МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ
КАФЕДРА БЕЗПЕКИ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
_____ О.Г. Корченко
«____»_____ 2020 р.

На правах рукопису
УДК 004.056.5

КВАЛІФІКАЦІЙНА РОБОТА

ЗДОБУВАЧА ОСВІТНЬОГО СТУПЕНЯ «МАГІСТР»

Тема: «Аналітична система оцінки безпеки хмарних сховищ»

Виконавець: Д.В. Дем'яненко

Керівник: А.О. Корченко

Нормоконтролер: О.О. Бурбела

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет: Кібербезпеки, комп'ютерної та програмної інженерії
Кафедра: Безпеки інформаційних технологій
Освітній ступень: «Магістр»
Спеціальність: 124 «Системний аналіз»
ОПП: «Консолідована інформація»

ЗАТВЕРДЖУЮ
Завідувач випускової кафедри
_____О.Г. Корченко
«____»_____ 2020 р.

.

ЗАВДАННЯ
на виконання магістерської кваліфікаційної роботи
Дем'яненко Діани В'ячеславівни

1. Тема: *Аналітична система оцінки безпеки хмарних сховищ*

затверджена наказом ректора від 01.10.2020 *№*1867.

2. Термін виконання: *з 05.10.2020* по 03.01.2021.

3. Вихідні дані: *проаналізувати особливості використання CloudComputing; визначити ризики та вразливості пов'язані з використанням концепції CloudComputing; проаналізувати моделі існуючих систем виявлення атак; створити модель системи аналізу та оптимізації на основі концепції CloudComputing; виконати експерименти для перевірки ефективності роботи системи аналізу.*

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці): *Аналіз вразливостей та ризиків, пов'язаних з використанням хмарних технологій. Розробка моделі аналітичної системи оцінки безпеки хмарних сховищ. Розробка та експериментальне дослідження системи аналізу безпеки хмарних сховищ.*

КАЛЕНДАРНИЙ ПЛАН

Виконання магістерської кваліфікаційної роботи

п/п	Завдання	Термін виконання етапів	Підпис керівника
1.	Уточнення завдання		
2.	Аналіз літературних джерел		
3.	Обґрунтування вибору рішення		
4.	Збір інформації		
5.	Аналіз вразливостей та ризиків, пов’язаних з використанням хмарних технологій		
6.	Розробка моделі аналітичної системи оцінки безпеки хмарних сховищ		
7.	Розробка та експериментальне дослідження системи аналізу безпеки хмарних сховищ		
8.	Оформлення презентації		
9.	Отримання рецензій		
10.	Захист в ЕК		

Здобувач Д.В. Дем’яненко

(підпис, дата)

Керівник кваліфікаційної роботи А.О. Корченко

(підпис, дата)

РЕФЕРАТ

Кваліфікаційна робота складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел і має ___ сторінок основного тексту, 9 рисунків, 9 таблиць, ___сторінок додатків. Список використаних джерел містить 53 найменування і займає 5 сторінок. Загальний обсяг роботи __ сторінок.

Метою роботи є розробка аналітичної системи оцінки безпеки хмарних сховищ для запобігання атакам.

Оскільки концепція CloudComputing може забезпечити необмежені обчислювальні ресурси, ця концепція є досить важливою у великих системах аналізу даних.

Робота враховує та досліджує основні переваги та недоліки використання концепції CloudComputing, розглядаючи можливі варіанти застосування в системах безпеки. Проаналізовано існуючі системи виявлення вторгнень та представлено їх переваги та недоліки.

В рамках роботи була розроблена модель системи аналізу та оптимізації файлів тривоги для систем виявлення атак з покращеними експлуатаційними характеристиками. Представлена система дозволяє вибрати лише унікальні атаки, що, в свою чергу, забезпечує високий рівень виявлення вторгнень. Ця функція досягається за допомогою алгоритму MapReduce. Аналітична система зберігає результати роботи в середовищі CloudComputing, що, в свою чергу, встановлює новий рівень цілісності та безпеки даних.

СИСТЕМА ВИЯВЛЕННЯ АТАК, СИСТЕМА АНАЛІЗУ ДАНИХ, CLOUDCOMPUTING, РОЗПОДІЛЕНІ ОБЧИСЛЕННЯ, РОЗПОДІЛЕНА СИСТЕМА, MAPREDUCE.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

СВА	–	система виявлення атак;
ПП	–	постачальники послуг;
КП	–	користувачів послуг;
ПІ	–	постачальниками інфраструктури;
IaaS	–	інфраструктура як послуга;
PaaS	–	платформа як послуга;
SaaS	–	програмне забезпечення як послуга.

ВСТУП

Актуальність хмарних технологій (CloudComputing) поширюються на всі сфери діяльності: бізнес, медицину, освіту та науку, медіа, фінанси та державний сектор. Зростаюча тенденція збільшення відсотка використання електронної комерції та користувачів Інтернету в цілому, які використовують її для здійснення покупок в Інтернет-магазинах, зберігання та резервного копіювання інформації з декількох пристроїв, розміщення інформації в Інтернеті, призводить до того, що використання фізичних ресурсів є неефективним і вимагають значних ресурсів для постійного підвищення продуктивності та технічного обслуговування. Тому доцільніше використовувати хмарні сервіси, які приносять користь кожному типу користувачів.

Користувачі хмарних технологій можуть отримувати обчислювальну потужність та потужність хмарного сховища рівно в обсязі, необхідному для роботи їхніх систем, тим самим уникаючи проблеми дефіциту ресурсів, що перетворюється на прямі економічні вигоди. Крім того, користувачі, яким доводиться обробляти велику кількість запитів, зможуть обробляти інформацію набагато швидше. Така гнучкість ресурсів та їх необмежена потужність роблять технології CloudComputing дуже цікавою та перспективною концепцією.

Хоча концепція хмарних технологій досить цікава, вона занадто молода і вимагає дуже детальних досліджень щодо її застосування та інтеграції з іншими системами. Слід зазначити, що в зарубіжних джерелах поняття CloudComputing активно вивчається, є значні результати і навіть практичні реалізації цієї концепції. Однак мало уваги приділяється застосуванню концепції CloudComputing та оцінці безпеки хмарних сховищ. CloudComputing може бути ефективним інструментом для створення систем аналізу безпеки даних, що є сьогодні дуже **важливим науковим завданням**.

Метою роботи є розробка аналітичної системи оцінки безпеки хмарних сховищ для запобігання атакам.

Для досягнення поставленої мети роботи потрібно вирішити наступні **задачі**:

– провести аналіз вразливостей та ризиків, пов’язаних з використанням Cloud Computing;

– розробити аналітичну систему аналізу файлів тривоги на основі концепції CloudComputing;

– провести експериментальне дослідження для визначення ефективності системи аналізу.

Наукова новизна. Удосконалено аналітичну систему оцінки безпеки хмарних сховищ за рахунок оптимізації файлів тривог, систем виявлення атак, використання розподілених обчислень, що дало можливість в автоматичному режимі аналізувати трафік, генерувати файл тривоги, передавати його на аналіз CloudComputing платформі, та на основі результатів формувати нові фільтри та правила для забезпечення більш надійного рівня безпеки.

Об’єкт дослідження: процес захисту інформації в хмарних технологіях.

Предмет дослідження: методи та засоби оцінки безпеки хмарних сховищ.

Галузь застосування: розроблена модель системи оцінки безпеки може бути використана як основа для створення практичної реалізації даної системи, що може бути застосована для захисту хмарних сховищ.

РОЗДІЛ 1
АНАЛІЗ ВРАЗЛИВОСТЕЙ ТА РИЗИКІВ, ПОВ'ЯЗАНИХ З ВИКОРИСТАННЯМ ХМАРНИХ ТЕХНОЛОГІЙ

1.1. Аналіз технології Cloud Computing

Історія хмарних технологій почалася дуже давно. Перші ідеї, що опосередковано вплинули на те, що згодом стало хмарними обчисленнями, відносяться до 70-х - 80-х років. Однак датою сучасної комп'ютерної історії був 2006 рік, коли Amazon представила свою інфраструктуру Інтернет-послуг, яка змогла забезпечити користувача не тільки хостингом, але й віддаленими обчислювальними можливостями клієнта. Новинку прийняли такі гіганти, як Google, Sun та IBM, і в 2008 році Microsoft заявила про свою зацікавленість у цій галузі [15].

Хмарні технології пропонують масштабовану інфраструктуру та програмне забезпечення без прямого підключення до фізичних машин, заощаджуючи при цьому витрати, енергію сервера та простої.

Хмарні технології - це здатність декількох фізичних серверів бути одним обчислювальним середовищем. Як правило, хмарні обчислення - це програми, доступ до яких здійснюється через Інтернет через браузер або інші мережеві програми, такі як FTP-клієнт.

Основна відмінність від звичайного методу роботи з програмним забезпеченням полягає в тому, що користувач не використовує ресурси свого комп'ютера або сервера своєї локальної мережі і потужність, що надається йому як послуга Інтернету. Користувач має повний доступ до власних даних та можливість працювати з ними з будь-якої точки світу та з будь-якого пристрою, але він не турбується управлінням операційною системою, програмним забезпеченням, обчислювальною потужністю, з якою відбувається ця робота.

Зберігання в хмарі не тільки даних, але і додатків змінює обчислювальну парадигму на традиційну модель клієнт-сервер, в якій веб-сайт користувача підтримує мінімально необхідну функціональність. Тож необхідність

інсталювати необхідні оновлення програмного забезпечення, перевірку вірусів та інші заходи технічного обслуговування покладається на постачальника хмарних послуг. Це також означає, що обмін даними, редагування версій та редагування стають набагато простішими, ніж коли програми та дані розміщуються на власних ПК.

Загалом, перші згадки про «хмарні технології» можна знайти у 1990-х. Активне використання цього терміна починається приблизно в 2006 році [12]. Точно визначити дату важко - вчені мають різні погляди на це. Л. Черняк зазначає, що Ерік Шмідт вперше у своєму виступі використав термін "хмара" та спробував дати описове визначення [12].

Ніколас Карр дещо розширив термін, зробивши аналогію головним чином між хмарною технологією та електричними мережами. Ідея настільки сподобалася вченим, що хмарну технологію порівняли з п'ятим інструментом [12].

Як визначено NIST, хмарні обчислення - це модель зручного мережевого доступу до загального фонду обчислювальних ресурсів (наприклад, мереж, серверів, файлів даних, програмного забезпечення та послуг), яку можна швидко забезпечити з мінімальними зусиллями управління та взаємодії з постачальником [46]

Хмарні обчислення - це розподілена обчислювальна технологія, при якій ресурси та потужність комп’ютера стають доступними користувачеві як веб-служба, тобто робоча станція на віддаленому сервері. Наприклад, якщо ви використовуєте електронну пошту на веб-сайті служби (наприклад, gmail), який дозволяє вам використовувати цю електронну пошту або обробку зображень вашого браузера через Picasa, ви використовуєте хмарну службу.

Хмарні сервіси - послуги, що надають користувачеві доступ до мережі до масштабованого та гнучко організованого пулу розподілених фізичних або віртуальних ресурсів, що надаються в режимі самообслуговування та адміністрування на його запит (наприклад, програмне забезпечення, дисковий простір, обчислювальна потужність тощо) [43].

Терміни "хмарна технологія" / "хмарний сервіс" із їхнім загальноприйнятим графічним поданням у формі "хмар" лише заплутують користувачів, адже їх структуру можна легко зрозуміти, якщо уявити її такою пірамідою.

Основою піраміди - «інфраструктура» - є набір фізичних пристроїв (сервери, жорсткі диски тощо), вона побудована на «платформі» - наборі послуг, а зверху - програмному забезпеченні, що надається на запит.

Хмарні обчислення - це певний основний вектор, отриманий в результаті синтезу ряду технологій та підходів.

Хмарні технології - це набір інструментів, які виконують обчислення за допомогою віддалених серверів та програм без безпосереднього залучення комп'ютерних ресурсів користувача. Не виключено, що в майбутньому комп’ютери матимуть лише один мікропроцесорний екран, і всі обчислення та потужність будуть розташовані та виконуватися віддалено на хмарних серверах.

Забезпечення декількох рівнів безпеки в "хмарі" забезпечує надійність інформації, що зберігається в службі. Технології цього типу популярні не тільки серед компаній, але також використовуються для державних цілей (Сінгапур, Італія, Молдова, Бельгія, Австрія, Словенія та Португалія). Україна також розглянула можливість використання "хмарних" послуг на державному рівні та представила проект закону "Про внесення змін до деяких законів України (про обробку інформації в хмарних обчислювальних системах)", який передбачає використання "хмар" при обробці публічної інформації, що належить державним інформаційним ресурсам , конфіденційна інформація та секретна інформація, що не становлять державної таємниці [20]. Тому важливим аспектом є аналіз можливостей та ресурсів існуючих "хмарних" служб з точки зору їхньої безпеки.

Відповідно до NIST 800-146 [46] CloudComputing - це модель, яка забезпечує легкий доступ до власної мережі до спільного пулу конфігурованих обчислювальних ресурсів (наприклад, мереж, серверів, сховищ, програм та послуг), які можуть надаватися та забезпечуватися з мінімальними витратами на

управління або зв’язуватися з постачальником послуг . Згідно з документом, основними особливостями "хмари" є:

- Самообслуговування на замовлення;
- Широкий мережевий канал;
- Підтримка ресурсів;
- Швидка масштабованість (гнучкість);
- Вимірюваність використання послуг.

Існує чотири моделі розгортання, а саме: приватна, публічна, суспільна та гібридна [8].

Приватна хмара - це інфраструктура, призначена для використання однією організацією, яка залучає декількох споживачів (наприклад, підрозділи однієї організації), а також, можливо, клієнтів та підрядників організації. Приватна хмара може належати, експлуатуватися і експлуатуватися самою організацією або третьою стороною (або будь-якою їх комбінацією) і може фізично існувати як усередині юрисдикції власника, так і поза нею.

Публічна хмара - це інфраструктура, якою може користуватися широка громадськість. Публічна хмара може належати, управляти та управляти комерційними, науковими та урядовими організаціями (або будь-якою їх комбінацією). Публічна хмара фізично існує в юрисдикції власника постачальника послуг.

Гібридна хмара - це поєднання двох або більше різних хмарних інфраструктур (приватної, публічної чи загальнодоступної), які залишаються унікальними об’єктами, але пов’язані за допомогою стандартних або приватних даних та прикладних технологій (наприклад, короткочасне використання). загальнодоступні хмарні ресурси для збалансування навантаження між хмарами).

Суспільна хмара - це тип інфраструктури, призначений для використання певною спільнотою споживачів з організацій, що мають спільні завдання (наприклад, місії, вимоги до безпеки, політики та відповідність різним вимогам). Публічна хмара може бути спільною власністю, управляти та експлуатувати одна

або кілька громадських організацій або третя сторона (або будь-яка їх комбінація) і може фізично існувати як в межах юрисдикції власника, так і поза нею.

Крім того, важливим кроком у впровадженні «хмарного середовища» є вибір архітектури. Деякі архітектури хмарних служб можуть бути виділені:

1. Інфраструктура як послуга (IaaS) - надання обчислювальних ресурсів на вимогу, на яких замовник може впроваджувати та запускати будь-яке програмне забезпечення, включаючи операційні системи та додатки. Як частина цієї архітектури, клієнт не керує базовою фізичною інфраструктурою та не контролює її, але має контроль над операційними системами та розгорнутими програмами [41, 62].

2. Платформа як послуга (PaaS) - надання хмарної платформи для реалізації програмного забезпечення на основі мов програмування та інструментів, що підтримуються хмарним провайдером. Клієнт не може керувати хмарною інфраструктурою (мережеве та серверне обладнання, операційні системи), але контролює реалізовані програми та, можливо, налаштування середовища [41, 63].

3. Програмне забезпечення як послуга (SaaS) - надання клієнту додатків, реалізованих у хмарній інфраструктурі провайдера. До програми можна отримати доступ з різних клієнтських пристроїв за допомогою "тонкого" клієнта, термінального клієнта або браузера. Клієнт не контролює робочі параметри та налаштування програми. Все зроблено під ключ [41, 61, 40].

Існує декілька видів діяльності пов’язаних з CloudComputing: *Постачальники Послуг* (ПП) роблять свої послуги доступними через Інтернет для *Користувачів Послуг* (КП). Метою CloudComputing є аутсорсинг забезпечення обчислювальної інфраструктури, яка необхідна для розміщення цих послуг. Інфраструктура пропонується «як сервіс» *Постачальниками Інфраструктури* (ПІ), переносячи тим самим інфраструктуру від ПП до ПІ, в результаті ПП можуть отримати гнучкість своїх рішень та зменшити видатки на їх обслуговування (див. рис. 1.1).

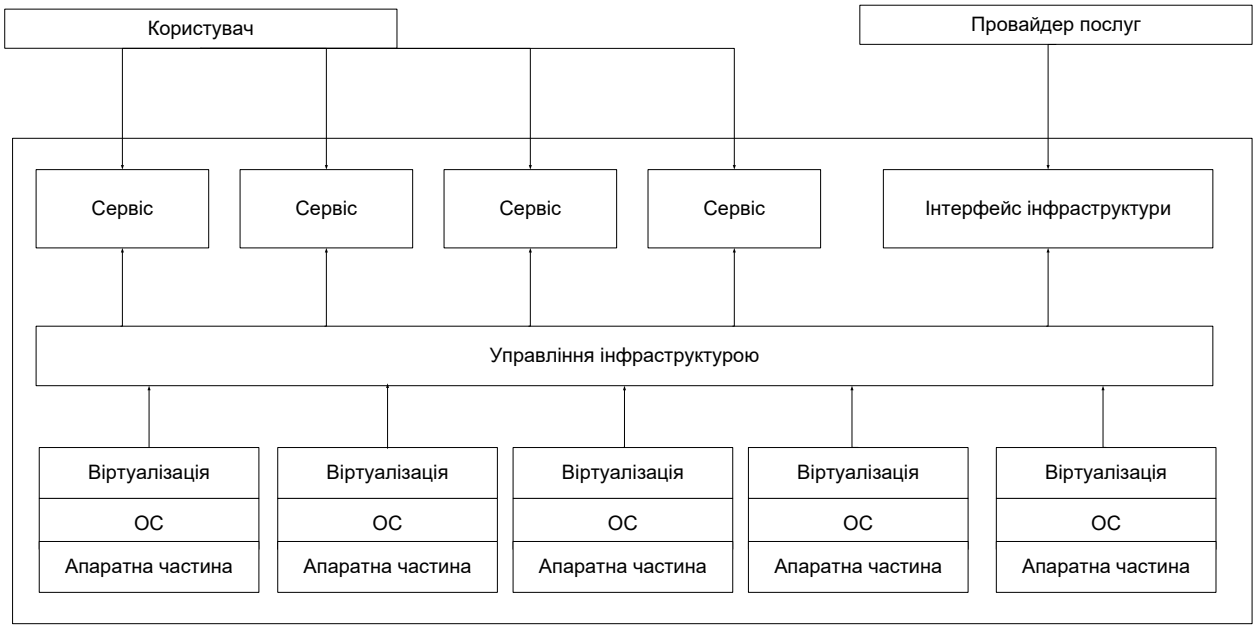


Рис.1.1. Ролі в Cloud Computing

1.2. Захист мереж за допомогою Cloud Computing

Розглядаючи аспекти використання CloudComputing для захисту локальних мереж та обробленої в них інформації, особлива увага приділяється досягненням Panda Security. Ця компанія однією з перших використала потенціал концепції CloudComputing для забезпечення інформаційної безпеки. Але основні досягнення компанії були досягнуті нещодавно, і це чітко підтверджує Virtual GateDefender Performa - пристрій захисту периметра віртуальної мережі, який забезпечує захист CloudComputing для віртуальних середовищ VMWare. Завдяки цій технології можна захистити зону за допомогою віртуальних серверів, вже встановлених у більшості мереж, щоб отримати найбільш ефективний захист від усіх типів IT-загроз. Крім того, це рішення зменшує фінансові витрати порівняно з традиційними рішеннями безпеки, що поширюються з відповідним обладнанням.

