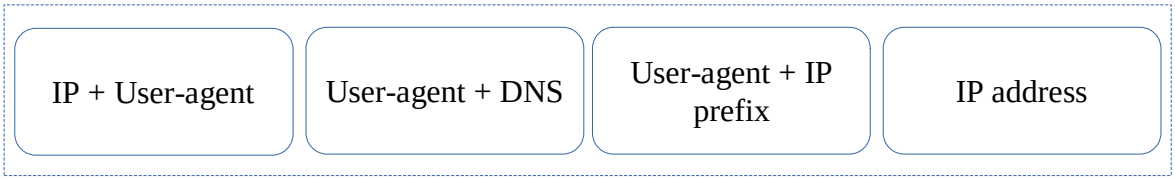


Рис. 1.1. Групи сеансів у процесі ідентифікації

Перша група містить сесії, об'єднані за принципом збігу IP адреси і User-Agent, друга група містить сесії, в яких збігаються User- Agent і хост, отриманий шляхом зворотного DNS запиту до IP адреси джерела кожного із запитів. Третя група складена з сесій з відповідними User- Agent і префіксом IP адреси, четверта група об'єднує сесії на базі тільки IP адреси. Кожен із записів перевіряється на входження в одну із сесій - кандидатів на основі пріоритету від першої до четвертої групи. Це дозволяє ефективно об'єднувати джерела запитів, що належать одній організації.

Іншими словами, можна умовно розділити підходи для класифікації сесій на такі категорії за використовуваними підходами:

- За адресою джерела;
- За адресою джерела і значенням поля User-Agent;
- За цифровим відбитком користувацького оточення;
- На основі встановленого токена сесії.

Адреса джерела є досить унікальним ідентифікатором сесії, оскільки легітимні користувачі не можуть простим способом його змінити і не хочуть цього робити. Існує проблема користувачів, які поділяють одну IP адресу, яка вирішується використанням додаткових параметрів, або кластеризацією за поведінковими характеристиками, що є важким завданням. Веб-роботи можуть підробляти адреси джерел, але це теж не просто зробити з тієї причини, що багато IP- адрес і проксі сервера є "засвіченими" і фігурують у базах даних, мають нехарактерну географічну прив'язку для користувачів веб-ресурсу (наприклад, масові запити на сайт Університету телекомунікацій з Франкфурту будуть підозрілими), а також великі блоки таких адрес коштують недешево.

У сучасній літературі з ідентифікації сесій дослідники дуже часто використовують комбінацію адреси джерела і значення поля User-Agent, таким чином поділяючи користувачів за NAT, що мають одну адресу, але використовують різні версії браузерів. Такий підхід теж не є надійним, хоча і показує прийнятні результати на великих вибірках. Використовується він масово через відсутність у більшості дослідників додаткової інформації про користувальницькі запити, крім логів веб-сервера.

Використання цифрового відбитка - це якісний метод ідентифікації, який, однак, вимагає вбудовування на сайт спеціалізованого програмного забезпечення та скриптів (наприклад, бібліотеку fingerprintjs2). Ймовірність, що два легітимних користувача, які поділяють одну IP адресу, матимуть однаковий цифровий відбиток, невелика. Існує проблема користувачів, які вимикають виконання скриптів, але така проблема характерна не для всіх веб-ресурсів.

Використання фіксованого ідентифікатора сесії є найбільш стійким методом поділу користувачів за сесіями. Таким ідентифікатором, наприклад, може служити окремий користувач. Звичайно, не всі ресурси можуть собі дозволити закривати свої дані для незареєстрованих користувачів через репутаційні витрати і бажання покращувати враження користувачів, однак ідентифікатор можна встановлювати і просто при першому заході. Для цього використовується комбінація з різних технік:

- Традиційні cookie файли;
- Локальне сховище переглядача;
- Зберігання даних у веб-кеші та історії запитів;
- Особливості HTML5, Silverlight, Java апплетів та інших технік.

Найбільш повний список таких підходів описано в бібліотеці evercookie. З розвитком стандартів веб з'являються всі нові і нові підходи.

Безумовно такі заходи покликані для визначення сесій не веб-роботів, а легітимних користувачів. Ми приймаємо за базис, що професійні веб-роботи здатні підробляти будь-які дані і підміняти сесії. Однак, спочатку, не всі веб-роботи на це здатних, по-друге, веб-роботи не здатні підробити свій ідентифікатор сесії за сесію легітимного користувача, що дозволяє надійно розділяти легітимних користувачів і використовувати це для виявлення веб-роботів.

В існуючих дослідженнях сесії також прийнято розбивати на блоки закінчених активностей за принципом часу простою. Найбільш поширений діапазон часу становить 30 хвилин, однак деякі дослідники моделей поведінки людини на веб-ресурсі стверджують, що слід використовувати проміжок в 1 годину.

Можна представити наступний умовний алгоритм розбиття сесій. Для кожного запиту відбувається перебір сесій з переліку активних сесій. Якщо проміжок між часом виникнення запиту і останнім запитом з даної активної сесії більше встановленого порогу, така сесія закривається. В іншому випадку відбувається перевірка на збіг адреси джерела запиту та сесії, а також значення поля User-Agent. Якщо запит не підійшов під жодну з відкритих сесій, створюється нова відкрита сесія і додається до переліку. Таким чином, після завершення роботи алгоритму утворюється список сесій, що включає в себе всі запити.

1.7.2 User-Agent, що змінюється

Заголовок HTTP, що містить версію браузеру, може змінюватися з течією часу. Для цього існує кілька причин:

- Легітимний користувач відкрив інший браузер;

- Зловмисник змінив значення заголовка;
- Інший легітимний користувач з аналогічною адресою джерела зайшов на ресурс (це можливо через технологію NAT, яку використовують провайдери).

Така поведінка впливає на визначення сесій користувача, якщо немає можливості отримати, встановлену скриптом, адресу сесії (зазвичай такі значення зберігаються в заголовку Cookie). Втім, фіксування сесії із заголовків не працює і для випадку відкриття другого браузера, але, по-перше, таких ситуацій незначна кількість, а, по-друге, для сайтів з автентифікацією кожному користувачеві буде присвоєно однакову користувацьку сесія, що дозволить об'єднати їх воєдино.

У випадку з використанням механізму ідентифікації сесій без надійного джерела визначення конкретної сесії (наприклад, якщо є доступ тільки до логів веб-сервера), доведеться приймати ризик, що за однією адресою джерела можуть ховатися кілька легітимних користувачів.

В існуючій літературі часто прийнято ідентифікувати сесії як пари "адреса джерела" і "значення User-Agent" за фіксований проміжок часу. Цей підхід дозволяє частково зменшити перекриття сесій для двох легітимних користувачів, які використовують одну і ту ж адресу джерела, оскільки існує великий набір різних версій браузера і параметрів оточення, що включаються в заголовок User-Agent. Однак, для веб-роботів, що змінюють заголовок в рамках своєї сесії, такий підхід навпаки, вносить зайві спотворення і поділяє одну сесію на велику кількість сесій.

Щоб боротися з цією проблемою, можна застосувати комбінацію з наступних підходів:

- Розділяти користувачів на основі адреси джерела;
- Розділяти користувачів за допомогою адреси і версії браузера;
- Аналогічно використовувати адресу і версію браузера, але об'єднувати сеанс для послідовних запитів з однаковими адресами джерела.

Перші два підходи очевидні, третій же є складним завданням з різними підходами до вирішення. По суті, він нагадує завдання з визначення пов'язаних

запитів з різними джерелами, яке виникає, коли веб-роботи зловмисників намагаються приховати свою активність використовуючи пули IP адрес, проксі-сервери та інші різні методи анонімізації.

Заголовок User-Agent використовується в завданні поділу на сесії з тієї причини, що він відображений практично в будь-яких логах веб-сервера (за замовчуванням відображається і в Nginx і в Apache). Якщо параметри логів модифіковані таким чином, що вони містять і інші заголовки - вони можуть використовуватися для того ж завдання, але, до них застосовні аналогічні обмеження, оскільки веб-роботи можуть змінювати їх вміст. Найкращим способом у цій ситуації буде вивід вмісту заголовка з унікальним ідентифікатором сеансу, встановлюваним веб-сервером (у цій ситуації зазвичай використовується Cookie). Встановити його можна, наприклад, з використанням наступного налаштування:
%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" \"%{session-q}C\"

Змінний User-Agent також можна використовувати як одну з характеристик виявлення веб-роботів. Справа в тому, що в цей заголовок включається операційна система, розрядність процесора і використання випадкової версії з іншою операційною системою є як ознакою того, що це інший легітимний користувач, так і те, що це веб-робот, що підробляє запити випадковим чином (особливо, якщо такі запити розташовані послідовно у вузькому проміжку часу або відповідають іншим встановленим правилам детектування підробки User-Agent в рамках одного джерела).

Поле User-Agent - це не просто назва і версія браузера. Як правило, там вказується ще й операційна система, розрядність процесора, мова, тип шифрування, модель мобільного пристрою, номер збірки та інші. Приклад значення, що використовується:

«Mozilla/5.0 (Linux; Android 6.0.1; SM-G920V Build/MMB29K) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/52.0.2743.98 Mobile Safari/537.36».

Наприклад, для *"Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36"* сайт "myip.ms" видає різні версії

браузера і вказує різні версії User-Agent, які він детектував. Крім цього, він містить список з 4881 версій User-Agent рядків, характерних для веб-роботів. Звичайно, це веб-роботи, які не приховують своєї присутності, але їх детектування теж несе користь, оскільки значна частина з них некоректно дотримується побажання і правил адміністраторів ресурсів щодо збору інформації, а також є веб-роботами зловмисників.

Роботизовані версії User-Agent можна детектувати різними способами. Досить повною базою є перелік регулярних виразів з проекту device-detector, який дозволяє не перераховувати кожну версію браузера веб-робота окремо, а пробігатися за списком з 237 правил.

1.8. Базові принципи виявлення веб-роботів

Базовим принципом, що лежить в основі системи виявлення веб-роботів, є коректна ідентифікація джерел запитів до веб-сервера.

Необхідно вирішити питання ідентифікації сесій, щоб отримати достатньо даних для класифікації кожного відвідування. Існує два основних підходи до виявлення веб-роботів. Відкладене виявлення включає сукупність аналітичних методів, заснованих на сигнатурному аналізі і машинному навчанні. Необхідно забезпечити цілісну модель поведінки веб-роботів на сайті, характеристик і сигнатур, властивих їх трафіку. Що стосується активного виявлення, воно включає в себе найпростіші методи фільтрації запитів і суто технічні пастки, що дозволяють виділяти автоматизовані засоби, або ускладнювати їм роботу.

Кожен веб-ресурс має певні вимоги та обмеження у використанні ресурсів, можливості показу CAPTCHA користувачеві та інших технічних прийомів. Необхідно оцінювати ризики збирання інформації веб-роботами та вартості побудови і підтримки системи виявлення та протидії. Виявлення веб-роботів базується на відмінності користувацької та роботизованої поведінки. Автоматизовані засоби завжди прагнуть систематизувати та впорядкувати алгоритм збору інформації та мінімізувати зайві дії на веб-ресурсі, до того ж,

розробники веб-роботів не мають інформації про статистичні характеристики користувацької поведінки на будь-якому веб-ресурсі. Даною інформацією володіє тільки власник ресурсу, і розробити програму, яка буде цю поведінку копіювати і одночасно вирішувати завдання масового збору інформації, не просто.

