МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРИЗОВАНИХ СИСТЕМ ЗАХИСТУ ІНФОРМАЦІЇ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____ С.В. Казмірчук

«_____»_____ 2020 р.

На правах рукопису
УДК 004.056.55(083.94)

ДИПЛОМНА РОБОТА

ВИПУСКНИКА ОСВІТНЬО СТУПЕНЯ «МАГІСТР»

Тема: Удосконалений програмний модуль для підвищення захищеності серверу від мережевих атак

Автор: Пергаменщик Н.В.

Науковий керівник: к.т.н., доц. Гулак Н.К.

Нормоконтролер: к.т.н., доц. Гулак Н.К.

Київ 2020

ВСТУП

Актуальність. Нові ІТ технології активно впроваджуються в усі сфери народного господарства. Поява глобальних і локальних мереж передачі даних надала користувачам комп'ютерів нові можливості для оперативного обміну інформацією. Розвиток Internet привів до використання глобальних мереж передачі даних в повсякденному житті практично кожної людини. У міру розвитку і ускладнення засобів, методів і форм автоматизації процесів обробки інформації підвищується залежність суспільства від міри безпеки використовуваних їм ІТ.

Інформаційна безпека має три основні складові: цілісність, доступність і конфіденційність. Цілісність означає захист точності і повноти інформації і програмного забезпечення. Конфіденційність належить до захисту чутливої інформації від несанкціонованого доступу. Доступність – це забезпечення доступності інформації і основних послуг для користувача в потрібний для нього час. Враховуючи актуальність інформаційної безпеки було вирішено створити інформаційну систему, що аналізує спираючись на поточний стан обчислювальної системи визначає, які типи атак було проведено на неї.

Захист інформації є однією з вічних проблем. Протягом історії способи розв'язання цієї проблеми визначались рівнем розвитку технологій. У сучасному інформаційному суспільстві технологія відіграє роль активатора цієї проблеми — комп'ютерні злочини стали характерною ознакою сьогодення. Комп'ютерними називають злочини, пов'язані з втручанням у роботу комп'ютера, і злочини, в яких комп'ютери використовуються як необхідні технічні засоби. Серед причин комп'ютерних злочинів і пов'язаних з ними викрадень інформації головними є такі:

- швидкий перехід від традиційної паперової технології зберігання та передавання інформації до електронної за одночасного відставання технологій захисту інформації, зафіксованої на машинних носіях;

• широке використання локальних обчислювальних мереж, створення глобальних мереж і розширення доступу до інформаційних ресурсів;

• постійне ускладнення програмних засобів, що викликає зменшення їх надійності та збільшення кількості уразливих місць. Сьогодні ніхто не може назвати точну цифру загальних збитків від комп'ютерних злочинів, але експерти погоджуються, що відповідні суми вимірюються мільярдами доларів. Серед основних статей варто виокремити такі:

• збитки, до яких призводить ситуація, коли співробітники організації не можуть виконувати свої обов'язки через непрацездатність системи (мережі);

• вартість викрадених і скомпрометованих даних;

• витрати на відновлення роботи системи, на перевірку її цілісності, на доробку уразливих місць тощо. Варто також враховувати й морально-психологічні наслідки для користувачів, персоналу і власників ІС та інформації. Що ж до порушення безпеки так званих «критичних» додатків у державному і військовому управлінні, атомній енергетиці, медицині, ракетно-космічній галузі та у фінансовій сфері, то воно може призвести до тяжких наслідків для навколишнього середовища, економіки і безпеки держави, здоров'я і навіть для життя людей.

Метою роботи є удосконалення програмного модулю для підвищення захисту серверу від мережевих атак за рахунок створення програмного продукту мовою програмування PHP яке надає можливість підвищити захист інформації на сервері.

Досягнення мети потребує розв'язання таких **задач**:

- аналіз сучасних технологій для взаємодії клієнта з сервером;
- проаналізувати існуючі залежності в програмній платформі PHPStorm;
- на основі аналізу визначити загрози серверу при мережевих атаках;
- розробка програмного продукту на основі мови програмування

PHP та програмної платформи PHPStorm.

Під ефективність будемо розуміти, що підвищується конфіденційність, цілісність користувачів , які будуть використовувати даний продукт.

Об'єкт дослідження: — інформаційний процес, відносно яких необхідно забезпечувати захист відповідно до поставленої мети захисту інформації.

Предмет дослідження: методи та засоби забезпечення конфіденційності та цілісності інформації.

Оцінка сучасного стану проблеми на основі вітчизняної та зарубіжної літератури. Атаки на бази даних кожний день тільки зростають, яскравим прикладом може послугувати злом баз даних виборців в США в штатах Арізона й Іллінойс. Найбільш важливими є бази даних банків, злом яких гарантує пропажу великої кількості грошей відвідувачів цього банку. Тому захист серверів й персональних даних є важливою темою будь якого веб-ресурсу.

Галузь застосування. Даний програмний продукт може використовуватись у компаній для захисту серверів й персональних даних їх співробітників, клієнтів.

Новизна одержаних результатів полягає в наступному:

На основі язика програмування PHP було вирішено проблему захисту сервера від мережевих атак не використовуючи ніяких налаштувань серверу.

Практична цінність. полягає у тому, що компанії зможуть використовувати цей програмний продукт для зберігання даних всіх своїх користувачів на основі симетрочного шифрування метода Blowfish. У разі злому бази компанії, зловмисники не отримають одразу всі конфіденційні дані компанії. Крім того їх сервер буде захищено від мережевих атак за допомогою різноманітних фільтрів. Через те, що цей програмний продукт написаний на мові PHP, кожен веб-ресурс, може скористатися їм.

Апробація. Основні положення роботи доповідалися та обговорювалися на конференції. Матеріали XVI науково-практичної конференції «Наука и іновація», 07.10.2020 р. -15.10.2020 р., Польща.

Розділ 1. НОРМАТИВНО ПРАВОВІ АКТИ

Для успішного й вдалого використання сучасних технологій необхідне ефективне та надійне управління не тільки інформаційними системами, але й також засобами їх безпеки, тому наразі виникає необхідність забезпечити інформаційну безпеку корпоративних процесів діяльності підприємств.

1.1.ISO/IEC 27005:2008

1.1.1.Обмеження за стандартом ISO/IEC 27005:2007

В стандарті ISO/IEC 27005:2007 були надані рекомендації для менеджменту риска ІБ в установах, фірмах, підприємствах, тощо.

Для створення й ефективного функціонування системи менеджменту інформаційної безпеки необхідно визначити області застосування і межі процесу менеджменту ризиків ІБ [3].

Дослідження організації представлено в таблиці 1.1.

Таблиця 1.1.

Параметри дослідження організації

Параметр дослідження	Зміст параметру
1.	2.
Оцінка організації	Дослідження організації виявляє характеристики елементів, що ідентифікують організації. Це стосується намірів, бізнесу, місій, значень і стратегій цієї організації. Вони повинні бути ідентифіковані разом з елементами їх розробки (наприклад, висновок субпідрядної договору). Труднощі цієї діяльності полягає в точному розумінні як структурована організація. Ідентифікація еѐ реальної структури забезпечить розуміння ролі і важливості кожного підрозділу в досягненні цілей організації.

Продовження таблиці 1.1.

1.	2.
Основні наміри організації	Основні наміри організації можуть бути визначені як мотив, навіщо вона існує (її поле діяльності, її ринковий сегмент, тощо).
Бізнес організації	Бізнес організації, що визначається службовцями і методиками ноу-хау дає можливість досягти своєї місії. Це є визначальним для поля діяльності організації .
Місія організації	При досягненні організацією своїх намірів досягається її місія. Щоб ідентифікувати місію повинні бути ідентифіковані надані послуги та / або виготовлені продукти відносно кінцевих користувачів.
Значення організації	Головний принцип значень - читкий кодекс поведінки ставився до здійснення бізнесу. Це може торкнутися персоналу, відносин із зовнішніми агентами (клієнтами, тощо), якістю продуктів або наданих послуг.
Структура організації	1.дрібна структура: кожен розділ знаходиться під владою менеджера підрозділу, відповідального за стратегічні, адміністративні та оперативні рішення щодо його модуля; 2. функціональна структура: функціональна влада здійснена на процедурах, природі роботи і іноді рішень або планування (наприклад, продукція, ІТ, людські ресурси, маркетинг і т.д.);
Організаційна структура	Структура організації представлена схематично в організаційній структурі. це подання має виділити направлення повідомлень і делегацію повноважень, але має також включати інші відносини, які навіть якщо вони не засновані на формальних повноважень є, проте, напрямками потоку інформації.
Стратегія організації	Це вимагає формального вираження керівних принципів організації. Керівництво визначає стратегію організації і необхідну розробку, щоб витягти вигоду з проблем загроз і планованих головних змін

Для здійснення цілого комплексу дій, які спрямовані на управління системою менеджменту інформаційної безпеки шляхом уніфікації і підвищення ефективності процесів ІБ необхідно читко визначити обмеження,

що зачіпають організацію і визначають орієнтацію її інформаційної безпеки [3]. Список обмежень наведено в таблиці 1.2.

Організація встановлює свої цілі, що фіксуються до конкретної мети. Ця ціль може бути виражена як стратегія дії або розробки з ціллю зменшення експлуатаційних витрат, покращення якості сервісу, тощо.

Таблиця 1.2.

Перелік обмежень при виборі СМІБ

Тип обмеження	Причини обмеження
1.	2.
Обмеження політичної природи	Вони можуть торкнутися урядових адміністрацій, громадських установ або в ширшому розумінні будь-якої організації, яка повинна застосовувати урядові рішення. Це рішення зазвичай відносно стратегічною або оперативної орієнтації, які повинні бути застосовані підрозділом урядових органів для прийняття рішень.
Обмеження стратегічної природи	Обмеження можуть з'явитися результатом запланованих або можливих змін до структур організації або орієнтації. Вони виражені в стратегічних або оперативних планах організації.
Територіальні обмеження	Структура організації і / або мета можуть ввести визначені обмеження, такі як розподіл сайтів по всій національній території або за кордоном.
Обмеження, що є результатом економічного та політичного клімату	Операційна діяльність організації може бути глибоко змінена специфічними подіями, такими як потрясіння або національні та міжнародні кризи
Структурні обмеження	Природа структури організації (дробової, функціональної або інший) може привести до певної організаційної політики безпеки і організації безпеки, адаптованої до структури.
Функціональні обмеження	Функціональні обмеження виникають безпосередньо із загальних або певної місій організації.
Обмеження відносно персоналу	Природа цих обмежень значно змінюється. Вона пов'язана з: рівнем відповідальності, вербуванням, кваліфікацією, навчанням, розумінням безпеки, спонуканням, доступністю, тощо.
Обмеження, що виникають в	Ці обмеження можуть слідувати з реструктурування або установки нової національної

результаті реєстрації організації	або міжнародної політики, накладає певні обмеження.
Обмеження, що пов’язані з методами	Методи, відповідні ноу-хау організації повинні будуть бути накладені для аспектів, таких як проектне планування, специфікації, розробка і так далі.
	Продовження
Обмеження культурної природи	Деякі звички роботи в організації привели до певної "культури" в межах організації, яка може бути несумісна з менеджментом безпеки. Ця культура - загальна структура довідкової інформації персоналу і можна визначити багатьма аспектами, включаючи освіту, машинну команду, професійний досвід, т.д.
	Продовження таблиці 1.2.
1.	2.
Бюджетні обмеження	У рекомендованих контролів безпеки може іноді бути дуже висока вартість. Не завжди доречно будувати інвестиції безпеки на рентабельності, фінансовим підрозділом організації взагалі потрібно економічного виправдання.
Обмеження, що є результатом існуючих раніше процесів	Прикладні проекти не обов'язково розроблені одночасно. Деякі залежать від існуючих раніше процесів. Навіть при тому, що процес може бути розподілен на підпроцеси, процес не обов’язково знаходиться під впливом всіх 9ід процесів іншого процесу.
Технічні обмеження	технічні обмеження взагалі являются результатом встановлених апаратних засобів, програмного забезпечення, ділянок пам'яті або сайтів, що поміщають процеси: - файлов; -загальної архітектури; -прикладного програмного забезпечення; -упаковки програмного забезпечення; -апаратних засобів; -мереж комунікації; -вбудування інфраструктур.
Фінансові обмеження	Реалізація контролів безпеки обмежується бюджетом, який може передати організація. Однак, фінансове обмеження, яке розглянуть, має всі ещѐ бути продовженим, оскільки про розподіл бюджету для безпеки можна домовитися на основі дослідження безпеки.

Екологічні обмеження	Екологічні обмеження є результатом географічної або економічної обстановки, в якій здійснені процеси: країна, клімат, природні ризики, географічна ситуація, економічний клімат і т.д.
Часові обмеження	Потрібно розглянути час, необхідний для того, щоб здійснити менеджмент безпеки, щодо спроможності модернізувати інформаційну систему; якщо час реалізації дуже тривалий, ризики для яких був спроектований менеджмент, можливо, зміниться. Час - коефіцієнт визначення для того, щоб вибрати рішення і пріоритети.
Обмеження за методами	Методи, відповідні ноу-хау організації, повинні використовуватися для проектного планування, специфікацій, розробка і так далі.

1.1.2. Ідентифікація та визначення цінності активів

Можна відрізнити два види активів (рис. 1.1). Щоб визначити цінність активу, організація спочатку повинна здійснювати ідентифікацію свої активи [3].

Рис. 1.1. Види активів

Зазвичай первинні активи - це основні процеси та інформаційна діяльність в області застосування. Можна також розглядати інші первинні активи, такі як процеси всередині організації, які будуть більш відповідними для складання політики інформаційної безпеки або плану безперервності бізнесу. Залежно від мети деякі дослідження не вимагатимуть вичерпного аналізу всіх елементів, складових сферу застосування. У таких випадках межі дослідження можуть бути обмежені ключовими елементами області застосування[3].

Типи первинних активів представлені в таблиці 1.3

Таблиця 1.3.

Типи та зміст первинних активів

Тип активу	Зміст активу
Бізнес-процеси	-втрата яких або деградація, позбавляють можливості виконувати місію організації; - які містять секретні процеси або процеси, що залучають приватну технологію; - які, якщо змінені, можуть сильно зачепити виконання місії організації; - які необхідні для організації, щоб виконати договірні, юридичні або регулюють вимоги.
Інформація	Більш широко, первинна інформація включає головним чином: - життєво важливу інформацію для здійснення місії або бізнесу; - персональну інформацію, яка може бути визначена державою щодо права приватного життя; - стратегічну інформацію для досягнення цілей, определѐнную стратегічними орієнтаціями; - інформацію високу вартість на збір якої, зберігання, обробку та передачу потрібно багато часу і / або залучення високих фінансових витрат.

1.1.3. Оцінки активів організацій.

В аналізі повинні бути ретельно оцінені, визначені або призначені значення на актив, так як кінцеве призначений значення виступає в визначення ресурсів, які будуть витрачені для захисту активу.

Критерії, які можуть використовуватися, щоб оцінити можливі наслідки представлені на рисунку 1.2. Критерії поділяються на два типа:

1.критерії з втрати конфіденційності, цілісності, доступності, неможливість відмови від авторства, спостережливості, автентичності або надійності активів;

2. критерії оцінки наслідків.