Panda Virtual GateDefender Performa пропонує високоефективний захист від усіх типів Інтернет-загроз та спаму. Рішення блокує неробочу або потенційно небезпечну мережеву діяльність за допомогою хмарної гібридної платформи

CloudComputing, а також збільшує можливості захисту та значно зменшує використання ресурсів.

Гібридна платформа хмарних обчислень, що використовується в новому віртуальному пристрої, поєднує приватну платформу хмарних обчислень, розміщену в корпоративній мережі, і забезпечує взаємодію та співпрацю між Panda GateDefender Performa та публічною платформою CloudComputing Cloud.

Таким чином, користувачі в повній мірі користуються перевагами CloudComputing завдяки постійному з'єднанню в реальному часі з колективним розумом - системою, яка автоматично виявляє, аналізує та класифікує 99,4% із 73 000 нових зразків шкідливого програмного забезпечення, які PandaLabs отримує щодня.

Використання гібридної платформи - це суттєвий крок вперед щодо безпеки периметральної мережі, оскільки вона надає компанії повний контроль над своїми даними, включаючи файли журналів та журнали, які набагато менш захищені без використання цієї функції. Це означає, що дані не надсилаються CloudComputing для аналізу, за винятком деяких випадків з високим рівнем шифрування.

Як бачимо, CloudComputing - це дуже ефективний інструмент захисту комп'ютерних мереж. Дослідження та використання CloudComputing для захисту користувачів від спам-атак також відомі, оскільки платформа CloudComputing має необмежену кількість ресурсів, це ефективний інструмент для аналізу електронної пошти та її фільтрації з небажаних спам-повідомлень, заощаджуючи при цьому матеріальні ресурси, забезпечуючи високу швидкість обробки та відокремлюючи від інфраструктури підприємство, що дозволяє не впливати на роботу інших послуг.

Однак використання CloudComputing як ключового елемента безпеки локальної мережі заважає той факт, що концепція має деякі вразливі місця та ризики, пов'язані з її використанням.

1.3. Дослідження проблем безпеки в середовищі Cloud Computing

Забезпечення безпеки в середовищі CloudComputing є дуже складним завданням. На додаток до звичайних труднощів із забезпеченням безпеки IT-систем, концепція CloudComputing несе додатковий рівень ризику, оскільки в багатьох випадках критичні послуги надаються зовнішньою організацією на замовленні. "Зовнішній" аспект аутсорсингу ускладнює підтримку цілісності та конфіденційності даних, забезпечуючи доступність даних та готовність до обслуговування, а також демонструючи відповідність нормативним вимогам. Насправді концепція CloudComputing передає більшу частину контролю над даними та транзакціями від організації замовника своєму постачальнику послуг CloudComputing, подібно до того, як організація передає деякі свої IT-операції аутсорсинговій компанії. Навіть основні завдання, такі як встановлення пакетів виправлень та налаштування між брандмауерами, можуть нести відповідальність постачальник послуг CloudComputing, а не кінцевий користувач.

Все це означає, що клієнти повинні узгодити довіру зі своїми постачальниками та оцінити ризики, пов'язані з впровадженням, розгортанням та розгортанням безпеки від їх імені. Такі відносини між постачальниками послуг CloudComputing та клієнтами, побудовані за принципом "довіра, але перевірка", мають велике значення. Клієнти залишаються в кінцевому підсумку відповідальними за відповідність нормативним актам та захист своїх критичних даних, навіть якщо відповідне навантаження було перенесене в середовище CloudComputing.

Дійсно, деякі організації віддають перевагу приватним або гібридним моделям загальнодоступним середовищам CloudComputing через ризики, пов'язані з аутсорсинговими послугами. Інші аспекти CloudComputing також вимагають значної переоцінки безпеки та ризиків. В середовищі CloudComputing важко знайти фізичне сховище. Процеси безпеки, які колись були видимими, тепер ховаються за рівнями абстракції. Ця відсутність прозорості може спричинити багато проблем із безпекою та дотриманням вимог.

Широкомасштабне колективне використання інфраструктури CloudComputing спричиняє значні відмінності між безпекою середовища

CloudComputing та безпекою більш традиційних IT-середовищ. У той же час балансування навантаження, зміни угод про рівень обслуговування (SLA) та інші аспекти сучасних динамічних IT-середовищ залишають ще більше місця для неправильної конфігурації, пошкодження даних та інших збоїв. Спільна інфраструктура вимагає високого ступеня стандартизації та автоматизації процесів, що допомагає підвищити рівень безпеки, виключаючи можливість помилок. Однак ризики, пов'язані з використанням спільної інфраструктури у великих масштабах, означають, що такі аспекти, як ізоляція, облікові дані та відповідність, повинні залишатися важливими в моделях CloudComputing.

Різні моделі середовища CloudComputing пропонують різні способи забезпечити користувача ресурсами своєї інфраструктури. Ці відмінності впливають на ступінь безпосереднього контролю над органами управління IT-інфраструктурою та розподілену відповідальність за забезпечення її безпеки.

У моделі SaaS велика частина відповідальності за управління безпекою покладається на постачальника послуг CloudComputing. SaaS пропонує декілька способів контролю доступу до веб-порталу, таких як керування обліковими записами користувачів, налаштування його на рівні програми та обмеження доступу з певних діапазонів IP або певних розташувань.

Модель середовища CloudComputing, така як Platform as a Service (PaaS), дозволяє клієнтам брати на себе більшу відповідальність за управління конфігурацією та безпеку. Сюди входять такі компоненти, як програмне забезпечення для підшивки, програмне забезпечення для баз даних та час роботи програми.

Модель Інфраструктура як послуга (IaaS) передає ще більший контроль та відповідальність за безпеку від постачальника послуг CloudComputing замовнику. У цій моделі клієнт отримує доступ до операційної системи, яка підтримує віртуальні зображення, мережеві інструменти та системи зберігання. Описані вище моделі CloudComputing представляють великий інтерес для організацій через їх гнучкість та економічну ефективність, але безпека цих моделей викликає велике занепокоєння.

Недавні дослідження галузевих аналітиків щодо поширення технології CloudComputing, а також статті в пресі підтвердили вищезазначені проблеми. Респонденти відзначають відсутність прозорості та контролю, а також стурбованість щодо захисту критичної інформації та надійності збереження офіційних даних у спільному середовищі, яке знаходиться під зовнішнім контролем. Вважається, що масове впровадження зовнішніх, широко використовуваних колективних і повністю відкритих платформ CloudComputing для критично важливих для бізнесу IT-послуг займе лише кілька років.

У короткостроковій перспективі більшість організацій вивчають способи використання сторонніх постачальників послуг CloudComputing. Ці середовища CloudComputing в основному будуть використовуватися для робочих навантажень з низьким рівнем ризику. Для безпеки таких вантажів може бути достатнім низький рівень універсальних гарантій. Для них головним визначальним фактором є ціна. Гібридні середовища CloudComputing забезпечують набагато більший ступінь гарантованого контролю. Такі робочі навантаження будуть передані у зовнішні середовища CloudComputing лише після того, як ці середовища почнуть пропонувати більш потужну та гнучку систему захисту.

Політика IBM досить цікава у цій галузі, наприклад, концепція IBM Security Framework була розроблена для опису безпеки ключових бізнес-активів та розгляду різних категорій ресурсів з точки зору бізнесу. Нижче наведено перелік ключових вимог безпеки для сучасного середовища CloudComputing корпоративного класу. Ці вимоги базуються на концепції безпеки IBM.

Організації повинні стежити за станом безпеки свого середовища CloudComputing. Ця вимога включає наскрізний контроль змін, управління зображеннями та інцидентами, а також доступ до звітів про інциденти для компаній-користувачів, а також до реєстрації та аудиту даних журналу для обчислювальних платформних служб.

Спостережливість може бути особливо важливою для дотримання нормативних актів. Закон Сарбанеса Окслі, HIPAA (Закон про переносимість та

відповідальність медичного страхування), європейські закони про конфіденційність та багато інших законів вимагають всебічного аудиту. Оскільки загальнодоступні середовища CloudComputing за визначенням є чорною рамкою для передплатників, ймовірно, що потенційні абоненти CloudComputing не зможуть продемонструвати відповідність нормативним актам.

На відміну від них, приватна або гібридна платформа може бути налаштована на відповідність цим вимогам. Крім того, у деяких випадках постачальники зобов'язані проводити аудит третіми сторонами, а не їхніми споживачами. Це додатково підвищує важливість підтримки належної прозорості в середовищі CloudComputing.

Загалом, організації часто повідомляють, що їм потрібні гнучкі SLA, які можуть бути адаптовані до їх конкретних обставин на основі їх досвіду у стратегічному аутсорсингу та традиційних керованих послуг. Організація повинна забезпечити, щоб уповноважені користувачі (власні та інші організації в ланцюзі поставок) мали необхідний доступ до даних та інструментів. Вони повинні бути впевнені, що несанкціонований доступ заблоковано. Зазвичай середовище CloudComputing обслуговує велику та різноманітну спільноту користувачів, що ще більше додає важливості цих механізмів управління.

Крім того, у середовищі CloudComputing з’являється новий рівень привілейованих користувачів: адміністратори постачальника послуг CloudComputing. Це підвищує важливість такої вимоги, як контроль за кожним привілейованим користувачем, включаючи запис його діяльності у відповідний журнал. Ця перевірка повинна включати фізичний моніторинг та попередні перевірки. Такі функції, як реєстрація на борту, повинні бути доступними для координації автентифікації та авторизації з корпоративними або сторонніми серверними системами. Виходячи зі стандартів, для спрощення підключення кінцевих користувачів як до корпоративних, так і до програм CloudComputing потрібен єдиний вхід. Це дозволить їм простий і швидкий доступ до послуг CloudComputing.

Дані та інформація. Більшість організацій вважають захист даних своїм найважливішим завданням безпеки. Поширені проблеми, що пов'язані з цими категоріями: способи зберігання та доступу до даних, необхідність забезпечення відповідності нормативним актам та аудиту, а також втрати, спричинені витоком даних. Усі конфіденційні або керовані дані, включаючи заархівовані дані, повинні бути належним чином розділені в інфраструктурі зберігання CloudComputing. Шифрування даних (і управління ключами шифрування), які передаються в середовище CloudComputing або зберігаються в центрі обробки даних постачальника послуг, є необхідною умовою забезпечення конфіденційності цих даних та відповідності нормативним вимогам. Шифрування мобільних даних та можливість безпечного обміну ключами шифрування між постачальником послуг CloudComputing та споживачем є важливою вимогою, яку часто ігнорують. Оскільки швидка та недорога передача даних через Інтернет все ще не використовується широко, організаціям часто потрібно передати CloudComputing портативному медіа-провайдеру. У цьому випадку дуже важливо, щоб дані були зашифровані, а доступ до ключів шифрування мав лише постачальник CloudComputing та клієнт.

При використанні технології CloudComputing можуть існувати значні обмеження щодо розміщення даних, що визначають місце розташування організації, тип даних, що використовуються в ній, і характер її операцій. Наприклад, кілька держав-членів Європейського Союзу (ЄС) категорично забороняють переміщення будь-яких нерозголошених персональних даних своїх громадян за їх межі. Влада кількох штатів США забороняє нерозголошення персональних даних своїх працівників іншим країнам. Крім того, впровадження середовища CloudComputing може порушити норми, що регулюють "експорт" зашифрованої інформації, а також створити серйозну загрозу інтелектуальній власності.

Перш ніж впроваджувати CloudComputing, юридичний відділ організації повинен ретельно проаналізувати всі вищезазначені вимоги та переконатися, що організація може контролювати географічне розташування своїх даних в

інфраструктурі провайдера. У секторах, де користувачі та дані можуть посилатися на різні чітко визначені класи ризику, такі як надання комунальних та фінансових послуг, відповідні організації повинні підтримувати загальноінфраструктурну класифікацію даних. Така класифікація даних визначатиме категорії людей, які мають доступ до даних, порядок шифрування та архівування даних та технології, що використовуються для запобігання втраті даних.

Усі типові вимоги до програмного забезпечення застосовуються до програм у середовищі CloudComputing. Однак ці вимоги стосуються і зображень, які підтримують ці програми. Постачальник послуг CloudComputing повинен підтримувати і дотримуватися безпеки процесу розробки. Крім того, користувачі довкілля потребують контролю над походженням зображень, а також над їх ліцензуванням та використанням. Операції з видалення та знищення зображень повинні виконуватися регламентовано, і забезпечується нерозголошення конфіденційних даних, що містяться на цих фотографіях. Визначення, перевірка та підтримка рівня захисту зображення відповідно до політики безпеки даного клієнта є дуже важливою вимогою, особливо у високорегульованих галузях. Організація потребує впевненості, що веб-служби, які вона публікує в середовищі CloudComputing, є безпечними, сумісними та відповідають її бізнес-правилам. Ключовою вимогою в цій галузі є використання найкращих стандартних безпечних методів розробки.

Мережі, серверу та кінцевим користувачам спільного середовища повинно гарантуватися, що домени всіх абонентів адекватно ізольовані один від одного і що жодні дані або інформація про транзакції не можуть просочуватися з однієї такої області в іншу. З цією метою клієнти повинні мати можливість налаштовувати довірені віртуальні домени або зони безпеки на основі політик безпеки. Оскільки контроль даних клієнтами стає все більш і більш обмеженим, клієнти очікують, що середовище CloudComputing включатиме такі інструменти, як системи виявлення та запобігання вторгненню. Клієнти побоюються не лише злому їх віртуальних доменів, але й можливості витоку даних, а також так

званого переміщення - зловживання домену клієнта зловмисником для нападу на третю сторону.

Передача даних зовнішньому постачальнику послуг розширює можливості DoS / DDoS-атак, що походять від організації або здійснюються через Інтернет. IT-середовище слід регулярно переглядати з точки зору захисту від загальних загроз та вразливостей. У спільному середовищі домовтеся про відповідальність усіх сторін за регулярний аналіз даних та оцінку безпеки. Організація повинна забезпечити контроль за будь-якими оцінками ризиків або впровадженням управління, які вона не виконує сама в контракті зі своїм постачальником. У випадках, коли постачальник CloudComputing надає каталоги зображень, клієнти хочуть, щоб ці зображення були в безпеці та добре захищені від пошкоджень та зловживань. Багато клієнтів також хочуть, щоб ці зображення мали криптографічну сертифікацію та захист.

Інфраструктура CloudComputing - включаючи сервери, маршрутизатори, пристрої зберігання даних, джерела живлення та інші компоненти, що підтримують роботу, повинна бути фізично захищена. Заходи безпеки у цій галузі включають належний контроль та моніторинг фізичного доступу за допомогою біометричних механізмів контролю доступу та закритих систем відеоспостереження.

1.4. Аналіз вразливостей Cloud Computing

Наступний перелік вразливостей є досить докладним [27], але є й може бути доповнений.

Вразливості автентифікації. Слабкі системи автентифікації, авторизації та обліку можуть сприяти несанкціонованому доступу до ресурсів, привілеям, неможливості відстеження, зловживанню ресурсами та інцидентам загалом тощо.

Фактори, що впливають:

• небезпечне зберігання даних CloudComputing для отримання доступу клієнтів;

• прогалини в дозволах;

• облікові дані зберігаються на тимчасовій машині.

Більше того, використання CloudComputing сприяє поширенню атак автентифікації, оскільки основні програми підприємства доступні в Інтернеті. Тому автентифікації на основі пароля буде недостатньо, і доведеться скористатися дворівневою моделлю доступу до ресурсів хмарних обчислень.

Вразливості служби. Можливі фактори впливу:

• замовник не може контролювати процес доставки;

• особа замовника була неправильно встановлена під час реєстрації;

• затримки часу між компонентами хмарних обчислень;

• виконання асинхронного копіювання ідентифікаційних даних;

• дозволи піддаються захопленню та відтворенню.

Віддалений доступ до інтерфейсу управління. Теоретично це дозволяє комп'ютеру користувача бути вразливим до погіршення безпеки системи в цілому, наприклад, через погану автентифікацію запитів та відповідей.

Уразливості в гіпервізорі. Атаки на шар гіпервізора здаються досить привабливими: насправді гіпервізор повністю контролює розподіл ресурсів між апаратним забезпеченням та віртуальними машинами, тому будь-які прогалини в цьому рівні є критично важливими для системи в цілому. Поразка гіпервізора пошкодить кожну віртуальну машину. Перші докази нападу на гіпервізор були представлені в [28], де автори представили концепцію віртуальних машин руткітів. Поки в найпопулярніших гіпервізорах (наприклад [29] та [30]) не було виявлено декількох уразливостей, вразливості дозволяли атакувати без кореневого доступу.

Відсутність ізоляції ресурсів. Споживання ресурсів одного клієнта може вплинути на ресурси, що використовуються іншим клієнтом. Поняття "інфраструктура як послуга" базується на архітектурі, де фізичні ресурси розподіляються між кількома віртуальними машинами і, отже, мають декількох клієнтів. Уразливості в гіпервізорі можуть призвести до несанкціонованого доступу до розподілених ресурсів. Наприклад, клієнтські віртуальні машини 1 і

2 мають власні віртуальні жорсткі диски, що зберігаються в одному загальному LUN (номер LU) у SAN. Клієнт 2 може перепризначити віртуальний жорсткий диск, щоб він міг переглядати та використовувати дані клієнта 1 у своїй віртуальній машині.

Вразливості інтерфейсу управління. Гіпервізор, що використовується в концепції інфраструктури як послуги в парадигмі CloudComputing, пропонує багатий API, що дозволяє провайдеру розробляти власні інструменти управління, звітування та звітування та інші корисні інтерфейси для своїх клієнтів. Уразливості в моделі безпеки гіпервізора або в "інтерфейсі управління" можуть призвести до несанкціонованого доступу до інформації про клієнта. У той же час уразливість на цьому рівні може дозволити зловмиснику маніпулювати ресурсами об’єкта в хмарних обчисленнях, спричиняючи відмову в обслуговуванні (наприклад, відключення запущених віртуальних машин), витік даних (наприклад, копіювання та передачу за межі віртуальних машин CloudComputing) та порушення даних (наприклад, заміна віртуальної машини модифікованою копією) або прямі фінансові втрати (наприклад, запуск декількох копій віртуальних машин).