Підбиваючи підсумок, можна сказати, що підхід до виявлення повинен базуватися на точності, надійності і враховувати тимчасовий фактор.

Висновок по першому розділу

В першому розділі були проаналізовані загрози та вектори атак що можуть бути направлені на веб-ресурси компаній з ціллю як виведення зі строю окремих сервісів так і інфраструктури в цілому. В розділі також було наведено те, що ціллю зловмисників може бути як виведення з ладу, так і збирання певної інформації стосовно обладнання та ПЗ.

Виходячи з наведеної в розділі інформації стає зрозуміло, що для збереження активів компанії, що найменш необхідно вживати заходів, направлених на зменшення кількості сервісів що доступні з мережі інтернет, з ціллю мінімізації випадків збирання інформації або проведення атак.

2. ДОСЛІДЖЕННЯ МЕТОДІВ АУТЕНТИФІКАЦІЇ

В наш час повсякденне життя людини і діяльність сучасної компанії складно уявити без використання великої кількості додатків і сервісів, які з метою забезпечення безпеки вимагають проведення аутентифікації. Це призводить до необхідності для користувачів запам'ятовувати і зберігати велику кількість ідентифікаторів, що підвищує ймовірність використання нескладних пральних комбінацій, однакових паролів в різних системах або їх втрати. У той же час зростання числа логінів і паролів до різних облікових записів вимагає від адміністратора додаткових зусиль і часу на їх періодичну заміну і відновлення.

2.1 Основні підходи

Кожен сервіс вимагає перевірки автентичності та авторизації. В архітектурі сервісів зазвичай є інтерфейсний шлюз, керуючий підключеннями з зовнішнього світу, який потім здійснює підключення до внутрішніх сервісів для обробки запиту. Як і де виконуються ці два важливі кроки, багато в чому залежить від середовища сервісу і від того, наскільки добре захищені служби. Існує чотири основні підходи до реалізації доступу до інфраструктури.

2.1.1 Глобальна аутентифікація і авторизація

Наведений варіант можливий у разі повної і абсолютної впевненості в тому, що інтерфейс користувача або сам користувач (наприклад, за допомогою URL запитів) зробили глобальну авторизацію в системі. Проблема полягає в тому, що при передачі авторизації в глобальний контекст концептуальна бізнес — логіка переміщається з серверних до зовнішні інтерфейсів.

2.1.2 Глобальна аутентифікація, авторизація сервісів

Цей підхід, ймовірно, найкраще підходить для більшості систем. Досить підтримувати базовий рівень аутентифікації на інтерфейсному рівні глобальних сервісів. Потім, коли інтерфейсний сервер викликає внутрішні сервіси для виконання реальної роботи, він може надати контекст безпеки. Це дозволяє внутрішнім сервісів не піклуватися про те, як хтось проходить перевірку автентичності, але як і раніше може підтримувати прийняття рішень бізнес-логіки про те, які дії дозволено виконувати цьому контексту безпеки.

Візьмемо аналогічну модель входу в настільний комп'ютер - Windows, Mac, Linux не має значення. Користувач вводить свої облікові дані, і утиліта входу в систему аутентифікує його, а потім створює процес робочого столу, призначений контексту користувача. Всі додатки, що запускаються з цього моменту пов'язані з контекстом користувача і файлової системою, мережею і т.ін. Доступ авторизується з використанням контексту одного користувача. Додатки та ядро не переймаються тим, як користувач отримав цей контекст безпеки, але вони напевно будуть тримати користувача в межах того, до чого йому дозволено доступ.

Аутентифікація може бути зупинена, але в міру того, які виклики здійснюються в серверні внутрішніх сервісів, контекст безпеки або користувача повинен супроводжуватися будь — якими викликами, щоб сервіси могли реалізувати свою власну бізнес — логіку того, що дозволено тому хто їх викликає.

Важливе зауваження: цей метод вимагає наявності серйозної аутентифікації і авторизації між сервісами. Необхідно переконатися, що при отриманні запиту на виконання будь — яких дій у внутрішньому сервісі із заданим контекстом безпеки контексту безпеки можна довіряти.

2.1.3 Перевірка авторизації служби

Основна ідея підходу полягає в аутентифікації в кожному сервісі або додатку окремо.

Дане рішення можливо, але воно найвірогідніше буде порушенням гарних багаторівневих архітектур, не виконувати аутентифікацію глобально. У разі невеликого набору сервісів це, ймовірно не велика проблема - але як тільки проект виросте до певних великих розмірів, наприклад 20 сервісів, реалізація даного методу аутентифікації і авторизації вже не здасться гарною ідеєю.

2.2. Основні протоколи аутентифікації

Незважаючи на те, що існує стільки власних методів аутентифікації, скільки систем, які їх використовують, вони в значній мірі є варіаціями декількох основних підходів:

- звичайна перевірка справжності HTTP;
- ключ API;
- багатофакторна аутентифікація;
- протокол OAuth.

Ці підходи майже завжди розроблялися для усунення обмежень у ранніх комунікаціях і інтернет - системах, зазвичай використовують існуючи архітектурні підходи з новими реалізаціями, щоб забезпечити аутентифікацію.

2.2.1 Звичайна перевірка справжності HTTP

Базова аутентифікація (Base Authentication) HTTP рідко рекомендується через властиві їй вразливостей безпеки.

Одним з рішень є звичайна перевірка справжності HTTP. При такому підході агент користувача HTTP просто надає ім'я користувача і пароль для перевірки

автентичності. Цей підхід не вимагає файлів cookie, ідентифікаторів сеансів, сторінок входу в систему і інших спеціальних рішень, а оскільки він використовує сам заголовок HTTP, немає необхідності в рукостискання або інших складних системах відповіді.

Проблема полягає в тому, що, якщо система не використовує SSL, аутентифікація передана у відкритому вигляді на небезпечних лініях, і це піддається атакам man-in-the-middle, де користувач може просто захопити дані входу в систему і аутентифікуватись через HTTP - заголовок copy-cat, прикріплений до шкідливого пакету.

Крім того, навіть якщо SSL застосовується, це призводить до уповільнення часу відповіді. І навіть ігноруючи це, в свою базової формі HTTP-запит жодним чином не шифрується. Він кодується в base64 і часто помилково оголошується зашифрованим через це.

Звичайна перевірка справжності при використанні HTTP-протоколу має своє місце. У внутрішній мережі, особливо в ситуаціях Інтернету речей, де швидкість не має значення, наявність базової системи аутентифікації HTTP прийнятно як баланс між вартістю реалізації та фактичної функцією. Однак в якості загального рішення перевірки автентичності, базову перевірку автентичності HTTP, слід використовувати рідко.

2.2.2 Ключ API

Ключі API (API-key) є галузевим стандартом, але не повинні розглядатися як цілісна міра безпеки.

Ключі API були створені як свого роду виправлення для ранніх проблем аутентифікації HTTP Basic Authentication і інших подібних систем. При такому підході кожному першому користувачеві присвоюється унікальне згенероване значення, що означає, що користувач відомий. Коли користувач намагається повторно увійти в систему, його унікальний ключ (іноді генерується з їх апаратної

комбінації і IP-даних, а іноді випадковим чином генерується сервером, який їх знає) використовується, щоб довести, що він той же користувач, що і раніше.

З одного боку, це дуже швидко. Здатність довести ідентичність один раз і рухатися далі дуже гнучка, і саме тому вона вже багато років використовується в якості підходу за замовчуванням для багатьох постачальників API. Крім того, настройка самої системи досить проста, а управління цими ключами після їх створення ще простіше. Це також дозволяє системам очищати ключі, тим самим видаляючи аутентифікацію після видалення ключа і забороняючи вхід в будь-яку систему, яка намагається використовувати віддалений ключ.

Проблема, однак, у тому, що ключі API часто використовуються для того, чим вони не є - ключ API не є методом авторизації, це метод аутентифікації. Оскільки будь — хто, хто робить запит сервісу, передає свій ключ, теоретично, цей ключ може бути підібраний так само легко, як і будь — яка передача по мережі, і якщо будь-яка точка у всій мережі небезпечна, вся мережа піддається впливу. Це робить ключі API не найбільш оптимальним варіантом реалізації - часто неправильно і принципово небезпечно організовані, вони, тим не менш, мають своє місце, коли належним чином захищені і затиснуті системами авторизації.

2.2.3 Багатофакторна аутентифікація

Багатофакторна аутентифікація (Multi-factor authentication, MFA) - це система безпеки, яка перевіряє особу користувача, вимагаючи кілька облікових даних. Замість того, щоб просто запитувати ім'я користувача і пароль, MFA вимагає додаткові облікові дані, такі як код зі смартфона користувача, відповідь на секретне питання, відбиток пальця або розпізнавання особи.

MFA є ефективним способом забезпечення підвищеної безпеки. Традиційні імена користувачів і паролі можуть бути вкрадені, і вони стають все більш уразливими для атак грубої сили. MFA створює кілька рівнів безпеки, щоб підвищити впевненість в тому, що користувач, запитувач доступ, насправді той, за

кого він себе видає. З MFA, кіберзлочинець може вкрасти одні облікові дані, але не зможе підтвердити особистість при додатковій перевірці.

Приклади багатофакторної аутентифікації включають використання комбінації цих елементів для аутентифікації:

- коди, що генеруються додатками для смартфонів;
- бейджи, USB-пристрої або інші фізичні пристрої;
- короткі токени, сертифікати;
- відбитки пальців;
- коди, надіслані на адресу електронної пошти;
- розпізнавання особи;
- сканування сітківки або райдужної оболонки ока;
- поведінковий аналіз;
- оцінка ризику;
- відповіді на питання особистої безпеки.

Коли справа доходить до MFA, зазвичай передбачається три типи факторів аутентифікації:

- те, що знає користувач (знання): пароль або PIN-код;
- те, що є у користувача (володіння): бейдж або смартфон;
- те, ким є користувач (успадкування): біометрія, відбитки пальців або розпізнавання голосу.

Останні рішення MFA включають додаткові чинники, враховуючи контекст і поведінку при аутентифікації. наприклад:

- місце знаходження користувача при спробі отримати доступ, наприклад в кафе або будинку;
- коли користувач намагається отримати доступ, наприклад, пізно вночі або протягом робочого дня;
- який пристрій використовується, наприклад смартфон або ноутбук;
- до якої мережі звертається користувач, приватної або публічної.

Часто званий адаптивною аутентифікацією, цей тип MFA враховує контекст для маркерів імен входу, які є незвичайними. Коли людина намагається

аутентифікуватися в незвичайному контексті, адаптивний MFA може посилити безпеку, запросивши додаткові облікові дані. Наприклад, якщо користувач входить в систему з кафе пізно вночі - а це не типово для цього користувача - інструмент MFA може запросити від користувача ввести код, відправлений на телефон користувача.

2.2.4 Протокол OAuth

OAuth об'єднує перевірку справжності та авторизацію, щоб забезпечити більш складний контроль області дії і допустимості.

OAuth технічно є не тільки методом аутентифікації, але і авторизації.

При такому підході користувач входить в систему. Потім ця система запросить аутентифікацію, зазвичай у вигляді токена. Потім користувач перенаправить цей запит на сервер перевірки автентичності, який або відхилить, або дозволить цю перевірку справжності. Звідси, маркер надається користувачеві, а потім сервісу. Такий маркер може бути перевірений в будь-який час незалежно від користувача ініціатором запиту для перевірки і може використовуватися протягом довгого часу зі строго обмеженим обсягом і терміном дії.