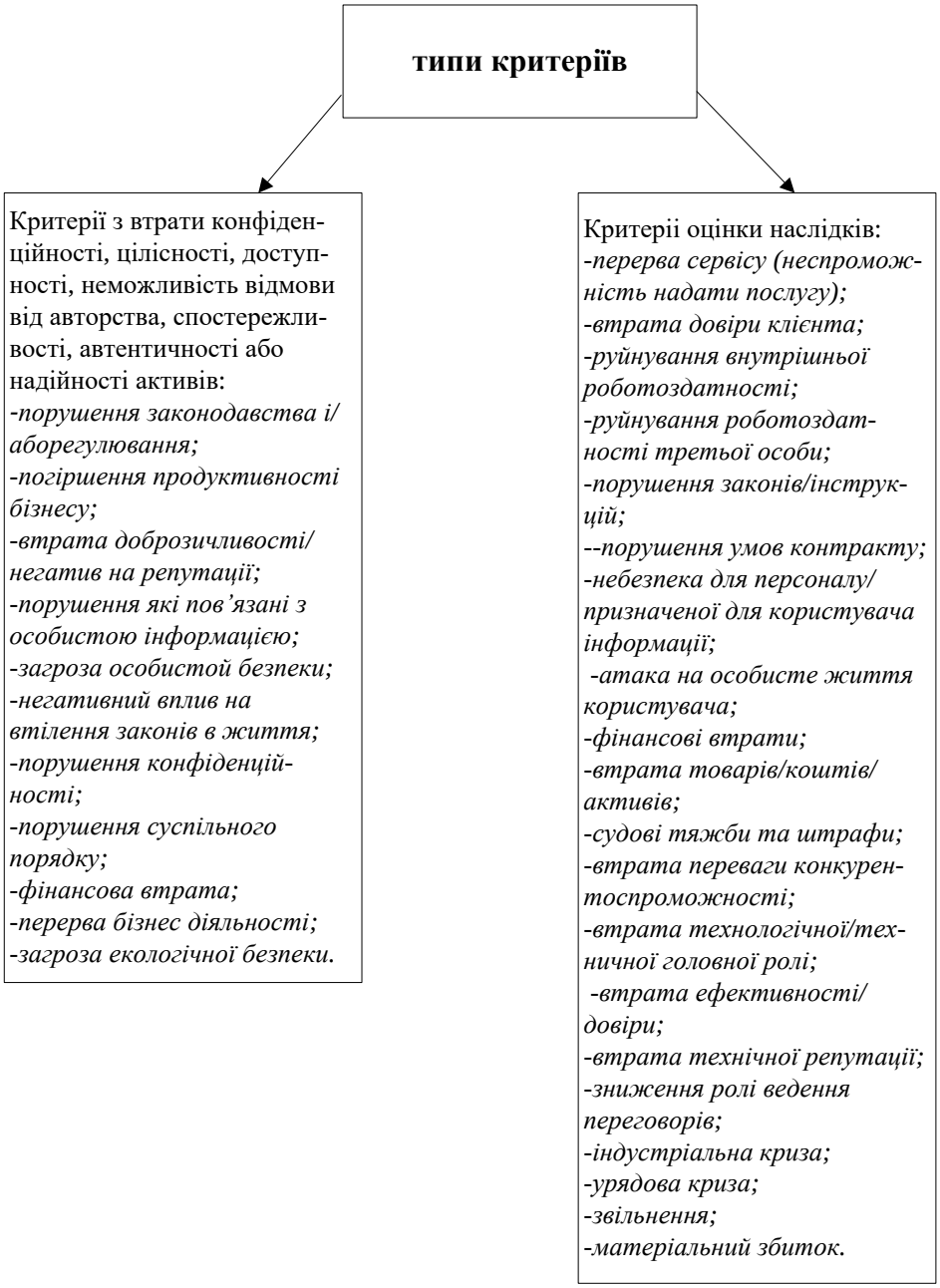

Рис. 1.2. Типи критеріїв оцінки збитків

Ці критерії - приклади проблем, які розглядаються для оцінки активу. Для того щоб виконати оцінки, організація повинна обрати критерії, які стосуються її типу вимог безпеки і бізнесу.

1.1.3. Оцінка впливу

Інцидент інформаційний безпеки може впливати на більше ніж один актив або тільки частина активу. Вплив пов'язано певною мірою успіху інциденту. Як наслідок, є суттєва відмінність між значенням активу і впливом, що випливають з інциденту. Впливом вважають наявність або безпосередній (експлуатаційний) ефект або майбутній (бізнес) ефект, який включає фінансові та ринкові наслідки. Безпосереднє (експлуатаційне) вплив є прямим або непрямим. Зміст цих впливів наведено на рисунку 1.3.

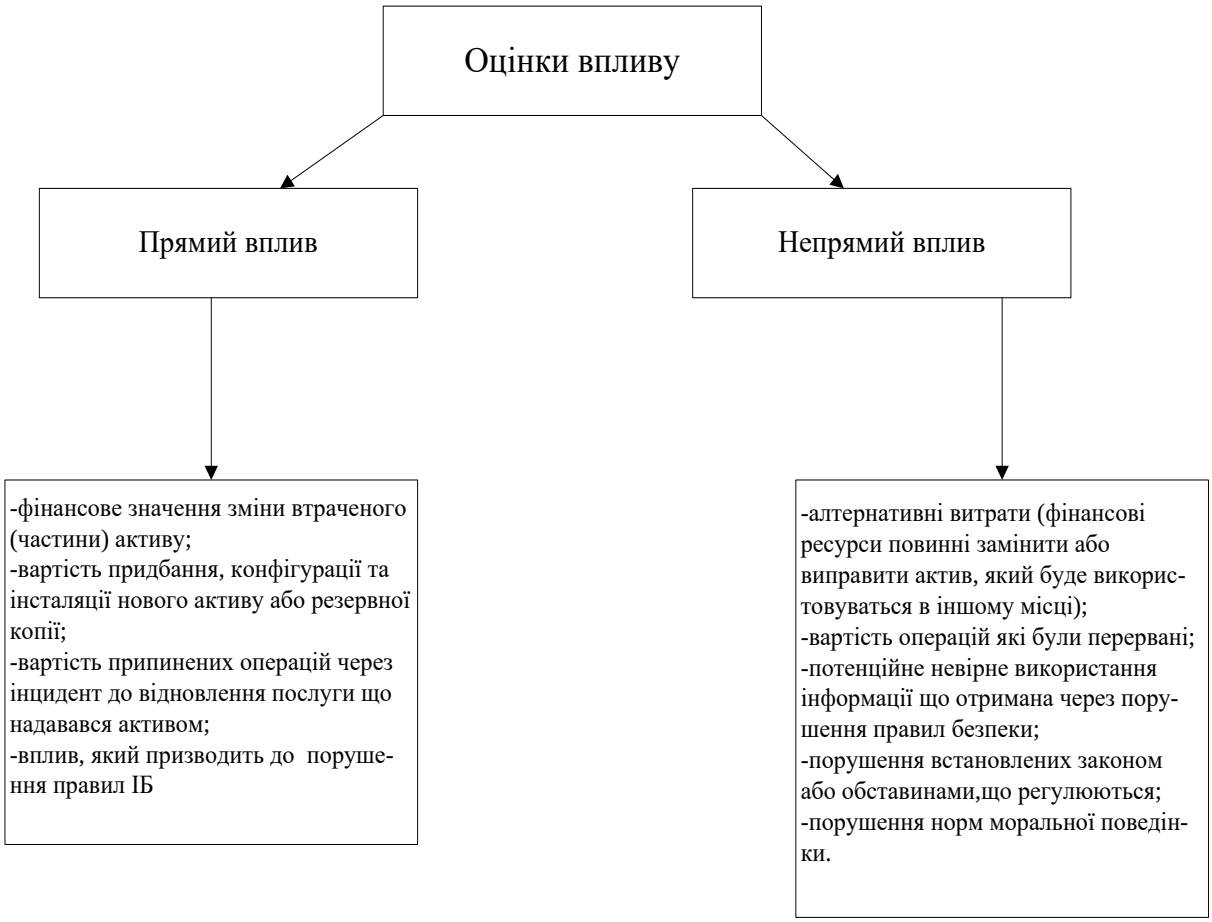

Рис. 1.3. Оцінки впливу.

1.2. Класифікація загроз за стандартом ISO/IEC 27005:2008

Загрози можуть бути навмисними, випадковими або екологічними (природними) і мати результат, наприклад, порушення або втрата основних сервісів. Наступний список показує релевантні для кожного типу загрози, де D (навмисні), A (випадкові елемент), E (екологічні). D використовується для

всіх навмисних акцій, націлених на інформаційні активи, А використовується для всіх людських дій, які можуть випадково пошкодити інформаційні активи і Е використовується для всіх інцидентів, які не засновані на людських діях (табл.1.4). Групи загроз не знаходяться в пріоритетному порядку [3].

Таблиця 1.4.

Класифікація загроз

Тип	Загрози	Позначення
1.	2.	3.
Фізичне пошкодження	Вогонь	A,D,E
	Пошкодження водою	A,D,E
	Забруднення	A,D,E
	Значний інциндент	A,D,E
	Знищення обладнання або носіїв	A,D,E
	Пил, корозія, обмерзання	A,D,E
Природні події	Кліматичні явища	E
	Сейсмічні явища	E
	Вулканічні явища	E
	Метеорологічні явища	E
	Повінь	E
Втрата необхідних сервісів	Відмова кондиціону-вання та водопостачання	A,D
	Втрачання електроенергії	A,D,E
	Відмова телекомунікаційного обладнення	A,D
Радіаційні несправності	Електромагнітна радіація	A,D,E
	Теплова радіація	A,D,E
	Електромагнітний імпульс	A,D,E
Компрометація інформації	Перехоплення і відправка	D

	компрометувати сигналу	
	Віддалений шпіонаж	D
	Підслуховування	D
	Крадіння носіїв або документів	D
	Крадіння обладнання	D

Продовження таблиці 1.4.

1.	2.	3.
Компрометація інформації	Відновлення інформації на носіях, відправлених на переробку або бракованих	D
	Виявлення	A,D
	Дані з ненадійних джерел	A,D
	Втручання в апаратні засоби	D
	Втручання в програмні засоби	D
	Виявлення позицій	D
Технічні відмови	Відмова обладнання	A
	Збій обладнання	A
	Обмеження інформаційної системи	A,D
	Програмний збій	A
	Порушення ремонтопридатності інформаційної системи	A,D
Несанкціоновані дії	Несанкціоноване використання обладнення	D
	Шахрайське копіювання програмного забезпечення	D
	Використання підробленого або скопійованого	A,D

	програмного забезпечення	
	Спотворення даних	D
	Незаконна обробка даних	D
Компрометація функцій	Помилка в використанні	A
	Зловживання правами	A,D
	Підробка прав	D
	Заперечення дій	D

Особливу увагу потрібно звернути на людський фактор, який відігріває не останню роль як джерело загрозі інформації (табл. 1.5.) [3].

Таблиця 1.5.

Мотивація людських загроз

Джерело загрози	Мотивація	Можливі наслідки
Хакер, крекер	Повстання Его Визов Статус Гроші	Хакерство Соціальна інженерія Вторгнення в систему, порушення Несанкціонований доступ в систему
Комп’ютерний злочинець	Руйнування інформації Незаконне розкриття інформації Грошово-кредитна вигода Неправомірне чергування даних	Комп'ютерний злочин (наприклад, кібер-переслідування) Шахрайське дія (наприклад, перегравання, наслідування, перехоплення) Інформаційне хабарництво Імітація Вторгнення в систему
Терорист	Помста Розвідка Руйнування Шантаж Політичні вигоди Освітлення в пресі	Бомба / Тероризм Інформаційна війна Системна атака (наприклад, распределѐнний відмову в обслуговуванні) Проникнення в систему Втручання в систему
Індустріальне шпигунство	Розвідка Его Цікавість	Вторгнення в систему Шантаж Перегляд секрету фірми

(компанії, іноземні уряди, інші урядові інтереси)	Грошово-кредитна вигода Помста Ненавмисні помилки і Упущення (наприклад, помилка введення даних, помилка програмування)	Неправильне комп'ютерне звернення Системний саботаж Інформаційне хабарництво Введення фальсифікованих даних, руйнування даних Перехоплення Шкідливий код (наприклад, вірус, логічна бомба, троянський кінь) Несанкціонований доступ в систему Системні помилки

В наступному підрозділі буде розглянуто вразливості ІБ.

1.3. Вразливості та методи їх оцінки.

Людський фактор є одним з слабких ланок, так як піддається емоціям. Людські джерела загрози відповідно перераховані в таблиці 1.6.[3]:

Таблиця 1.6.

Фактори людських загроз

Джерело загроз	Мотивація	Можливі наслідки
1.	2.	3.
Хакер, крекер	Повстання Его Виклик Статус Гроші	Хакерство Соціальна інженерія Вторгнення с систему, порушення Несанкціонований доступ в систему
Комп’ютерний злочинець	Порушення інформації Незаконне розкриття інформації Грошово-кредитна вигода Неправомочне чередування да-них	Комп'ютерний злочин Шахрайська дія Інформаційне хабарництво Імітація, вторгнення в систему
Продовження таблиці 1.6.		
1.	2.	3.
Терорист	Помста Розвідка Руйнування	Бомба/Тероризм Інформаційна війна Системна атака

	Шантаж Політичні вигоди Освітлення у пресі	(наприклад, розподілѐна відмова в обслуговуванні) Проникнення в систему Втручання в систему
Індустріальне шпигунство (компанії, іноземні уряди, інші урядові інтереси)	Конкурентоздатна перевага Економічне шпигунство	Економічна розвідка Інформаційна крадіжка Вторгнення в персональні дані Соціальна розробка Проникнення в систему Несанкціонований доступ в систему (доступ до секретної, приватної, і/або пов'язаній з технологією інформації)
Інсайдер (погано навчені, розсерджені, зловмисні, недбалі, нечесні або звільнені службовці)	Розвідка Его Цікавість Грошово-кредитна вигода Помста Ненавмисні помилки і упущення (наприклад, помилка введення даних, помилка програмування)	Напад на службовця Шантаж Перегляд секрету фірми Неправильне комп'ютерне поводження Шахрайство і крадіжка Інформаційне хабарництво Введення фальсифікованих даних, руйнування даних Перехоплення Шкідливий код (наприклад, вірус, логічна бомба, троян-ський кінь) Продаж персональної інформації Системні помилки Вторгнення в систему Системний саботаж Несанкціонований доступ в систему

1.4. Уразливості і методи для оцінки уразливості

У комп'ютерній безпеці, уразливість — це нездатність системи протистояти реалізації певної загрози або сукупності загроз [4]. Тобто, це недоліки в комп'ютерній системі, завдяки яким можна навмисно порушити її цілісність і викликати невірну роботу.

Уразливість виникає в результаті допущених помилок програмування, недоліків, допущених при проектуванні системи, ненадійних паролів, вірусів та інших шкідливих програм, скриптових і SQL-ін'єкцій. Деякі уразливості відомі тільки теоретично, інші ж активно використовуються і мають відомі експлойти.

Наведемо приклади вразливості в різних областях безпеки включаючи приклади загроз, які могли б експлуатувати цю уразливість. Перелік може забезпечити довідку під час оцінки загроз і уразливості, визначити відповідні інцидентні сценарії (табл.1.7.). В деяких випадках інші загрози можуть експлуатувати також цю уразливість [3].

Таблиця 1.7.

Приклади вразливості та загроз

Тип	Приклад вразливості	Приклади загроз
1.	2.	3.
Апаратні засоби	Недостатнє обслуговування / дефектна інсталяція з носіїв даних	Пролом в ремонтопридатності інформаційної системи
	Вади схем для періодичних замін	Руйнування устаткування або носіїв
	Сприйнятливість до вологості, пилу, забрудненню	Пил, корозія, обмерзання
	Чутливість до електромагнітної радіації	Електромагнітна радіація
	Вади ефективного контролю внесення змін конфігурації	Помилка у використанні
	Сприйнятливість до змін напруги	Втрата джерела живлення
	Сприйнятливість до температурних змін	Метеорологічне явище

Продовження таблиці 1.7.