Відсутність ізоляції репутації. Дії одного клієнта впливають на репутацію іншого клієнта.

Уразливості шифрування. Ці вразливості включають можливість зчитування даних під час передачі, таких як атаки, слабка автентифікація, обробка самопідписаних сертифікатів тощо.

Неможливість обробки даних в зашифрованому вигляді. Статичне шифрування даних не є складним завданням, але, незважаючи на останні досягнення в галузі гомоморфного шифрування [31], мало шансів, що це шифрування буде підтримуватися під час обробки інформації. Брюс Шнайєр підрахував, що пошук в Інтернеті за допомогою зашифрованих ключових слів є цілком розумним, але використання цього алгоритму збільшить кількість процесорного часу, необхідного приблизно на трильйон. Це означає, що

протягом тривалого часу клієнтам CloudComputing не залишається іншого вибору, як довіряти постачальникам при використанні їхніх сховищ.

Недосконала процедура управління ключами. Інфраструктура CloudComputing вимагає управління та зберігання різних типів ключів; наприклад: ключі сеансу для захисту даних під час передачі (наприклад, ключі SSL), ключі шифрування файлів, пари ідентифікації ключів провайдера CloudComputing, пари ключів ідентифікації клієнта, маркер та сертифікати [33]. Оскільки віртуальні машини не мають фіксованої апаратної інфраструктури, а вміст хмарних обчислень зазвичай географічно розподілений, це ускладнює використання стандартних елементів керування, таких як апаратний модуль безпеки (АМБ), для генерації ключів. Приклад:

• АМБ повинен бути фізично захищений (від крадіжки). Тому стає дуже важко використовувати такий модуль у розподілених системах. Ключові стандарти управління, такі як PKCS # 10, PKCS # 11, не передбачають жодної можливості їх використання в розподілених системах.

• Інтерфейс управління ключами, доступний через Інтернет, є більш вразливим до атак, ніж ті, що використовуються локально, оскільки рівень безпеки знижується при використанні каналу зв'язку.

Проблеми генерації ключів. Поєднання стандартних системних зображень, технологій віртуалізації та відсутність пристроїв введення означає, що віртуалізована система має набагато нижчий рівень ентропії, ніж фізична система [34]. Це означає, що зловмисник на одній віртуальній машині може забрати ключі шифрування, створені на інших віртуальних машинах, оскільки джерела ентропії, що використовуються для генерації випадкових чисел, можуть бути подібними. Цю проблему не важко вирішити, але якщо її не врахувати при проектуванні системи, це може мати серйозні наслідки.

Неточне моделювання використання ресурсів. CloudComputing особливо схильний до виснаження ресурсів. Хоча багато постачальників дозволяють клієнтам заздалегідь резервувати ресурси, алгоритм розподілу ресурсів може не працювати належним чином у таких ситуаціях:

• неточності в моделюванні навантаження та процесі доставки ресурсів можуть призвести до несправності системи;

• збої в ресурсах безпеки.

Можливість зондування внутрішньої мережі хмарних обчислень. Клієнти CloudComputing можуть виконувати сканування портів та інші тести, спрямовані на інших клієнтів у внутрішній мережі.

1.5. Ризики використання Cloud Computing

Проблеми з ізоляцією. Характерною особливістю хмарних обчислень є велика кількість користувачів та розподіл ресурсів. Обчислювальна потужність, сховище та мережеві ресурси спільно використовуються багатьма користувачами. Цей клас ризику включає:

• несправність механізмів розподілу ємності;

• збій механізмів розподілу пам'яті;

• несправність механізмів маршрутизації.

Ймовірність цього інциденту залежить від моделі хмарних обчислень, ризик менший для приватних платформ CloudComputing та вищий для загальнодоступних (публічних) платформ. Наслідки можуть включати втрату цінних або конфіденційних даних або пошкодження репутації клієнтів.

Порушення інтерфейсу управління. Клієнтський інтерфейс управління громадською платформою CloudComputing доступний через Інтернет і забезпечує доступ до великої кількості ресурсів. Тому слід бути дуже обережним, щоб захистити інтерфейс. Звичайно, цей ризик впливає на постачальників більше, ніж на споживачів послуг.

Збір даних під час передачі. Хмарні обчислення завдяки своїй розподіленій архітектурі забезпечують більшу передачу даних, ніж традиційні інфраструктури. Наприклад, вам потрібно обмінятися даними для синхронізації різних віртуальних машин, переміщення віртуальних машин між різними апаратними середовищами, здійснення зв'язку між різними компонентами системи та іншими платформами хмарних обчислень.

Витік даних під час завантаження та завантаження. Цей ризик схожий на попередній, але стосується моменту передачі інформації між клієнтом та платформою хмарних обчислень.

Небезпечне або неефективне видалення даних. Коли клієнт змінює постачальника послуг, масштабує систему або розповсюджує обладнання, дані можуть залишатися доступними в різних місцях. Заповнення запиту на видалення даних не означає, що його більше не можна читати, єдиним способом безпечного видалення даних є фізичне знищення носія. Однак ця опція може бути не прийнятою більшістю постачальників послуг, оскільки ці носії можуть зберігати дані від інших активних користувачів. При використанні шифрування даних ризик цієї загрози значно зменшується.

Економічна відмова в обслуговуванні (EDOS). Є кілька сценаріїв, коли ресурси клієнта можуть використовуватися іншими сторонами з економічними наслідками:

• викрадення даних: зловмисник використовує обліковий запис і використовує ресурси клієнта для власної вигоди або для заподіяння економічної шкоди клієнту;

• клієнт не визначив ліміт використання платних ресурсів та максимальне завантаження цих ресурсів;

• зловмисник використовує відкритий доступ до клієнтських ресурсів - наприклад, коли клієнт оплачує HTTP-запит, велика кількість запитів (DDoS) може спричинити економічні втрати.

Втрата ключів шифрування. Це передбачає розкриття секретних ключів (SSL, шифрування файлів тощо), втрату або пошкодження цих ключів або їх несанкціоноване використання, що може призвести до несанкціонованого доступу.

Системи сканування. Сканування портів та систем, картографування мережі, хоч і не прямі, але загрози. Ці інструменти можна використовувати для збору інформації, корисної для атаки на систему.

Порушення ядра. Кожна архітектура CloudComputing базується на вузькоспеціалізованій платформі, програмному рівні (базовій системі), який знаходиться над апаратним забезпеченням і відповідає за управління ресурсами на різних рівнях абстракції. Наприклад, для платформ хмарних обчислень IaaS це може бути гіпервізор. Як і будь-яке інше програмне середовище, системне ядро може бути вразливим до різних вразливостей безпеки, які можуть бути використані зловмисником.

Ризик зміни юрисдикції. Дані клієнтів можуть належати до різних юрисдикцій, деякі з яких можуть бути під загрозою. Якщо центри обробки даних розташовані в країнах із високим ризиком, наприклад, за відсутності верховенства права та непередбачуваності правових рамок та органів влади, у країнах, які не дотримуються міжнародних угод, дані користувачів можуть бути втрачені або скомпрометовані.

Ризик захисту даних. CloudComputing створює ряд загроз для захисту даних. Користувачеві CloudComputing може бути важко контролювати, як обробляється його інформація. В будь-якому випадку лише користувач буде нести відповідальність за обробку його даних, навіть якщо всі процеси відбуваються у віддаленій системі. Ця проблема ще більше посилюється взаємодією між різними платформами CloudComputing. З іншого боку, деякі постачальники CloudComputing надають інформацію про обробку даних, яку вони виконують, деякі з яких мають сертифікат SAS70.

Ризик ліцензування. Умови ліцензування, такі як ліцензування робочого місця та перевірка ліцензії в Інтернеті, можуть перестати працювати в середовищі хмарних обчислень.

Слід зазначити, що перелік згаданих ризиків та вразливостей не є вичерпним і може бути доповнений залежно від особливостей використання платформи, але наведені основні моменти, характерні для більшості випадків.

Подальша робота дозволить запропонувати аналітичну систему оцінки безпеки хмарних сховищ на основі виявлення атак, яка може ефективно аналізувати велику кількість попереджень, оптимізувати результати та

представляти результати користувачеві, тоді як аналітична система має архітектуру хмарних обчислень, яка забезпечує високу продуктивність, швидкість аналізу та збої системи.

Щоб система могла працювати на новій платформі розподілених обчислень Hadoop, також відомій як платформа CloudComputing, буде запропонований алгоритм Map / Reduce.

Як видно з вищевикладеного, CloudComputing - це дуже перспективна технологія, яка може використовуватися в різних аспектах IT. Одним із додатків CloudComputing може бути інформаційна безпека. Щоб краще зрозуміти використання CloudComputing в інформаційній безпеці та побудувати модель системи оцінки безпеки на основі CloudComputing, нам потрібно зрозуміти основні типи традиційних СВА, проаналізувати їх переваги та недоліки та зробити висновки про те, як CloudComputing може вдосконалити ці системи.

РОЗДІЛ 2
РОЗРОБКА МОДЕЛІ АНАЛІТИЧНОЇ СИСТЕМИ ОЦІНКИ БЕЗПЕКИ ХМАРНИХ СХОВИЩ

2.1. Дослідження моделей систем виявлення атак

СВА аналізує інформацію про мережеву активність та виявляє зловмисні дії. Методи, що використовуються СВА для виявлення вторгнень, поділяються на два різні типи залежно від критеріїв: виявлення ненормальної поведінки [38,44,45,46] та виявлення шахрайства.

Використовуючи метод виявлення аномалій, система повинна побудувати еталонний профіль на основі історії мережі та порівняти його з профілями в певний час, з яких можна зробити висновки про вторгнення. З іншого боку, метод виявлення шахрайства заснований на принципі аналізу сигнатур атак [50,52], вхідна інформація порівнюється з сигнатурою атаки, і в разі виявлення конвергенції відбувається атака. Більшість СВА працюють шляхом виявлення шахрайства, наприклад Snort [53]. Ця система може використовуватися як система запобігання проникненню. Snort використовує мову керування правилами, яка використовує як метод підпису, так і аномалії. Snort є стандартом СВА і є основним інструментом у багатьох наукових експериментах [36,55].

Системи виявлення вторгнень (СВА) - це програмні чи апаратні системи, які автоматизують процес перегляду подій, що відбуваються в комп’ютерній системі або мережі, аналізуючи їх з точки зору безпеки. Зі збільшенням кількості мережевих атак аналіз витрат і вигод стає необхідним доповненням до інфраструктури безпеки.

Виявлення вторгнень - це процес моніторингу та аналізу подій, що відбуваються в комп’ютерній системі чи мережі. Вторгнення визначаються як спроби порушити конфіденційність, цілісність, доступність або обхід механізмів безпеки комп'ютера або мережі. Проникнення можуть бути спричинені зловмисниками, які отримують доступ до систем іззовні, або авторизованими користувачами систем, які намагаються отримати додаткові дозволи, яких у них

немає. СВА - це програмні чи апаратні пристрої, які автоматизують процес моніторингу та аналізу мережевих або системних подій з метою виявлення вторгнень.

Принцип дії СВА такий: система отримує інформацію про подію (аналіз трафіку, журнал запитів тощо), аналізує дані, події та створює відповідь на основі аналізу - результат може відрізнятися від найпростішого звіту або активного втручання у виявлення вторгнень.

СВА стають важливим доповненням до інфраструктури безпеки будь-якої організації. Технології виявлення вторгнень не забезпечують повну безпеку системи. Однак використання СВА у поєднанні з іншими заходами безпеки робить інфраструктуру безпеки організації досить безпечною та стійкою до атак. Використання СВА допомагає досягти кількох цілей:

• здатність реагувати на атаку;

• можливість розпізнати дію чи подію як атаку, а потім здійснити дію, щоб заблокувати джерело.

Коли зловмисники атакують вашу систему, вони зазвичай виконують попередню роботу. Першим етапом атаки, як правило, є зондування або перевірка системи чи мережі на наявність можливих точок входу. У мережі без СВА зловмисник може ретельно проаналізувати систему з мінімальним ризиком виявлення та покарання. При такому необмеженому доступі зловмисник в кінцевому підсумку може знайти вразливість та використати її для отримання необхідної інформації.

Та ж сама мережа з СВА, що виконує постійний моніторинг мережевої активності, представляє для атакуючого більш важку проблему. Хоча атакуючий і може переглядати мережу на уразливості, СВА виявить сканування, ідентифікує його, як підозріле, може виконати блокування доступу атакуючого до цільової системи і сповістить персонал, який у свою чергу може виконати відповідні дії для блокування доступу атакуючого. Навіть наявність простої реакції на зондування мережі буде означати підвищений рівень ризику для

атакуючого і може перешкоджати його подальшим спробам проникнення в мережу.

Типи СВА

СВА можливо класифікувати за наступними характеристиками:

- За архітектурою:
  - централізовані;
  - розподілені.
- За джерелом отримання інформації:
  - network-based;
  - host-based ;
  - application-based.
- За способом аналізу:
  - виявлення зловживань (misuse detection);
  - виявлення аномалій (anomaly detection).
- За способом обробки:
  - interval-based (або пакетний режим);
  - real-time.
- За способом реакції:
  - активні;
  - пасивні.

Архітектура СВА

Основними архітектурними компонентами СВА є:

- host - система, на якій встановлена СВА;
- target - система, над якою проводиться моніторинг.

Спільне розташування host і target (централізовані)

Перші СВА мали спільне розташування – СВА була встановлена на системі за якою проводився контроль. Даний спосіб взаємодії мав свої недоліки та переваги.

Перевага: Система є автономною, немає необхідності підтримувати зв'язок з віддаленою системою.

Недолік: Виникає загроза компрометації самої СВА, оскільки, виконавши успішну атаку на цільову систему, зловмисник має змогу деактивувати СВА.

Поділ host і target (розподілені)

З розвитком технологій стало можливим роздільне використання СВА та системи над якою виконується моніторинг.

Переваги: Є можливість створити СВА не доступною для зловмисника, тому, атакуючи цільову систему (target), порушник майже не має змоги скомпрометувати СВА.

Недоліки: Якщо СВА є єдиною для великої кількості target систем, виконавши атаку на СВА, всі target системи позбавлені захисту.

Сучасні СВА, як правило, складаються з наступних компонентів:

- сенсор, який відстежує події в мережі або системі;
- аналізатор подій, виявлених сенсорами;
- компонента прийняття рішення.

Способи управління

- Централізоване управління - весь моніторинг, виявлення та звітність управляються безпосередньо з єдиної системи. У цьому випадку існує єдина консоль СВА, яка пов'язана з усіма сенсорами, розташованими в мережі.
- Частково розподілене управління - моніторинг та визначення управляються з локально керованого вузла з ієрархічною звітністю в одну чи більше центральних розташувань.
- Повністю розподілене управління - моніторинг та визначення атаки робляться в точці аналізу.

Спосіб обробки інформації

Швидкість реакції вказує на час, що минув між подіями, які були виявлені монітором, аналізом цих подій і реакцією на них.

- Interval-based - реакція відбувається через певні проміжки часу, інформаційний потік від точок моніторингу до інструментів аналізу не є безперервним.
- Real-Time - обробляють безперервний потік інформації від джерел.

Інформаційні джерела

Network-Based – ці СВА виявляють атаки, перехоплюючи та аналізуючи мережеві пакети. Прослуховуючи мережевий сегмент, мережевий СВА може відображати мережевий трафік від кількох хостів, підключених до сегмента мережі, і таким чином захищати ці хости..

Network-based СВА асто складається з безлічі датчиків, розташованих у різних точках мережі. Ці пристрої сканують мережевий трафік, виконують аналіз локального трафіку та генерують звіти про атаки для центральної консолі управління. Багато з цих датчиків розроблені для роботи у "невидимому" режимі, щоб ускладнити зловмисникові виявлення їх присутності та місцезнаходження..

Переваги:

- Кілька оптимально розташованих network-based СВА можуть переглядати велику мережу.
- Network-based СВА зазвичай є пасивними пристроями, які прослуховують мережевий канал без впливу на нормальне функціонування мережі. Таким чином, звичайно, буває легко модифікувати топологію мережі для розміщення network-based IDS.
- Network-based IDS можуть бути зроблені практично невразливими для атак або навіть абсолютно невидимими для атакуючих.

Недоліки:

- Для network-based СВА може бути важко обробляти всі пакети великої або завантаженої мережі, отже, вони можуть пропустити певні атаки, що починалися при великому об’ємі трафіку.

- Network-based СВА не можуть аналізувати зашифровану інформацію. Ця проблема зростає ще більше, якщо організація (і атакуючі) використовують VPN.
- Більшість network-based СВА не можуть визначити, чи була атака успішної; вони можуть лише визначити, що атака була розпочата. Це означає, що після того, як СВА визначить атаку, адміністратор повинен вручну досліджувати кожен атакований хост для визначення, чи відбувалося реальне проникнення.
- Деякі network-based СВА мають проблеми з визначенням мережевих атак, які включають фрагментовані пакети. Такі фрагментовані пакети можуть призвести до того, що IDS буде функціонувати нестабільно.

Host-Based CBA - для інформації, зібраної на одному комп'ютері. Це вигідне розташування дозволяє заснованому на хості СВА аналізувати дії з високою ймовірністю та точністю, визначаючи лише ті процеси та користувачів, які мають відношення до конкретної атаки операційної системи. Більше того, на відміну від мереж, хости «бачать» наслідки атаки, оскільки вони можуть безпосередньо отримувати доступ до системної інформації, файлів даних та системних процесів, на які спрямована атака.

Host-based СВА зазвичай використовують інформаційні джерела двох типів:

- результати аудиту ОС;
- системні події.

Результати аудиту операційної системи, як правило, генеруються на рівні ядра операційної системи і є більш детальними та краще захищеними, ніж системні події. Однак системні події значно менші та менш численні, ніж результати аудиту, і тому їх легше аналізувати. Деякі host-based СВА ризначені для підтримки централізованої інфраструктури управління та отримання звітів про СВА, які можуть дозволяти одній консолі управління відстежувати кілька хостів. Інші створюють повідомлення у форматі, сумісному із системами управління мережею.

Переваги:

- Можливість визначити атаки, які не можуть бачити network-based системи, оскільки є можливістю стежити за подіями локально.
- Host-based СВА часто можуть функціонувати в оточенні, в якому мережевий трафік зашифрований.
- На функціонування host-based СВА не впливає наявність у мережі комутаторів.
- Коли host-based СВА працюють з результатами аудиту ОС, вони можуть надати допомогу у визначенні троянських програм або інших атак, які порушують цілісність ПЗ.

Недоліки:

- Host-based СВА використовують обчислювальні ресурси хостів, за якими вони спостерігають, що впливає на продуктивність системи.