Це принципово більш безпечна і потужна система, ніж інші підходи, в значній мірі тому, що вона дозволяє м'яко встановлювати область (тобто, які системи дозволяє користувачеві аутентифікувати ключ) і дійсність (тобто ключ не повинен бути навмисно відкликаний системою, він автоматично стане застарілим згодом).

Як і у всьому, є деякі основні плюси і мінуси цього підходу. З одного боку, він має явну перевагу, коли справа доходить до рівня безпеки, який він може запропонувати, і з цієї причини OAuth швидко стає фактичним вибором для тих, хто вважає за краще уникати ключів API. OAuth можна легко налаштувати, і аутентифікація буде працювати неймовірно швидко.

2.2.5 Особливості OAuth

OAuth - це свого роду протокол протоколів або мета-протокол, що означає, що він забезпечує корисну відправну точку для інших протоколів (наприклад, OpenID Connect, NAPS і UMA). Це аналогічно тому, як WS-Trust використовувався в якості основи для WS-Federation, WS-SecureConversation і т. ін.

Спочатку роботи з OAuth важливо враховувати, що він вирішує ряд важливих потреб, які є у більшості постачальників API, в тому числі:

- делегований доступ;
- зниження обміну паролями між користувачами і третіми особами (так званий "пароль анти-патерн");
- анулювання доступу.

Коли використовується шаблон захисту від паролів, і користувачі діляться своїми обліковими даними зі стороннім додатком, єдиний спосіб скасувати доступ до цього додатка - змінити пароль. Отже, всі інші делеговані права доступу також скасовуються. З OAuth користувачі можуть заборонити доступ до певних додатків, не втрачаючи змоги відкрити іншу програму, яка повинна продовжувати діяти від їх імені.

У OAuth існує чотири основних актора:

- власник ресурсу (Resourse Owner): сутність, яка контролює дані, що надаються API, зазвичай кінцевий користувач,
- клієнт (Client): мобільний додаток, сайт та ін. Що хоче отримати доступ до даних від імені власника ресурсу,
- сервер авторизації (Authorization Server): служба маркерів безпеки (STS) або, сервер OAuth, який видає маркери,
- сервер ресурсів (Resource Server): служба, що надає дані, API.

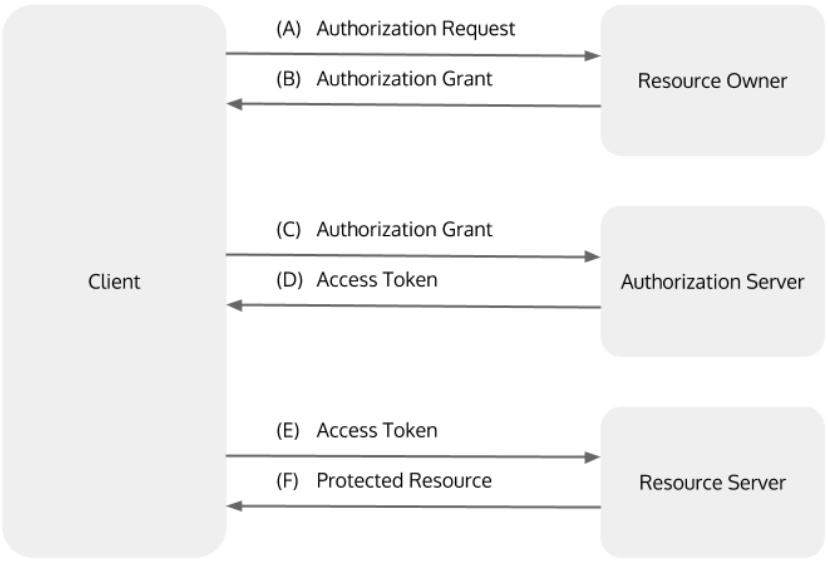


Рис. 2.1. Абстрактний опис процесу авторизації в Oauth

OAuth визначає оточення. Оточення по суті - дозвіл або делеговані права, які власник ресурсу дозволяє клієнту використати від його імені. Клієнт може запросити певні права, але користувач може надати тільки деякі з них. Права, запитувані клієнтом, часто відображаються на будь-якому екрані призначеного для користувача інтерфейсу. Однак така сторінка не може бути представлена користувачеві. Якщо користувач вже надав клієнтові певні права, ця сторінка буде пропущена.

У OAuth існують наступні види токенів:

- маркери доступу: це маркери, представлені API;
- маркери поновлення: вони використовуються клієнтом для отримання нового маркера доступу від AS;
- маркер ідентифікатора: визначає OpenID Connect.

Прикладом маркера доступу виступає сеанс, який створюється для користувача при вході на веб-сайт. Поки цей сеанс дійсний, користувач може продовжувати взаємодіяти з веб-сайтом без необхідності повторного входу в систему. Після того, як цей сеанс закінчився, користувач може отримати новий, увійшовши знову з паролем. Маркери поновлення схожі на паролі в цьому порівнянні. Крім того, як і паролі, клієнт повинен зберігати маркери поновлення в

безпеці. Вони повинні зберігатися в безпечному сховищі облікових даних. Втрата цих токенів потребує відкликання всіх дозволів, які були отримані користувачами.

Порядок того, як система передає токени серед сервісів, безумовно, вплине на загальну безпеку. Існує два різні способи передачі токенів:

- за значенням;
- за посиланням.

Вони аналогічні тому, як мова програмування передає дані, ідентифіковані змінними.

Середовище виконання або скопіює дані в стек при виклику функції, що викликається (за значенням), або відправить покажчик на дані (за посиланням). Аналогічним чином маркери будуть або містити всі ідентифікаційні дані в міру їх передачі, або бути посиланням на ці дані.

У випадках, коли токен передається по посиланню, необхідно пам'ятати, що знадобиться спосіб разименувати токен. Зазвичай це робиться за допомогою API, що викликає нестандартну кінцеву точку, що надається сервером OAuth.

OpenID Connect заснований на OAuth. OpenID Connect забезпечує протокол, організовуючи діяльність сервісів. Профіль цього також додає нові кінцеві точки, потоки, типи маркерів, області та багато іншого. OpenID Connect (який часто скорочується OIDC) був розрахований на застосування в мобільній сфері. Види маркерів, які він визначає, повинні бути JWT, які були також розроблені для сценаріїв з низькою пропускною здатністю. При використанні OAuth, доступний як делегований доступ, так і доступ до можливостей системи. Це ознаменує меншу кількість залежностей і зниження рівня складності.

2.2.6 OAuth 2.0

В оновленій версії OAuth 2.0 реалізовано більше потоків OAuth, щоб забезпечити кращу підтримку додатків, які не ґрунтуються на браузері. Це основна критика OAuth від клієнтських додатків, що не були засновані на браузері. Наприклад, в OAuth 1.0 десктопні додатки або додатки для мобільних телефонів

повинні були вказати користувачеві відкрити браузер для потрібної служби, пройти аутентифікацію в службі і скопіювати маркер зі служби назад в додаток. Основна критика тут спрямована проти користувацького досвіду. З OAuth 2.0 тепер існують нові способи отримання авторизації користувача в додатках.

OAuth 2.0 більше не вимагає, щоб клієнтські програми мали криптографію, додаток може зробити запит, використовуючи тільки виданий маркер по HTTPS.

Підписи OAuth 2.0 набагато менш складні. Немає більше спеціального розбору, сортування або кодування.

Токени доступу OAuth 2.0 є "недовговічними". Як правило, маркери доступу OAuth 1.0 можуть зберігатися протягом року або більше. У протоколі OAuth 2.0 крім токенів доступу передбачені токени поновлення. Передбачається, що маркери доступу можуть бути недовговічними (тобто заснованими на сеансі), а маркери поновлення можуть бути "часом життя". Користувач може використовувати маркер поновлення для отримання нового маркера доступу, а не для повторної авторизації в додатку.

Нарешті, OAuth 2.0 повинен мати чіткий поділ ролей між сервером, відповідальним за обробку запитів OAuth, і сервером, що обробляють авторизацію користувача.

2.2.7 OpenID Connect

OpenID Connect - OpenID Connect 1.0 - це простий рівень ідентифікації поверх протоколу OAuth 2.0. Це дозволяє клієнтам перевіряти особистість кінцевого користувача на основі аутентифікації, що виконується сервером авторизації, а також отримувати базову інформацію про профілі кінцевого користувача сумісним і REST — подібним чином.

OpenID Connect дозволяє клієнтам всіх типів, включаючи веб-, мобільні і JavaScript-клієнти, запитувати і отримувати інформацію про аутентифіковані сеанси і кінцевих користувачів. Набір специфікацій є розширюваним, дозволяючи учасникам використовувати додаткові функції, такі як шифрування

ідентифікаційних даних, виявлення постачальників OpenID і управління сеансами, коли це має сенс для них.

OpenID Connect визначає токени ID. Вони призначені для клієнта. На відміну від маркерів доступу і маркерів поновлення, непрозорих для клієнта, маркери ідентифікаторів дозволяють клієнту знати, серед іншого:

- як користувач пройшов перевірку автентичності (який тип облікових даних був використаний);
- коли користувач пройшов перевірку автентичності;
- різні властивості авторизованого користувача (наприклад, ім'я, прізвище та таке інше).

Це корисно, коли компанія вимагає трохи інформації для настройки призначеного для користувача інтерфейсу. Якщо вашому клієнтові потрібні дані про користувача, йому видається маркер ID і попереджаються проблеми в майбутньому.

2.2.8 SAML і OpenID

Метою Security Assertion Markup Language (SAML) є надання способу для реалізації технології єдиного входу (Single Sign On - SSO), більш сучасним стандартом є протокол OpenID описаний вищє, який за принципом роботи схожий з OAuth, але його метою є аутентифікація користувачів, а не безпечне надання особистих даних.

Проведемо порівняння стандартів SAML і OpenID. Аутентифікація в них проходить аналогічно тому, як проходить отримання даних від користувача в OAuth, а саме: користувач бажає здійснити вхід на програмному клієнта, для цього він перенаправляється до провайдера ідентифікації, де він підтверджує свою особистість, потім він перенаправляється назад, а програмний клієнт отримує його ідентифікатор.

З OpenID дуже просто приймати всіх користувачів, що приходять з різних провайдерів ідентифікації. З іншого боку, для підтримки взаємодії з SAML

провайдерами зазвичай потрібно написання додаткового коду і їх список обмежений лише явно обраними.

Ще однією відмінністю є те, що при аутентифікації через OpenID програмний клієнт отримує його ідентифікатор, якому він може довіряти, але не може довіряти іншим даними про користувача, наприклад його імені, і адресою електронної пошти.

З іншого боку, існує явна довіру між програмним клієнтом і SAML провайдером ідентифікації. Ми можемо отримати всю інформацію про користувача включаючи його ім'я та e-mail, і цієї інформації можна буде довіряти, просто через відносини довіри. Це означає, що ми просто повинні довіряти, що провайдер ідентифікації якимось чином перевірив всі дані. Якщо ж користувач приходить з SAML токеном випущеним невідомим провайдером, то програмний клієнт просто відкидає його.

OpenID заснований на принципі REST побудови API, а дані подаються у вигляді JSON веб токенів (JWT). SAML заснований на XML.

2.3 Аналіз SSO-технологій

Для аналізу було вибрано наведені нижче параметри в якості основних індикаторів захищеності протоколу і реалізації його редиректів. Крім цього, підтримуваний крос-доменний транспорт буде далі основним принципом, за яким конкретна реалізація протоколу буде ставитися до тієї чи іншої моделі логіна.