1.	2.	3.
Апаратні засоби	Незахищіне зберігання	Крадіжка носіїв або документів
	Недолік в обережності при знищенні	Крадіжка носіїв або документів
	Неконтрольоване копіювання	Крадіжка носіїв або документів
Програмне забезпечення	Відсутність або недостатнє програмне тестування	Зловживання правами
	Відомі недоліки в програмному забезпеченні	Зловживання правами
	Немає виходу з системи при залишенні робочої станції	Зловживання правами
	Передача або багатократне використання носіїв даних без належного стирання	Зловживання правами
	Мале число ревізій	Зловживання правами
	Неправильний розподіл прав доступу	Зловживання правами
	Широко розповсюджене програмне забезпечення	Спотворення даних
	Застосування застосовних програм до фальшивих даних в термінах часу	Спотворення даних
	Складний призначений для користувача інтерфейс	Помилка у використанні
	Вади в документуванні	Помилка у використанні
	Встановлено невірний параметр	Помилка у використанні
	Некоректні дати	Помилка у використанні
Мережа	Вади ідентифікуючих і розпізнавальних механізмів для призначеної для користувача аутентифікації	Підробка прав
	Незащищѐнные таблиці паролів	Підробка прав
	Поганий менеджмент паролями	Підробка прав
	Запущені непотрібні служби	Незаконна обробка даних
	Недопрацьоване або нове програмне забезпечення	Програмний збій

Продовження таблиці 1.7.		
1.	2.	3.
Мережа	Неясні або неповні специфікації для розробників	Програмний збій
	Вада ефективний контролю внесення зміна	Програмний збій
	Неконтрольоване завантаження і використання програмного забезпечення	Підробка програмного забезпечення
	Вади в процедурі резервного копіювання	Підробка програмного забезпечення
	Вади фізичного захисту будівлі, дверей і вікон	Крадіжка носіїв або документів
	Відмова менеджменту від перевірки звітів	Несанкціоноване використання устаткування
	Нестача доказу посилки або отримання повідомлення	Заперечення дій
	Незахищена лінія зв'язку	Підслуховування
	Незахищений чутливий трафік	Підслуховування
	Погана спільна проводка	Відмова телекомунікаційного устаткування
	Єдина точка відмови	Відмова телекомунікаційного устаткування
	Вади ідентифікації і аутентифікація відправника і одержувача	Підробка прав
	Небезпечна мережева архітектура	Віддалене шпигунство
	Передача паролів у відкритому виді	Віддалене шпигунство
	Неадекватний менеджмент мережею (здатність системи протистояти помилкам маршрутизації)	Насиченість інформаційної системи
	Незащищѐнные підключення загальнодоступної мережі	Несанкціоноване використання устаткування
Персонал	Відсутність персоналу	Порушення доступності персоналу

	Неадекватні процедури вербування	Знищення устаткування або носіїв недостатнє навчання безпеки
	Помилка у використанні неправильне використання програмного забезпечення	Помилка у використанні
		Продовження таблиці 1.7.
1.	2.	3.
Персонал	Вади розуміння безпеки	Помилка у використанні
	Нестача механізмів моніторингу	Незаконна обробка даних
	Неконтрольована робота зовнішнім штатом або персоналом що прибирає	Крадіжка носіїв або документів
	Вада політики для вірного використання носіїв передачі даних і обміну повідомлень	Несанкціоноване використання устаткування
Сайт організації	Неадекватне і недбале використання фізичного контролю доступу до будівлі і приміщень	Знищення устаткування або носіїв інформації
	Місце розташування в області, сприйнятливій до затоплення	Нестабільна потужність мережі
	Повінь	Втрата джерела живлення
	Нестача фізичного захисту створення, дверей і вікон	Крадіжка устаткування
	Вади формальної процедури для призначений для користувача реєстрації і де-регистрації	Зловживання правом

1.5 Список нормативних документів щодо інформаційної безпеки в Україні.

- Закон про інформацію від 02.10.1992 № 2657-XII.[3]
- Закон про основи національної безпеки України .[3]
- Закон України про захист інформації в інформаційно - телекомунікаційних системах.[3]
- Закон України про захист персональних даних від 01.06.2010 №2997-VI.[3]
- НД ТЗІ 1.1-002-99 Загальні положення щодо захисту інформації в компю’терних системах від несанціаонованого доступу[3]
- НД ТЗІ 2.5-010-03 Вимоги до захисту інформації WEB-сторінки від несанкціонованого доступу[3]
- ГСТУ СУІБ 2.0/ISO/IEC 27002:2010 Інформаційні технології. Методи захисту. Звід правил для управління інформаційною безпекою.(ISO/IEC 27003:2005, MOD)[3]

Розділ 2. СУЧАСНІ МЕТОДИ ЗАХИСТУ ІНФОРМАЦІЇ ТА ШИФРУВАННЯ

2.1 Захист інформації в Інтернеті

DARPA по завданню міністерства оборони США приступило до проекту по створенню експериментальної мережі передачі пакетів. Ця мережа, названа ARPANET, призначалася на початку для вивчення методів забезпечення надійного зв'язку між комп'ютерами різних типів. Багато методів передачі даних через модеми були розроблені в ARPANET. Тоді ж були розроблені і протоколи передачі даних у мережі - TCP / IP. TCP / IP - це безліч комунікаційних протоколів, які визначають, як комп'ютери різних типів можуть спілкуватися між собою.[1,4]

Експеримент із ARPANET був настільки блискучий, що багато організацій захотіли увійти в неї, з метою використання для щоденної передачі даних. ARPANET перетворилася з експериментальної мережі в робочу мережу. Відповідальність за адміністрування мережі взяло на себе DCA, в даний час зване DISA. Але розвиток ARPANET на цьому не зупинилися; Протоколи TCP / IP продовжували розвиватися й удосконалюватися.

Один із перших стандартів для протоколів TCP / IP, що ввійшов у Military Standarts, тобто у військові стандарти, і всі, хто працював у мережі, зобов'язані були перейти до цих нових протоколів. Для полегшення переходу DARPA звернулася з пропозицією до керівників фірми Berkley Software Design - упровадити протоколи TCP / IP у UNIX. З цього і почався союз UNIX і TCP / IP.[4]

Через деякий час TCP / IP був прийнятий у звичайний, тобто в загальнодоступний стандарт, і термін Internet увійшов у загальний ужиток, з ARPANET виділилася MILNET, що стала відноситися до DDN міністерства оборони США. Термін Internet став використовуватися для позначення єдиної мережі: MILNET плюс ARPANET. І хоча ARPANET припинила

своє існування, мережа Internet існує, її розміри набагато перевищують початкові, тому що вона об'єднала безліч мереж в усьому світі. Хостом в мережі Internet називаються комп'ютери, що працюють в багатозадачній операційній системі (Unix, VMS), підтримують протоколи TCPIP і надають користувачам які-небудь мережеві послуги.

Новини про спостереження за громадянами, які поширили світ завдяки Едварду Сноудену, сприяли величезному зростанню популярності шифрування. Тим часом, методи таємного передавання конфіденційної інформації тривалий час супроводжували людей.

2.2 Загрози програмних модулів на сервері

Дослідження безпеки інформації в складі державної інформаційної політики припускає вирішення цілого комплексу питань загальнонаукової властивості. При цьому стає важливим не лише з’ясування сутнісних рис, функцій державної інформаційної політики в області інформаційної безпеки, але і виявлення основних чинників, що впливають на неї. Науковці, використовуючи міждисциплінарний комплексний підхід при розробці різних аспектів проблеми безпеки, позитивний світовий і вітчизняний досвід її забезпечення, у своїх роботах розширили дослідне поле, запропонували рекомендації щодо зміцнення безпеки країни. Оскільки питання національної безпеки України є предметом окремих досліджень, то у даній роботі наша увага буде зосереджена на розгляді інформаційної безпеки як складової державної інформаційної політики. За даними статистики WASC (Web Application Security Consortium), більше 13% сайтів можуть бути скомпрометовані повністю автоматично, 80-96% з яких мають високий ступінь вразливості, 86% – середній ступінь вразливості, 37% – низький . Літературні джерела видані з інтервалом майже в 10 років, наочно ілюструють зміни в підходах до захисту Web-ресурсів. [4]Проаналізувавши наукову та спеціалізовану літературу було визначено, що всі загрози інформаційної безпеки Web-додатків можна розділити на кілька категорій:

1. Міжсайтовий скриптинг – XSS (атака на користувача спрямована на виконання в його браузері довільного сценарію) - впровадження шкідливого JavaScript–коду на сторінку Webсистеми, яка піддається атаці . При завантаженні сторінки браузер автоматично виконує JavaScript-код, збираючи дані автентифікації входу і відправляючи їх на сайт зловмисника.

2. SQL-ін'єкція – впровадження шкідливого SQL-коду в тіло HTTPзапиту до Webдодатку . Коли, в URL рядку можна виконати, за рахунок підміни SQL-запиту до бази даних і дізнатися пароль адміністратора. SQL-ін'єкція типу: «http: //localhost/path/user.php? id = -2 + UNION + SELECT + 1,2,3,4,5, concat (user_email, 0x3e, user_passwd), 7,8,9 , 10,11 + from + users- ».

3. CRLF-атака – техніка модифікації HTTP-заголовків запиту . Можна виділити 2 види: • CRLF-ін'єкція – використання ASCII-представлення комбінації CR + LF (перенесення каретки + новий рядок) для формування «шкідливих» URL. • Розшарування HTTP-запиту. З його допомогою зловмисник може сформувати URL, який підмінить собою відповідь сервера, а також, ініціювавши внутрішню помилку, можливо буде побачити як інформацію про сервер Webдодатків, так і службову інформацію.

4. XXE (XML eXternal Entity)-атаки. При валідації XML-документа парсером (об'єктно-орієнтованою скриптовою мовою програмування, створеною для генерації HTMLсторінок на Web-сервері з підтримкою CGI) за допомогою схеми DTD, всі її директиви обов'язково повинні бути виконані. Якщо правилами визначено, що у вхідному документі допускаються спецсимволи XML – ризик бути атакованим дуже великий.

5. CSRF (Cross Site Request Forgery) – міжсайтова підробка запитів . Дає можливість зловмиснику скористатися автентифікаційними даними відповідної особи (cookies) і провести від її імені будь-яку шкідливу операцію.

6. DDoS-атака – велика кількість зовнішніх запитів (часто безглуздих або неправильно сформульованих) з різним трафіком (за різними портами), що викликає переповнення пам'яті та подальшу відмову в обслуговуванні системи, за цей час можна безперешкодно отримати доступ до ресурсу і

отримати root . Таблиця 1 Web-атаки та способи запобігання Тип загрози Способи запобігання Міжсайтовий скриптинг – XSS Заборона на вкладення HTML-сторінок, екранування спецсимволів - <-, -> ‖, передача всіх cake з прапором HttpOnly. SQL-ін’єкція Повсюдне використання ORM в Web-додатку, клієнтська і серверна валідація даних, тестування Web-додатків, екранування «очищених» параметрів, що надходять в неперевірені SQLзапити, тестування додатка інструментами на знаходження потенційних SQL-ін’єкцій. CRLF-атака Обов’язкове кодування CRLF-послідовності до передачі в HTTPзаголовки, а також повне кодування переданих даних. XXE (XML eXternal Entity)-атаки Ретельне налаштування XML-парсерів, використання XML Schema замість DTD для валідуючого парсеру. CSRF (Cross Site Request Forgery) Встановлення HttpOnly-прапора передачі cookie, використання одноразових сесійних token і відправка прихованої форми, щоб token не можна було підробити, перевірка реферерів при HTTPзапити до Web-додатку. DDoS-атака Виходу з цієї ситуації фактично немає, однак наслідки DDoS-атак і їх ефективність можна істотно знизити за рахунок правильного налаштування маршрутизатора, брандмауера і постійного аналізу аномалій в мережевому трафіку. [5]

2.2.1 Захист від CSRF

Усі запити повинні захищатися, які змінюють дані на сервері, а також запити, які повертають персональні чи інші делікатні дані.

Найбільш простим способом захисту від даного типу атак є механізм, коли веб-сайти повинні вимагати підтвердження більшості дій користувача і перевіряти поле HTTP_REFERER, якщо воно вказане в запиті. Але цей спосіб може бути небезпечний, і використовувати його не рекомендується [5,6].

Іншим поширеним способом захисту є механізм, при якому з кожною сесією користувача асоціюється додатковий секретний унікальний ключ, призначений для виконання запитів. Секретний ключ не повинен передаватися у відкритому вигляді, наприклад, для POST-запитів ключ слід передавати в

тілі запиту, а не в адресі сторінки. Браузер користувача посилає цей ключ в числі параметрів кожного запиту, і перед виконанням будь-яких дій сервер перевіряє цей ключ. Перевагою даного механізму, в порівнянні з перевіркою Referer, є гарантований захист від атак CSRF. Недоліками ж є вимога можливості організації сесії користувачів і вимога динамічної генерації HTML-коду сторінок сайту.

Специфікація протоколу HTTP / 1.1 [6] визначає безпечні методи запитів, такі як GET, HEAD, які не повинні змінювати дані на сервері. Для таких запитів, при відповідно сервера специфікації, немає необхідності застосовувати захист від CSRF.

Може виникнути бажання підстрахуватися і додати ключ в кожен запит, але слід мати на увазі, що специфікація HTTP / 1.1 [6,7] допускає наявність тіла для будь-яких запитів, але для деяких методів запиту (GET, HEAD, DELETE) семантика тіла запиту не визначена і повинна бути проігнорована. Тому ключ може бути переданий тільки в самому URL або в HTTP-заголовку запиту. Необхідно захистити користувача від нерозсудливо поширення ключа в складі URL, наприклад, на форумі, де ключ може виявитися доступним зловмисникові. Тому запити з ключем в URL не слід використовувати в якості адреси для переходу, тобто виключити перехід на таку адресу клієнтським скриптом, перенаправленням сервера, дією форми, гіперпосиланням на сторінці і т. П. З метою приховування ключа, що входить в URL. Їх можна використовувати лише як внутрішні запити скриптом з використанням XMLHttpRequest або обгорткою, наприклад AJAX.

Суттєвим факт того, що ключ (CSRF-токен) може бути призначений не для конкретного запиту або форми, а для всіх запитів користувача взагалі. Тому досить витоку CSRF-токена с URL, що виконує просте дію або який не виконує дію зовсім, як захисту від підробки запиту позбавляється будь-яка дія, а не тільки те, з яким пов'язаний став відомим URL.

Існує більш жорсткий варіант попереднього механізму, в якому з кожним дією асоціюється унікальний одноразовий ключ. Такий спосіб складніший в реалізації і вимогливий до ресурсів. Спосіб використовується деякими сайтами та порталами, такими як Livejournal, Rambler і ін. На 2016 рік не було відомостей про перевагу більш жорсткого варіанту в порівнянні з варіантом, в якому використовується єдиний для кожної сесії секретний ключ [7].

2.2.1.1 Наслідки CRLF-ін'єкції

Наслідки CRLF-ін'єкції варіюються і можуть включати в себе всі наслідки використання XSS і розкриття конфіденційної інформації. Таким способом можна деактивувати деякі обмеження системи безпеки, такі як фільтри XSS і Same Origin Policy в браузерах жертви, роблячи їх вразливими до шкідливих атак.

2.2.1.2 Як запобігти CRLF / HTTP-ін'єкції заголовків в веб-додатках

Найкращий метод запобігання - це не використовувати введення даних користувачем безпосередньо в заголовок відповіді. Якщо це неможливо, ви завжди повинні використовувати функцію для кодування спеціальних символів CRLF. Ще одна хороша практика безпеки - це оновлення мови програмування до версії, яка не дозволяє робити ін'єкції CR і LF всередині функцій, які встановлюють HTTP-заголовки. Задля цього потрібно подивись таблицю вразливостей Рис. 2.1

Classification	ID / Severity
PCI v3.2	6.5.1
OWASP 2013	A1
OWASP 2017	A1
CWE	93
CAPEC	105
WASC	24
HIPAA	164.306(a), 164.308(a)
ISO27001	A.14.2.5
CVSS:3.0	CVSS:3.0: AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:L/A:H
Netsparker	Medium

Рис. — 2.1 —Таблиця класифікації вразливостей і їх ступеня тяжкості

2.2.2 Як запобігти XXE-атаки

Навчання розробників має велике значення для виявлення і пом'якшення наслідків XXE. Крім того, запобігання XXE зобов'язує:

- По можливості використовуйте менш складні формати даних, такі як JSON, і уникайте сериализации конфіденційних даних.
- Виправте або поновіть все процесори і бібліотеки XML, які використовуються додатком або в базовій операційній системі. Використовуйте перевірки залежностей. Оновлення SOAP до SOAP 1.2 або вище.
- Вимкніть зовнішню сутність XML і обробку DTD у всіх синтаксичних анализаторах XML в додатку відповідно до таблиці OWASP «Запобігання XXE».
- Реалізуйте позитивну («білий список») перевірку, фільтрацію або очищення вхідних даних на стороні сервера, щоб запобігти ворожі дані в документах, заголовках або вузлах XML.

- Переконайтеся, що функція завантаження файлів XML або XSL перевіряє вхідний XML з використанням перевірки XSD або аналогічної.
- Інструменти SAST можуть допомогти виявити XXE в вихідному коді, хоча ручна перевірка коду - найкраща альтернатива для великих і складних додатків з багатьма інтеграціями.
- Якщо ці елементи управління неможливі, розгляньте можливість використання віртуальних виправлень, шлюзів безпеки API або брандмауерів веб-додатків (WAF) для виявлення, моніторингу та блокування атак XXE

2.2.3 Способи захисту від DDos-атак.