Application-Based CBA - є підмножиною host-based СВА, які аналізують події, що надійшли до програми. Найбільш загальними джерелами інформації, що використовуються application-based СВА є журнали транзакцій програми.

Можливість безпосередньо взаємодіяти з додатком, з певним доменом або використовувати специфічні для програми знання дозволяє application-based СВА виявляти підозрілу поведінку авторизованих користувачів, які перевищують їх права доступу. Такі проблеми можуть виникати лише під час взаємодії користувача з додатком.

Переваги:

- Application-based СВА можуть аналізувати взаємодію між користувачем та програмою, що часто дозволяє відстежити неавторизовану діяльність конкретного користувача.
- Application-based СВА часто можуть працювати в зашифрованих середовищах, так як вони взаємодіють з додатком у кінцевій точці транзакції, де інформація представлена вже в незашифрованому вигляді.

Недоліки:

- Application-based СВА можуть бути більш вразливі, ніж host-based, оскільки журнали додатки не так добре захищені, як результати аудиту ОС, що використовуються host-based СВА.
- Application-based СВА часто аналізують події на рівні абстракції, на якому зазвичай неможливо визначити «троянські програми» або інші подібні атаки, пов'язані з порушенням цілісності ПЗ.

Способи аналізу інформації

Є два підходи до аналізу подій для визначення атак:

- визначення зловживань (misuse detection);
- визначення аномалій (anomaly detection).

Визначення зловживань – детектори аналізують діяльність системи, аналізуючи подію або набір подій, щоб відповідати заздалегідь визначеним шаблонам, що описують відому атаку. Шаблон відповідності, відомий під час атаки, називається підписом, а визначення зловживання іноді називають "визначенням підпису". Кожен підпис може відповідати певній атаці, але існують також можливі підходи, які використовують один підпис для визначення групи атак.

Переваги:

- Це ефективний метод виявлення атак і водночас має низьку ймовірність помилкових спрацьовувань.
- Цей метод дозволяє швидко і надійно визначити конкретний засіб атаки або технологію. Це може допомогти адміністратору скоригувати заходи безпеки.
- Дозволяють адміністраторам, незалежно від рівня їхньої експертизи щодо безпеки, ініціювати процедури обробки інцидентів.

Недоліки:

- Можна ідентифікувати лише ті атаки, підписи яких є в базі даних, вам слід постійно оновлювати базу даних, щоб отримати підписи нових атак.
- Багато детекторів шахрайства розроблені для використання лише конкретних підписів, що унеможливлює виявлення загальних варіантів атак..

Визначення аномалій - детектори аномалій виявляють незвичну поведінку на хості або в мережі. Атаки змінюють звичайний режим роботи системи, що дозволяє ідентифікувати атаки, аналізуючи систему та порівнюючи її зі стандартом. Метрики та технології, що використовуються для виявлення аномалій, включають:

- Визначення допустимого порогу. У цьому випадку основні атрибути поведінки користувача та системи визначаються кількісно. Для кожного атрибута визначено рівень, який встановлено як дійсний. Такі атрибути поведінки можуть визначати кількість файлів, доступних користувачеві за певний проміжок часу, кількість невдалих спроб входу в систему, кількість процесорного часу, що використовується процесом тощо. Цей рівень може бути статичним або евристичним.
- Статистичні метрики:
  - параметричні, за яких передбачається, що розподіл атрибутів профілю відповідає конкретному зразку;
  - непараметричні, при яких розподіл атрибутів профілю є результат, виходячи з набору значень історії, які спостерігалися за певний період часу.
- Метрики, які ґрунтуються на правилах, аналогічні непараметричним статистичним метрикам в тому, що зразки специфікують як правила, а не як чисельні характеристики.
- Нейронні мережі.
- Генетичні алгоритми.
- Моделі імунних систем.

Найчастіше використовуються перші дві технології. На жаль, детектори аномалій, що базуються на них, часто мають велику кількість повідомлень про помилки, оскільки нормальні моделі поведінки користувачів або системи можуть характеризуватися високим ступенем невизначеності. Незважаючи на ці недоліки, дослідники припускають, що засновані на аномалії СВА здатні

виявляти нові форми атак, на відміну від СВА, заснованих на підписах, які базуються на схемі попередніх атак.

Крім того, деякі форми виявлення аномалій генерують вихідні дані, які в подальшому можуть бути використані як джерело інформації для методу виявлення шахрайства. Наприклад, шукач аномалій на основі значення колеса може генерувати графік, що показує "нормальну" кількість файлів, доступних певному користувачеві; детектор шахрайства може використовувати цей сюжет як частину підпису виявлення, який говорить, що "якщо кількість файлів, доступних для даного користувача, перевищує цю" звичайну "діаграму більш ніж на 10%, слід спрацьовувати попереджувальний сигнал".

Переваги:

- СВА, засновані на визначенні аномалій, виявляють несподівану поведінку, таким чином, мають можливість визначити симптоми атак без знання конкретних деталей атаки.
- Детектори аномалій можуть створювати інформацію, яка надалі буде використовуватися для визначення сигнатур для детекторів зловживань.

Недоліки:

- Підходи визначення аномалій зазвичай створюють велику кількість помилкових сигналів.
- Підходи визначення аномалій часто вимагають певного етапу навчання системи, під час якого визначаються характеристики нормальної поведінки.

Розглянемо використання мережевих систем виявлення атак більш детально.

2.2. Мережеві системи виявлення атак

Програмні системи, які контролюють вміст трафіку, що надходить через мережу, називаються мережевою системою виявлення вторгнень (NIDS). Вони працюють на рівні мережі відповідно до моделі OSI і контролюють встановлені зв’язки, аналізують структуру та вміст мережевих пакетів. Система NIDS аналізує весь трафік як на одному комп'ютері, так і на виділеному сервері (шлюз,

маршрутизатор, зонд). Коли виявляється атака, NIDS має механізм реагування на цей тип загрози. Діапазон його можливостей досить широкий: від надсилання попереджувальних повідомлень на робочу консоль (електронна пошта, телефон), до блокування облікового запису, відключення, переналаштування брандмауера або маршрутизатора.

Механізм контролю та аналізу встановленої статистики з'єднань дозволяє виявити спробу сканування системи або атаку відмови в обслуговуванні (одночасно кілька підключень до відкритих служб). Контроль вмісту трафіку здійснюється шляхом пошуку конкретних послідовностей даних (підписів), що надсилаються в мережевому пакеті. Наприклад, якщо пакет, що містить послідовність даних "81 F1 1 березня 2011 р. 9B 81 F1 01", перехоплюється під час з'єднання, встановленого із сервером Microsoft SQL, робиться спроба операції "переповнення буфера". Цю вразливість було виявлено в службі дозволу Microsoft SQL Server 2000 та Microsoft Desktop Engine (MSDE) 2000.

Системи виявлення вторгнень мають власну базу знань, в якій збираються відомі типи мережевих атак. Вони також дозволяють користувачам розробляти та додавати нові описи мережевих інцидентів.

Потенційно складним моментом у роботі NIDS є надійність визначення IP-адреси зловмисника. Зловмисник може легко імітувати атаку сторонніх хостів. Потім, згідно зі сценарієм, NIDS визначатиме IP-адреси з підроблених зловмисником мережевих пакетів, в результаті чого будуть вживатися превентивні дії, але не проти зловмисника. Для підвищення надійності захисту необхідно контролювати всю архітектуру мережі, розміщуючи датчики в кожному сегменті мережі (див. Рис.2.1).

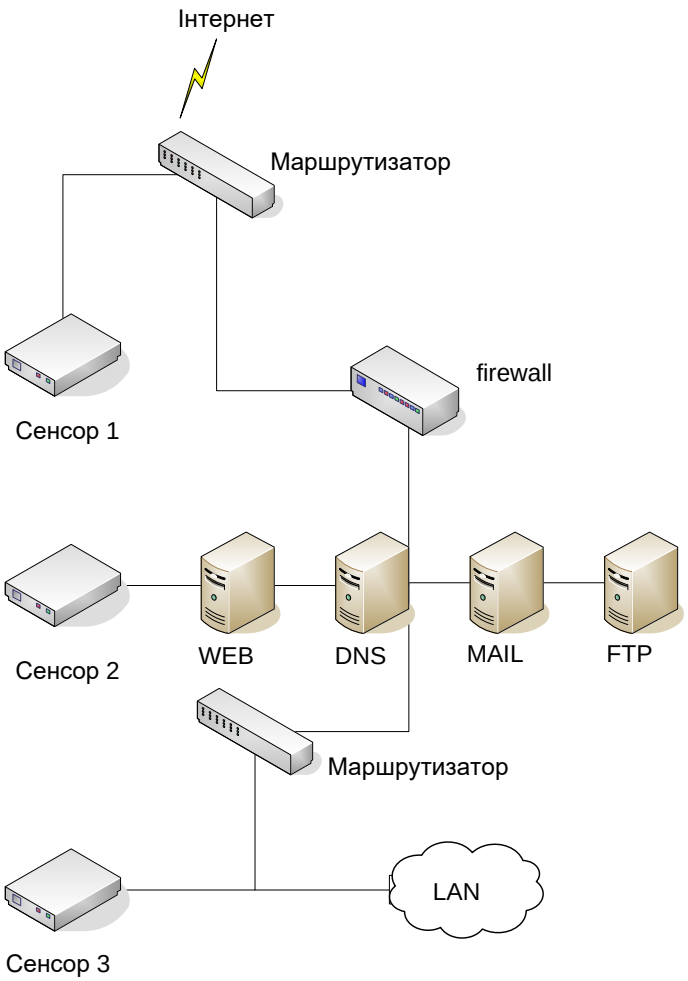


Рис. 2.1. Розміщення сенсорів в мережі

Сенсор 1 знаходиться в зоні максимальної потенційної загрози мережі. Він аналізує весь вхідний та вихідний трафік та високу ймовірність великої кількості помилкових спрацьовувань. При збільшеному навантаженні мережі NIDS може не справлятися з усім потоком трафіку, а ефективність методів аналізу зменшиться, наприклад, за рахунок зменшення кількості перевірених підписів.

Сенсор 2 аналізує трафік на сервері. Тут вхідний трафік фільтрується між екраном мережі. З правильно налаштованим брандмауером це безпечніша область мережі. Через зменшення трафіку зменшується кількість помилкових тривог. Сенсор 2 повинен бути налаштований відповідно до специфіки серверів.

Сенсор 3 аналізує трафік у локальній мережі, теоретично це найбільш безпечна зона. Необхідно звертати увагу на будь-які ненормальні дії в мережі.

Кількість помилкових спрацьовувань у цій області має бути найменшою, тому звертайте більше уваги на повідомлення сенсора 3.

Далі буде проведено порівняння програмних продуктів виявлення вторгнень у мережу.

2.3. Порівняння програмних засобів виявлення вторгнень

Всього включено 5 систем виявлення вторгнень. Для порівняння були розглянуті лише системи з відкритим кодом. Зверніть увагу, що цей список не є вичерпним і включає лише вібіркові системи.. У табл. 2.1. наведена коротка інформація про кожну систему.

Таблиця 2.1

Некомерційні системи виявлення атак

Назва системи	Виробник
AAFID	Purdue University, West Lafayette, IN, USA
ASAX	University of Namur, Belgium
Prelude	Yoann Vandoorselaere Laurent Oudot
SHADOW	Naval Surface Warfare Center, Dahlgren Division
Snort	Martin Roesch

Для порівняння систем використовувались такі критерії:

Рівень моніторингу системи. Вказує, на якому рівні збираються захищені системні дані для виявлення вторгнень. Існують системні та мережеві джерела. Рівні ядра, процесу та програми розділені в системних джерелах. В одному випадку СВА може мати власні модулі моніторингу, які отримують інформацію в режимі реального часу від первинних джерел - каналу даних у мережі, системної шини мережевого вузла, ядра вузла ОС та вбудованих модулів.

В іншому випадку СВА може використовувати лише вторинні джерела інформації - журнали вузлів ОС, програми, мережеві дані, зібрані іншим способом (на інших вузлах). У цьому випадку ймовірність отримання

спотворених даних вища і навмисно нижча за швидкість виявлення можливої атаки.

Використовуваний метод виявлення. Метод виявлення також є ключовим критерієм порівняння. Методи поділяються на формальні та евристичні. Формальні методи базуються на механізмах визначення поточного стану системних ресурсів та оцінки ризиків, пов’язаних із цими станами. У цих методах міра "безпеки" повинна бути визначена для кожного стану одного ресурсу. Існуючі системи використовують евристичні методи виявлення, які виявляють і класифікують атаки на основі неповної інформації та з певною ймовірністю неправильного виявлення та класифікації.

Пристосованість до невідомих атак. Визначає, чи застосовуваний метод виявляє раніше невідомі атаки. Експертні методи вимагають постійного оновлення бази знань про атаки та описи нових атак, здійснених експертами. Методи виявлення ненормальної поведінки об'єкта дозволяють виявити невідомі атаки.

Управління. Вказує організацію управління системою. Управління може бути централізованим, розподіленим. Крім того, може бути можливим дистанційне управління СВА. У випадку повністю розподіленого управління всіма компонентами СВА слід керувати окремо. Завдяки повністю централізованому управлінню, усіма компонентами СВА можна керувати з одного вузла.

Архітектура. Визначає компонування архітектури СВА. Архітектура може бути нерозподіленою або розподіленою. Ця риса визначає швидкість реакції СВА на атаку. Завдяки нерозподіленій архітектурі СВА може збирати та аналізувати інформацію про поведінку об’єктів у обмеженому сегменті системи. Таким чином, розподілена архітектура СВА дозволяє розширити горизонт моніторингу системи, який визначає час оцінки безпеки системи та виявлення атак.

Формування відповідної реакції на атаку. Визначає наявність у системі вбудованих механізмів відповідної реакції на атаку, крім факту її реєстрації. Можливі наступні методи реагування:

- розрив з'єднання з атакуючим об'єктом;
- блокування його на між мережевому екрані;
- відстеження шляху проникнення атакуючого об'єкта в мережу.

Захищеність. ін визначає ступінь захисту СВА від атак на його компоненти, включаючи захист переданої інформації, стійкість до часткової несправності компонентів або їх злому..

Постають питання питання щодо наявності прогалин у компонентах СВА, безпеки каналів передачі даних між ними та авторизації компонентів у СВА.

Нижче в табл. 2.2 наведені результати порівняння розглянутих систем за обраними критеріями.

Таблиця 2.2

Результати порівняння розглянутих систем за обраними критеріями

	AAFID	**ASAX**	**Prelude**	**SHADOW**	**Snort**
Рівень спостереження за системою	системний	системний	системний, мережевий	мережевий	мережевий
Метод виявлення	експертний	експертний	експертний	експертний	експертний, статистичний аналіз
Адаптивність	-	-	-	-	+
Керування	централізоване	централізоване	централізоване	розподілене	централізоване
Архітектура	розподілена	Не розподілена	розподілена	розподілена	Не розподілена
Реакція	-	-	+	-	+
Захист	-	-	SSL, Libsafe	SSH	-

AAFID. Система базується на автономних агентах виявлення вторгнень. Найцікавіше в системі - це її архітектура. Ця система базується на роботах Кросбі та Спаффорда, які запропонували використовувати автономні агенти, що

працюють на основі генетичних алгоритмів та адаптуються до поведінки користувачів. Ідея використання генетичних алгоритмів не реалізована, але архітектурні ідеї реалізовані в системі AAFID. Основними елементами системи є: агенти, фільтри, трансивери, монітори.

Система AAFID повністю розподілена. Будь-який вузол мережі може розмістити будь-яку кількість агентів, які спостерігають за подіями, що цікавлять цей вузол. Агент - Автономний програмний компонент системи виявлення атак, робота якого залежить лише від операційної системи, в якій вона працює. Це означає, що ефективність агента не залежить від інших компонентів СВА. Всі агенти на одному вузлі передають зібрану інформацію на один трансивер. Трансивер є частиною системи виявлення, яка контролює запуск і зупинку агентів на конкретному вузлі.

Кожен вузол з агентами має один приймач. Більше того, трансивери можуть виконати деяке зменшення інформації, отриманої від агентів, у більш загальному вигляді. Трансивери ведуть отриману інформацію до одного або декількох моніторів. Кожен монітор відстежує та взаємодіє з кількома трансиверами. Монітори працюють на рівні системи в цілому, тому вони можуть аналізувати отриману інформацію з урахуванням співвідношення подій у різних сферах системи.

ASAX. Система ASAX була розроблена в Університеті Намура, Бельгія. Система призначена для аналізу журналів UNIX.

Завдання побудови системи було розділено на дві основні задачі: розробка уніфікованого та гнучкого формату запису журналу, який можна було б правильно перетворити на оригінальні журнали операційної системи, та розробка ефективної, ефективної та зручної мови для опису та ідентифікації складних структур шаблонів у записах журналів. Перше завдання було вирішене Siemens-Nixdorf AG, друге завдання Університетом Намура в Бельгії. В результаті дослідницької роботи мова RUSSEL була розроблена як модель для аналізу потоків даних загалом та для аналізу журналів безпеки зокрема. Представлений модуль складається з правил виявлення несанкціонованого

доступу до системи на основі порогового завдання для кількості невдалих спроб входу. Для підвищення продуктивності використовуються спеціальні модулі, які надсилають запис до журналу аналізатора. Завдяки цьому одна копія запису проходить усі модулі правил, і можна використовувати систему для обробки даних журналу в режимі реального часу.

SHADOW. Система SHADOW була розроблена в Центрі військово-морської війни. Система спочатку була розроблена в технології з відкритим кодом. SHADOW працює на Linux і складається з двох типів компонентів: датчика мережевого трафіку та аналізатора. Вони можуть бути на одному вузлі або на різних вузлах. Датчик базується на бібліотеці libpcap та утиліті tcpdump, які широко доступні на різних платформах. Завдання датчика - збирати та архівувати мережевий трафік відповідно до типів протоколів (встановлюється адміністративно під час конфігурації датчика). Аналізатор збирає дані з датчиків і застосовує до них спеціалізовані фільтри, що містять підписи відомих атак та підписи, встановлені користувачем. Датчики та аналізатори повинні розташовуватися у виділених вузлах.

Prelude - це система з відкритим кодом. Система була розроблена з самого початку як гібридна СВА, яка може допомогти адміністратору мережі контролювати активність як на рівні мережі, так і на вузлах. Система розподілена і складається з наступних елементів:

- мережеві датчики - різні датчики, які аналізують дані на рівні мережі на основі досвіду. Датчики генерують повідомлення про виявлення аномалій і надсилають їх в модуль управління. Prelude має власну базу даних підписів атак, а також може використовувати підписи формату Snort, що дозволяє ефективно оновлювати базу даних підписів атак;
- системні датчики - різні датчики рівня системи, які аналізують журнали реєстрації операційної системи, програми. Датчики генерують повідомлення про виявлення аномалій і надсилають їх в модулі управління.