- Крос-доменний транспорт
- Автоматичне перенаправлення
- Варіативність параметра redirect_uri
- Домени кінцевих точок протоколів

2.3.1 Крос-доменний транспорт

Основні канали передачі даних від провайдера до клієнта, в контексті веб-протоколів фактично є крос-доменної передачею даних в браузері. OAuth і OpenID по документації використовують для обміну даними тільки перенаправлення (Redirects). Але крім перенаправлень широко використовуються й інші способи, головним з яких є PostMessage - спеціальний метод з HTML5 для безпечного обміну повідомленнями між двома різними доменами. Flash вже є застарілим способом, який використовувався до поширення PostMessage API. При цьому флеш-об'єкти впроваджується в обидва вікна і зв'язуються для обміну даними через власний механізм LocalConnection Flash API. Інший застарілий транспорт Fragment, який разом з Flash ще використовується, але багато в чому лише для забезпечення сумісності, влаштований так, що передане іншому домену повідомлення встановлюється в URL Fragment допоміжного вікна або фрейму, звідки воно разом з повним URL вже є доступним для читання всім в межах домену. Під терміном Fragment об'єднано кілька варіацій такого методу, наприклад, rmr або ifpc, використовувані в Google API і бібліотеці Shindig. Cookie позначає посилену версію протоколу OAuth з кодом авторизації, який передається через заголовок Set-Cookie замість URL.

Очевидно, що описані в стандартах перенаправлення (Redirects) використовуються як транспорт абсолютно всіма протоколами. PostMessage, Fragment і Flash є допоміжними механізмами, які в основному використовуються для різних складних веб-додатків на стороні клієнта в рамках відповідного варіанту реалізації OAuth. Такі транспорти, втім, описують передачу даних тільки на клієнтській стороні, перед цим дані запитуються з сервера і передаються на вхід Javascript-обробника в браузері або за допомогою редиректів, або за допомогою XMLHttpRequest-запиту (XHR). При цьому, OpenID-додатки не використовують складних видів транспорту, покладаючись тільки на редіректи (Redirects). Посилений Cookie-транспорт з усіх розглянутих SSO-сервісів використовується

тільки браузером Google Chrome в процесі входу до облікового запису користувача для синхронізації, а його аутентифікація побудована цілком на Google OAuth 2.

2.3.2 Автоматичні перенаправлення

Кожен запит в рамках OpenID для підтвердження автентичності користувача, а також кожен запит на авторизацію за допомогою OAuth може бути направлений користувачеві явно для підтвердження у вигляді вікна або діалогу, а може бути і оброблений автоматично, якщо запит клієнта був дозволений раніше. Цікаво зауважити, що всі SSO-сервіси при цьому підтримують спеціальний прапор, який вказує провайдеру, що запит слід перенаправити за допомогою редиректу в будь-якому випадку, навіть якщо запит ще не підтверджений (перенаправити з повідомленням помилки). Такий параметр передбачений для деяких випадків, коли потрібно дуже швидко і без участі користувача дізнатися, чи підтверджений запит, і відразу отримати токен в разі успіху. Подібна поведінка описана в стандарті OpenID, але вона не вказана в жодній з версій OAuth, проте відомо, що і OAuth-провайдери теж використовують цей прапор.

Фактично це означає, що впровадження SSO-протоколу, що підтримує такий прапор, призводить до появи Open Redirect'a, що вважається вразливістю середньої критичності. Атакуючому досить лише зареєструвати свого SSO-клієнта і адресу для перенаправлення, а потім сконструювати URL для запиту на авторизацію. Широко відомі проблеми, пов'язані з небезпекою співіснування SSO-протоколів разом з Open Redirect'амі, однак SSO-протоколи реалізують можливість Open Redirect'a самі по собі через свої особливості.

2.3.3 Варіативність параметра redirect_uri

Виконуючи запит на аутентифікацію або авторизацію, клієнт вказує свій URL, куди провайдер направить браузер після обробки запиту. У разі OAuth це

oauth_callback / redirect_uri, а в OpenID такий параметр openid.return_to. Як правило, redirect_uri повинен збігатися із заздалегідь вказаним в налаштуваннях клієнта значенням для надійності, але іноді розробниками протоколу використовуються менш суворі перевірки. В залежності від того, чим суворіша перевірка redirect_uri, тим менше вразливий протокол. Facebook використовував перевірку на рівні static domain, дозволяючи використовувати URL в межах зареєстрованого домену або будь-яких його піддоменів і залишаючи таким чином багато можливостей для атак, згодом ця перевірка була замінена на більш сувору типу tail, як і у Google OpenID, коли redirect_uri повинен починатися з зареєстрованого значення, дозволяючи перенаправлення лише на URL з дочірніми шляхами або з додатковими параметрами в рядку запиту. Найбільш сувора перевірка static в OAuth від Google, яка пропускає тільки URL, повністю ідентичний зареєстрованому, хоча при цьому можливо ретранслювати повідомлення про помилки на всі піддомени (subdomains) і на всі дочірні шляхи (tail). Microsoft OAuth вживає всі URL, які знаходяться на тому ж домені або будь-якому піддомені.

2.3.4 Домени кінцевих точок

Іноді окремі частини веб-сервісів, потенційно небезпечні або містять призначений для користувача контент виносяться на власні піддомени, або взагалі на спеціальні ізольовані домени, подібно до того, як переклад сторінки в Google Translate відображається через спеціальний домен-пісочницю googleusercontent.com. Останній критерій показує розташування кінцевих точок кожного протоколу, і виявляється, що SSO-сервіс майже завжди розміщується на найважливішому піддомені провайдера, пов'язаному с обліковим записом користувача SSO: наприклад, обидва SSO-протоколу від Google винесені на accounts.google.com. Ймовірно, це пов'язано з тим, що перш ніж авторизувати або відмовити в запиті, кінцева точка протоколу зобов'язана знати, який саме користувач надіслав запит, тобто знати його Cookie з ідентифікатором його сесії. В такому випадку, SSO-протоколи реалізують можливість Open redirect'a на

найважливіших піддоменів провайдера з усіх можливих, що набагато значніше, ніж, наприклад, open redirect на домені сервісу для скорочення посилань провайдера.

2.4 Моделі логіна

Дослідження можливих застосувань open redirect'ов SSO-протоколів в цілому було б не дуже зручне через те, що додатки, що використовують єдиний вхід, сильно різняться між собою, як, наприклад, веб-додатки та нативні програми. Разом з цим, реалізації протоколу теж різні для різних типів додатків. Очевидно, що і в стандартах OpenID і OAuth теж даються класифікації додатків. Для дослідження роботи перенаправлень вони незручні тим, що не враховують використовуваний транспорт, тобто те, які саме типи перенаправлень використовуються. Тому далі ми визначимо 4 різних сценарію роботи SSO-протоколу, що грунтуються на тому, через який транспорт передаються повідомлення.

2.4.1 Стандартна модель

Додатки першого типу є веб-додатками або браузерних додатками і використовують для передачі даних тільки самі перенаправлення HTTP, або звичайні HTTP-запити, слідуючи стандартам OAuth або OpenID. Наприклад, до цієї групи належать authorization grant flow з OAuth 2, а також implicit flow, якщо в останньому не використовуються ніякі проксі-iframe для передачі токенів. У той же час OAuth-клієнт може використовувати скрипти і sdk від сервіс-провайдера, якщо вони виконуються на домені клієнта.

2.4.2 Проксі-модель

Наступна група включає веб-додатки і браузерні додатки, що використовують для отримання даних на стороні клієнта спеціальні проксі-вікна (iframe). Проксі, виконуваний на домені провайдера, отримує дані від провайдера і передає їх на стороні браузера потрібного клієнта, використовуючи різні механізми на кшталт postMessage API, Flash та інших. Такий підхід використовується не тільки для браузерних додатків, але і для веб-додатків з серверною частиною: тоді облікові дані, отримані від проксі, передаються з клієнтської сторони на сервер для подальшої роботи та опрацювання. При цьому проксі-фрейм не призначений для виконання API-запитів на сервер.

2.4.3 Модель Контролер

Деякі браузерні додатки використовують iframe-контролер для отримання облікових даних. На відміну від фрейму-проксі, який відразу завантажується з токеном в URL або всередині своєї сторінки, контролер отримує облікові дані через XHR-запит на домен провайдера. Такий фрейм не повинен перезавантажуватися, оскільки призначений для виконання різних API-запитів і не тільки для аутентифікації або авторизації.

2.4.4 Нативна модель

В останній групі знаходяться різні типи встановлених додатків: нативні і мобільні. Такі додатки, на відміну від трьох попередніх сценаріїв, виконуються поза браузером, і тому використовують для підтримки SSO-протоколу або вбудований, або зовнішній браузер.

Висновок до другого розділу

Під час опрацювання матеріалів по другому розділу було встановлено, що ефективним вирішенням описаних у першому розділі загроз та проблем є використання технології єдиного входу (Single Sign-On, SSO).

Single Sign-On - механізм, за допомогою якого єдина дія по аутентифікації і авторизації користувача може дозволити йому доступ до всіх комп'ютерів і систем, до яких цей користувач має дозвіл на доступ, без необхідності введення безлічі паролів.

Основними перевагами технології SSO є: зменшення кількості паролів в системі, забезпечення необхідного рівня безпеки без заподіяння незручностей користувачам, зменшення часу на введення інформації для аутентифікації, зниження витрат на IT-службу за рахунок зменшення кількості запитів по відновленню логінів і паролів, а також завдяки можливості фактичного зменшення кількості інтерфейсів що матимуть прямий доступ до мережі інтернет.

3. ТЕХНОЛОГІЯ УПРАВЛІННЯ АКТИВАМИ IBM SECURITY ACCESS MANAGER ДЛЯ ОРГАНІЗАЦІЇ НАСКРІЗНОЇ АУТЕНТИФІКАЦІЇ

IBM Security Access Manager або ISAM допомагає спростити доступ для користувачів, підвищуючи безпеку розгортання мобільних, хмарних і веб-технологій та Інтернету речей тощо, ISAM надає можливість автоматизувати доступ користувачів до всіх корпоративних додатків, використовуючи політики і профілі доступу. Рішення забезпечує єдину реєстрацію в таких типах додатків, як Microsoft Windows, Web, Java, Mainframe, і може бути встановлено практично на будь-яких кінцевих пристроях мережі.

Для розгортання рішення можна вибрати локальну середу, віртуальне або апаратний пристрій або контейнерну середу Docker. ISAM допомагає знайти золоту середину між зручністю використання і безпекою за рахунок керування доступом з урахуванням ризиків, єдиного входу в систему, інтегрованих засобів управління доступом, об'єднання облікових записів і мобільної багатофакторної ідентифікації.

3.1 Налаштування інтегрування ISAM з SAML 2.0

Після встановлення ISAM у той спосіб який є найбільш прийнятним, треба звернути увагу на необхідність зміни паролю, та при необхідності мови інтерфейсу (на жаль Українська мова на цей час ідсутня) ISAM (у горі з правого боку)

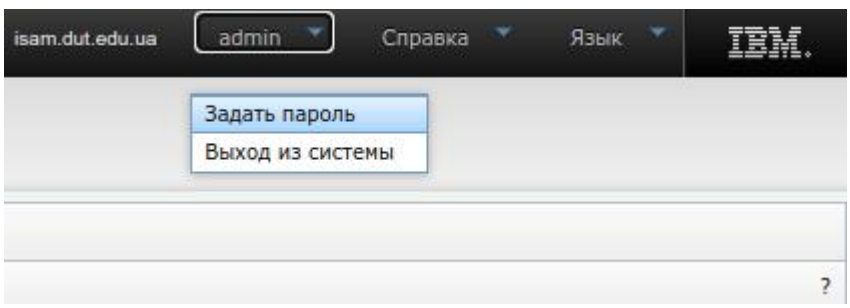


Рис. 3.1. Варіант зміни паролю за допомогою інтерфейсу.