DDos-атака не має ніякого відношення до операційної системи DOS.

Distributed Denial Of Service Attack – так виглядає повний варіант назви - це виведення сервера з ладу методом відправлення на нього лавини запитів - нехай і абсолютно безглуздих. Сервер не витримує навантаження і «падає». У зв'язку з цим перестають функціонувати, наприклад, сайти, які звертаються до даного сервера, бази даних і так далі.

DDos-атака - найбільш ефективний і поширений метод. Таким чином можна, наприклад, «обрушити» Інтернет-магазин конкурентів. Адже на його відновлення може піти чимало часу, а це - втрачена потенційний прибуток.При цьому хакери не мають на меті зламати сервер і отримати до нього доступ. Самі зловмисники, звичайно, не можуть організувати таку потужну лавину помилкових запитів: занадто багато ресурсів для цього доведеться задіяти. Атаку здійснюють комп'ютери звичайні користувачів, які, як правило, навіть не підозрюють про це. На машину жертви встановлюється спеціальне шкідливе програмне забезпечення, яке і включається в призначений час. Тисячі, а то і мільйони комп'ютерів по всьому світу починають посилати запити, а також заражати машини інших користувачів, так що кількість запитів при «сприятливому» збігу обставин може зростати в геометричній прогресії.

Видів DDos-атак існує безліч. Але краще знати, як від них захиститися.

- Завжди містити ПО для забезпечення безпеки сервера в актуальному стані;

- передбачати можливість резервного копіювання даних і перенесення їх на інший сервер;

- мати можливість віддалено перезавантажити будь-сервер в аварійній ситуації;

- мати в наявності запасний мережевий інтерфейс, за яким можна буде отримати доступ до атакується системі;

- переклад ресурсів в «хмару». Компанії, що спеціалізуються на хмарних технологіях і пропонують перенесення сервера в хмару, мають куди більш потужні засоби протидії хакерам, ніж підприємство малого чи середнього бізнесу;

- інструктувати співробітників, як вести себе в підозрілої ситуації. DDos-атака починається не миттєво і часто її можна перехопити ще на початковій стадії.

На жаль, але стовідсоткових методів сьогодні від DDos-атак не існує. Хоча б тому, що завжди потрібно брати до уваги горезвісний людський фактор. Навіть не так налаштований маршрутизатор або брандмауер може послужити джерелом бід. Тому до попередження DDos-атак потрібно підходити комплексно: з огляду на як програмні аспекти, так і апаратні і організаційні.

1. Передбачення. Часто DDos-атаки здійснюються або економічними конкурентами, або з релігійних, етнічних, політичних причин. Тому, якщо організація знаходиться в зоні ризику - краще заздалегідь передбачити ймовірність атаки.

2. Нарощування обчислювальної потужності. При дійсно серйозною DDos-атаці цей метод може не спрацювати, але, якщо запитів не надто багато або атака знаходиться на самому початку, краще мати потужний сервер, який буде чинити опір як можна довше. Зрештою, за цей час можна буде застосувати інші заходи протидії.

3. Атака. Суть методу в тому, що весь атакуючий трафік перенаправляється в зворотну сторону. Якщо в розпорядженні є досить потужний сервер, атака захлинеться і обернеться проти зловмисників.

4. Спеціальне ПО. На ринку існує маса спеціальних пропозицій з протидії DDos-атакам. Має сенс вивчити їх все, щоб вибрати найбільш підходящий варіант. Подібного роду ПО коштує недешево, але витрати окупляться при першій же критичної ситуації.

5. Розосередження серверів. «Не звалювати все яйця в одну корзину» - так можна охарактеризувати цей метод. Тобто рознести активні частини сервера з можливістю їх дублювання. Навіть якщо якась частина ресурсів виявиться недоступною, інша частина буде функціонувати.

6. Відведення активного IP-адреси або доменного імені від ресурсів, які можуть бути схильні до DDos-атакам.

7. Фільтрація і блокування трафіку. Часто при цьому важко відокремити «чистий» трафік від «поганого», але можна відсікати тільки другорядні запити. Втім, якщо зловмисник використовує першорядні запити, цей метод виявиться слабким захистом.

8. Оскільки сьогодні комп'ютерний тероризм досяг небувалого раніше розмаху і, схоже, в подальшому ситуація не покращиться, існує достатня кількість фірм, що пропонують вистежити джерело атаки і провести відповідні дії.

9. Ліквідація вразливостей. Відразу після відбиття атаки або навіть під час неї потрібно шукати і усувати уразливі місця в системі.

Корпорація Google пропонує всім, хто піддався DDos-атакам, продовжувати відображати вміст їхніх сайтів навіть під час самої атаки. Правда, в даний час цей сервіс знаходиться на стадії тестування, але в майбутньому планується ввести його як постійну

2.2.4 SQL ін'єкція як спосіб злому сайтів та програм

2.2.4.1 Визначення SQL ін’єкцій

SQL ін'єкція — один із способів злому сайтів та програм, що мають бази даних, заснований на впровадженні в запит довільного SQL-коду. Впровадження SQL, залежно від типу СКБД , може дати атакуючому виконати запит до бази даних, отримати можливість запису та читання локальних файлів та виконання довільних команд на сервері. Атака типу впровадження SQL може бути можлива за некоректної обробки вхідних даних, що використовуються в SQL-запитах.Розробник застосунків, що працюють з базами даних, повинен знати про таку уразливість і вживати заходів протидії впровадженню SQL.[14]

2.2.4.2 Принцип атаки

Припустимо, серверне ПЗ, отримавши вхідний параметр id, використовує його для створення SQL-запиту. Розглянемо такий PHP-скрипт:

```
$id = $_REQUEST['id'];
$res = mysql_query("SELECT * FROM news WHERE id_news = $id");
```

Якщо переданий параметр id на сервер дорівнює 5 (наприклад так: *http://example.org/script.php?id=5*), то виконається такий SQL-запит:

```
SELECT * FROM news WHERE id_news = 5
```

Але якщо зловмисник передасть як параметр id рядок -1 OR 1=1 (наприклад, так: *http://example.org/script.php?id=-1+OR+1=1*), то виконається запит:

```
SELECT * FROM news WHERE id_news =-1 OR 1=1
```

Зміна вхідних параметрів шляхом додавання в них конструкцій мови SQL викликає зміну в виконанні SQL-запиту.[14]

Впровадження в рядкові параметри

Серверне ПЗ, отримавши запит на пошук даних у новинах параметр search_text, використовує його в наступному SQL-запиті (тут параметри екрануються лапками) :

```
$search_text = $_REQUEST['search_text'];
$res = mysql_query("SELECT id_news, news_date, news_caption, news_text,
news_id_author FROM news WHERE news_caption
LIKE('%$search_text%') ");
```

При запиті виду http://example.org/script.php?search_text=Test ми отримаємо виконання такого SQL-запиту:

```
SELECT id_news, news_date, news_caption, news_text, news_id_author
FROM news WHERE news_caption LIKE('%Test%')
```

Але, запровадивши в параметр search_text символ лапки ми можемо кардинально змінити поведінку SQL-запиту. Наприклад, передавши як параметр search_text значення ') +and+ (news_id_author='1, ми змусимо виконати запит:[14]

```
SELECT id_news, news_date, news_caption, news_text, news_id_author
FROM news WHERE news_caption LIKE('%') and (news_id_author='1%')
```

Використання UNION

Мова SQL дозволяє об'єднувати результати декількох запитів за допомогою оператора UNION. Це надає зловмисникові можливість отримати несанкціонований доступ до даних.

Розглянемо скрипт відображення новини (ідентифікатор новини, яку необхідно відобразити, передається в параметрі id) :

```
$res = mysql_query("SELECT id_news, header, body, author FROM news
WHERE id_news = ". $_REQUEST['id']);
```

Якщо зловмисник передасть як параметр id конструкцію -1 UNION SELECT 1,username, password,1 FROM admin, це викличе виконання SQL-запиту

```
SELECT id_news, header, body, author FROM news WHERE id_news =-1
UNION SELECT 1,username,password,1 FROM admin
```

Оскільки новини з ідентифікатором -1 завідомо не існує, з таблиці news не буде вибрано жодного запису, проте в результат потраплять записи, несанкціоновано відібрані з таблиці admin внаслідок ін'єкції SQL.[14]

Використання UNION + group_concat()

У деяких випадках хакер може провести атаку, але не може бачити більше однієї колонки. У разі MySQL зломщик може скористатися функцією:

group_concat(col, symbol, col)

яка об'єднує кілька колонок в одну. Наприклад, для прикладу цього вище виклик функції буде таким:

-1 UNION SELECT group_concat(username, 0x3a, password) FROM admin

Екранування хвоста запиту

$res = mysql_query("SELECT author FROM news WHERE id=". $_REQUEST['id']." AND author LIKE ('a%') ");

Відображає ім'я автора новини згідно ідентифікатора id лише за умови, що ім'я починається з літери a, і впровадження коду з використанням оператора UNION складне.

У таких випадках, зловмисниками використовується метод екранування частини запиту за допомогою символів коментаря (/*або--в залежності від типу СКБД).[14]

У цьому прикладі, зловмисник може передати в скрипт параметр id зі значенням**-1 UNION SELECT password FROM admin/***, виконавши таким чином запит

SELECT author FROM news WHERE id=-1 UNION SELECT password FROM admin/* AND author LIKE ('a%')

в якому частина запиту (*AND author LIKE ('a%')*) позначена як коментар і не впливає на виконання.

Розщеплення SQL-запиту

Для розділення команд в мові SQL використовується символ **;** (*крапка з комою*), впроваджуючи цей символ до запиту, зловмисник отримує можливість виконати декілька команд в одному запиті, однак не всі діалекти SQL підтримують таку можливість.

$id = $_REQUEST['id'];
$res = mysql_query("SELECT * FROM news WHERE id_news = $id");

Зловмисником передається конструкція, що містить крапку з комою, наприклад 12;INSERT INTO admin (username, password) VALUES ('HaCkEr', 'foo'); то в одному запиті будуть виконані 2 команди[14]

SELECT * FROM news WHERE id_news = 12;
INSERT INTO admin (username, password) VALUES ('HaCkEr', 'foo');
і в таблицю admin буде несанкціоновано доданий запис HaCkEr.

2.2.4.3 Методика атак типу впровадження SQL-коду

Пошук скриптів, які уразливі для атаки

На цьому етапі зловмисник вивчає поведінку скриптів сервера з метою виявлення їх аномальної поведінки при маніпуляції вхідними параметрами. Маніпуляція відбувається всіма можливими параметрами:

- Даними, переданими через методи GET і POST
- Значеннями [HTTP-Cookie]
- HTTP_REFERER (для скриптів)
- AUTH_PASSWORD та AUTH_USER (при використанні аутентифікації)

Як правило, маніпуляція зводиться до підстановки в параметри символу одинарної (рідше подвійний або зворотної) лапки.

Аномальною поведінкою вважається будь-яка поведінка, при якому сторінки, одержувані до і після підстановки лапок, розрізняються (і при цьому немає повідомлення неправильний форматі параметрів).[14]

Найчастіші приклади аномальної поведінки:

- виводиться повідомлення про різні помилки;
- при запиті даних (наприклад, новини або списку продукції) запитувані дані не виводяться взагалі, хоча сторінка відображається і т. д.

Слід враховувати, що відомі випадки, коли повідомлення про помилки, в силу специфіки розмітки сторінки, не видно в браузері, хоча і присутні в її HTML-коді.

2.2.4.4 Захист від атак типу впровадження SQL-коду

Для захисту від цього типу атак необхідно ретельно фільтрувати вхідні параметри, значення яких будуть використані для побудови SQL-запиту.

Фільтрація рядкових параметрів

Припустимо, що код, який генерує запит (на мові програмування Паскаль), виглядає так:

```
statement:= 'SELECT * FROM users WHERE name = "' + userName + '";';
```

Щоб впровадження коду було неможливо, для деяких СКБД, в тому числі, для MySQL, потрібно брати в лапки всі рядкові параметри. У самому параметрі замінюють лапки на \", апостроф на \', зворотну косу риску на \\ (це називається **«екранувати спецсимволи»**). Це можна робити таким кодом:[14]

```
<?php
$query = "SELECT * FROM users WHERE
user='".mysql_real_escape_string($user)."';";
?>
```

Фільтрація цілочисельних параметрів

Візьмемо інший запит:

```
$query = 'SELECT * FROM users WHERE id = ' + id + ';';
```

У цьому випадку поле **id** має числовий тип, і його найчастіше не беруть в лапки. У такому випадку допомагає перевірка - якщо змінна **id** не є числом, запит взагалі не повинен виконуватися.[14]

Для PHP цей метод буде виглядати так:

```
$query = 'SELECT * FROM users WHERE id = '. intval($id);
```

Усікання вхідних параметрів

Для внесення змін в логіку виконання SQL-запиту потрібно впровадження достатньо довгих рядків. Так, мінімальна довжина такого рядка

у наведених вище прикладах становить 8 символів («**1 OR 1=1**»). Якщо максимальна довжина коректного значення параметра невелика, то одним з методів захисту може бути максимальне усікання значень вхідних параметрів.[14]

Наприклад, якщо відомо, що поле **id** у вищенаведених прикладах може приймати значення не більше 9999, можна «відрізати зайві» символи, залишивши не більше чотирьох:

statement:= 'SELECT * FROM users WHERE id = ' + LeftStr(id, 4) + ';';

2.2.5 Захист серверу від XSS атак

XSS — це тип вразливості інтерактивних інформаційних систем у вебі. XSS виникає в той момент, коли на сторінки, які були згенеровані сервером, з якоїсь причини потрапляють користувацькі скрипти. Специфіка подібних атак полягає в тому, що замість безпосередньої атаки сервера зловмисники використовують вразливий сервер для атаки на користувача. Для терміну використовують скорочення «XSS», щоб не було плутанини з каскадними таблицями стилів (абревіатура «CSS»). Довгий час програмісти не приділяли їм належної уваги, вважаючи їх безпечними. Однак ця думка помилкова: на сторінці або в HTTP-Cookie можуть бути досить вразливі дані (наприклад, ідентифікатор сесії адміністратора). На популярному сайті скрипт може влаштувати DoS-атаку.

Сьогодні веб-додатки потребують захисту від різноманітних атак. Більшість користувачів використовують мови програмування клієнтської сторони, такі як JavaScript, але розповсюдження такого підходу також викликає проблему вразливості додатку до атак типу XSS (Cross Site Scripting). Метою міжсайтового скриптингу може бути крадіжка cookies користувача за допомогою вбудованого на сервери скрипта з подальшою вибіркою необхідних даних та використання їх для подальших атак та злому, вилучення даних із форм для передачі персональних даних користувача зловмиснику, або

ж для проведення DDoS-атаки . Зловмисник здійснює атаку не на пряму, а за допомогою вразливостей веб-сайту впроваджуючи свій спеціальний Java Script код. У користувачів цей код відображається як частина сайту. Чіткої класифікації для міжсайтового скриптингу не існує, однак експертами по всьому світу прийнято виділяти три основних типи [8,9]:

- Постійний XSS. Один із найбільш небезпечних типів вразливостей, так як дозволяє зловмиснику отримати доступ до сервера і вже з нього керувати шкідливим кодом (видаляти, модифікувати). Кожного разу при зверненні до сайту виконуватиметься заздалегідь завантажений код, працюючий в автоматичному режимі. В основному таким вразливостям піддаються форуми, портали, блоги де присутня можливість коментування в HTML без обмежень. Сам шкідливий код може бути вбудований як в текст так і в картинки/рисунки.

- Непостійний XSS. В цьому випадку шкідливий скрипт виступає в ролі запиту жертви до зараженого веб-сайту. Цей принцип працює по наступній схемі:

1) Зловмисник заздалегідь створює URL-посилання, яке буде містити шкідливий код та відправляє його жертві.

2) Вона направляє цей URL-запит на сайт (переходячи по посиланню)

3) Сайт автоматично бере дані з шкідливого скрипта та підставляє у вигляді модифікованої URL-відповіді жертві. 4) У результаті в браузері у жертви виконується даний код, а зловмисник отримує всі cookies користувача.