Модулі управління - процеси, які отримують і обробляють повідомлення сенсорів. Розрізняються такі види модулів управління:

• Модулі документації - відповідає за реєстрацію повідомлень в журналах реєстрації або базах даних MySQL, PostgreSQL;

• Модулі реагування - аналізує повідомлення і генерує можливу відповідь, реакцію системи на атаку.

Snort. Система Snort - це класичний продукт з відкритим кодом. Розробку розпочав один автор, але завдяки відкритій архітектурі та коду, система почала досить швидко розвиватися за допомогою інших розробників, і, крім того, вона інтегрувалася з іншими програмними продуктами, такими як бази даних зберігання журналів виявлення, аналізатори журналів. Це, перш за все, означає, що система виявляє атаки виключно на основі аналізу мережевого трафіку. Основним методом виявлення атак, що використовується в системі, є виявлення шахрайства на основі опису підписів атак. Система використовує просту мову для опису підписів атак, яка повністю описана в документації і дозволяє системним адміністраторам доповнювати базу даних підписів своїми підписами. Кожне правило цієї мови складається з двох частин: умови застосування та умови дії.

Оскільки ця система стала досить популярною і вважається еталоном для СВА, ми зупинимось на детальному описі системи виявлення атак Snort.2.3.1.

Аналіз системи виявлення атак Snort

Snort - це мережева IDS, здатна в режимі реального часу аналізувати трафік, що передається по контрольованому інтерфейсу, з метою виявлення спроб вторгнення або спроб пошуку вразливостей (таких, як переповнення буферу, сканування портів, CGI-атаки, ідентифікація операційної системи, ідентифікація версій використовуваних мережевих сервісів тощо). Гнучкість і зручність Snort ґрунтуються на 3 положеннях:

- Мовою правил, використовуваної для опису властивостей підозрілого і потенційно небезпечного трафіку;
- Механізмі оповіщення про виявлення атаки;

- Модульній архітектурі коду, аналізатора трафіка, заснованої на концепції модулів.

Зверніть увагу, що існують процедури декодування мережевого трафіку, що працюють від каналу до рівня програми. На даний момент Snort підтримує декодування для інтерфейсів Ethernet, SLIP та PPP.

Розглянемо найважливіший елемент з точки зору користувача - мову правил. Правила встановлюються у файлі конфігурації Snort. Їх синтаксис досить простий:

ACTION PROTO IP_ADDR1 PORT1 DIRECTION IP_ADDR2 PORT2 [(OPTIONS)]

ACTION. Існує три основні директиви, які визначають, що робити далі, коли ви знайдете мережевий пакет, який відповідає правилу: передача, журнал та попередження.

Директива про передачу просто говорить вам ігнорувати пакет. Директива log визначає, що пакет повинен бути переданий вибраною користувачем процедурою документації для подальшого запису у файл журналу. Директива оповіщення генерує повідомлення, коли виявляє пакет, який відповідає правилу в заданому користувачем порядку, а потім передає пакет у процедуру документації для подальшого аналізу.

Ви також можете використовувати ще дві директиви: активацію та динаміку. Вони дозволяють одному правилу викликати інше. Наприклад, якщо виявлено пакет з очевидними ознаками атаки переповнення буфера, може знадобитися сформувати повідомлення про атаку та записати ще кілька пакетів у файл журналу для подальшого аналізу. Ця функціональність досягається наданням активних та динамічних директив. Крім того, ви можете визначити власні директиви, пов’язуючи їх з однією або кількома процедурами документації. Наприклад, визначення

```
ruletype doublealert
{
    type alert
```

```
        output alert_eventlog: LOG_AUTH LOG_ALERT
        output database: log, mysql, user = snort dbname = snort host =
        localhost
}
```

створює нову директиву doublealert, що генерує повідомлення з наступною передачею його eventlog і записом інформації про виявлений пакеті в базу даних MySQL.

PROTO. В даний час для аналізу доступні три протоколи і, відповідно, припустимі три значення цього параметра - tcp, udp, icmp. У майбутньому, можливо, з'явиться підтримка ARP, IPX, IGRP, GRE, RIP, OSPF та інших.

IP_ADDR. Snort не має механізму для розпізнавання імен (з міркувань продуктивності), тому для завдання хостів необхідно використовувати їх IP-адреси. Ключове слово any дозволяє задати всі можливі адреси для під мереж вказуються CIDR-блоки.

Символ «!» інвертує умову, тобто ! 192.168.3.0/24 означає будь-яка IP-адресу, що не належить підмережі 192.168.3.0/24. Крім того, можна задавати списки адрес, перераховуючи їх через кому і укладаючи в квадратні дужки: [192.168.2.0/24, 192.169.3.54/32].

PORT. Визначення номерів портів здійснюється точно так, як і в Linux-утиліті ipchains. Тобто, крім єдиного номера порту можна задати діапазон портів через двокрапку, наприклад, 6000:6010 - порти з 6000 по 6010 включно, :1024 - порти з 1 по 1024, 1024: - порти з 1024 по 65536. Як і у випадку IP-адрес, символ «!» інвертує умова, а ключове слово any позначає всі порти.

DIRECTION. Цей оператор дозволяє визначити напрямок руху пакету:

-> (Одностороннє) - правило буде застосовуватися тільки до пакетів, що йде з IP_ADDR1 на IP_ADDR2;

<> (Двостороннє) - напрям руху пакету ролі не грає.

OPTIONS. Укладені в круглі дужки параметри є необов'язковою частиною правила і, одночасно найважливішою частиною системи виявлення вторгнення. Параметри можуть визначати текст, який повідомляє про загрозу повідомлення,

задавати додаткові дії при спрацьовуванні правила та додаткові умови на відповідність аналізованих пакетів цьому правилу.

Параметри відокремлюються один від одного крапкою з комою, а ключове слово параметра відокремлюється від його аргументу двокрапкою. В даний час існує 24 параметри, але їх кількість постійно збільшується від версії до версії.

Параметри дозволяють створювати правила для перехоплення практично будь-яких пакетів, які якось можуть загрожувати безпеці. А якщо врахувати, що Snort може перехоплювати пакети на канальному рівні, то його застосування особливо цікаво на вузлах, захищених за допомогою firewall, так як відфільтровані firewall пакети все одно будуть знаходитися в полі зору Snort.

Параметри, значення яких мають сенс при відповідності аналізованого пакету всім умовам:

msg - містить текст повідомлення;

logto - задає альтернативний файл для запису в нього вмісту пакета;

session - цей параметр дозволяє включити вилучення даних користувача з TCP-сесії, наприклад, для подальшого аналізу того, які команди вводив користувач під час telnet-сесії;

resp - якщо пакет відповідає правилу, то Snort виконає одну з зазначених дій - наприклад, закриє з'єднання, відправивши TCP-RST-пакет одному з вузлів.

react - блокує задані в правилі web-сайти, закриваючи з'єднання з ними та / або відправляючи задане повідомлення браузеру, з якого була зроблена спроба зайти на сайт.

Нижче наводяться деякі приклади визначення правил.

log tcp any any -> 192.168.1.0/24 6000:6010

Згідно з цим правилом всі пакети, адресовані на зазвичай використовувані порти хостів деякої під мережі, будуть записуватися в журнал.

alert tcp! 192.168.1.0/24 any -> 192.168.1.0/24 any (msg: "IDS004 - SCAN-NULL Scan"; flags: 0; seq: 0; ack: 0;)

Таке правило виявляє спробу так званого NULL-сканування портів.

alert tcp! 192.168.1.45/32 any -> 192.168.1.45/32 80 (msg: "IIS-_vti_inf"; flags: PA; content: "_vti_inf.html"; nocase;)

Адресовані web-серверу пакети, що містять в собі запит до файлу _vti_inf.html, розглядаються як спроба скористатися однією з вразливостей Internet Information Server, при виявленні таких пакетів відбудеться генерація повідомлення про цю подію, а сам пакет запишеться в лог-файл.

alert tcp any any <> any 6688 (msg: "Napster Client Data"; flags: PA; content: ". mp3"; nocase; resp: rst_all)

У разі виявлення запиту до Napster-сервера з'єднання примусово закривається. Як видно, за допомогою Snort організувати фільтрацію небажаного трафіку можна більш ефективно, ніж просто закриваючи відповідні порти на брандмауера, оскільки є можливість ввести додаткову умову на вміст пакетів.

alert tcp any 80 <> 192.168.1.0/24 any (content-list: "*.txt"; msg: "***"; react: block, msg;)

Цим правилом блокується доступ на web-сайти, адреси яких перераховані у файлі *.txt. При виявленні запиту до небажаного сервера з'єднання з ним закривається, генерується повідомлення «***», яке крім запису в лог відправляється і браузеру. Таким чином, Snort може виконувати функції web-фільтра.

activate tcp! 192.168.1.0/24 any -> 192.168.1.0/24 143 (flags: PA; content: "| E8C0FFFFFF | \ bin |"; activates: 1; msg: "IMAP buffer overflow!";) dynamic tcp! 192.168.1.0/24 any -> 192.168.1.0/24 143 (activated_by: 1; count: 50;)

Директива activate нічим не відрізняється від alert, за винятком того, що в розділі опцій правила activate завжди присутня опція activates, призначена для виклику на виконання іншого правила. Викликати можна тільки правило dynamic. Директива dynamic, у свою чергу, нічим не відрізняється від директиви log. Вона також призначена для запису пакетів в лог, але має при цьому два додаткових параметри: activated_by, яка асоціює правило dynamic з правилами activate, а також count, яка вказує для якої кількості пакетів повинне

відпрацювати правило, тобто скільки пакетів, наступних за перехопленим пакетом, повинні бути записані в балку. Наведене вище правило activate аналізує пакети на предмет спроби реалізації атаки на переповнювання буфера і, в разі виявлення такої, викликає правило dynamic для запису в лог файл наступних 50 пакетів. Якщо атака була успішною, то, аналізуючи згодом файл, можна встановити, яких саме збитків було завдано.

На сайті www.snort.org завжди можна скласти для своїх цілей набір вже готових правил. На сайті вони розбиті на кілька груп: BackDoor Activity, BackDoor Attempts, Backdoor Sig. Based, Exploits, Finger, FTP, ICMP, MISC, NetBios, RPC, RServices, Scans, SMTP, Sysadmin, TELNET, Virus - SMTP Worms, Web-CGI, Web-ColdFusion, Web-FrontPage, Web-IIS, Web-Misc .

Аналізуючи лог можна створити свою сигнатуру:

«052499-22:27:58.403313 192.168.1.4:1034 -> 192.168.1.3:143 TCP TTL: 64TOS: 0x0 DF *** PA * Seq: 0x5295B44E Ack: 0x1B4F8970 Win: 0x7D78 90 90 90 90 90 90 90 90 90 90 90 90 90 90 EB 3B .....; 5E 89 76 08 31 ED 31 C9 31 C0 88 6E липня 2010 6E 0C ^. V.1.1.1 .. n.. N. B0 0B 89 F3 8D 6E серпня 1989 E9 8D 6E 0C 89 EA CD 80 ..... n. ... n. .... 31 DB 89 D8 40 CD 80 90 90 90 90 90 90 90 90 90 1 ... @ ..... 90 90 90 90 90 90 90 90 90 90 90 E8 C0 FF FF FF ..... 2F 62 69 6E 2F 73 68 90 90 90 90 90 90 90 90 90 / bin / sh ..... »

рядок / bin / sh явно виглядає підозріло. Таким чином, ця стрічка і кілька попередніх їй байт і будуть являти собою сигнатуру експлойта. Нове правило для Snort буде виглядати приблизно так:

alert tcp any any -> 192.168.1.0/24 143 (content: "| E8C0 FFFF FF | / bin / sh"; msg: "New IMAP Buffer Overflow detected!";)

До складу дистрибутива Snort входять специфікації, а також рекомендації з розробки модулів, призначені для сторонніх авторів. В даний час Snort підтримує три види модулів: детектуючі, препроцесора і модулі виведення інформації.

Детектуючі модулі призначені для виявлення будь-якого єдиного аспекту пакету і, таким чином, розширюють набір можливих параметрів, що визначають умови на відповідність правилу. Так, наприклад, параметр flags, що задає умови на наявність або відсутність в пакеті встановлених TCP-прапорів, реалізований за допомогою детектуючого модуля.

Код препроцесорів викликається на виконання після декодування пакету і перед виконанням детектуючого коду. Через механізм препроцесорів можлива модифікація пакетів, наприклад, для повторного складання TCP-потоку, для дефрагментації пакетів, нормалізації HTTP-запитів, можливий додатковий аналіз поза основним детектуючого коду, можливо також виконання завдань по збору статистики.

Зокрема, процедура визначення сканування портів реалізована саме за допомогою препроцесора. Під «скануванням портів» цей препроцесор розуміє відправку з одного і того ж IP-адреси SYN-, FIN-, NULL-, SYNFIN-або XMAS-пакетів контрольованим вузлом більше, ніж на P портів, за час менше T, або на один порт відразу декількох хостів. Користь від застосування цього препроцесора полягає в тому, що при виявленні атаки немає необхідності генерувати повідомлення про атаку для кожного пакету, а можна зробити це для всієї атаки в цілому.

РОЗДІЛ 3
РОЗРОБКА ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ СИСТЕМИ АНАЛІЗУ БЕЗПЕКИ ХМАРНИХ СХОВИЩ

3.1. Архітектура системи аналізу

Проаналізувавши основні аспекти концепції CloudComputing та основні моделі СВА, ми пропонуємо загальну архітектуру СВА та її модернізацію для використання аналітичної системи на основі концепції CloudComputing з можливістю аналізу великої кількості інформації про тривоги. Ця система - це система інтеграції тривог, побудована на інфраструктурі хмарних обчислень. Рис. 3.1. показує загальну архітектуру системи виявлення вторгнення що працює на основі порівняння за шаблоном.

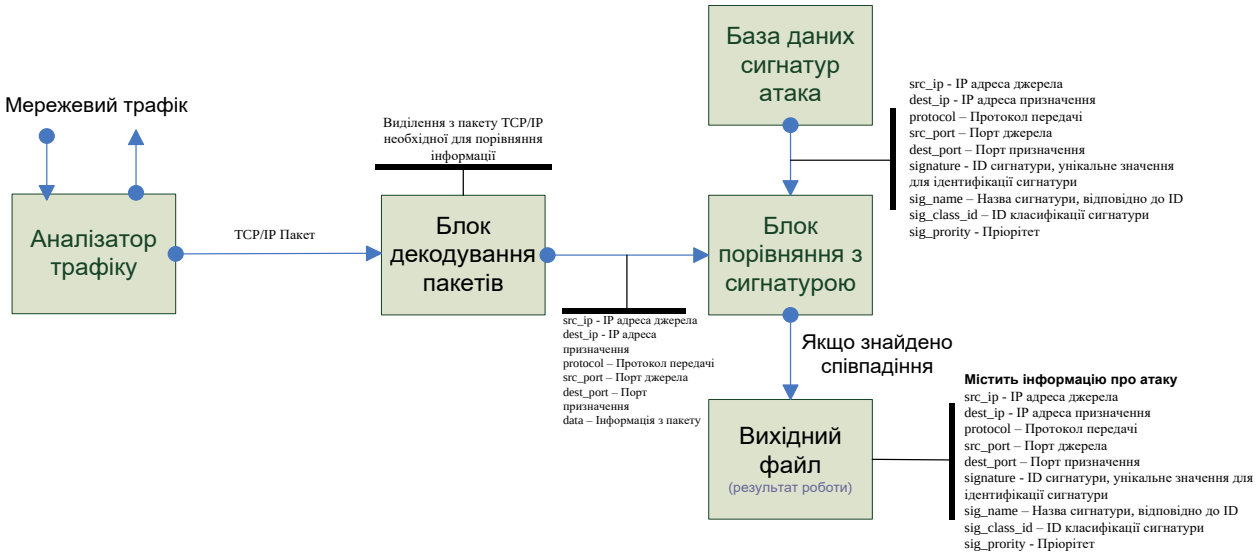


Рис.3.1. Архітектура СВА на основі порівняння за шаблоном

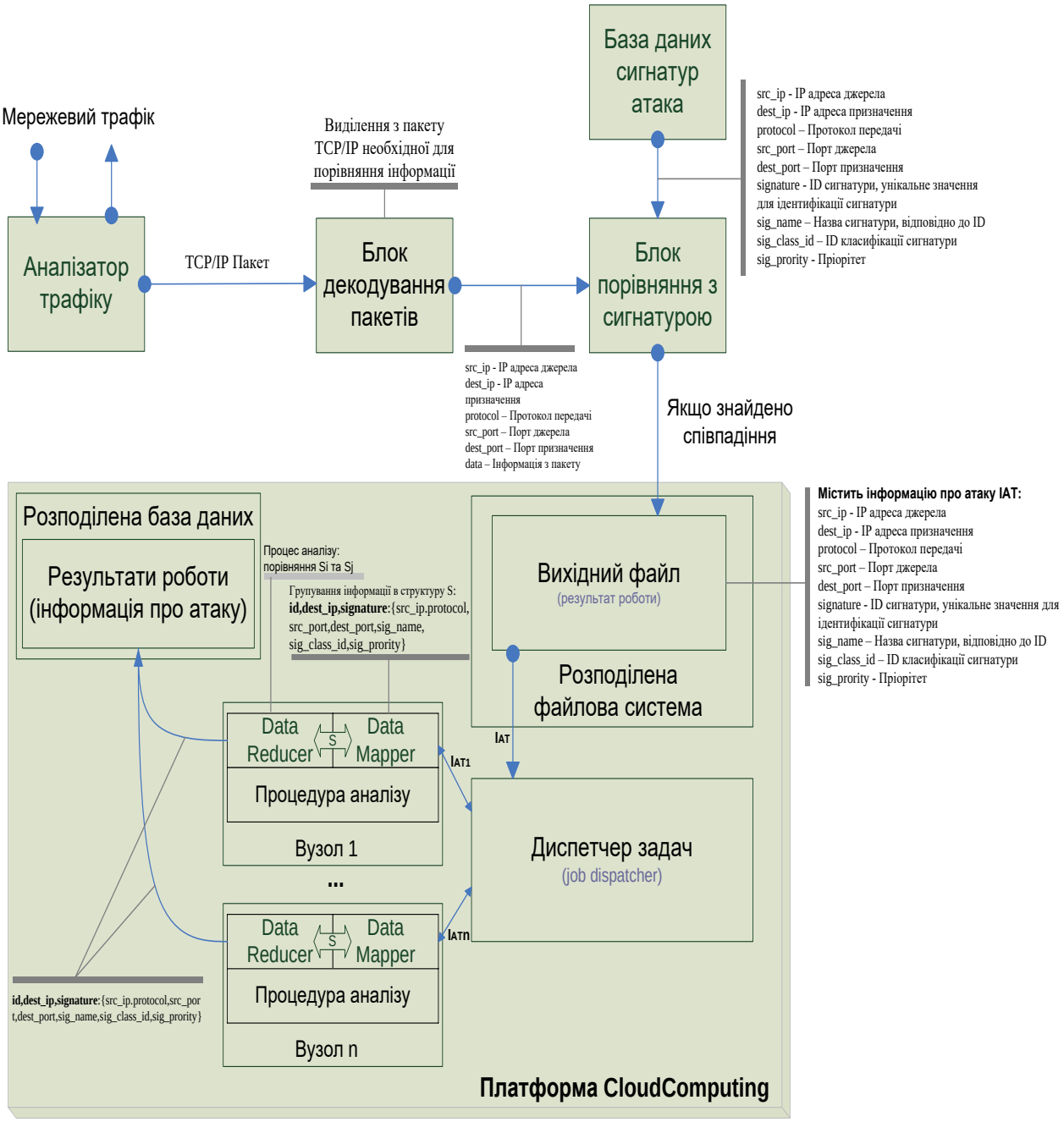


Рис. 3.2. Удосконалена архітектура системи аналізу на основі концепції Cloud Computing

СВА, генератор сигналу тривоги

Генератор тривог - це програмне забезпечення, призначене для роботи в мережі та виявлення спроб вторгнення та аналізу мережевої активності. Програмне забезпечення повинно подавати оригінальний твір у текстовому файлі. У нашому випадку цей продукт - СВА Snort. Ця система має широкий

спектр функцій, можливість створювати правила користувачем і результати роботи в текстовому файлі.