Після зміни паролю, та проведення налаштувань у відповідності до офіційних керівництв, для створення, та конфігурування взаємодії з SAML 2.0 треба виконати наступні дії.

Адреса 192.168.88.199 наведена як приклад.

Знаходячись в інтерфейсі керування перейти за посиланням https://192.168.88.199/mga/federation та з лівого боку у горі треба натиснути “Додати”

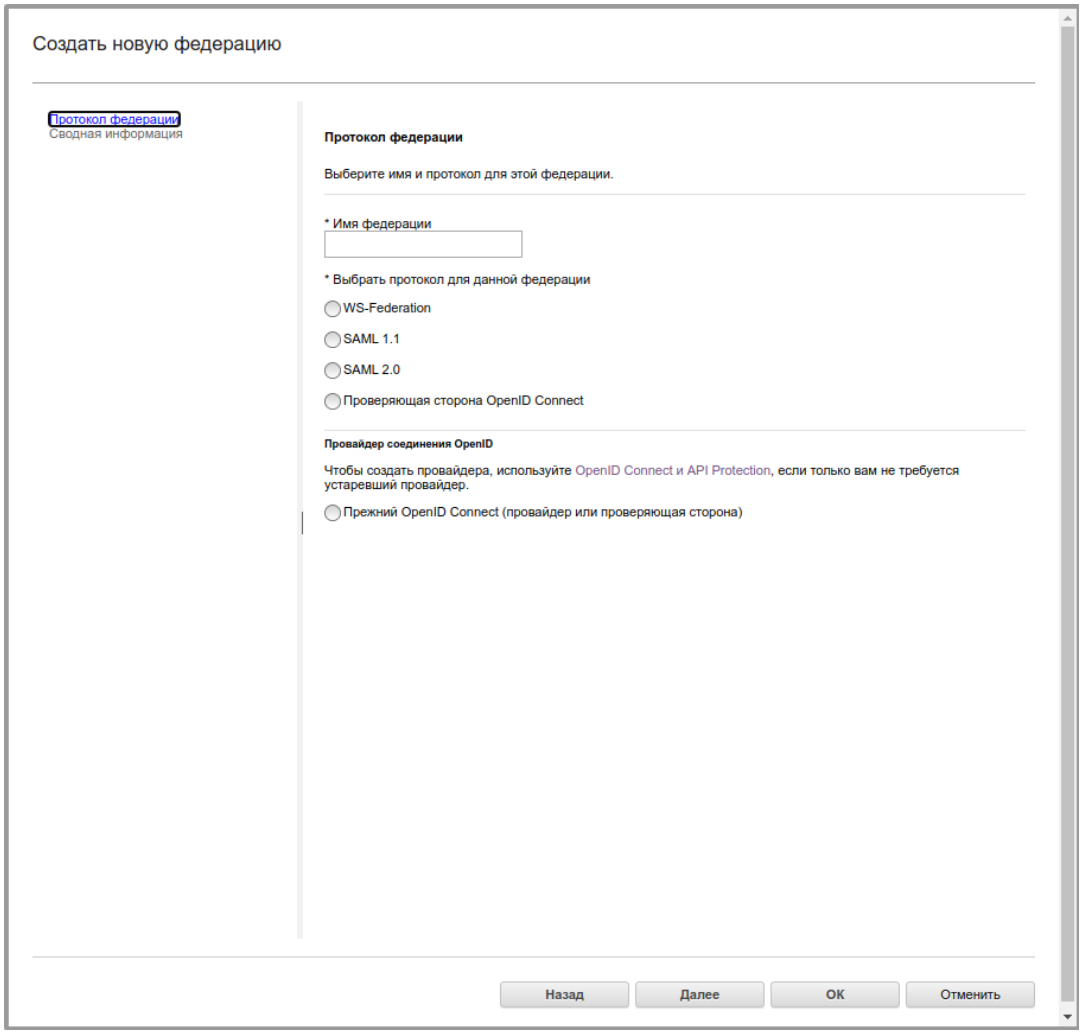


Рис. 3.2 Приклад вікна що виникає при створенні федерації

У вікні «Створити нову федерацію» на екрані «Протокол федерації» вкажіть ім'я в полі «Ім'я федерації», для параметра «Виберіть протокол для цієї федерації» виберіть SAML 2.0 і натисніть «Далі».

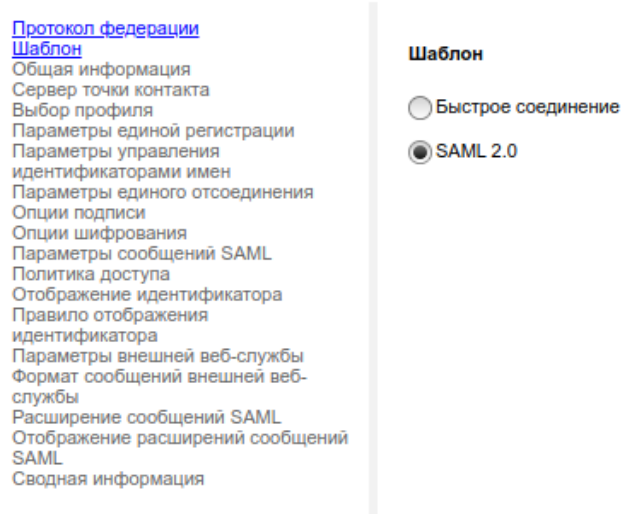


Рис 3.3. Приклад вибору на етапі “Шаблон”

Як зазначено в Рис. 3.3 на етапі “Шаблон” необхідно обрати SAML 2.0 після чого, на екрані «Загальна інформація» вкажіть ім'я для «Назва компанії», для «Визначте свою роль» виберіть «Постачальник послуг» та натисніть «Далі». На екрані “Сервер точки контакта” необхідно вказати URL-адресу у форматі:

https: // <ім'я хоста зворотного проксі-сервера>: <номер порту зворотного проксі-сервера> / <ім'я з'єднання>

Після чого необхідно натиснути кнопку “Далі”

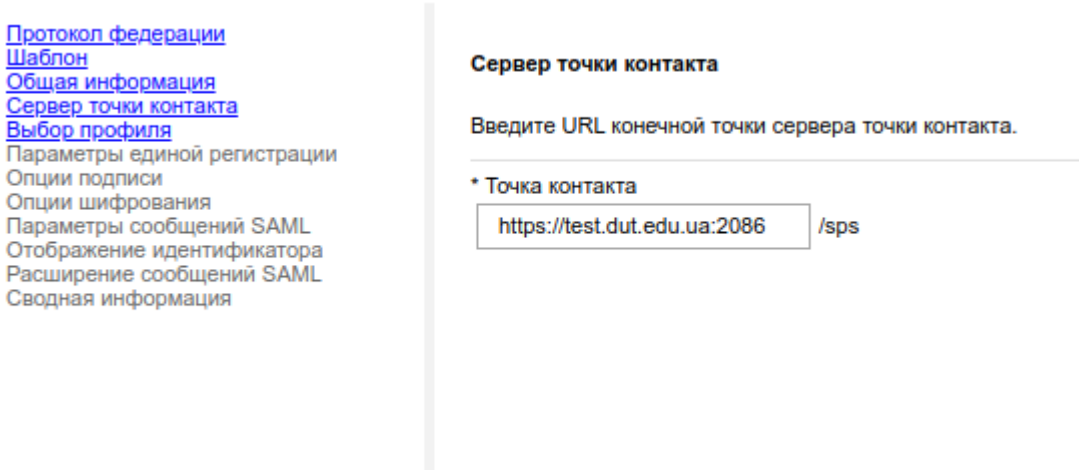


Рис. 3.4 Приклад заповнення URL у формі “Сервер точки контакта”

На наступному екрані, “вибору профілю” залиште обраної єдиний вхід для веб-браузера та натисніть «Далі».

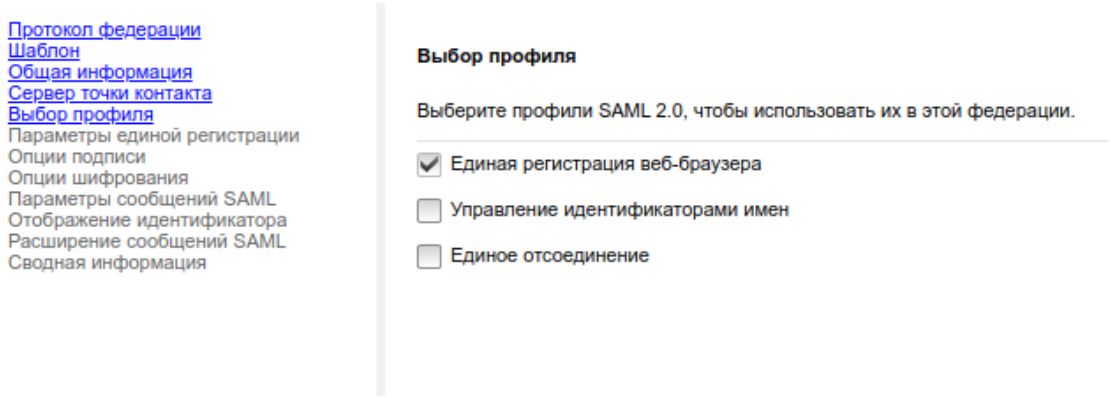


Рис. 3.5. Приклад заповнення форми “Вибір профілю”

Наступним кроком є заповнення форми налаштувань єдиного входу, де у розділі «Підтримувані прив'язки» відміченим є тільки «POST HTTP». У розділі NameID за замовчуванням списку, виберіть urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress.

Встановіть прапорець «Вимагати підпису вихідних запитів аутентифікації SAML» і натисніть «Далі».

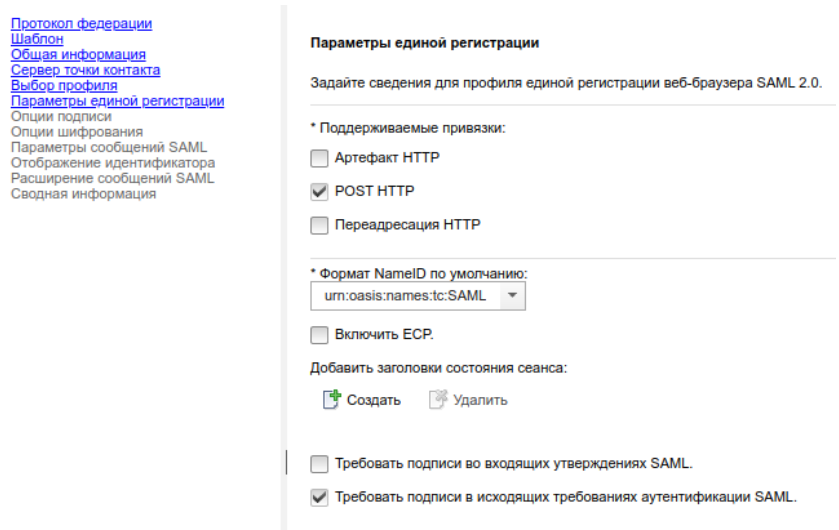


Рис. 3.6. Приклад заповнення форми “Параметри єдиної реєстрації”

Наступна форма, «Параметри підпису», в цій формі потрібно у списку «База даних сертифікатів» обрати rt_profile_keys, а у списку «Метка сертифіката» обрати server і натиснути далі.