- на основі DOM-моделі. В цьому варіанті можливе використання як постійного так і непостійного XSS.

Для виявлення XSS вразливостей можна використовувати різноманітні спеціалізовані сервіси, які в автоматичному режимі проведуть сканування сторінки [9]. Хоча даний метод не дає повної гарантії успіху, тому рекомендовано перевіряти сторінки в ручному режимі та обов’язково вилучити всі вразливі до даного типу атак спецсимволи, в яких прописуються всі зарезервовані мовою html-запити та теги.

Серед методів боротьби з XSS можна виділити декілька основних [9]:

1) Якщо на сайті присутній користувацький вхід, то необхідно виконувати шифрування.

2) Якщо шифрування виконати не можливо по певним причинам, варто використовувати перевірку введення (валідацію). Вона зазвичай використовує білі списки, а не чорні. Наприклад для того, щоб скласти список усіх шкідливих протоколів необхідно просто внести у список усі безпечні протоколи і заборонити усе що відсутнє у ньому. Це забезпечить захист навіть при появі нових шкідливих протоколів.

3) Зашифрування HTML на стороні клієнта за допомогою JavaScript. Оскільки JavaScript не має в наявності API для зашифрування HTML необхідно створити його власноруч.

4) Безпечна обробка даних повинна виконуватися не лише на стороні веб-сервера а й на стороні клієнта

5) Використання JQuery. Найпоширеніша форма XSS в JQuery - це коли ви передаєте введення користувача селектору JQuery. Веб-розробники часто використовують location.hash і передають його селектору, що спричинить XSS, оскільки JQuery буде представляти HTML. jQuery розпізнав цю проблему і випрацював їх логіку вибору, щоб перевірити, чи починається введення з хеша. Тепер jQuery візуалізує HTML лише у тому випадку, якщо перший символ є < Якщо ви передаєте недовірені дані селектору JQuery, переконайтесь, що ви правильно вимкніть значення за допомогою функції jsEscape, наведеної вище.

6) Використання PHP. У PHP є вбудована функція для шифрування сутностей, відомих як htmlentities. Необхідно викликати цю функцію, щоб уникнути введення даних у контексті HTML.

7) Використання CSP (Content Security Policy). CSP є останньою лінією захисту від міжсайтового скриптингу. Якщо захист від xss не спрацював по певним причинам використовується політика безпеки для пом’якшення xss, обмеживши можливості зловмисника. CSP дозволяє керувати різними речами,

наприклад, чи можна завантажувати зовнішні сценарії та чи виконуватимуться вбудовані сценарії. Для розгортання CSP потрібно включити заголовок відповіді HTTP під назвою ContentSecurity-Policy зі значенням, що містить вашу політику.

На завершення варто зауважити що запобігання xss атакам є надзвичайно важливим для забезпечення цілісності даних веб-додатку. Основними з методів боротьби з даним типом атак є використання JQuery, PHP, CSP, шифрування HTML, перевірка введення. А отже варто використати дані методи при захисті власного веб-додатку.

2.3 Шифрування

Шифрування це процес перетворення читаного людиною тексту або інформації в іншу форму в незрозумілий рядок символів для його приховування - передача незахищеними каналами або зберігання в цій формі. Зрозумілий текст це явний текстом. Перетворений текст результату називається секретним текстом, зашифрованим текстом або криптограмою. Шифрування здійснюється за допомогою спеціальних математичних функцій, які також називаються криптографічними алгоритмами шифрування.[2] Зворотна операція, яка спрямована на перетворення секретного тексту в загальнодоступний, називається дешифрування або дешифрування. Илюстрацією шифрування є Рис. – 2.2.

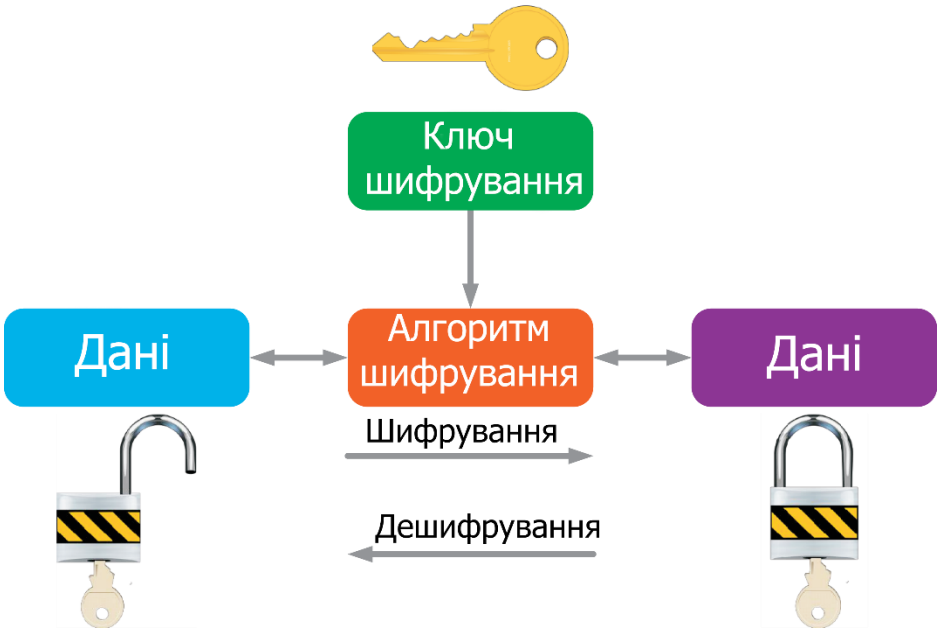

Рис. – 2.2 – Схема шифрування

Шифрування використовується для приховування інформації від небажаного або несанкціонованого доступу. Це особливо корисно, коли потрібно відправити повідомлення від відправника одержувачу таким чином, щоб неможливо читати його третім особам.[5,6]

Згідно з джерелами, римський імператор Гай Юлій Цезар використовував шифр Цезаря з воєнними цілями. Шифр Цезара - це дуже простий шифр, що змінює код. Кожна буква загальнодоступного тексту замінюється іншою літерою того ж алфавіту - відокремлена від нього певною кількістю місць в алфавіті. Окремі літери явного тексту мають свої постійні еквіваленти. Кожного разу застосовувалася зміна вправо. Перші літери алфавіту використовувалися для шифрування тексту, в якому відбувалися літери з кінця алфавіту. Щоб прочитати зашифроване повідомлення, достатньо було перемістити певну кількість символів вліво. У цьому випадку ключ дешифрування - це кількість символів. Нижче наведено короткий приклад.[4,6]

Зашифрований текст: "dwdnxmhpbrvzlflh"

Клавіша: замінити кожну літеру на третю букву алфавіту

Алфавіт: a b c d e f g h i j k l m n o p p r r t t v v w x y z

Зміна дешифрування: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Таким чином, розшифроване повідомлення має такий зміст: "Ми атакуємо на світанку".

Ключ є одним з основних елементів шифру. Він використовується для шифрування та дешифрування тексту. У шифрі Цезар ключовою функцією є кількість символів, за якими відбувається зсув.

У багатьох шифрах ключову роль відіграє ключ або пароль, за допомогою якого явний текст шифрується з метою його шифрування. З іншого боку, той самий пароль використовується для отримання відкритого тексту із зашифрованого тексту. У комп'ютерному шифруванні ключ являє собою послідовність бітів. Ефективність шифрування багато в чому визначається довжиною ключа. У великому спрощенні, довжина ключа - це кількість можливих комбінацій клавіш. Якщо простір можливих клавіш невеликий, агресор може спробувати всі ключі по черзі. Ця дія називається брутфорс. Метод шифрування вважається порушеним, коли хтось виявляє спосіб атакувати даний шифр, який є більш ефективним, ніж брутфорс, і тому намагається випробувати всі можливі ключі. Тому потрібно брати досить довгі ключі, щоб нападники при бутфорсі не мали шансів на успіх.[8,5]

Шифр Цезаря не забезпечує майже жодного захисту. Він забезпечує лише 26 клавіш. Спроба їх (навіть вручну) займає дуже мало часу. Тому ви можете швидко прочитати зашифровані тексти, створені за допомогою цього методу

У комп'ютерному шифруванні можна визначити довжину ключа на основі кількості бітів (наприклад, 40-розрядний ключ, 56-бітний, 128-бітовий, 256-бітовий). Разом з довжиною ключа кількість можливих комбінацій збільшується - так само, як і у випадку з паролем. 128-бітове шифрування в мільярд разів сильніше, ніж 40-бітове шифрування.

Енігма - роторна електромагнітна шифруюча машина, що використовується німцями під час Другої світової війни.

Повідомлення були зашифровані вже в давнину. Прикладом цього є код Цезаря, згаданий вище. Можна розділити історію криптографії на три епохи. У першій, яка тривала до початку XX століття, вона була зашифрована вручну або з використанням механічних дисків.

Дуже відомим прикладом шифрувальної машини з другої епохи криптографії є Enigma. Вона стала знаменитою під час Другої світової війни, коли вона служила збройним силам Третього рейху та його союзникам, щоб тримати радіостанції в таємниці від союзників. З німецької сторони алгоритми шифрування Enigma вважалися дуже складними. Польські криптологи - Маріан Реєвський, Генрік Зигальський та Єжи Ружицький - зуміли розшифрувати спочатку повідомлення, що передаються за допомогою німецького шифрувальника. З документацією польських математиків подальшу роботу з розшифрування постійно вдосконаленої Енігми проводили англійські вчені на чолі з Аланом Тьюрінгом.[7,9]

Протягом тривалого часу розробка криптологічних методів була в основному обумовлена військовими застосуваннями. З одного боку, криптографи намагалися шифрувати вміст повідомлення найефективнішим способом, а з іншого - криптоаналітики спробували розшифрувати перехоплені повідомлення якомога швидше. В даний час дослідження методу шифрування проводяться набагато ширше.

Більшість методів шифрування пов'язані з труднощами вирішення математичних задач, які є об'єктом дослідження в області теорії чисел. Говорячи розмовно, принцип шифрування ґрунтується на принципі, що виконувати деякі дії дуже просто, але дуже важко скасувати і відновити початковий стан. Наприклад, легко розбити керамічну вазу, кинувши її з великої висоти, але набагато важче склеїти шматки.Подібні явища існують і в області математики і є предметом дослідження в рамках теорії чисел. [4,5,9]

2.3.1 Опис асиметричних і симетричних методів шифрування

Симетричні методи використовують один і той же ключ для дешифрування і шифрування повідомлень. Перш ніж передавати конфіденційну інформацію, відправник і одержувач повинні спільно визначити секретний ключ і надати йому захищений канал. Симетричні методи шифрування включають Шифп Цезаря. Іншими симетричними шифрами є AES, One-Time і 3DES.

У асиметричних методах шифрування, як відправник, так і одержувач мають окрему пару ключів. Ця пара складається з відкритого ключа та закритого ключа. Перший з них є публічним і доступним для громадськості, а інший повинен зберігатися в таємниці.

Відправник використовує відкритий ключ одержувача для шифрування повідомлення. Одержувач може прочитати його за допомогою секретного приватного ключа. Асиметрія випливає з того, що дані, зашифровані відкритим ключем згаданої пари, можуть бути розшифровані тільки за допомогою приватного ключа пари.

На практиці це виглядає так (див Рис. – 2.3). Коли користувач хоче отримувати конфіденційні повідомлення в зашифрованому вигляді електронною поштою, він створює пару ключів зі спеціальним програмним забезпеченням. Потім він розкриває відкритий ключ або відправляє його людям, які мають намір надіслати йому секретні повідомлення. Відправник і приймач виконують інші заходи, описані вище. Агресор не може встановити ключ розшифрування протягом розумного часу на основі відкритого ключа, який він має у своєму розпорядженні. Одним з найбільш популярних асиметричних методів шифрування є RSA .[10,7]

Рис. – 2.3 – схема асиметрчного шифрування

Найбільшим недоліком симетричних алгоритмів шифрування є необхідність використання одного і того ж ключа для дешифрування і шифрування повідомлень(див Рис. – 2.4). Тому, перш ніж зробити секретну комунікацію, відправник і одержувач повинні обміняти ключ. Ключ повинен передаватися через захищений канал. У більшості випадків люди, які спілкуються один з одним, не мають такого варіанту. Канал передачі, який вони використовують, піддається атакам, і агресор має можливість перехопити секретний ключ.

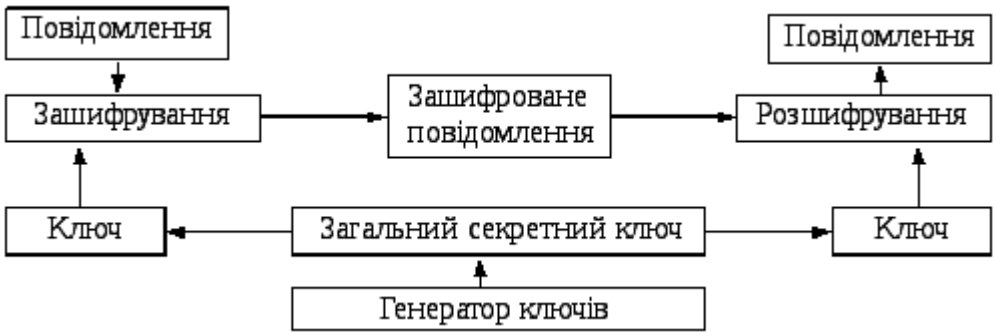

Рис. – 2.4 – схема симетричного шифрування

Переваги алгоритмів асиметричного шифрування включають більш високий рівень безпеки і зручності використання. Ці методи не вимагають передачі секретних ключів або розкриття їх будь-кому. Додатковою перевагою асиметричних шифрів є можливість їх використання для створення цифрових підписів.

Недоліком асиметричних шифрів є великий обсяг роботи, необхідний для шифрування і дешифрування повідомлень, що призводить до низької продуктивності цих методів. Як правило, симетричні алгоритми дозволяють захищати явне і читати секретні повідомлення набагато швидше. Різниця в продуктивності може бути особливо важливо при обробці великих обсягів даних. У цьому випадку асиметричні методи шифрування не працюють.

Властивості асиметричних криптографічних систем відкривають для них широкий спектр можливостей. В даний час використовуються асиметричні методи, наприклад, при шифруванні електронної кореспонденції, а також у криптографічних протоколах, таких як SSL / TLS. Він дуже популярний серед інших Протокол HTTPS, що дозволяє безпечно здійснювати зв'язок веб-браузера з сервером. Крім того, асиметричні методи часто використовуються для цифрового підписання транзакцій або документів.

У деяких ситуаціях симетричних алгоритмів цілком достатньо, і вам не потрібно використовувати складну асиметричну криптографію. Це той випадок, коли обставини дозволяють надійно встановити ключ через спілкування з особами (наприклад, на особистій зустрічі). Симетричні методи також корисні в середовищах, в яких центральний пристрій керує всіма ключами (наприклад, у закритих банківських системах).

Оскільки блок управління все ще має всі ключі, немає необхідності визначати приватні та відкриті ключі. Асиметрична криптографія не потрібна, якщо потрібно захистити лише одного користувача. Наприклад, симетричний шифр достатній для шифрування приватних файлів. Асиметричні алгоритми зазвичай краще підходять для середовищ, що використовуються кількома або більше користувачами.

Можливе загальне використання симетричних і асиметричних елементів криптографії. Це рішення називається гібридним шифруванням. Правильні дані шифруються симетричним методом, а ключ, що використовується для цієї мети, запобігає перехопленню за допомогою відкритого ключа одержувача. Таким чином, уникається обробка великих обсягів даних неефективними

асиметричними методами. Вони потрібні лише для шифрування та дешифрування ключа. Одержувач прочитає його за допомогою свого приватного ключа, отримавши можливість розшифровувати все повідомлення.

Популярний симетричний алгоритм шифрування даних DES був створений у співпраці з IBM з NSA . Метою було створення безпечного стандарту шифрування для обміну інформацією між адміністративними одиницями США. Через низьку довжину ключа (лише 56 біт) DES в даний час більше не забезпечує високий рівень безпеки, необхідний у деяких додатках. Однак можна штучно збільшити довжину ключа шляхом шифрування даних повторно за допомогою алгоритму DES. Такі каскадні варіанти алгоритму Triple DES, TDES і 3DES все ще використовуються в секторі банківських послуг.