Платформа Cloud Computing

Платформа CloudComputing повинна забезпечувати використання алгоритму MapReduce та підтримувати розподілені файлові системи. Прикладом такої платформи є віртуалізація двох служб Apache Java: Hadoop [42] і HBase [43]. Hadoop - це комбінація Google MapReduce [39] та Google File System [41], структура, заснована на цій технології, яка підтримує обробку великої кількості інформації в кластерах. Hadoop прозоро забезпечує додатки надійністю та швидкістю операцій з даними. Hadoop реалізує обчислювальну парадигму, відому як MapReduce. Відповідно до цієї парадигми, програма розбита на велику кількість невеликих завдань, кожне з яких може виконуватися на кожному з вузлів кластера. Hadoop також має розподілену файлову систему, яка використовується для зберігання обчислювальних даних у вузлах кластера, що забезпечує дуже високу сукупну пропускну здатність кластера. Ця система дозволяє легко масштабувати вашу програму до тисяч вузлів та петабайт даних.

Вузли, або ще їх називають інформаційні вузли, бмінюються блоками інформації завдяки протоколу блоків, що підтримується Hadoop. Вузли постійно з'єднуються між собою для балансування навантаження, контролю потоку даних і реплікації даних.

Розподілена файлова система (HDFS)- е файлова система, яка може взаємодіяти з кількома вузлами в мережі. У цій системі вузли мають доступ до одного дискового простору.

Диспетчер задач (MapReduce) складається з одного Трекера задач та декількох Трекерів завдань. Трекер задач контролює програмне забезпечення клієнта та формує робочі завдання для доступних вузлів Трекерів завдань в робочому кластері.

Також для реалізації CloudComputing платформи можна використовувати готовий продукт від компанії Amazon, Amazon Elastic MapReduce, готовий продукт Amazon, Amazon Elastic MapReduce, який спеціально розроблений для

аналізу великих обсягів інформації. Реалізація поєднує послугу для запуску додатків MapReduce та розподілену файлову систему для зберігання та обробки результатів.

Розподілена база даних (HBase) підтримує таблиці з одним мільярдом рядків і одним мільйоном стовпців.

Використання алгоритму MapReduce в СВА

MapReduce - це платформа для обчислення певних наборів розподілених завдань за допомогою великої кількості комп'ютерів (вузлів), що утворюють кластер. MapReduce складається з двох етапів: Map і Reduce. Крок Map попередньо обробляє введення. Для цього один із комп'ютерів (головний вузол - батьківський вузол) отримує вхідні дані від завдання, ділить їх на частини та передає іншим комп'ютерам (робочі вузли - робочий вузол) для початкової обробки. Цей крок отримує свою назву від однойменної функції вищого порядку. Основний вузол отримує відповіді від працюючих вузлів і на їх основі створює результат - рішення даної проблеми.

Перевага MapReduce полягає в тому, що він забезпечує розподілену операцію попередньої обробки та згортки. Операції попередньої обробки не залежать одна від одної і можуть виконуватися паралельно (хоча на практиці це обмежується джерелом вхідного сигналу та / або кількістю використовуваних процесорів). Подібним чином, багато робочих вузлів можуть конвертувати - потрібно лише, щоб усі результати попередньої обробки з одним конкретним значенням ключа оброблялися одночасно одним робочим вузлом. Хоча цей процес може бути менш ефективним, ніж більш послідовні алгоритми, MapReduce може застосовуватися до великих обсягів даних, які можуть бути оброблені великою кількістю серверів. Так, MapReduce можна використовувати для сортування петабайт даних, що займає лише кілька годин.

Паралельність також забезпечує деяке відновлення після часткової відмови сервера: якщо робоча станція, яка виконує операцію попередньої обробки або згортки, не вдається, її робота може бути переміщена на іншу робочу станцію (за умови, що доступні дані для операції). Фреймворк в основному базується на

функціях відображення та скорочення, що широко використовуються у функціональному програмуванні, хоча насправді семантика фреймворку відрізняється від прототипу.

На рис. 3.3. представлена архітектура MapReduce.

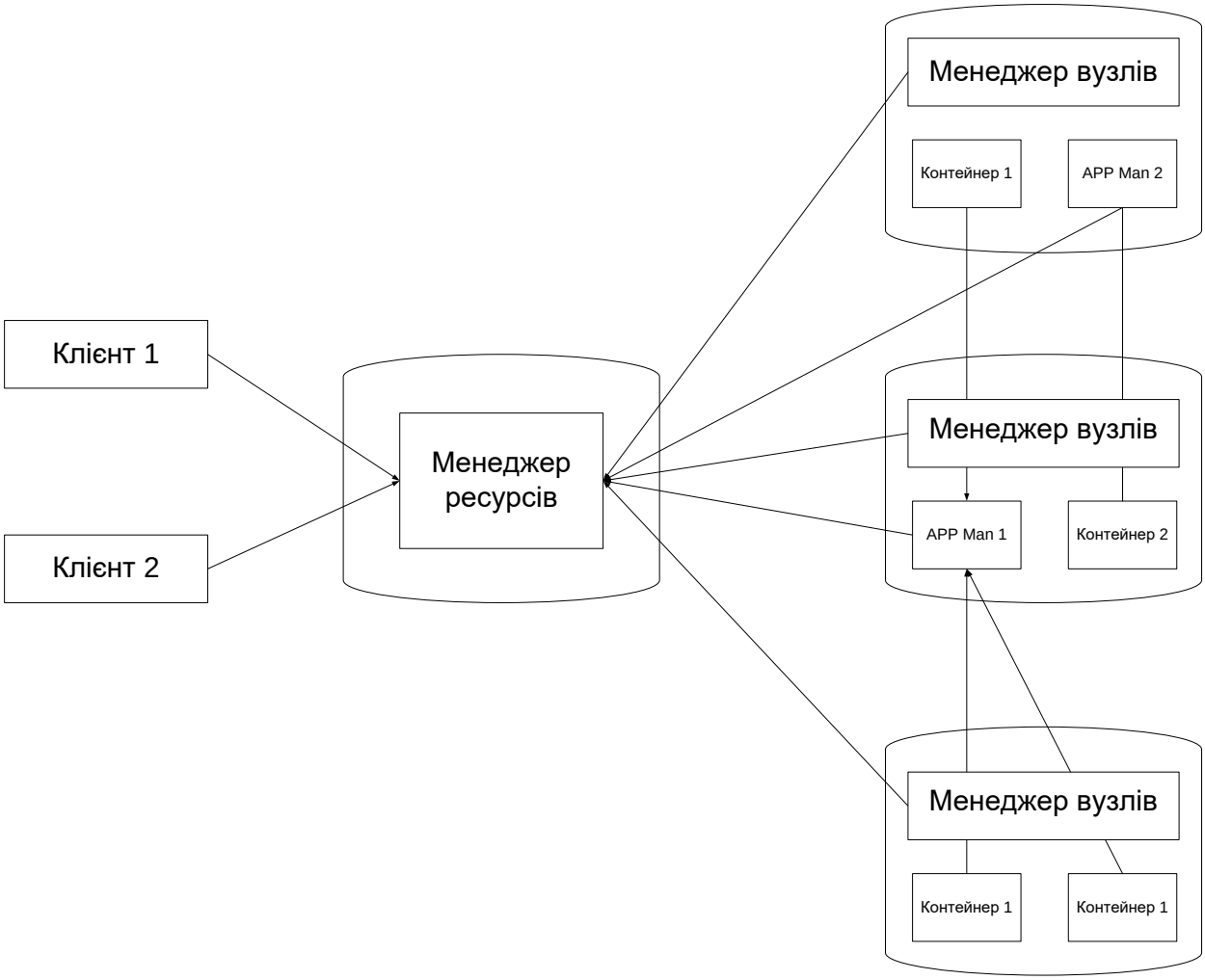


Рис. 3.3. Архітектура роботи MapReduce

MapReduce – новітня технологія, що має наступні переваги:

Масштабованість. Поділ завдань управління ресурсами та прикладних програм дозволяє простіше та ефективніше розширити кластер. Масштабованість особливо важлива у світлі сучасних апаратних тенденцій - Hadoop тепер може бути розгорнутий у кластері з 4000 машин. Однак у 2009 році 4000 машин середнього розміру (тобто 8 ядер, 16 ГБ пам'яті, 4 ТБ дискового простору) вимагали лише вдвічі більше ресурсів, ніж 4000 машин у 2011 році (16 ядер, 48 ГБ пам'яті, 24 ТБ дискового простору) . Більше того, з точки зору

експлуатаційних витрат, було вигідніше працювати в ще більшій групі з 6000 машин і більше.

Доступність.

• ResourceManager використовує Apache Keeper для обробки збоїв. Коли ResourceManager не працює, подібний процес може швидко розпочатися на іншому комп'ютері, оскільки статус кластера було збережено у Keeper. У цьому випадку всі заплановані та запущені програми будуть лише перезапущені.

• ApplicationMaster - підтримуються точки відновлення на рівні програми. ApplicationMaster може відновити збережений стан у разі відмови.

Сумісність протоколу. Це дозволить різним версіям клієнтів і серверів Hadoop взаємодіяти між собою. На додаток до вирішення багатьох існуючих проблем оновлення, майбутні випуски дозволять послідовне оновлення коду без загальних простоїв системи - дуже велике досягнення в системному адмініструванні.

Інноваційність і гнучкість. Головною перевагою запропонованої архітектури є те, що MapReduce по суті стає зручною для користувача бібліотекою. Обчислювальні системи (ResourceManager та NodeManager) повністю незалежні від особливостей MapReduce. Клієнти зможуть одночасно використовувати різні версії MapReduce в одному кластері. Це стає тривіальним, оскільки для кожної програми запускається окрема копія ApplicationMaster. Це дає вам можливість виправити помилки, вдосконалення та нові функції, оскільки повне оновлення кластера більше не є обов'язковою процедурою.

Використання Hadoop Distributed File System

Сучасні тенденції у розвитку веб-додатків та експоненціальне зростання інформації, яку вони обробляють, призвели до попиту на файлові системи, орієнтовані на високу продуктивність, масштабованість, надійність та доступність. Такі гіганти пошукової індустрії, як Google та Yahoo, не змогли уникнути цієї проблеми.

Специфіка програм та обчислювальної інфраструктури Google, побудованих на величезній кількості недорогих серверів із властивими

постійними перебоями, призвела до розвитку власної закритої розподіленої файлової системи Google File System (GFS). Ця система призначена для автоматичного відновлення після аварій, високої стійкості до несправностей, високої пропускної здатності при доступі до даних у потоковому режимі. Система призначена для роботи з великими обсягами даних, включаючи великі розміри файлів, тому GFS оптимізовано для відповідних операцій. Зокрема, для спрощення впровадження та підвищення продуктивності GFS не реалізує стандартний інтерфейс POSIX.

Відповіддю GFS став проект Hadoop з відкритим кодом з розподіленою файловою системою Hadoop. Проект активно підтримується та розвивається Yahoo. Проведемо порівняльний аналіз термінів, що використовуються в цих системах, визначимо їх сумісність та зосередимо увагу на HDFS більш детально (див. табл. 3.1).

Таблиця 3.1

Порівняння HDFS та GFS

	HDFS	GFS
Головний сервер	NameNode	Master
Підпорядкований сервер	DataNode	Chunk Server
Операція Append та SS	-	+
Автоматичне відновлення	-	+
Мова реалізації	Java	C++

HDFS - розподілена файлова система, яка використовується в проекті Hadoop. HDFS-кластер, в першу чергу складається з NameNode-сервера і DataNode-серверів, які зберігають безпосередньо дані. NameNode-сервер, керує простором імен файлової системи і доступом клієнтів до даних. Щоб розвантажити NameNode-сервер, передача даних здійснюється тільки між клієнтом і DataNode-сервером.

*Secondary NameNode.*Основний NameNode-сервер фіксує всі транзакції, пов'язані зі зміною метаданих файлової системи в log-файлі, під назвою EditLog. При запуску основного NameNode-сервера, він зчитує образ HDFS (розташований у файлі FsImage) і застосовує до нього всі зміни, накопичені в

EditLog. Потім записується новий образ, вже із застосованими змінами і система починає роботу вже з чистим log-файлом. Слід зауважити, що дану роботу NameNode-сервер виконує неодноразово при його першому запуску. У подальшому подібні операції покладаються на вторинний NameNode-сервер. FsImage і EditLog в кінцевому підсумку зберігаються на основному сервері.

Механізм реплікації. При виявленні NameNode-сервером відмови одного з DataNode-серверів (відсутність heartbeat-повідомлень від неї), запускається механізм реплікації даних:

- Вибір нових DataNode-серверів для нових реплік.
- Балансування розміщення даних по DataNode-серверів.

Аналогічні дії здійснюються в разі пошкодження реплік або у разі збільшення кількості реплік, властивих кожному блоку.

Стратегія розміщення реплік. Дані зберігаються у вигляді послідовності блоків фіксованого розміру. Копії блоків (репліки) зберігаються на декількох серверах, за замовчуванням - трьох.

Їх розміщення відбувається наступним чином:

- Перша репліка розміщується на локальному ноді.
- Друга репліка на інший ноді, в цій же стійці.
- Третя репліка на довільній ноді іншої стійки.
- Інші репліки розміщуються довільним чином.

При читанні даних клієнт обирає найближчу до нього DataNode-сервер з реплікою.

Цілісність даних. Ослаблена модель цілісності даних, реалізована у файловій системі, не гарантує ідентичність реплік. Тому HDFS надсилає клієнтам перевірку цілісності даних. Під час створення файлу клієнт обчислює контрольні суми кожні 512 байт, які потім зберігаються на сервері DataNode. Під час читання файлу клієнт отримує доступ до даних і контрольної суми. І у випадку їх розбіжності є посилання на іншу тиражування.

Запис даних. При записі даних в HDFS використовується підхід, що дозволяє досягти високої пропускної здатності. Додаток веде запис в потоковому

режимі, при цьому HDFS-клієнт кешує записувані дані в тимчасовому локальному файлі. Коли у файлі накопичуються дані на один HDFS-блок, клієнт звертається до NameNode-сервера, який реєструє новий файл, виділяє блок і повертає клієнтові список datanode-серверів для зберігання реплік блоку. Клієнт починає передачу даних блоку з тимчасового файлу, перший DataNode-сервер зі списку. DataNode-сервер зберігає дані на диску і пересилає наступному DataNode-серверу в списку. Таким чином, дані передаються в конвеєрному режимі і реплікуються на необхідній кількості серверів. Після закінчення запису, клієнт повідомляє NameNode-сервер, який фіксує транзакцію створення файлу, після чого він стає доступним в системі.

Видалення даних. У силу забезпечення цілісності даних (на випадок відміни операції), видалення у файловій системі відбувається за певною методикою. Спочатку файл буде перенесено в спеціально відведену для цього / trash директорію, а вже після закінчення певного часу відбувається його фізичне видалення:

- Видалення файлу з простору імен HDFS.

- Звільнення пов'язаних з даними блоків.

Недоліки технології:

- Відсутність автоматичного запуску головного сервера в разі його збою (дана функціональність реалізована в GFS)

- Відсутність операцій append (передбачається у версії 0.19.0) і snapshot (дані функціональності також реалізовані у GFS)

Система аналізу тривог

Аналізатор нормалізує файл журналу тривоги СВА та відправляє його для подальшої обробки. Кожен сигнал тривоги містить велику кількість параметрів, аналізатор виділяє лише ті параметри, які необхідні, і передає їх для подальшої обробки.

Процедура аналізу складається з DataMapper та DataReducer.

DataMapper використовується для роботи з вхідними даними. Результатом процедури є список пар типів - ключ, значення. Тоді фреймворк

CloudComputing збирає всі пари з однаковим ключем і ділить їх на групи залежно від ключа.

DataReducer використовується для прив'язки даних до DataMapper. Потім результати вносяться в базу даних.

3.2. Процедура інтеграції тривоги

Кореляція тривог - це процес аналізу, який генерує звіт для мережі на основі попереджень, генерованих СВА. Ряд запропонованих підходів включає багатофазний аналіз послідовності тривоги. Наприклад, модель, запропонована Андерсоном та Вальдесом, представляє процес кореляції як сукупність подій низького рівня, використовуючи дані атак та подібні метрики для об'єднання подій в одну атаку. Цей підхід залежить від пулу вхідних даних, який включає детальний опис характеристик безпеки та пріоритетів тривоги, формат перевірки тривоги.

Хоча деякі методи кореляції тривожності були описані вище, єдиної думки щодо того, що це за процес і як його слід оцінювати, немає. На щастя, Фредерік запропонував досить відповідний підхід до інтеграції тривожності. Результати його досліджень підтверджують, що його підхід має досить високий рівень зниження швидкості. Спираючись на його концепцію, ми визначимо основні етапи поєднання тривожності та застосуємо їх у нашій аналітичній системі

Як показано на рис.3.4, процес об’єднання полягає в перевірці того, чи може raw-alert та meta-alert бути об’єднані. Спочатку raw-alert з ключем K1 та значення V1 та V2 визначається як перший сигнал тривоги та переміщається в meta-alert. Наступна надходить тривога з ключем K2. Оскільки ключі тривоги не співпадають, процесор розподілу переносить її в meta-alert. Після формування третього сигналу тривоги з ключем K1 та значенням V1, V3, він переміщається в meta-alert, але не повністю, а лише нове значення V3, що відноситься до ключа K1.