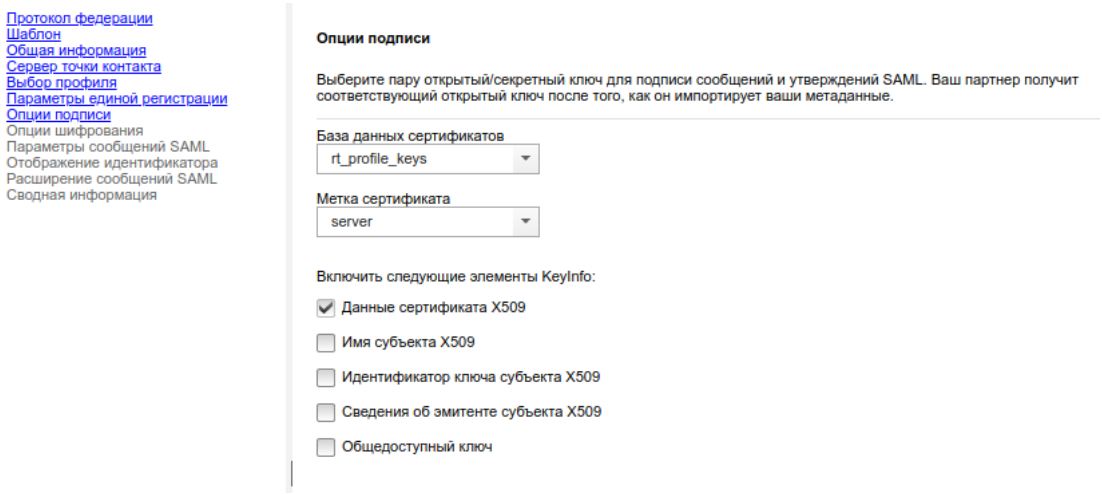
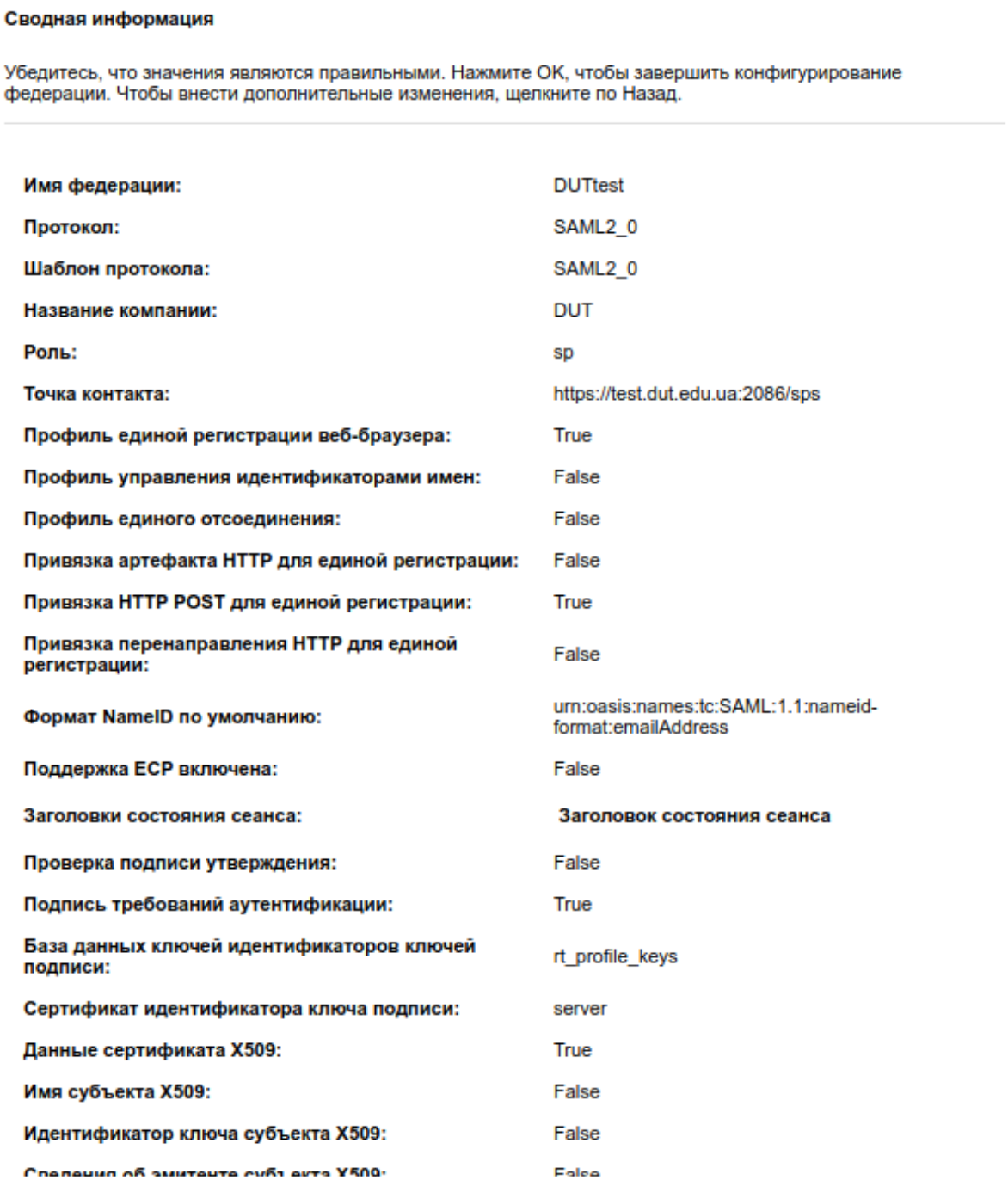


Рис. 3.7. Приклад заповнення форми “Параметри єдиної реєстрації”

У формі параметри шифрування, необхідно обрати “База даних сертифікатів” та «Метка сертифіката» такі самі як і на попередньому етапі. На наступних етапах, “Параметри повідомлень SAML”, “Відображення ідентифікатору” та “Розширене повідомлення SAML” параметри залишаємо за замовченням, та натискаємо кнопку “Далі”. У формі “Зведена інформація” переглядаємо обрані параметри, та натискаємо кнопку “Ок”



Сводная информация

Убедитесь, что значения являются правильными. Нажмите OK, чтобы завершить конфигурирование федерации. Чтобы внести дополнительные изменения, щелкните по Назад.

Имя федерации:	DUTtest
Протокол:	SAML2_0
Шаблон протокола:	SAML2_0
Название компании:	DUT
Роль:	sp
Точка контакта:	https://test.dut.edu.ua:2086/sps
Профиль единой регистрации веб-браузера:	True
Профиль управления идентификаторами имен:	False
Профиль единого отсоединения:	False
Привязка артефакта HTTP для единой регистрации:	False
Привязка HTTP POST для единой регистрации:	True
Привязка перенаправления HTTP для единой регистрации:	False
Формат NameID по умолчанию:	urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress
Поддержка ECP включена:	False
Заголовки состояния сеанса:	**Заголовок состояния сеанса**
Проверка подписи утверждения:	False
Подпись требований аутентификации:	True
База данных ключей идентификаторов ключей подписи:	rt_profile_keys
Сертификат идентификатора ключа подписи:	server
Данные сертификата X509:	True
Имя субъекта X509:	False
Идентификатор ключа субъекта X509:	False
Сведения об эмитенте субъекта X509:	False

Рис. 3.8. Приклад форми “Зведена інформація”

Після чого треба виділити створену “Федерацію” та натиснути “експорт” для збереження файлу у форматі .xml на комп'ютері.

3.2 Налаштування постачальника довіреної служби хмарної аутентифікації RSA

Щоб налаштувати службу перевірки автентичності SAML агента для IBM Security Access Manager необхідно виконати наступні дії:

Увійдіть в консоль адміністрування хмари і перейти до “каталогу додатків” через меню “додатки”, знайдіть IBM Security Access Manager і натисніть “Додати”, щоб додати з'єднувач.

На сторінці «Основна інформація» вкажіть назву програми в полі «Ім'я» і натиснути «Далі».

На сторінці профілю з'єднання необхідно імпортувати метадані, що були експортовані на етапі “Налаштування інтегрування ISAM з SAML 2.0” та зберегти їх.

В якості URL-адреси підключення необхідно вказати URL-адресу у форматі. *https: // <ім'я хоста зворотного проксі-сервера>: <номер порту зворотного проксі-сервера> / <ім'я вузла> / sps / <ім'я федерації> / saml20 / <логін>*

У формі “Підпис відповіді SAML” необхідно створити і завантажити файл certificateBundle.zip. З zip-архіву необхідно завантажити ключі cert.pem і private.key у відповідні поля, після чого встановити прапорець “Включити сертифікат”.

Наступним кроком є заповнення форми «Постачальник послуг» URL-адресу ACS:

https: // <ім'я хоста зворотного проксі-сервера>: <номер порту зворотного проксі-сервера> / <ім'я вузла для федерації> / sps / <ім'я федерації> / saml20 / login

Аудиторія (ідентифікатор об'єкта постачальника послуг): *https: // <ім'я хоста зворотного проксі-сервера>: <номер порту зворотного проксі-сервера> / <ім'я вузла для федерації> / sps / <ім'я федерації> / saml20*

В розділі ідентифікації користувача, у списку тип ідентифікації виберіть Адреса електронної пошти. У списку “Властивість” виберіть поштову скриньку або sAMAccountName як значення в залежності від типу властивості, яка налаштована для прийому в IBM Security Access Manager.

На сторінці “Доступ користувача” необхідно налаштувати параметри політики доступу і натиснути “Далі”, після чого на наступній сторінці треба “Зберегти і завершити”. Останнім кроком треба опублікувати зміни, після чого в “каталогу додатків” що в меню “додатки”, необхідно знайдіть IBM Security Access Manager і експортувати метадані для подальшого налаштування RSA SecurID Access в якості партнера федерації в IBM Security Access Manager.

3.3 Налаштування партнера федерації IBM Security Access Manager

Для налаштування партнера необхідно перейти, як приклад, за посиланням https://192.168.88.199/mga/federation# обрати необхідну федерацію, після чого натиснути “Партнери” та у вікні що з’явилось натиснути “Додати”. Наступним кроком необхідно виконати імпорт метаданих що були експортовані з хмари, після чого, у формі «Параметри єдиного входу» в поле «Цільовий URL-адрес за замовчуванням» вкажіть URL-адрес, для якого дозволений доступ через федерацію, і натисніть «Далі», усі наступні форми можна залишити за замовченням, після натискання кнопки “Ок” треба задіяти зміни.