Розширений стандарт шифрування AES був створений як наступник алгоритму DES на вимогу Американського Федерального Агентства NIST (Національний інститут стандартів і технологій). У 2000 році він прийняв його як стандарт, що використовується в державній адміністрації. Спочатку алгоритм називався Rijndael з фрагментів імен його творців, Vincent Rijman і Joan Daemen.

Шифр доступний у трьох базових версіях - AES-128, AES-192 і AES-256. Цифри в назвах відносяться до довжини ключа. Таким чином, новий алгоритм використовує набагато довші ключі, ніж у DES. Метод AES забезпечує дуже високий рівень безпеки. Алгоритм є загальнодоступним і може бути реалізований в апаратних засобах або програмному забезпеченні без необхідності сплачувати ліцензійні збори. Варіанти AES-192 і AES-256 дозволені в США для шифрування документів державного рівня з найвищим рівнем класифікації.

RSA - асиметричний метод, який можна використовувати не тільки для шифрування, але й для створення цифрових підписів. Він був розроблений

трьома криптографами в Університеті MIT. Назва шифру походить від перших букв імен його авторів.

Безпека даних, зашифрованих за допомогою RSA, ґрунтується на складності поширення великих чисел у прості множники. Безпека багато в чому визначається довжиною ключа, тому він не повинен бути занадто малим.

У порівнянні з вищеописаними методами 3DES і AES алгоритм RSA набагато повільніше - принаймні тисяча разів. Тому він використовується в основному тільки для обміну симетричним ключем. Відповідні дані захищені симетричним методом. Це рішення поєднує в собі переваги обох методів - зручне обміну ключами і високу ефективність шифрування.

Twofish - симетричний алгоритм шифрування. Наступник методу Blowfish, який досі використовується. Довжина ключа 128, 192 або 256 біт. Автори Twofish не патентували цей алгоритм, поширюючи його у відкритому доступі. Тому вона доступна всім для вільного використання.

Алгоритм Blowfish працює на 64-бітному вхідному блоці відкритого тексту, який ми обробляємо в 16 раундах (Рис. – 2.6). Максимальна довжина ключа - 448 біт. Шифрування працює таким чином:

Розділений на 2 етапи:

1. Підготовчий — формуємо ключі шифрування по секретному ключу.

- Ініціалізація масивів P і S за допомогою секретного ключа K

    1. Ініціалізація фіксованим рядком P1-P18, що складається з шістнадцяткових цифр мантиси числа пі.
    2. Проводиться операція XOR над P1 з першими 32 бітами ключа K, над P2 з другими 32-бітами і так далі. Якщо ключ K коротше, то він накладається циклічно.

- Шифрування ключів і таблиць замін

3. Алгоритм шифрування 64-бітного блоку, використовуючи початкові ключі P1-P18 і таблицю замін S1-S4, шифрує 64 бітну нульовий (0x0000000000000000) рядок. Результат записується в P1, P2.
4. P1 і P2 шифруються зміненими значеннями ключів і таблиць замін. Результат записується в P3 і P4.
5. Шифрування триває до зміни всіх ключів P1-P18 і таблиць замін S1-S4.

2. Шифрування тексту отриманими ключами і F(x) (Рис. – 2.5), з попереднім розбиттям на блоки по 64 біти. Якщо неможливо розбити початковий текст на блоки по 64 біти, використовуються різні режими шифрування для побудови повідомлення, що складається з цілого числа блоків. Сумарна необхідна пам'ять 4168 байт: P1-P18: 18 змінних по 32 біта; S1-S4: 4x256 змінних по 32 бита.[11,9]

Розшифрування відбувається аналогічно, тільки P1-P18 застосовуються у зворотному порядку.[11,9]

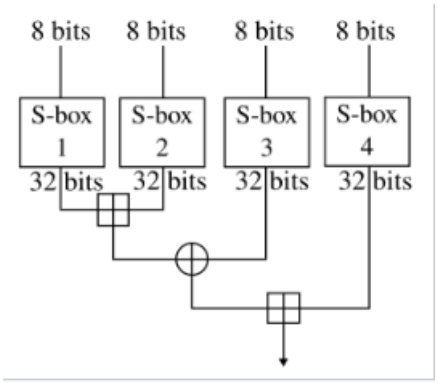

Рис. – 2.5 - Функція F (x) в Blowfish

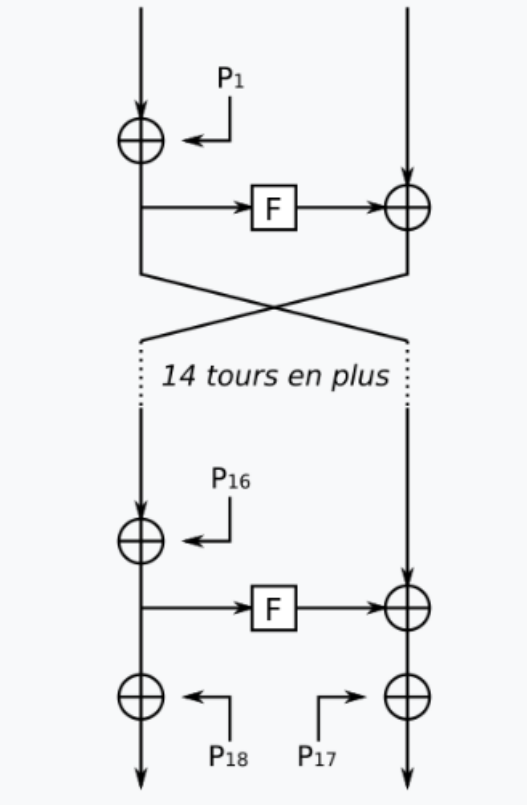

Рис. – 2.6 – Мережа Фейстеля при шифруванні

Сьогодні AES, Twofish, Blowfish і 3DES вважаються сучасними і безпечними алгоритмами шифрування.

Можна зашифрувати файли, папки, розділи, цілі диски, а також передачу даних, електронну кореспонденцію та багато інших елементів. Численні програми (деякі з них безкоштовні) дозволяють криптографічно захищати файли, носії інформації та потоки інформації в мережі. Навіть деякі утиліти стиснення даних пропонують шифрування. Наприклад, безкоштовний 7-Zip підтримує криптографічний алгоритм AES з ключем довжиною 256 біт. Популярне, умовно-безкоштовне WinRAR шифрує за допомогою методу AES з 128-бітовим ключем. Для шифрування дискових ресурсів доступно багато програм. Найбільш функціональними є крос-платформний інструмент VeraCrypt, наступник відмовився від проекту TrueCrypt. Він працює в середовищі Windows, Linux і Mac OS. Вона дозволяє шифрувати розділи і диски у фоновому режимі, а також створювати файли-контейнери, які збирають дані, підключені до файлової системи, як віртуальні диски. Як і його

попередник, інструмент підтримує алгоритми шифрування AES, Serpent і Twofish, що дозволяє їх каскадувати. Однак творці збільшили з тисячі до більше трьохсот тисяч число ітерацій при обчисленні хеш-значень для створення ключа. Ключі шифрування генеруються функцією PBKDF2 з рядком з порушенням 512 біт.

Найбільш відомим програмним забезпеченням для шифрування всіх видів даних є PGP. Американський фізик Філ Циммерман створив інструмент PGP, в якому він реалізував код RSA. Закінчена програма опублікована на Usenet, що робить її доступною для всіх для вільного використання. Інноваційною особливістю цього рішення була можливість приєднання електронної пошти з цифровим підписом, що підтверджує ідентичність відправника.

Через незрозумілу ситуацію, коли проект PGP належить компанії-розробнику McAfee, був створений стандарт OpenPGP. Він заснований на вихідному коді PGP 5.x, який був опублікований в книзі і в такому вигляді був експортований з США. Правове регулювання забороняє шифрувати за допомогою ключа довжиною більше 40 біт. Крім того, алгоритми, що використовуються в програмному забезпеченні PGP, наприклад, RSA, вже були запатентовані. Що ще гірше, були помилкові чутки, що PGP мав прогалини, які можна використовувати пізніше.

Поряд з GnuPG з'явилася перша реалізація стандарту OpenPGP. Він був розроблений як безкоштовна альтернатива PGP і опублікований під ліцензією GNU GPL. З часом було створено багато розширень стандарту OpenPGP, які забезпечують більшу функціональність, ніж PGP.

2.3.1.1 Важливі відмінності симетричного и асиметричного шифрування

- Симетричний алгоритм гарний для передачі великих обсягів шифрованих даних. Асиметричний алгоритм, за інших рівних, буде істотно повільніше. Крім того, для організації обміну даними по асиметричному

алгоритму або обом сторонам повинні бути відомі відкритий і закритий ключ, або таких пар повинно бути дві (по парі на кожну сторону).

- Симетричні алгоритми зазвичай будуються на основі деяких блоків з математичними функціями перетворення. Тому модифікувати такі алгоритми легше. Асиметричні ж алгоритми зазвичай будуються на деяких математичних задачах, наприклад. RSA побудований на завданню зведення в ступінь і взяття по модулю. Тому їх практично неможливо або дуже складно модифікувати.
- З точки зору злому (компрометації) пароля, асиметричний алгоритм легший, так як серверу достатньо змінити пару ключів і розіслати створений публічний ключ. У разі симетричного шифрування, постає питання про те, як передати наступний пароль. Однак і ці обмеження обходяться, наприклад, на обох сторонах ключі постійно генеруються по одному і тому ж алгоритму, тоді питання стає в збереженні цього алгоритму в секреті.[12,10]

2.5 Захист серверу через шифрування передачі даних в Інтернеті

Для шифрування передачі даних в Інтернеті використовується протокол SSL, який шифрує інформацію, що передається з веб-браузера, на сервер і в зворотному напрямку. Починаючи з версії 3, SSL розробляється як TLS. Він захищає лише канал передачі. При отриманні даних цільовий сервер його розшифровує і зберігає в локальній файловій системі.Шифрування TLS в даний час використовується в основному у зв'язку з HTTPS - протоколом зв'язку для безпечної передачі даних у World Wide Web. Коли користувач Інтернету викликає веб-сайт через SSL, він використовує ті самі елементи HTTP, які він зазвичай посилає на веб-сервер. Різниця полягає в тому, що на цей раз дані обмінюються зашифрованим каналом. Таким чином, ім'я HTTPS, розміщене на початку URL-адреси (замість HTTP), не означає іншого протоколу, але сигналізує шифрування.Більшість веб-серверів підтримують TLS 1.0, деякі також мають SSLv2 і SSLv3 з різними алгоритмами

шифрування. Тим часом майже всі веб-браузери віддають перевагу TLS з RSA і шифруванням AES. Також використовується SSL на серверах електронної пошти, веб-додатках і комунікаціях між серверами.

2.5.1 Цифрові підписи

Цифрові підписи використовуються для безпечного проведення онлайн-транзакцій. Вони дозволяють перевіряти електронні документи на їх походження та автентичність. Вони підтверджують ідентичність спілкуються людей. По суті, вони являють собою цифрову версію печатки, яка одночасно виступає як підпис.

Варто зазначити, що цифровий підпис забезпечує ще більшу безпеку, ніж традиційна підпис. Вона гарантує одержувачу не тільки того, що відправник повідомлення є особою, для якої він забезпечує (гарантію достовірності), але й гарантує недоторканність листування. Таким чином, він засвідчує, що документ досягне одержувача в тому вигляді, в якому його відправив відправник (він не буде перехоплений і змінений у дорозі).

Цифрові підписи вимагають інфраструктури відкритого ключа , яка надає послуги з аутентифікації і видає сертифікати для підтвердження відповідності використовуваних ключів.

Для того, щоб користувач міг одягнути документи з цифровим підписом, він повинен мати ключі підпису. У цифрових підписах, як правило, застосовуються алгоритми асиметричного шифрування. Для кожного учасника зв'язку повинна бути згенерована окрема пара ключів, що складається з закритого ключа та відкритого ключа. Відкритий ключ дозволяє перевірити цифровий підпис. Приватний ключ, однак, використовується для підписання і повинен зберігатися в безпечному місці. Він не може потрапити в руки третіх осіб.

Щоб зробити цифровий підпис, потрібно створити хеш-значення з надісланого повідомлення та підписати його приватним ключем. Повідомлення і підпис потім відправляються адресату, без необхідності

шифрувати вміст документа. Мова йде не про конфіденційність, а про підтвердження вашої ідентичності та гарантування недоторканності.

Одержувач перевіряє автентичність підпису, перевіряючи значення хешування за допомогою відкритого ключа. Якщо процедура успішна, то можна вважати, що повідомлення надходить від власника закритого ключа і його вміст не маніпулюється під час передачі.

Сертифікат - це набір даних, який підтверджує певні властивості об'єктів або людей. Достовірність і недоторканність сертифіката можна перевірити за допомогою криптографічних методів. Цифровий сертифікат містить дані, необхідні для його перевірки.[8,12]

2.5.2 Інфраструктура відкритого ключа

Інфраструктура публічних ключів надає ресурси, необхідні для надання послуг сертифікації. До його складу входять правила, що визначають умови видачі сертифікатів. Завданням цієї інфраструктури є також перевірка ідентичності осіб або організацій, які подають заявку на отримання сертифіката. Зазвичай він публікує інформацію про видані та відкликані сертифікати. Крім того, PKI повинна мати методи для підготовки таких сертифікатів, які є надійними і не підвищують сумніви користувачів. Важливо, щоб не було доступу з Інтернету до обладнання, яке використовується для створення сертифікатів.

2.6 Висновки до другого розділу

Виходячи з нашого аналізу, ми з’ясували, що кожен веб додаток, сервер может стати підлягати нападкам злочинців, тому ми віришили, що потрібно удосконалити наш веб додаток захистом від Ddos й XSS.

Також ми з‘ясували, що для великої кількості даних, шифрувати краще симетричними шифрами, тоді це не займе великий проміжок часу и сервер не

буде зависати при великої кількості людей в онлайні. Було обрану шифр блочний симетричний шифр Blowfish, через те, що немає жодного вдалого випадку взлому цього шифру, з‘ясували що бази даних не є бездоганно захищенні, і тому для того, щоб зберегти конфеденційність даних, було вирішено тримати дані в базі в шифрованому виді. Це зроблено для того, щоб зловмисник, який тим чи іншим способом зміг отримати доступ до бази, не на руках конфеденційні та персональні дані людей

В рамках цієї діпломной роботи було вирішено використовувати утиліту itables й функцію **filter_sanitize_encoded()** задля захисту серверу,а також тримати ключ також у базі, але у замаскованому виді, ключі будуть унікальні через те, що ми будем використовувати ключи в якості хешування пароля користувача функцією MD5.

Розділ 3. ПРОГРАМНА РЕАЛІЗАЦИЯ ПРОДУКТУ, ВДОСКОНАЛЕННЯ ПРОГРАМННОГО МОДУЛЯ В ЯКОМУ ВИКОРИСТАНО СИМЕТРИЧНЕ ШИФРУВАННЯ BLOWFISH ПЕРСОНАЛЬНИХ ДАНИХ

3.1 Вибір технологій для розробки програмного продукту

Для розробки такого продукту нам буде потрібен набір технологій нового покоління. Вибір технологій відіграє велику роль у написанні програмного продукту, адже з участю сучасних технологій ми можемо розробити ресурс, який буду працювати з великою швидкістю використовуючи мінімально написаного кода. Все залежить від стандартних бібліотек, які вкладаются в ту чи іншу мову програмування, не говорячи про сторонні бібліотеки, які ми можемо доставити собі самі, для розвязвування тих чи інших задач. Крім того середа розробки відіграє не менш важливу роль.

Так як нам потрібно шифрувати данні які знаходятся на стороні сервера. Було вирішено використовувати таку мову програмування як PHP. PHP - це мова програмування, яка використовується в основному для створення динамічних веб-сайтів. Історія цієї мови почалася, коли Rasmus Lerdorf створив його як набір інструментів для обслуговування власного сайту. Проте, PHP отримав справжню популярність, коли двома ізраїльськими програмістами, Зеєвом Сураським і Анді Гутмансом, зацікавився проект. Вони вирішили переписати PHP-код з нуля за допомогою вже існуючої спільноти PHP. Вони оголосили про PHP 3.0 як наступника PHP / FI. Найважливішою особливістю цієї програми була її модульність і легка можливість розширити функціональність мови.