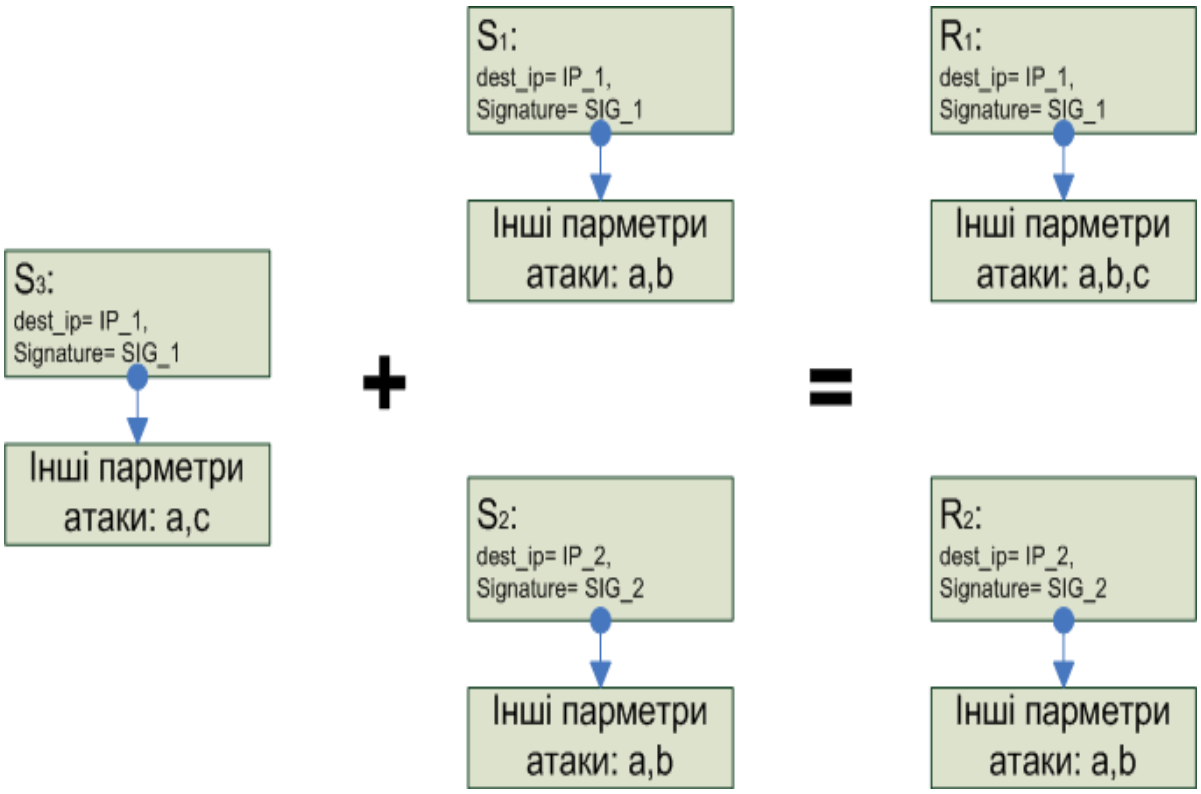


Рис. 3.4. Процедура інтеграції сигналу тривоги

Процес у цій формі відповідає класичній ідеї. У майбутньому Snort CBA буде використовуватися із посиланням на базу даних MySQL, а описаний вище підхід застосовуватиметься в процесі інтеграції. Сповіщення, генеровані Snort, зберігаються в базі даних MySQL і розподіляються в таких таблицях: "подія" (пов'язана з таблицею "підпис"), "iphdr", "tcphdr", "udphdr", "icmphdr". У цих 5 таблицях поля sic та cid - це складні ключові поля, що використовуються для ідентифікації тривоги, але вони не є необхідними для зв'язування тривог. При виборі даних з основних таблиць процес інтеграції поєднує всі дані і поміщає результат у таблицю "final_alert".

3.4. Інтеграція CBA в Cloud Computing

Рис 3.5. показує послідовність роботи CBA з аналізатором лог файлів. Лог файл – файл з записами тривог, генерований аналізатором. Regular Parser,

Analysis Procedure, Data Mapper, Data Reducer . База даних – розподілена база даних в CloudComputing архітектурі. Мета файл - файл, що передається в розподілену файлову систему. Всі мета-дані проміжні результати роботи Диспетчера задач між Data Mapper та Data reducer.

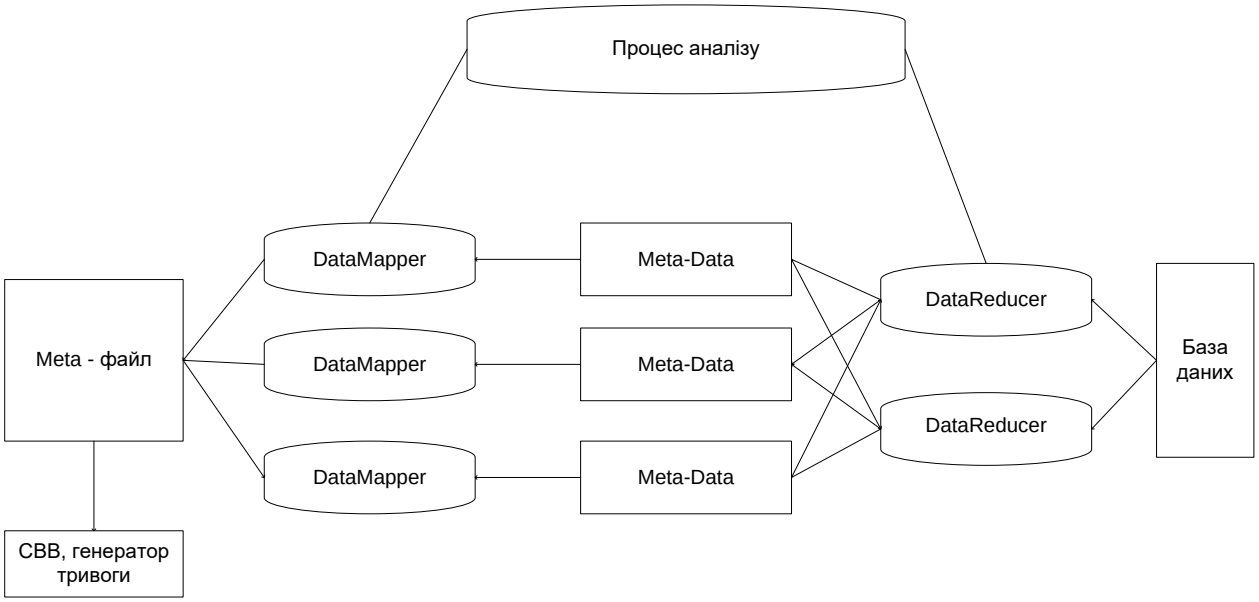


Рис. 3.5. Послідовність роботи СВА з аналізатором лог файлів

На початку процедури, генератор тривоги збирає підозрілі пакети та записує інформацію у файл журналу. Однак файл журналу не підготовлений, тому перший компонент є парсером, він витягує необхідну інформацію з файлу і створює базовий метафайл. Наступним кроком є відправка метафайлу до розподіленої системи, де він буде розділений на частини та відправлений на обробку до різних вузлів. Диспетчер завдань запускає DataMapper і призначає завдання кожному вузлу.

Основна мета DataReducer - зменшити надмірність та сукупну інформацію. Правило зменшення: якщо будь-який із двох сигналів тривоги, які мають однакову IP-адресу призначення та однаковий підпис, представлений як один сигнал тривоги, це означає, що два сигнали інтегровані в один, а їх атрибути поєднані. Наприклад, у метафайлі є 6 сигналів тривоги, як показано в таблиці 3.1,

таблиця 3.2 буде результатом обробки цього файлу. I1 та I2 - одна і та ж спроба, оскільки зловмисник виконував однакові дії, але в різний час. Після аналізу I1 та I2 вони падають до R1. Різні атрибути I3 та I4 повинні поєднуватися як R2 {b, c}, оскільки вони мають однакову адресу призначення та підпис. Наступний приклад: Є два хости, які використовують однакові методи для атаки на одну ціль. Всі основні атрибути різні, тому атаки не можна поєднувати. Наприклад, I5 та I6 передаються як R3 та R4. Процес показаний на Рис.3.6. Нарешті, система зберігає результати у розподіленій базі даних.

Таблиця 3.2

Вхідні сигнали тривоги

ID	IP_DST	Сигнатура	Параметри
I1	IP_1	A	a,b
I2	IP_1	A	a,b
I3	IP_2	A	a,b
I4	IP_2	A	a,c
I5	IP_3	A	a,b
I6	IP_3	B	a,b

Таблиця 3.3

Вихідні сигнали тривоги

ID	IP_DST	Сигнатура	Параметри
1	2	3	4
1	2	3	4
R1	IP_1	A	a,b
R2	IP_2	A	a,b,c
R3	IP_3	A	a,b
R4	IP_3	B	a,c

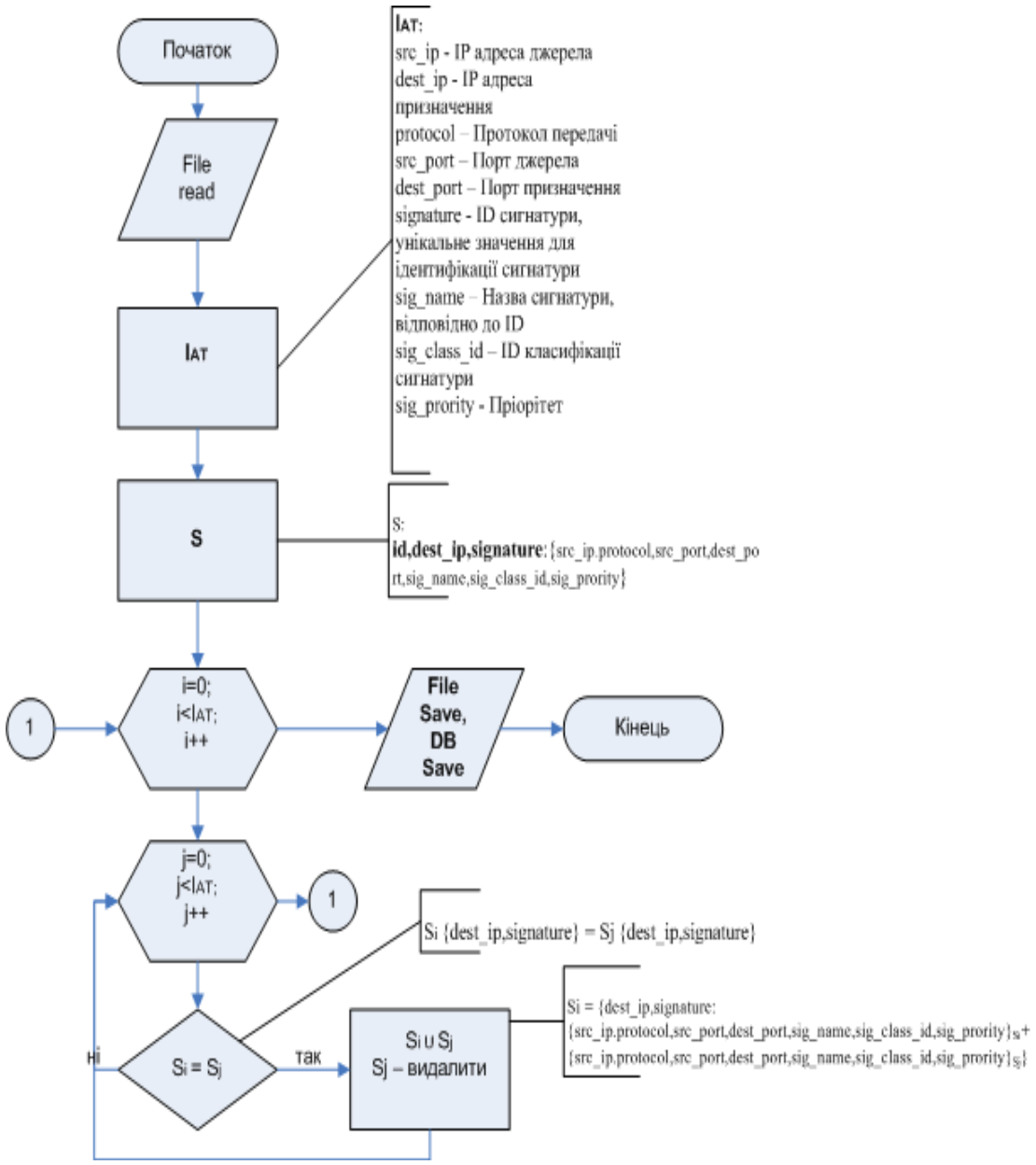


Рис. 3.6. Алгоритм роботи системи аналізу

Створення нової структури файлів попереджень виділяє ключову інформацію, яка є критично важливою для визначення атаки та ігнорування несуттєвої інформації, вилучаючи її з нової структури. (див. табл. 3.4).

Таблиця 3.4

Структура вхідного файлу для початку аналізу

ip_dst	IP адреса призначення
signature	ID сигнатури, унікальне значення для ідентифікації сигнатури
sig_name	Назва сигнатури, відповідно до ID сигнатури
sig_class_id	ID класифікації сигнатури
sig_priority	Пріоритет сигнатури
ip_src	IP адреса джерела
ip_proto	TCP/IP протокол
port_dst	Порт призначення
port_src	Порт джерела

Ця інформація буде використана для створення нової структури.

Для отримання результатів та оцінки ефективності нашої аналітичної системи проводились експерименти. В результаті ми отримали такі параметри: час аналізу файлів тривоги та отриману кількість сигналів тривоги. Для проведення випробувань використовувались наступні системи (див. табл. 3.5). Результати розрахунку представлені в табл. 3.6. Вхідні дані для обох систем були підготовлені за допомогою тривожних файлів, створених лабораторією MIT Лінкольна [47].

Таблиця 3.5

Тестові системи

Параметри	Тестова система 1	Тестова система 2
Концепція	Локальні віртуалізована ресурси	Розподілені обчислення
Обчислювальний пристрій	Intel Core i5 , 3,3 GHz	20EC2 обчислювальних пристрої (8x2.5 EC2 віртуальних ядер)
Оперативна пам'ять	3,5 Гб	7 Гб
Пристрій зберігання	7200 RPM Sata	-
Оперційна система	Apache Hadoop MapReduce	Elastic MapReduce

Таблиця 3.6

Результати експерименту

Вхідна кількість тривог	Час аналізу (сек.)		Вихідна кількість тривог	Рівень зменшення
	Тестова система 1	Тестова система 2		
280	1,4	4,1	30	89,28%
380	1,9	4,2	11	97,10%
440	2,1	4,2	9	97,95%
750	3,0	4,8	16	97,86%
1100	4,8	5,2	33	97,00%
1700	8,6	5,5	15	99,11%
2100	15,5	5,9	16	99,23%
3390	19,6	6,5	75	97,97%
5815	381,1	7,1	72	98,76%
6340	390,5	8,3	80	98,76%
12670	680,3	9,5	79	99,37%

Головна різниця між системами – централізована та розподілена архітектура. Інша вагома різниця в тому, що Snort дає змогу генерувати записи в базу даних MySQL напряму і потім отримувати результати з неї, в нашій системі результати подаються з Amazon S3 системи, яка отримує їх напряму з Snort лог файлу.

З результатів досить чітко зрозуміло, що Тестова система 1 ефективна, коли кількість тривог для обробки не перевищує 1100, але цей метод потребує значно більше часу, якщо кількість тривог перевищує 5000. В свою чергу, для нашої системи на виконання даного аналізу необхідно лише 4-9 секунд. Слід зауважити, що використання більшої кількості інстансів дасть змогу обробляти інформації ще швидше. Але, якщо кількість тривог буде менша за 1000, наша система потребує набагато більше часу, ніж традиційна. Це пов’язано з тим, що велика кількість обчислювальних вузлів потребують більше часу для синхронізації та обміну даними між собою.

ВИСНОВКИ

Таким чином, детально проаналізувавши концепцію CloudComputing можна зробити висновок шо дана концепцепція є досить цікавим інструментов для застосування в різних сферах IT технологій. Особливо велий інтерес дана концепція має в сфері захисту інформації, оскільки можливості концепцепції CloudComputing при достатньому рівні вивчення, можут запропонувати кардинально нові рішення для забезпечення безпеки.

В роботі були проаналізовані аспекти безпеки використання Cloud Computing, розглянуті найбільш поширені ризики за вразливості що пов’язані з Cloud Computing. Було виконано аналіз моделей СВА та порівняння програмних реалізацій.

Побудована модель системи аналізу що працює на основі концепції Cloud Computing. Можна зробити висновок що система аналізу на основі концепції CloudComputing має менший час обробки при роботі з великим об’ємом інформації, це досягається розподіленими обчисленням. Також використання даного алгоритму дає смогу скоротити розмір файлу тривоги СВА що дасть змогу для зручного аналізу.

Розроблена система дозволяє в автоматичному режимі, аналізувати трафік, генерувати файл тривоги, передавити його на аналіз CloudComputing платформі, та основі результатів формувати нові фільтри, та правили, для забезпечення більш надійного рівня безпеки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hayes B. CloudComputing // Communications of the ACM.-2008.- P.9–11
2. Google app engine [Електронний ресурс]: – Режим доступу: http://google.com.ua/appengine
3. Weiss A. Computing in the clouds// networker.-2007.- P.16–25
4. Keynote K. Massively distributed systems: From grids and p2p to clouds. // In The 3rd International Conference on Grid and Pervasive Computing - gpc-workshops, 2008. 25P.
5. Райхман Е. Л., Азгальдов Г. Г. Експертные методы в оценке качества товаров. Москва: Економика. 1974. 151 с.
6. Саати Т. Л. Целочисленные методы оптимизации и связанные с ними Екстремальные проблемы. Москва: Мир, 1973.302 с.
7. Саати Т. Л. Принятие решений. Метод анализа иерархий. Москва: Радио и связь, 1989. 316 с.
8. Системы обнаружения и предотвращения вторжений (IPS/IDS). – URL: https://www.anti-malware.ru/security/ids-ips
9. Milojicic D. Cloud computing: Interview with russ daniels and franco travostino // IEEE Internet Computing, 2008.- P.7–9
10. Geelan J. Twenty one experts define cloud computing.Virtualization [Електронний ресурс]: – Режим доступу: http://virtualization.sys-on.com/node/612375
11. Gartner Group Gartner’s hype cycle report [Електронний ресурс]:
12. – Режим доступу: http:// www.gartner.com /
13. Foster I. What is the grid? - a three point checklist [Електронний ресурс]: – Режим доступу:http://www.gridtoday.com/02/0722/100136.html
14. Gruman G. , Knorr E. What CloudComputing really means [Електронний ресурс]: – Режим доступу: http://www.infoworld.com/article/08/04/07/15FE-cloud-computing-reality 1.html- Назва з екрану

15. McFedries P. The cloud is the computer // IEEE Spectrum Online.-2008

16. Bragg R. Cloud computing: When computers really rule [Електронний ресурс]: – Режим доступу: http://www.technewsworld.com/story/63954.html

17. Безопасность как головная боль облачных вычислений [Електронний ресурс] / А. Иванов. – Режим доступу: World Wide Web. – URL: http://www.cnews.ru/reviews/free/saas/articles/articles12.shtml

18. Безопасность облачных вычислений [Електронний ресурс]. – Режим доступу: World Wide Web. – URL: http://ru.thales-esecurity.com/solutions/by-business-issue/cloud-computing-security

19. Haaff B. CloudComputing - the jargon is back! [Електронний ресурс]:

– Режим доступу: http://Cloud Computing.sys-con.com/node/613070

20. Watson P., Lord P., Gibson F. CloudComputing for e-science with carmen.// 2008.-P.1-5

21. Jha S., Merzky A., Fox G. Using clouds to provide grids higher-levels of abstraction and explicit support for usage modes [Електронний ресурс]: – Режим доступу: http://grids.ucs.indiana.edu/ptliupages/publications/cloud-grid-saga.pdf, 2020.