3.4 Налаштування федерації IBM Security Access Manager

3.4.1 Конфігурування зворотного проксі IBM Security Access Manager

Для налаштування зворотного проксі, для прикладу, треба перейти за посиланням https://192.168.88.199/wga/reverseproxy після чого треба натиснути кнопку “Створити”

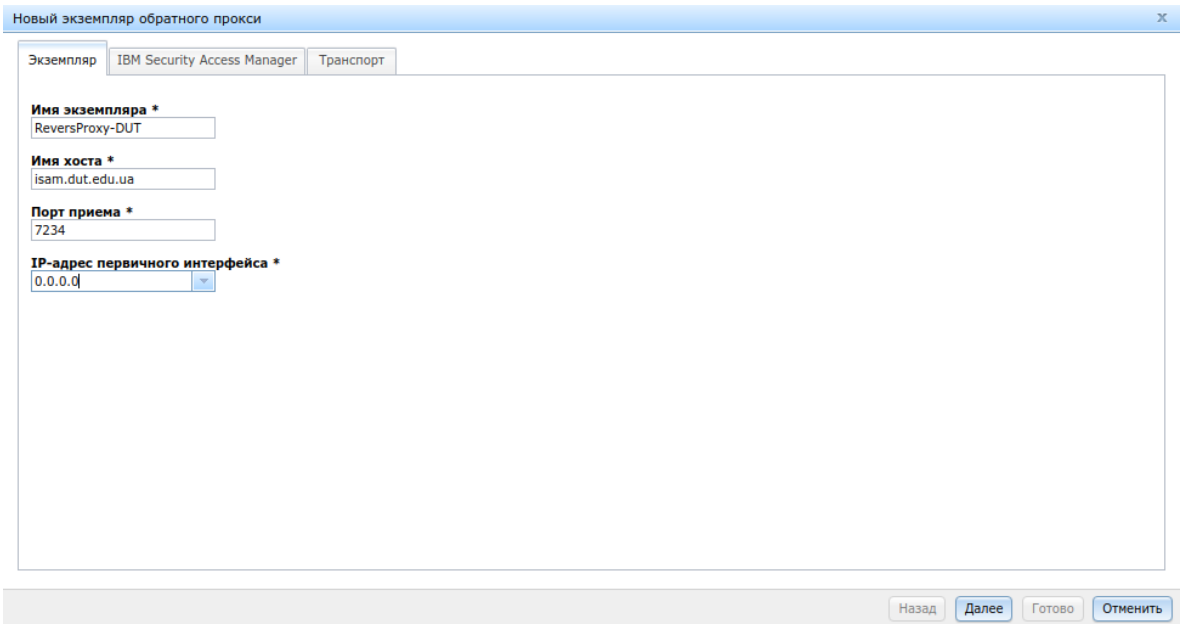


Рис. 3.9. Приклад вікна створення нового реверс проксі

У формі, що наведена на Рис. 3.8 необхідно вказати ім'я в полі «Ім'я примірника», в списку «IP-адреса» для основного інтерфейсу виберіть IP-адреса зі списку , а також зазначити порт прийому після чого натиснути «Далі».

На вкладенці “IBM Security Access Manager” треба задати пароль, який слід запам’ятати, та натиснути «Далі».

На наступному етапі, на вкладенці “Транспорт” треба зазначити порти, які будуть слухатись, слід звернути увагу, на наведеній інформації з 1 розділу, де зазначаються можливі проблеми в забезпеченні безпеки в разі використання не захищених з’єднань. Керуючись цими знаннями порт 80 активувати не будемо.

Після завершення конфігурування слід натиснути кнопку “Готово”, в разі коректного налаштування, в верхній частині екрану ми отримаємо повідомлення, приведене на Рис. 3.9.

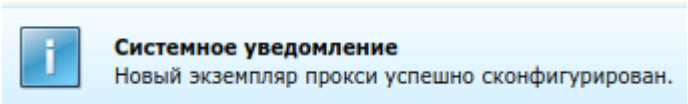


Рис. 3.10. Повідомлення про створення нового екземпляру реверс проксі

На наступному крокові, необхідно обрати зворотний проксі-сервер, створений вище, і натисніть «Управління»> «Конфігурація»> «Редагувати файл конфігурації» та привести його до вигляду, як приклад, наведеному на Рис. 3.10

Після чого необхідно зберегти внесені зміни, та перезапустити обраний зворотній проксі сервер.

```
#
# FILENAME
#       webseald.conf
#
# DESCRIPTION
#       Configuration file for the Access Manager WebSEAL server (webseald)
#


##############################
# WEBSEAL GENERAL
##############################
[server]

# WebSEAL server instance name. Typically, this is based on the hostname of the
# machine and the instance name of the server.
server-name = isam.dut.edu.ua-ReversProxy-DUT

# If web-host-name is set WebSEAL will use this for the server's hostname.  If
# left unset WebSEAL will attempt to automatically determine the server's
# hostname.  On systems with many hostnames, interfaces or WebSEAL instances
# the automatic determination may not always be correct requiring this manual
# setting.
# web-host-name = www.webseal.com
web-host-name = isam.dut.edu.ua
#----------------------
# THREADS AND CONNECTIONS
#----------------------

# Number of WebSEAL worker threads
# The number of configured worker threads specifies the number of
# concurrent incoming requests that can be serviced by this server
# instance. Choosing the optimal number depends on the quantity
# and type of traffic on your network. Modifying this value should
# be done carefully to ensure optimal performance. Please consult
# the WebSEAL Administration Guide for further information.
worker-threads = 300
```

Рис. 3.11. Приклад відредагованого файлу розширеної конфігурації

3.4.2 Конфігурування федерації IBM Security Access Manager

Щоб застосувати конфігурацію перевіряючьої сторони і Агента SSO до федерації IBM Security Access Manager потрібно перейти, як приклад, за посиланням https://192.168.88.199/wga/reverseproxy обрати екземпляр зворотного проксі-сервера, який був доданий в попередніх умовах, і натисніть «Управління» > «Конфігурація AAC і федерації» > «Управління федерацією» та натиснути “додати”, повинна відобразитись форма як на Рис. 3.11.

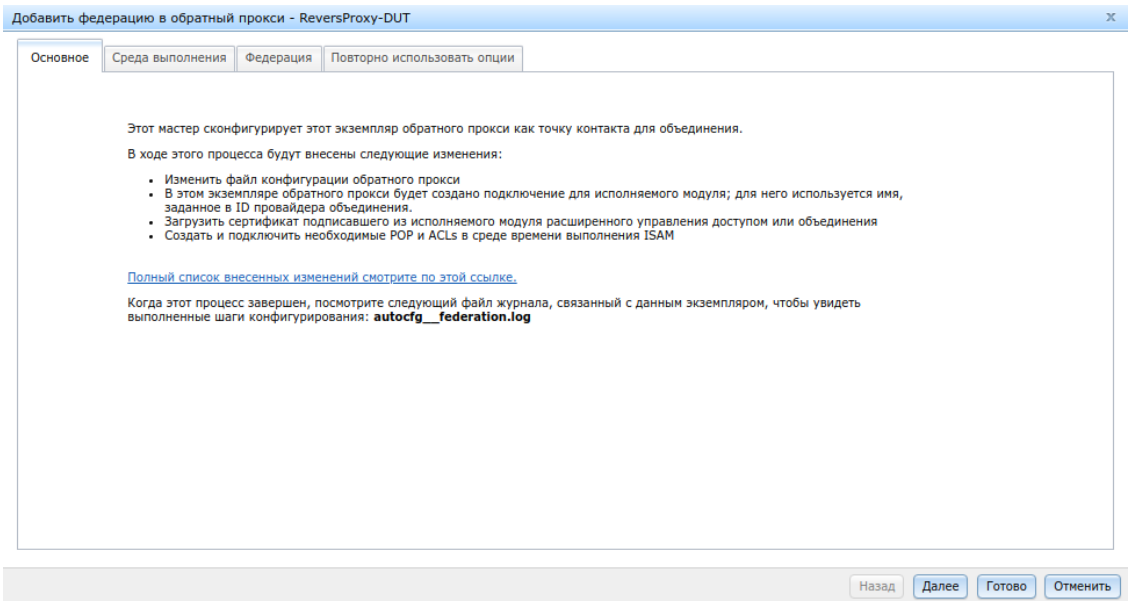


Рис. 3.12. Форма додавання федерації в зворотній проксі

Далі, на вкладенці “Середа виконання” треба вказати ім’я користувача, а також пароль. Крім цього повинні бути зазначені валідні значення в поля “Ім’я хосту” та “Порт” приклад наведений на Рис 3.12.

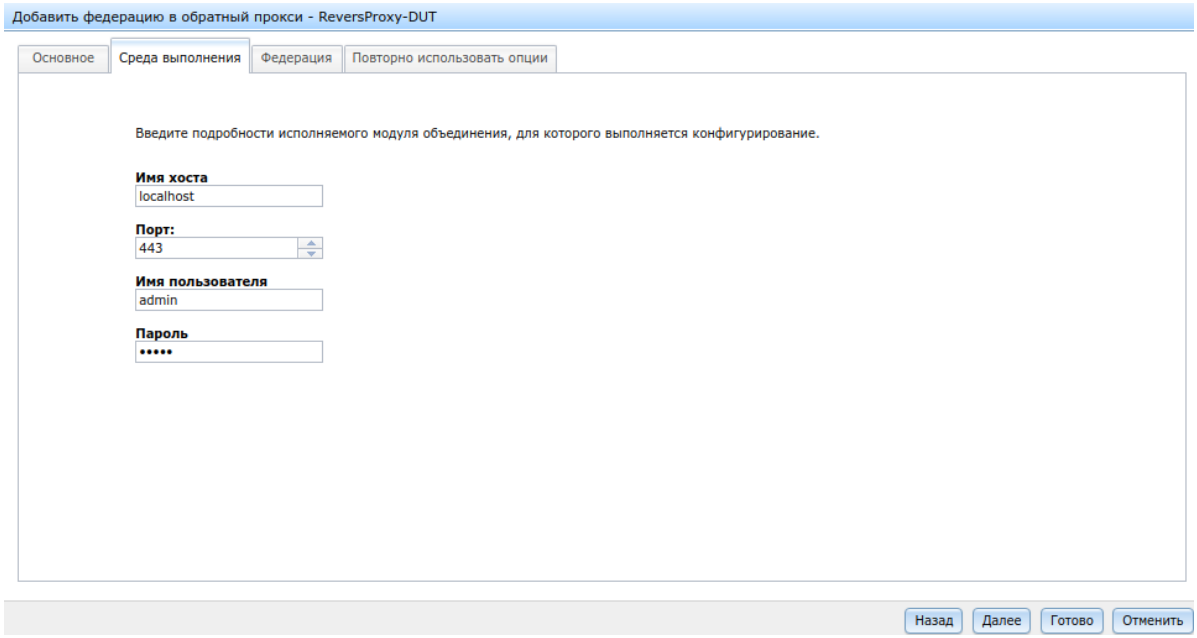


Рис. 3.13. Приклад заповнення вкладенки “Середа виконання”

На наступному кроці, з випадаючого меню необхідно обрати екземпляр федерації, та на останньому кроці, без змін форми треба натиснути кнопку

“Готово”. Після збереження конфігурації необхідно перезавантажити екземпляр зворотного проксі.

3.4.3 Налаштування профілю точки контакту IBM Security Access Manager

Для налаштування зазначеного етапу, як приклад, необхідно перейти за посиланням https://192.168.88.199/mga/poc

На сторінці «Точка контакту» необхідно обрати «Поточний профіль» або, “Ім'я користувача Access Manager та розширені атрибути”, що дозволяють тільки відомим заздалегідь зареєстрованим користувачам виконувати єдиний вхід в систему постачальника послуг, також можна обрати “облікові дані Access Manager”, що дозволяють всім користувачам, які пройшли перевірку автентичності, входити в систему постачальника послуг.