PHP - це мова програмування, яка значно відрізняється від інших конкуруючих мов. Одним з його переваг є те, що він безкоштовний, кожен може завантажити його для встановлення та використання без будь-яких обмежень. Мову PHP легко вивчати і набагато швидше, ніж інші мови. Більш того, можна вставляти PHP-код безпосередньо в HTML-документ, що дуже

допомагає у створенні скриптів. PHP дозволяє працювати з багатьма базами даних і чудово працює з базою даних MySQL. Крім того, інтерпретатор PHP може виступати як модуль найпопулярнішого сервера Apache, що дає йому величезну перевагу над іншими мовами.

Створюючи додаток, що є метою цієї роботи, я буду базуватися на версії PHP 5, яку з попередніх версій відрізняє той факт, що вона з'явилася в абсолютно новій моделі об'єктно-орієнтованого програмування. Основними перевагами цієї моделі є "легкість перекладу індивідуальних вимог з області застосування в індивідуальні кодові модулі і можливість повторного використання коду". Але крім PHP нам будут потрібні ще знання з HTML та MySQL, які згадували вище. Звісно ми могли крім всіх ціх технологій використати JS, але він нам нідочого, а JS - javascript лише трохи зробить наші діх плавнішими. Звісно є допоміжні бібліотеки для JS, які вчать його розмовляти з сервером, але в рамках цієї роботи використовувати іх не будем, так як PHP швиде зробе те, що може зроботи JS з його бібліотеками, адже PHP з самого початку створили для розмови з сервером.[11,12]

HTML - це мова розмітки Hyper Text Markup. HTML є мовою програмування, винайденою вченим Тімом Бернерсом-Лі, метою якого було полегшити дослідникам різних університетів доступ до наукових праць інших вчених. Через кілька років виявилося, що створення цієї мови стало фундаментом для створення Інтернету, який ми знаємо сьогодні.

HTML - це мова програмування, що дозволяє реалізовувати веб-сайти, але це мова опису сторінки, а не зовнішній вигляд її окремих елементів. У HTML визначено певний набір стилів, який використовується на таких веб-сайтах, як заголовки, абзаци, списки, таблиці. Більш того, в ньому є певні елементи форматування символів, наприклад, шрифт, і кожен елемент має своє ім'я і існує у вигляді маркерів або тегів. Веб-браузер після завантаження файлу читає теги, а потім форматування тексту і графічних елементів відображає їх на екрані. Важливою особливістю HTML, яка сприяла

популярності системи WWW та Інтернету, є незалежність операційної системи та використовуваного комп'ютерного обладнання. .[26,13]

Як я вже згадував вище, програма, написана на HTML, є рядком так званого тега, які є символами певного значення. Теги HTML - це переважно англійські слова або їх скорочення, наприклад,

<b> - жирний текст

<img> - малюнок

<p> - параграф

Мітки в основному парами і розміщуються в кутових дужках, початковій мітці і кінцевій мітці, якій передує знак "/", наприклад

<html> і </ html> - початок і кінець програми

<body> і </ body> - початок і кінець вмісту сторінки

CSS використовується для створення зовнішнього вигляду веб-сайту, написаного в HTML. Він визначає кольори, ефекти, положення, метод відображення і навіть анімацію окремих елементів у ньому. CSS був створений для того, щоб відокремити структуру HTML-документа від форми його подання. Таке розділення збільшує обсяг доступності сайту, зменшує складність коду та полегшує зміни у структурі документа. Препроцесори та рамки використовуються для істотного спрощення та прискорення редагування стилів: CSS Framework - це набір готових стилів, що спрощує і прискорює створення сторінок. Одним з найбільш популярних є Bootstrap. Завдяки цьому набагато простіше створювати чуйні сторінки. Bootstrap також містить готові шаблони для типографіки, форм, кнопок навігації та інших елементів інтерфейсу. Попередні процесори, такі як Less.js або Sass, вводять елементи CSS-елементів, відомі з мов програмування, тобто функцій, умовних операцій або циклів. Звичайно, такий код, згенерований препроцесором, не може бути належним чином прочитаний і інтерпретований браузером, тому його слід спочатку перетворити (перенести) в результуючий код CSS.[26,15]

MySQL - це система управління реляційними базами даних. Відомий і цінується в першу чергу завдяки його неймовірній продуктивності і швидкості. Це чудово для веб-проектів, але не тільки - він також успішно використовується у великих IT-проектах таких організацій, як NASA. Противники MySQL часто говорять, наскільки їм не вистачає багатьох посилань, які мають реальні, великі системи баз даних. З мого досвіду я знаю, що деякі з цих людей навіть не розрізняють версію системи, яку пропонує компанія MySQL AB (виробник MySQL).[16,19]

Загальні функції MySQL:

- Написано на C та C ++ (продуктивність!).
- API для багатьох мов програмування: C, C ++, Eiffel, Java, Perl, PHP, Python, Ruby, Tcl.
- Повна багатопоточність, використовуючи потоки ядра. Це означає, що MySQL працюватиме на багатопроцесорній машині, якщо вона є.
- Необов'язкова підтримка транзакцій.
- B-дерева зі стиснутими індексами. Вам буде корисно, якщо ви маєте багато баз. Досить сказати, що воно значно впливає на час пошуку і вилучення даних (рядків) з бази даних.
- Велика кількість типів даних у стовпцях. Числа, рядки символів, двійкові об'єкти (BLOB), дата і час, типи переліку, набори. Варто відзначити, що в MySQL ми можемо налаштувати дану колонку на певну кількість даних, які ми маємо намір зберігати в ній (наприклад, TINYINT, а не INT), таким чином ми отримуємо більш високу продуктивність і знижуємо споживання пам'яті (включаючи використання диска). Можна визначити певні типи даних як національні (наприклад, різні стандарти кодування).
- Підтримка пунктів агрегування та групування SQL.
- Зовнішні з'єднання (LEFT & RIGHT).
- Команда SHOW дозволяє переглядати інформацію про бази даних,

таблиці та індекси. Команда EXPLAIN, описує роботу оптимізатора запитів.

- Дуже проста (з точки зору адміністратора) система безпеки. Усі паролі зашифровані.

- Підключення до сервера здійснюється через: TCP / IP, ODBC, JDBC.
- Розташування (в мовному сенсі) сервера.

В якості середовища розробки я використовував таку програму як PHPStorm.

PhpStorm - це повноцінне, багатоплатформне середовище програмування, яке дозволяє працювати з PHP-додатками. Він має функцію підказки синтаксису: класи, функції, методи, індекси таблиць і імена змінних. Редактор підтримує документацію, створену у форматі PHPDoc, надає можливість рефакторингу коду, включаючи перейменування файлів, функцій, констант, класів, методів, параметрів або змінних. У PhpStorm ви зможете налагоджувати додатки (Zend Debugger і Xdebug), а також JavaScript і виконувати модульні тести. Програма має підтримку Zend Framework і інструментів командного рядка Symfony, що підтримують управління проектами.[18,20]

Що відрізняє PhpStorm від конкуренції - це підтримка більшості популярних систем керування версіями: Git, CVS, SVN, Mercurial і Perforce. Програма також має інтеграцію з GitHub.com. PhpStorm також може використовуватися для управління базами даних.

PhpStrom включає середовище WebStrom, що пропонує зручні інструменти для створення HTML-коду, JavaScript і CSS-стилів. Програма в режимі редагування HTML-коду здатна шукати відповідні елементи класу CSS. PhpStorm розпізнає синтаксис Leaner CSS (LESS) і SASS / SCSS, що дозволяє спростити управління каскадними стилями.

Крім PHPStorm нам була потрібна программа, яка змогла би розгорнути сервери на локальній машині. Однією з таких програм є OpenServer, яку ми й обрали.

Open Server Panel - це портативна серверна платформа і програмне середовище, створена спеціально для веб-розробників з урахуванням їх рекомендацій і побажань. Програмний комплекс має багатий набір серверного програмного забезпечення, зручний, багатофункціональний продуманий інтерфейс, має потужні можливості з адміністрування та налаштування компонентів. Платформа широко використовується з метою розробки, налагодження і тестування веб-проектів, а так само для надання веб-сервісів в локальних мережах.[17,21]

Хоча спочатку програмні продукти, що входять до складу комплексу, не розроблялись спеціально для роботи один з одним, така зв'язка стала дуже популярною серед користувачів Windows, в першу чергу через те, що вони отримували безкоштовний комплекс програм з надійністю на рівні Linux серверів.

За час свого існування Open Server зарекомендував себе як першокласний і надійний інструмент необхідний кожному веб-майстру.

3.2 Фаєрвол iptables задля захисту серверу

Iptables є утилітою, яка виконує функції брандмауера, ця утиліта нам знадобится для реалізації захисту сервера від DDos атаки. Її настройка проводиться в командному рядку, за допомогою правил iptables можна дозволяти або блокувати проходження трафіку. Коли відбувається спроба встановлення з'єднання з поточної машиною, iptables переглядає список правил в списку, щоб зрозуміти, як потрібно вчинити в цьому випадку. Якщо правила немає, то виконується дія за замовчуванням. Як правило, itpables передвстановлюють на всіх Linux-дистрибутивах. Щоб оновити утиліту, або

встановити її, якщо з якихось причин вона відсутня в базовій поставці, потрібно скористатися наступною командою:

sudo apt-get install iptables

Існують і графічні інструменти-альтернативи iptables, наприклад Firestarter, але і робота в командному рядку не є дуже вже складною. Однак слід дотримуватися особливої обережності під час налаштування iptables через віддалене ssh-з'єднання, оскільки одна невірна команда може заблокувати можливість з'єднання з віддаленим сервером - доведеться якимось чином вносити зміни в налаштування машини фізично отримавши до неї доступ.

Типи правил

Існує три типи правил iptables - input, forward і output.

Input - Такі ланцюжки використовуються для контролю поведінки вхідних з'єднань. Наприклад, якщо користувач спробує підключитися до сервера по SSH, то iptables порівняє його IP-адресу зі своїм списком, щоб дозволити або заборонити доступ.

Forward - Правила цього типу використовуються для обробки вхідних повідомлень, кінцевий пункт призначення яких не є поточним сервером. Наприклад, в разі маршрутизатора, до нього підключаються багато користувачів і додатки, але дані не надсилаються на сам маршрутизатор, вони лише передаються йому, щоб він міг перенаправити їх адресату. Якщо ви не займаєтеся налаштуванням маршрутизації або NAT, то правила цього типу використовувати в роботі не будете.

Output - Такі ланцюжки використовуються для вихідних з'єднань. До прміеру, якщо користувач намагається отправінг запит ping до сайту 1cloud.ru, iptables вивчить ланцюжок правил, щоб зрозуміти, що потрібно робити в разі ping і цього сайт, і тільки потім дозволить або заборонить з'єднання.

Важливий момент

Навіть в разі пинга зовнішніх хостів, потрібно не тільки відправити пакети до них, а й отримати відповідь. При роботі з iptables важливо пам'ятати, що багато протоколів передачі даних вимагають двосторонньої комунікації. Тому потрібно налаштовувати правила відповідним чином - випадки, коли новачки забувають дозволити роботу з сервером по SSH трапляються дуже часто.

3.3 Особливості шифрування методом BlowFish, використовуючи PHP.

Оскільки PHP є сучасною мовою, то компанія, яка займается її розробкою в одній з своїх версій випустила нові розширення, які дозволяють робити шифрування і хешування різними методами.

За допомогою таких розширень як Mcrypt і Mhash, що реалізують досить велике число алгоритмів шифрування. При такому підході скрипт спочатку шифрує дані для зберігання, а потім дешифрує їх при запиті.

Задля захисту ключа метода шифрования Blowfish, було вирішено використати хешування

Хеш-функція (англ. Hash function від hash - «перетворювати в фарш», «мішанина»), або функція згортки, - функція, що здійснює перетворення масиву вхідних даних довільної довжини в (вихідну) бітову рядок встановленої довжини, що виконується певним алгоритмом. Перетворення, вироблене хеш-функцією, називається хешированием. Вихідні дані називаються вхідним масивом, «ключем» або «повідомленням». Результат перетворення (вихідні дані) називається «хешем», «хеш-кодом», «хеш-сумою», «зведенням повідомлення». [23,22]

Хеш-функції застосовуються в наступних випадках:

- при побудові асоціативних масивів;
- при пошуку дублікатів в серіях наборів даних;
- при побудові унікальних ідентифікаторів для наборів даних;

- при обчисленні контрольних сум від даних (сигналу) для подальшого виявлення в них помилок (що виникли випадково або внесених навмисно), що виникають при зберіганні і / або передачі даних;
- при збереженні паролів в системах захисту у вигляді хеш-коду (для відновлення пароля за хеш-коду потрібно функція, яка є зворотною по відношенню до використаної хеш-функції);
- при виробленні електронного підпису (на практиці часто підписують не саме повідомлення, а його «хеш-образ»);

MD5 (алгоритм Message-Digest 5) - криптографічний алгоритм, розроблений Роном Рівестом (співзасновником RSA), який є популярною криптографічною функцією ярлика, який генерує 128-бітну абревіатуру з потоку даних будь-якої довжини.[27,24]

У 2004 році був знайдений спосіб створення зіткнення MD5, що знижує його безпеку в деяких додатках (наприклад, файли підпису).

Завдяки відомим криптоаналітичним атакам функція MD5, безумовно, не повинна використовуватися в додатках, що вимагають опору зіткненням, наприклад, у цифровому підписі.

MD5 ("Ala has a cat") = 91162629d258a876ee994e9233b2ad87

Навіть невелика зміна тексту (в даному випадку перетворення а у у) викликає (з дуже високою ймовірністю) зовсім іншу абревіатуру MD5

MD5 ("Ala has cats") = 6a645004f620c691731b5a292c25d37f

Поширене використання MD5 полягає в створенні ярликів всіх типів файлів, доступних для загального доступу (як правило, в Інтернеті), завдяки чому особа, яка завантажила файл із Інтернету, може негайно перевірує (генеруючи ярлик MD5 на його копіюванні та порівнянні результатів) або ж файл яка була опублікована його автором або виникли спотворення під час самого процесу завантаження даних. Значення, що публікується в цьому випадку, має форму 32-символьного числа в шістнадцятковій системі числення.

Результат MD5 для архіву "linux-2.6.10.tar.bz2", розмір 35 МБ: cffcd2919d9c8ef793ce1ac07a440eda

3.4 Результати розробки програмного продукту

Основною метою нашої роботи є удосконалення прогромного модулю, який використовується задля захисту персональних даних.

Тому спочатку поговоримо про удосконалення, а вже потім про сам програмний модуль. Задля захисту серверу від Ddos й XSS атаки ми розробили спеціальний програмний код.

3.4.1 Захист від DDos атаки

Запропонований нами метод призначений для захисту від атаки з декількох десятків або сотень комп'ютерів частими запитами на ваш сайт. Принцип дії захисту від DDoS-атаки - блокувати IP-адреси, частота звернень з яких свідомо більше, ніж зазвичай потрібно для роботи з вашим сайтом. Передбачається, що на сервері встановлений і працює фаєрвол iptables. Захист вбудовується в php-код сторінок сайту і використовує таблицю MySQL для зберігання тимчасових даних.