22. Miguel L. Bote L., Yannis A. Dimitriadis Grid characteristics and uses: a grid definion.// 2004.- P.291–298.

23. Kurdi H., Li M., Al-Raweshidy H. A classification of emerging and traditional grid systems// Distributed Systems Online.-2008.

24. Stockinger H. Defining the grid: a snapshot on the current view//The Journal of Supercomputing.-2007.

25. Kemal A. Delic, Martin A. Walker Emergence of the academic computing clouds // ACM Ubiquity.-2008.

26. Rochwerger B., Breitgand D., Levy E. The reservoir model and architecture for open federated CloudComputing // IBM Systems Journal.

27. Wladawsky-Berger I. Cloud computing, grids and the upcoming cambrian explosion in it [Електронний ресурс]: – Режим доступу:

http://www.ogf.org/OGF22/- Назва з екрану, 2020.

28. Varia J. Amazon white paper on cloud architectures [Електронний ресурс]: – Режим доступу: http://aws.typepad.com/aws/2008/07/white-paper-on.html

29. Flexiscale web site [Електронний ресурс]: – Режим доступу: http://www.flexiscale.com

30. King S., Chen P., Wang Y. Implementing malware with virtual machines // 2006

31. Secunia [Електронний ресурс]: – Режим доступу: http://secunia.com/advisories/37081, 2020.

32. Gentry M. [Електронний ресурс]: – Режим доступу: http://delivery.acm.org, 2020.

33. Schneier F. [Електронний ресурс]: – Режим доступу: http://www.schneier.com/blog/archives/homomorphic_enP.html

34. Bechere A., Stamos A., Wilcox N. [Електронний ресурс]: – Режим доступу: http://www.slideshare.net/astamos/cloud-computing-security - Назва з екрану, 2020.

35. Andersson D., Fong M., and Valdes A. Heterogeneous Sensor Correlation: A Case Study of Live Traffic Analysis // Third Ann. IEEE Information Assurance Workshop, 2002

36. Attig M. and Lockwood J. A Framework for Rule Processing in Reconfigurable Network Systems" // 13th Annual IEEE Symposium, 2005

37. Chang F., Dean J., Ghemawat S. Bigtable: A Distributed Storage System for Structured Data // OSDI, 2006

38. Denning An Intrusion Detection Model // IEEE Transaction on Software Engine, 1987

39. Dean J. , Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters // OSDI 2004, 2004

40. EC2, "Amazon Elastic Compute Cloud" [Електронний ресурс]: – Режим доступу:

http://www.amazon.com/gp/browse.html?node=201590011 - Назва з екрану, 2019.

41. Ghemawat S., Gobioff H., Leung S. The Google File System // SOSP, 2003

42. Borthakur D. The Hadoop Distributed File System [Електронний ресурс]: – Режим доступу: http://lucene.apache.org/hadoop

43. Hbase: Bigtable-like structured storage for Hadoop hdfs [Електронний ресурс]: – Режим доступу: http://wiki.apache.org/lucene-hadoop/Hbase

44. Javitz H.S. , Valdes A. The NIDES Statistical Component Description and Justification // Technical Report, 1994

45. Ko P., Ruschitzka M., Levit K. Execution Monitoring of Security-Critical Programs in Distributed Systems: A Specification-Based Approach // IEEE Symp.Security and Privacy, 1997

46. Kruegel P. , Vigna G. Anomaly Detection of WebBased Attacks // CCS, 2003

47. MIT Lincoln Laboratory [Електронний ресурс]: – Режим доступу: www.ll.mit.edu/IST/ideval/data/data_index.html, 2019.

48. S. Lohr Google and I.B.M . Join in 'Cloud Computing' MySQL, The open source database [Електронний ресурс]: – Режим доступу: http://www.MySQL.com, 2019.

49. Neumann P.G., Porras P.A. Experience with EMERALD to Date // First USENIX Workshop Intrusion Detection and Network Monitoring, 1999

50. Porras P., Fong M., Valdes A. A Mission-Impact-Based Approach to INFOSEC Alarm Correlation // the Recent Advances in Intrusion Detection, 2002

51. Valeur F., Vigna G., Kemmerer R. A Comprehensive Approach to Intrusion // IEEE Transactions on Dependable and Secure Computing, 2004

52. Felix Wu laboratory [Електронний ресурс]: – Режим доступу: http://www.cs.ucdavis.edu/%7Ewu/tcpdump/, 2018.

53. Wu Y. S., Foo B., Mei Y. Collaborative intrusion detection system (CIDS): a framework for accurate and efficient IDS// Computer Security Applications Conference, 2019.

Додаток А

Слайди презентації

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ
КАФЕДРА БЕЗПЕКИ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ «МАГІСТР»

Тема: «Аналітична система оцінки безпеки хмарних сховищ»

Автор: Д.В. Дем'яненко

Керівник: А.О. Корченко

Актуальність

Сьогодні хмарні технології (CloudComputing) поширюються на всі сфери діяльності: бізнес, медицину, освіту та науку, медіа, фінанси та державний сектор. Зростаюча тенденція збільшення відсотка використання електронної комерції та користувачів Інтернету в цілому, які використовують її для здійснення покупок в Інтернет-магазинах, зберігання та резервного копіювання інформації з декількох пристроїв, розміщення інформації в Інтернеті, призводить до того, що використання фізичних ресурсів є неефективним і вимагають значних ресурсів для постійного підвищення продуктивності та технічного обслуговування. Тому доцільніше використовувати хмарні сервіси, які приносять користь кожному типу користувачів.

Хоча концепція хмарних технологій досить цікава, вона занадто молода і вимагає дуже детальних досліджень щодо її застосування та інтеграції з іншими системами. Слід зазначити, що в зарубіжних джерелах поняття CloudComputing активно вивчається, є значні результати і навіть практичні реалізації цієї концепції. Однак мало уваги приділяється застосуванню концепції CloudComputing в оцінці безпеки хмарних сховищ. CloudComputing може бути ефективним інструментом для створення систем аналізу безпеки даних, що є сьогодні дуже **актуальним завданням**.

Мета та задачі роботи

Метою роботи є розробка аналітичної системи оцінки безпеки хмарних сховищ для запобіганням атакам.

Для досягнення поставленої мети роботи потрібно вирішити наступні **задачі**:

- провести аналіз вразливостей та ризиків, пов'язаних з використанням Cloud Computing;
- розробити аналітичну систему аналізу файлів тривоги на основі концепції CloudComputing;
- провести експериментальне дослідження для визначення ефективності системи аналізу.

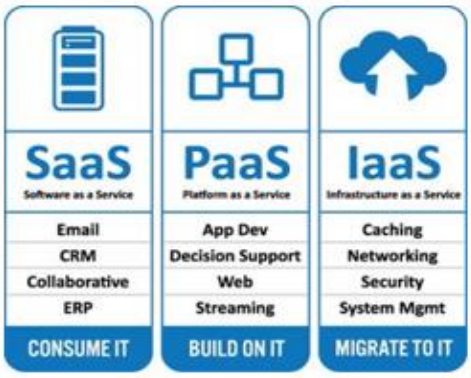
Новизна

Удосконалено аналітичну систему оцінки безпеки хмарних сховищ за рахунок оптимізації фйлів тивог, систем виявлення атак, використання позподілених обчислень, що дало можливість в автоматичному режимі аналізувати трафік, генерувати файл тривоги, передавати його на аналіз CloudComputing платформі, та на основі результатів формувати нові фільтри та правила для забезпечення захисту від атак та надійного рівня безпеки.

Об'єкт дослідження: процес оцінки безпеки хмарних сховищ.
Предмет дослідження: системи аналізу безпеки хмарних сховищ

Технологія Cloud Computing

Моделі розгортання



Моделі сервісних послуг

Результати порівняння розглянутих систем

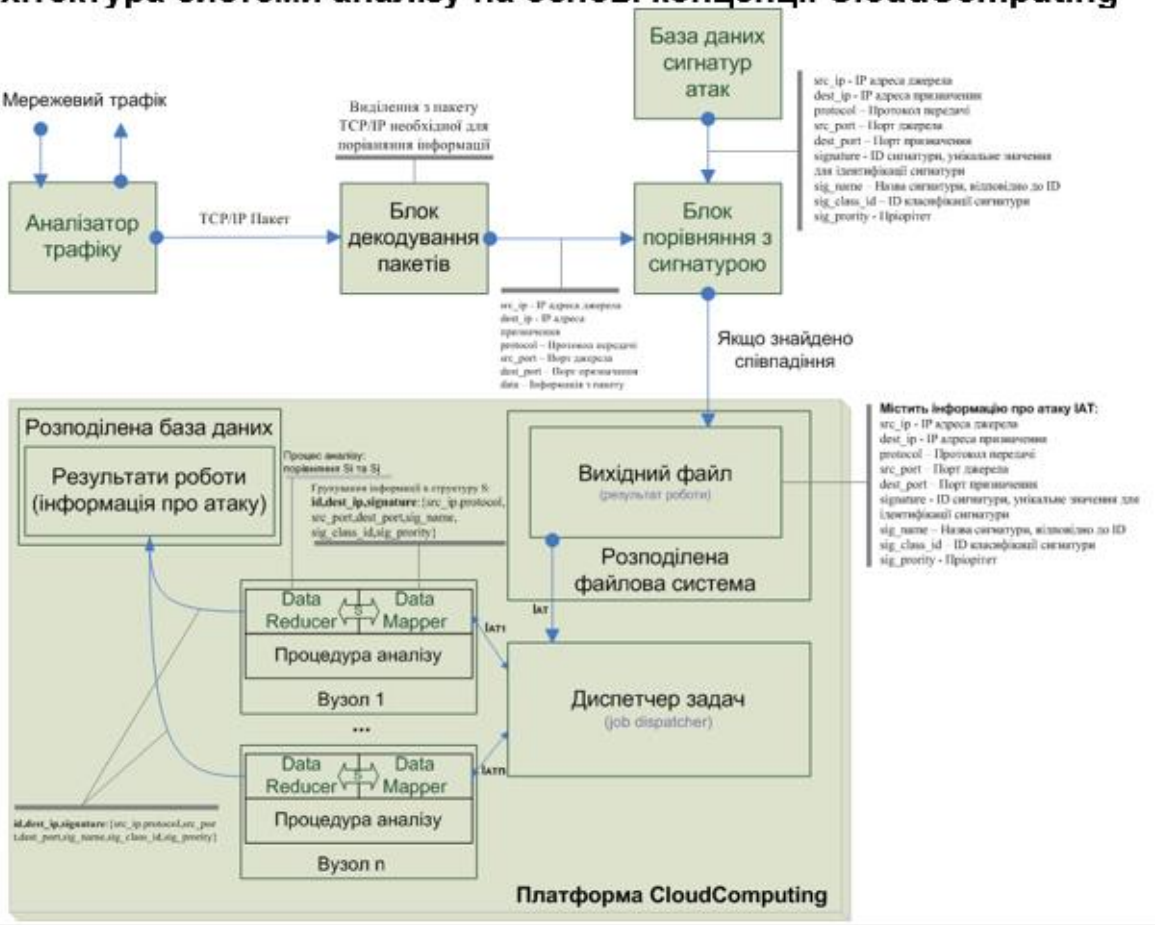
	AAFID	ASAX	Prelude	SHADOW	Snort
Рівень спостереження за системою	системний	системний	системний, мережевий	мережевий	мережевий
Метод виявлення	експертний	експертний	експертний	експертний	експертний, статистичний аналіз
Адаптивність	-	-	-	-	+
Керування	централізоване	централізоване	централізоване	розподілене	централізоване
Архітектура	розподілена	Не розподілена	розподілена	розподілена	Не розподілена
Реакція	-	-	+	-	+
Захист	-	-	SSL, Libsafe	SSH	-

Загрози, властиві архітектурам

Тип архі-тектури	Компонент	Загроза	К	Ц	Д	С
IaaS	SLA – угода про рівень надання послуг	Моніторинг та виконання SLA, моніторинг атрибутів QoS	+			
	Апаратне забезпечення	Фізичні атаки на апаратне забезпечення, втрата або компроментація даних на виведених із експлуатації або замінених пристроях зберігання даних	+		+	
	Мережі та Інтернет підключення	DDOS Атака "Людина в середині" (MITM). IP-підробка. Сканування портів. Безпека DNS.	+		+	
	Хмарне ПЗ	Атаки проти XML. Атаки на веб-сервіси		+	+	
	Візуалізація	Модифікації хоста VM, DoS-атака, мобільність VM, зв'язок між VM				
PaaS		Налаштування за замовчуванням	+	+		
		Відсутній SSL протокол і недоліки його реалізації	+	+		
		Некоректне призначення ролей доступу до даних	+			
SaaS		DoS, блокування облікового запису, переповнення буфера			+	
		XSS, слабкий контроль доступу, перевищення повноважень	+			
		Крадіжка сесії, перехоплення пакетів даних, проникнення в мережевий трафік	+			

Архітектура системи на основі порівняння за шаблоном

Архітектура системи аналізу на основі концепції CloudComputing



Вхідні дані для початку аналізу

Вхідні дані	
dest_ip	IP адреса призначення
signature	ID сигнатури, унікальне значення для ідентифікації сигнатури
sig_name	Назва сигнатури, відповідно до ID сигнатури
sig_class_id	ID класифікації сигнатури
sig_priority	Пріоритет сигнатури
src_ip	IP адреса джерела
protocol	TCP/IP протокол
dest_port	Порт призначення
src_port	Порт джерела

Процедура аналізу і оптимізації

Приклад процесу аналізу і оптимізації

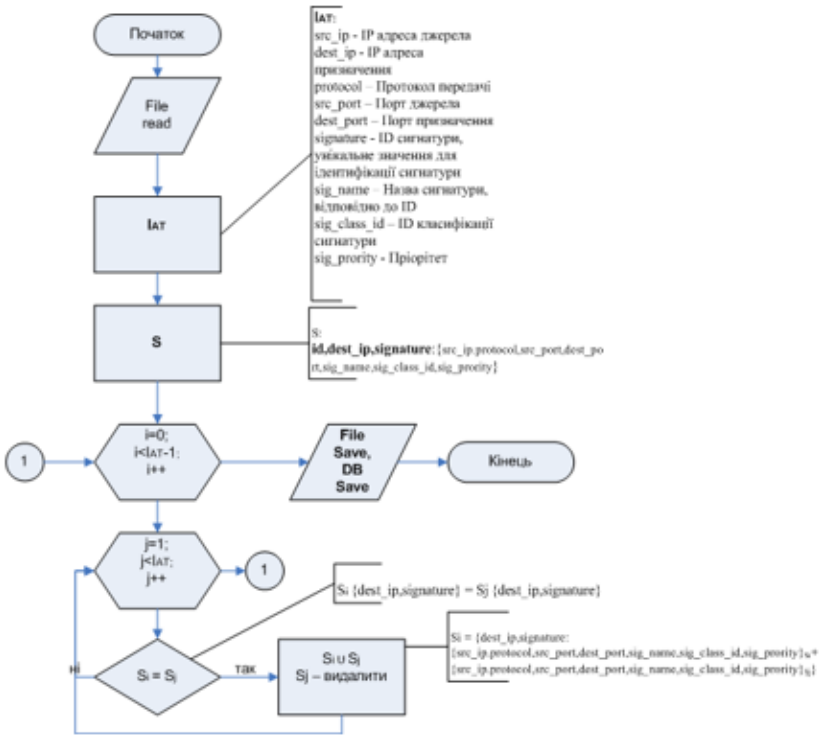
Вхідні сигнали тривоги

ID	IP_DST	Сигнатура	Параметри
I1	IP_1	A	a,b
I2	IP_1	A	a,b
I3	IP_2	A	a,b
I4	IP_2	A	a,c
I5	IP_3	A	a,b
I6	IP_3	B	a,b

Вихідні сигнали тривоги

ID	IP_DST	Сигнатура	Параметри
R1	IP_1	A	a,b
R2	IP_2	A	a,b,c
R3	IP_3	A	a,b
R4	IP_3	B	a,c

Алгоритм роботи системи аналізу



Експериментальні дані

Параметри	Тестова система 1	Тестова система 2
Концепція	Локальний віртуалізований ресурс	Розподілені обчислення
Обчислювальний пристрій	Intel Core i5 , 3,3 GHz	20EC2 обчислювальні пристрої (8x2.5 EC2 віртуальних ядер)
Оперативна пам'ять	3,5 Гб	7 Гб
Пристрій зберігання	7200 RPM Sata	-
Операційна система	Apache Hadoop MapReduce	Elastic MapReduce

Результати експерименту

Вхідна кількість тривог*	Час аналізу (сек.)		Вихідна кількість тривог	Рівень зменшення
	Тестова система 1	Тестова система 2		
280	1,4	4,1	30	89,28%
380	1,9	4,2	11	97,10%
440	2,1	4,2	9	97,95%
750	3,0	4,8	16	97,86%
1100	4,8	5,2	33	97,00%
1700	8,6	5,5	15	99,11%
2100	15,5	5,9	16	99,23%
3390	19,6	6,5	75	97,97%
5815	381,1	7,1	72	98,76%
6340	390,5	8,3	80	98,76%
12670	680,3	9,5	79	99,37%

**В якості вхідних сигналів тривог використовувались підготовлені файли тривог від MIT Lincoln Laboratory*

Висновки

1. Були проаналізовані аспекти безпеки використання концепції CloudComputing та проведено аналіз можливих ризиків та вразливостей її використання, що дало можливість порівняти переваги і недоліки кожної з них.

2. Удосконалено аналітичну систему оцінки безпеки хмарних сховищ, що дало можливість в автоматичному режимі аналізувати трафік, генерувати файл тривоги, передавати його на аналіз CloudComputing платформі, та на основі результатів формувати нові фільтри та правила для забезпечення захисту від атак та надійного рівня безпеки.

3. Виконано експериментальні дослідження, в результаті яких була обґрунтована ефективність розробленої системи аналізу. Розроблена система дозволяє в автоматичному режимі, налізувати трафік, генерувати файл тривоги, передавати його на аналіз CloudComputing платформі, та основі результатів формувати нові фільтри, та правила, для забезпечення більш надійного рівня безпеки.

Представлена система дозволяє вибрати лише певні атаки, що, в свою чергу, забезпечує високий рівень виявлення вторгнень. Ця функція досягається за допомогою алгоритму MapReduce. Аналітична система зберігає результати роботи в середовищі CloudComputing, що, в свою чергу, встановлює новий рівень цілісності та безпеки даних.