Для доступу до ресурсу за допомогою системи єдиного входу необхідно використовувати URL складений наступним чином:

https://<isam_hostname>:<port_number>/<junction_name>/sps/<federation_name>/saml20/logininitial?RequestBinding=HTTPPost&PartnerId=<provider_ID>&NameIdFormat=Email&Target=https://<target_application_location>

Де:

isam_hostname - ім'я хоста зворотного проксі-сервера.

port_number - номер порта зворотного проксі-сервера.

junction_name - ім'я вузла, налаштованого для зворотного проксі-сервера.

federation_name - ім'я федерації, створеної для постачальника послуг.

provider_ID - ідентифікатор постачальника посвідчень.

target_application_location - додаток, в який користувач може увійти за допомогою системи єдиного входу.

Висновок по третьому розділу

Система IBM Security Access Manager або ISAM крім того, що спрощує взаємодію користувачів з корпоративними мережевими ресурсами, істотно знижує навантаження на системних адміністраторів і технічну підтримку. Використання SSO в якості єдиної точки входу підвищує швидкість доступу до даних, а сам механізм аутентифікації автоматизується. Для підвищення рівня безпеки оброблюваної в системі інформації в ISAM реалізована підтримка багатофакторної і адаптивної аутентифікації з використанням всіх можливих сучасних чинників. Даний продукт спрямований на зниження витрат бізнесу, підвищення загального рівня безпеки внутрішньої мережі підприємства і зручності роботи користувачів.

Крім цього слід зазначити, що ISAM, як і інші продукти IBM гарно документовані, що надає змогу, фактично без певного досвіду, самостійно у відносно невеликий проміжок часу, сконфігурувати та застосувати ISAM.

ВИСНОВКИ

Під час виконання магістерської роботи було проаналізовано сучасний стан проблем та загроз безпеці веб ресурсів корпоративного сегменту. Під час опрацювання розділу була приділена увага автоматизованим атакам, та зазначено, що для збереження активів компанії, що найменш необхідно вживати заходів, направлених на зменшення кількості сервісів що доступні з мережі інтернет з ціллю мінімізації випадків збирання інформації або проведення атак.

В продовження першого розділу, у другому, були досліджені основні методи, етапи, протоколи аутентифікації, а також проведений аналіз SSO-технологій. Було встановлено, що ефективним вирішенням описаних у першому розділі загроз та проблем є використання технології єдиного входу (Single Sign-On, SSO), основними перевагами якої зокрема є зменшення кількості паролів в системі, забезпечення необхідного рівня безпеки без заподіяння незручностей користувачам, а також зниження витрат на IT-службу за рахунок зменшення кількості запитів по відновленню логінів і паролів, та завдяки можливості фактичного зменшення кількості інтерфейсів що матимуть прямий доступ до мережі інтернет.

Практична частина, що полягала в конфігурування IBM Security Access Manager для організації наскрізної аутентифікації ґрунтувалась на теоретичних знаннях отриманих в перших двох розділах, та за ціль мала:

- Налаштування інтегрування ISAM з SAML 2.0
- Налаштування постачальника довіреної служби хмарної аутентифікації RSA
- Налаштування партнера федерації IBM Security Access Manager
- Налаштування федерації IBM Security Access Manager
- Налаштування профілю точки контакту IBM Security Access Manager

ПЕРЕЛІК ПОСИЛАНЬ

1. Retail e-commerce sales in the United States from 2017 to 2023 [Електронний ресурс] URL: https://www.statista.com/statistics/272391/us-retail-e-commerce-sales-forecast/

2. Common Attack Pattern Enumeration and Classification [Електронний ресурс] URL: https://capec.mitre.org/

3. The WASC Threat Classification Online [Електронний ресурс] URL: http://projects.webappsec.org/w/page/13246978/Threat%20Classification

4. OWASP [Електронний ресурс] URL: https://www.owasp.org/index.php/Main_Page

5. OWASP Top 10 – 2017 [Електронний ресурс] URL: https://wiki.owasp.org/images/9/96/OWASP_Top_10-2017-ru.pdf

6. Komarova A. et al. Comparison of Authentication Methods on Web Resources //International Conference on Intelligent Information Technologies for Industry. – Springer, Cham, 2017.

7. ICS vulnerabilities: 2018 in review [Електронний ресурс] URL: https://www.ptsecurity.com/upload/corporate/ww-en/analytics/ICS-vulnerabilities-2019-eng.pdf

8. User-Agent [Електронний ресурс] URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent

9. A Method for Web Robots Control [Електронний ресурс] URL: http://www.robotstxt.org/norobots-rfc.txt

10. Phillip Griffin. Biometric Electronic Signatures //ISSA Journal. November 2017

11. NIST SP 800-30. [Електронний ресурс] URL: https://csrc.nist.gov/publications/detail/sp/800-30/rev-1/final

12. Thomas R. Peltier. Information Security Risk Analysis. – 2005. CRC Press Taylor & Francis Group. 6000 Broken Sound Parkway NW, Suite 300.

13. English L.P. Information Quality Management: The Next Frontier // Quality Congress AsQs. Annual Quality Congress Proceeding. 2001

14. EU Regulaton 910/2014 of 23 July 2014 eIDAS (Electronic Identification, Authentication and Trust Services). [Електронний ресурс] URL: http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32014R0910&from=EN

15. User authentication guidance for information technology systems. ITSP 30.031.v.3. Government od Canada. April 2018. [Електронний ресурс] URL: https://www.cse-cst.gc.ca/en/system/files/pdf_documents/itsp.30.031v2-eng_0.pdf

16. Harrison, M. (2004). Human error analysis and reliability assessment. Workshop on Human Computer Interaction and Dependability, 46th IFIP Working Group 10.4 Meeting, Siena, Italy, July 3–7, 2004.

17. Petsas T. et al. Two-factor authentication: is the world ready?: quantifying 2FA adoption //Proceedings of the eighth european workshop on system security. – ACM, 2015. – С. 4.

18. Wiefling, Stephan; Lo Iacono, Luigi; Dürmuth, Markus. "Information web-site on Risk-based Authentication". Risk-based Authentication.

19. E.J. Goh, H. Shacham, N. Modadugu, and D. Boneh. SiRiUS: Securing remote untrusted storage. In Proc. Network and Distributed Systems Security (NDSS) Symposium 2003, pages 131–145, 2013.

20. В. Г. Олифер, Н. А. Олифер «Сетевые операционные системы» Серия: Учебник для вузов. Издательство: Питер, 2008 г. ISBN 978-5-91180-528-9

21. National Computer Security Center. A guide to understanding discretionary access control in trusted systems. Fort George Gordon Meade. 1987.

22. Делимся опытом по интеграции SSO средствами SAML 2.0 [Електронний ресурс] URL: https://habr.com/ru/company/true_engineering/blog/193662/

23. https://ru.wikipedia.org/wiki/SAML

24. Щербаков А.Ю. Современная компьютерная безопасность. Теоретические основы. Практические аспекты. Учебное пособие. – М.: Книжный мир, 2009.

25. https://www.ibm.com/ru-ru/products/verify-for-workforce-iam

26. https://www.ibm.com/support/pages/ibm-security-access-manager-90-documentation-pdfs-part-1-2

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

ВІДГУК РЕЦЕНЗЕНТА
на магістерську роботу

студентки Кушнір Ірини Михайлівни
на тему: «Технології управління активами корпоративної інформаційної системи на базі рішення IBM Security Access Manager»

Актуальність: У зв'язку з все більшим використанням віддаленого доступу до корпоративних мереж та активів під час пандемії COVID-19, використання систем єдиного входу стає все більш актуальним. Зазначені системи окрім зменшення ризику компрометації та вторгнення, надають користувачам можливість використання одного паролю для входу до великої кількості сервісів та систем, що в свою чергу зменшує ризик атак що базуються на людському факторі. Виходячи з зазначеного вище, тема магістерської роботи є актуальною та своєчасною.

Позитивні сторони:

1. На основі проведеного аналізу, в роботі було встановлено загрози веб ресурсам корпоративного сегменту, вплив автоматичних систем, та проблеми виявлення сканування, та протидії таким системам.
2. Було досліджено методи та протоколи автентифікації, що в контексті роботи є логічним та послідовним кроком для аргументації використання SSO технологій.
3. Запропоновано варіант налаштування системи IBM SAM, для організації єдиного входу з описанням використання реверс — проксі.
4. Текст викладено достатньо грамотно, послідовно. Сформульовано чіткі та змістовні висновки. Графічний матеріал оформлено якісно. Список науково-технічної літератури свідчить про вміння користуватись матеріалами за темою магістерської роботи.

Недоліки:

Запропонований варіант технології управління активами на базі рішень ISAM бажано було б показати на прикладі конкретного підприємства.

Висновок: робота заслуговує оцінки "**відмінно**", а студентка — Кушнір Ірина Михайлівна гідна присвоєння кваліфікації 2149.2 професіонал із організації інформаційної безпеки, викладач вищих навчальних закладів.

Якість магістерської роботи	
Виконано на замовлення підприємства	
Виконано за тематикою НДР	
Виконано з макетом	
Виконано з застосуванням ЕОМ та МПТ	√
Має практичну цінність	√
Проект-частина комплексної теми	

Підпис рецензента _____ _____
(П.І.Б.)

_____
(посада, науковий ступінь, вчене звання)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

ПОДАННЯ
ГОЛОВІ ДЕРЖАВНОЇ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ МАГІСТЕРСЬКОЇ РОБОТИ

Направляється студент Кушнір І.М. до захисту магістерської роботи
(прізвище та ініціали)
спеціальності 125 Кібербезпека
освітньо-професійної програми Інформаційна та кібернетична безпека
(шифр і назва спеціальності)
на тему: «Технології управління активами корпоративної інформаційної системи на базі рішення IBM Security Access Manager».

Магістерська робота і рецензія додаються.

Директор інституту _____ Савченко В.А.
(підпис) (прізвище та ініціали)

Довідка про успішність

Кушнір ІМ. за період навчання в інституті
(прізвище та ініціали студента)
ННІЗІ з 2019 року по 2021 рік повністю виконав навчальний план за напрямом підготовки, спеціальністю з таким розподілом оцінок за:
національною шкалою: відмінно ____%, добре ____%, задовільно ____%;
шкалою ECTS: A ____%; B ____%; C ____%; D ____%; E ____%.

Секретар інституту, факультету (відділення) _____ Черниш О.В.
(підпис) (прізвище та ініціали)

Висновок керівника магістерськоїї роботи

Студентка ***Кушнір І.М.*** обрала тему роботи, метою якої було дослідити зміст технології управління активами корпоративної інформаційної системи на базі рішення IBM Security Access Manager та розробити варіант технології управління їх захистом на підприємстві. Перелік використаних джерел свідчить про вміння магістром розбиратись в наукових питаннях та застосовувати їх при дослідженнях. Під час виконання магістерської роботи Кушнір І.М. показала відмінну теоретичну та практичну підготовку, вміння самостійно вирішувати питання і робити висновки. Роботу виконувала сумлінно, акуратно та вчасно за планом.

Все це дозволяє оцінити виконану магістерську роботу студентки Кушнір Ірини Михайлівни на оцінку **«відмінно»** та присвоїти їй кваліфікацію 2149.2 професіонал з організації інформаційної безпеки, викладач вищих навчальних закладів.

Керівник магістерської роботи _____ Гайдур Г.І.
(підпис) (прізвище та ініціали)
“____” _____ 2020 року

Висновок кафедри про магістерську роботу

Магістерська робота розглянута. Студент Кушнір І.М.
(прізвище та ініціали)
допускається до захисту даної магістерської роботи в Державній екзаменаційній комісії
Завідувач кафедри Інформаційної та кібернетичної безпеки
(назва)

_____ Гайдур Г.І.
(підпис) (прізвище та ініціали)

“____” _____ 2020 року