Отже, створюємо таблицю access_log. Нам потрібно тільки два поля, для зберігання IP-адреси і часу доступу (Рис – 3.1)

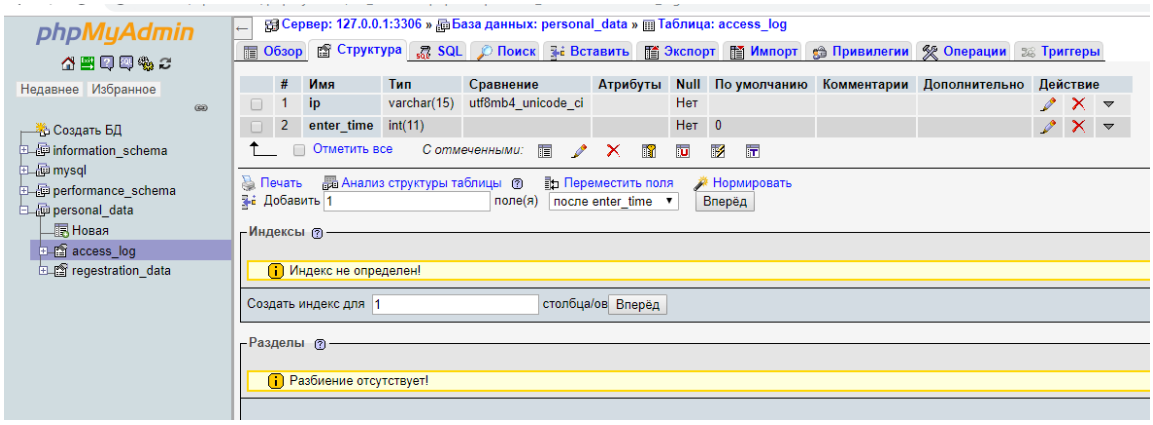

Рис. – 3.1 – структура таблиці access_log

Далі пишимо декілька функцій для фільтрації трафіку. Все, що робить функція **check_ddos()** - при кожному зверненні до захищається сторінці сайту

пише в таблицю час звернення і IP-адреса клієнта. При цьому кожен раз робиться вибірка з $ rec_limit останніх звернень з даного IP, і якщо час між першим і останнім зверненнями в цій вибірці менше значення змінної $ time_limit (в секундах) - викликається функція iptables_ban_ip, і даний IP блокується файерволом. В даному прикладі, якщо з однієї IP буде більш, ніж 4 звернення в секунду - IP блокується. Сама таблиця тимчасових даних постійно очищається від зайвих даних і містить тільки останні звернення.

Функція для запису особливо настирливих IP-адрес в блек-лист брандмауера **- iptables_ban_ip()**

3.4.2 Захист від XSS атаки

Э декілька варіантів захисту від XXS атаки.

Захист на стороні сервера

- Кодування керуючих HTML-символів, JavaScript, CSS і URL перш ніж можна буде в браузері. Для фільтрації вхідних параметрів можна використовувати такі функції: filter_sanitize_encoded (для кодування URL), htmlentities (для фільтрації HTML).
- Кодування вхідних даних. Наприклад за допомогою бібліотек OWASP Encoding Project, HTML Purifier, htmLawed, Anti-XSS Class.
- Регулярний ручної і автоматизований аналіз безпеки коду і тестування на проникнення. З використанням таких інструментів, як Nessus, Nikto Web Scanner і OWASP Zed Attack Proxy.
- Вказівка кодування на кожній web-сторінці (наприклад, ISO-8859-1 або UTF-8) до будь-яких користувальницьких полів.
- Забезпечення безпеки cookies, яка може бути реалізована шляхом обмеження домену та шляхи для прийнятих cookies, установки параметра HttpOnly, використанням SSL.
- Використання заголовка Content Security Policy, що дозволяє задавати список, в який заносяться бажані джерела, з яких можна

довантажувати різні дані, наприклад, JS, CSS, зображення та ін.

Захист на стороні клієнта

- Регулярне оновлення браузера до нової версії.
- Установка розширень для браузера, які будуть перевіряти поля форм, URL, JavaScript і POST-запити, і, якщо зустрічаються скрипти, застосовувати XSS-фільтри для запобігання їх запуску. Приклади подібних розширень: NoScript для FireFox, NotScripts для Chrome і Opera.

Так як зі сторони клієнта ми повливати ніяк не здатні, тому будемо захищати на стороні сервера. Ми будемо використовуввати функцію **filter_sanitize_encoded().** За допомогою цієї функції ми зможемо фільтрувати данні, які надходять з ріних запросів до нашого серверую.

3.4.3 Захист даних на сервері через програмний модуль методом Blowfish

Оскільки бази даних не є бездоганно захищинні, зловмисники використовуючи різні варіанти атаки на базу. І в ті рази коли в них це спрацьовує, то вони отримують усю базу в не зашифрованому виді, усі дані користувачів можуть попасти в откритий доступ, а дані можуть бути різні, від Імені та адреса до банківских карт та документів. Саме тому було віришено тримати в базі лише зашифровану інформацію, а в той час колі користувачу буде потрібно розшифровувати його в іншому запросі.

Шифрування Blowfish підрозумовує під собою шифрування з ключем, цей ключ зловмисники не повинні отримати аж в ніякому разі, так як перебираючі методи шифрування, він може в один момент знайти наш метод шифрування й отримати усі дані користувачів. Саме з цим було питання що роботи з ключем. Тут є 2 шляхи вирішення проблеми.

Перший шлях прямує видачею ключів самим користувачам, і щоб вони при заході к собі в профіль вводили також ключ шифрування, у разі не правильного ключа, він не отримає своїх даних.

Другий шлях це утримування ключів прямо в базі, але при цьому

замаскувати його, для того щоб зловмисники не зрозуміли що це ключ від шифрування. Для цього використовується хешування.

Так як користувачі не люблять робити багато дій, було вирішено пійти другим шляхом.

Так як ключ повинен бути різним й не примітним хешування для цього ідеально підходить. Ми добавили в базу ще одне поле в якому буде ключ в вигляді хешування паролю користувача по алгоритму MD5.

Плюсами цього методу э заплутаність зловмисника у базі й мінімальними діями для користувача при роботі с ресурсом.Ми отримали таку структуру бази даних (Рис. – 3.2).

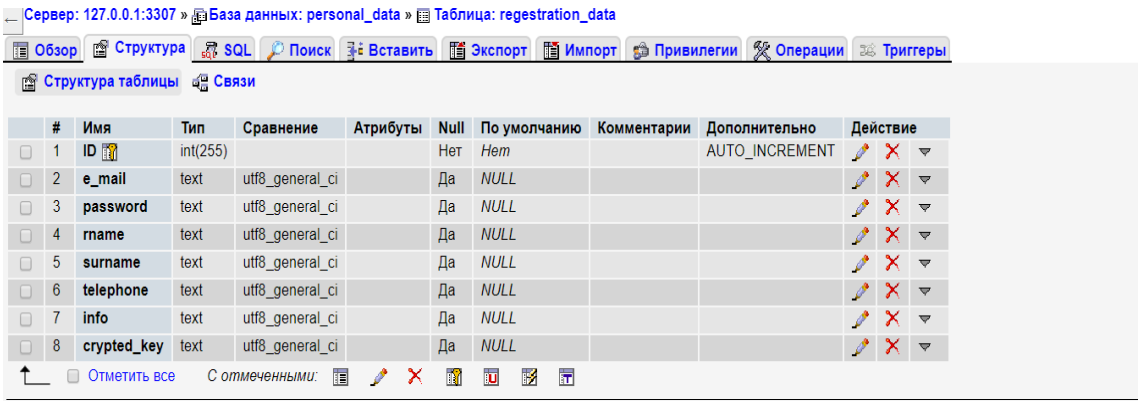

Рис. – 3.2 – структура бази даних

Для того щоб більш зрозуміло рядок в якому ми тримаємо ключ було названо “crypted_key”, в любу хвилину ми можемо перейменувати це поле, для того, щоб замаскувати його, наприклад назвати його “info2” .Ось в такому вигляді ми маємо базу даних (Рис. – 3.3)

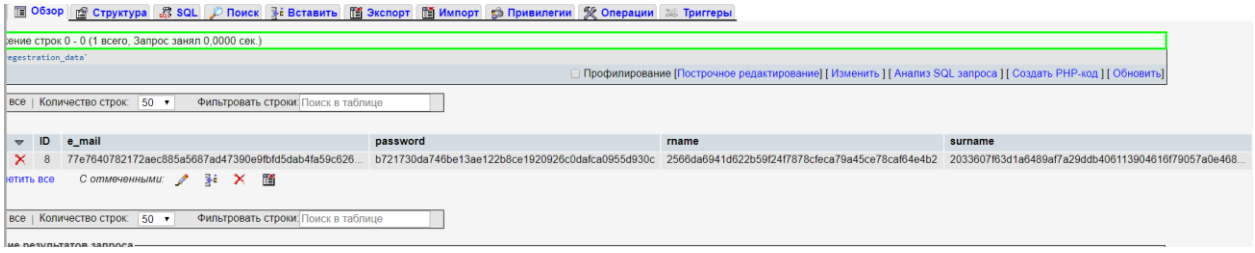

Рис. – 3.3 – база даних

Усі дані зашифровані, що робить безпечним роботу з ресурсом в якому є таке шифрування.

Для користувача це виглядає як звичайна форма реєстрації(Рис. – 3.4)

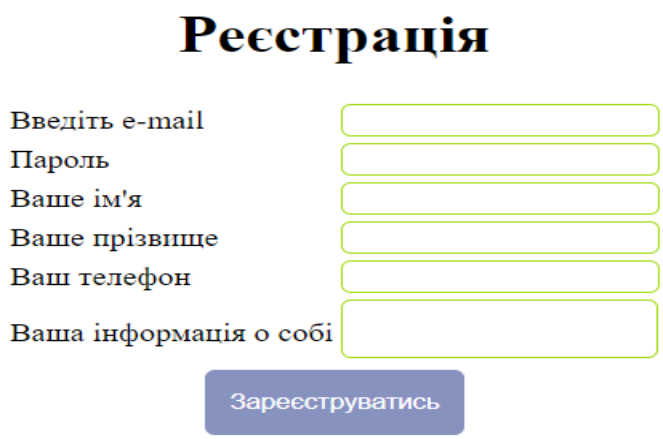

Рис. – 3.4 – форма реєстрації

Шифрування здійснювалось через штатні методи PHP.

Для шифрування й дешифрування ми використали методи з бібліотек Mcrypt й Mhash:

- mcrypt_encrypt (string $cipher , string $key , string $data , string $mode [, string $iv]) : string
- mcrypt_decrypt (string $cipher , string $key , string $data , string $mode [, string $iv]) : string
- mcrypt_get_iv_size (string $cipher , string $mode) : int
- mcrypt_create_iv (int $size [, int $source = MCRYPT_DEV_URANDOM]) : string
- pack (string $format [, mixed $...]) : string
- md5 (string $str [, bool $raw_output = FALSE]) : string
- bin2hex (string $str) : string

mcrypt_encrypt/decrypt – функція, яка приймає 4 обов’язкових параметрів й 1 необов’язковий.

$cipher - Одна з констант MCRYPT_ciphername чи назва алгоритма у вигляді строкового рядка.

$key - Ключ, з яким будуть шифруватися дані. Якщо довжина ключа не відповідає вимогам шифру, то буде повернуто FALSE і видано попередження.

$data - Дані, які будуть зашифровані за допомогою шифру cipher і режиму mode. Якщо розмір даних не кратний розміру блоку, то вони будуть доповнені Симв '\ 0'. Розмір тексту який повертається може бути більше розміру вихідних даних data.

$mode - Одна з констант MCRYPT_MODE_modename, або одна з наступних рядків: "ecb", "cbc", "cfb", "ofb", "nofb" і "stream".

$iv - Використовується для ініціалізації в режимах CBC, CFB, OFB, а також в деяких алгоритмах в режимі STREAM. Якщо переданий IV розмір не підтримується режимом зчеплення або IV не був переданий, а режим зчеплення його вимагає, функція згенерує попередження про помилку і поверне FALSE.[25,26,27]

За допомогою цієї функції ми будемо шифрувати дані.

mcrypt_get_iv_size – функція в яку передається 2 параметри.

Повертає розмір не започатковано вектора для відповідної комбінації шифру і режиму.

$cipher - Одна з констант MCRYPT_ciphername чи назва алгоритма у вигляді строкового рядка.

$mode - Одна з констант MCRYPT_MODE_modename, або одна з наступних рядків: "ecb", "cbc", "cfb", "ofb", "nofb" і "stream".

Ініціалізувалися вектор ігнорується в режимі ECB, так як він там не потрібен. Вам знадобиться один і той же започаткований вектор як для шифрування, так і для дешифрування, інакше все ваше шифрування перетвориться на гарбуз.

mcrypt_create_iv - функція в яку передається також 2 параметри.

Створює ініціалізуючий вектор (Initialization Vector або IV) з випадкового джерела

$size - розмір IV.

$source - Джерело IV. Джерело може бути задано однією з констант: MCRYPT_RAND (системний генератор випадкових чисел), MCRYPT_DEV_RANDOM (читає дані з / dev / random) або MCRYPT_DEV_URANDOM (читає дані з / dev / urandom). До версії 5.3.0, на Windows підтримувався тільки MCRYPT_RAND. Зверніть увагу, що до PHP 5.6.0 значенням за замовчуванням було MCRYPT_DEV_RANDOM.

Pack – функція, яка приймає одинобов'язковий параметр.

Упаковує задані аргументи в бінарний рядок відповідно до формату в параметрі format.

$format - Параметр format задається у вигляді рядка і складається з кодів формату і опціонального аргументу повторення. Аргумент може бути цілочисельним, або * для повторення до кінця введених даних. Для a, A, h, H число повторень визначає те, скільки символів взято від одного аргументу даних, для @ - це абсолютна позиція для розміщення таких даних, для всього іншого число повторень визначає як багато аргументів даних було оброблено і упаковано в результуючий бінарний рядок .[29,30]

md5 – функція з одним обов'зково приймаючим параметром й одним необов'зковим.

Обчислює MD5-хеш рядка str, використовуючи алгоритм MD5 RSA Data Security, Inc. і повертає цей хеш.

$str – текстовий рядок.

$raw_output - Якщо необов'язковий аргумент raw_output має значення TRUE, то повертається бінарний рядок з 16 символів.

bin2hex - Перетворює бінарні дані в шістнадцяткове подання

$str – текстовий рядок.

Таким чином, викори стовуючі усі функції описані вище, ми можемо програмно реалізувати шифрування персональних даних.

Ось так виглядають зашифрованні e-mail, ключ та розшифрованний e-mail(Рис. – 3.5)

Зашифрований e-mail - 1e2e81799c03188a05aa40dfff421301
Ключ - 81dc9bdb52d04dc20036dbd8313ed055
Розшифрований e-mail - wert@ra

Зашифрований e-mail - 049729b5178898e382ff3d33fb9a1ca9cf824f9924f52ec21b6dba22be83300c
Ключ - 2d592cd6f6338d99d879cddeb9e51219
Розшифрований e-mail - masterNP@rambler.ru

Зашифрований e-mail - 7294503839fa45d6bdd0a16da1477ff946f75b141ba6f145
Ключ - d89f3a35931c386956c1a402a8e09941
Розшифрований e-mail - fasdfsa@wrqw

Рис. - 3.5 – ітогові дані

Для створення програмного продукта використовувались такі технології:

- HTML
- PHP
- CSS
- MYSQL

А теперь про кожен файл, який відповідає за роботу програмного продукта, окремо.

Index.php – головна сторінка реєстрації та авторизації, тут користувачі можуть зареєструватись і в той же час пройде перевірка на оригінальність e-mail, або авторизуватись.

Папка parts – використовуючи технологію PHP ми можем підключати файли в яких є лише частинка всієї сторінки, таким чином це спрощує написання коду, так як в одному файлі ми можему держати логіку, а в іншому її візуальну картину.

functions.php – в цьому файлі містятся усі функції, які не входять в стандартний пакет язика, тут ми писали власноруч свої функції.

connection.php – цей файл відповідає за підключення до бази данних, один с важніх файлів, так як там знаходятся данні з доступами до самої бази, якщо цей файл попаде в руки зловмисника, то він отримає усю базу одразу.

auth.php – файл який відповідає за перевірку унікальності e-mail. В тому разі, якщо e-mail новий, то тут шифрують усі дані із форми і записуются в базу даних.

profile.php – файл, який відповідає за авторизацію користувача, тут користувачі, якщо прошли перевірку, то вони зможуть побачи ті дані, які вводили при реєстрації.

style.css – стилі, які ми використовували задля наших форм реєстрації та авторизації.

ddosDefense – файл, який відповідає за захист северверу від DDOS атаки

В результаті написанні програмного продукту ми створили можливість шифрувати персональні дані методом Blowfish.

Розлобленний програмний продукт являє собой захист персональних даних та може використовуватись компаніми та користувачами задля збереження данних в базі даних в шифрованому виді методом blowfish.

ВИСНОВКИ

Ми вирішили основу актуальну проблему сьогодення, захист даних які знаходятся в базі. В ході досягнені мети нашої дипломної роботи були вирішені поставлені задачі.

- Було проаналізовано існуючі сучасні технології, які дали змогу взаємодіяти клієнту з сервером.
- Проаналізували існуючі залежності в програмній платформі PHPStorm.
- Визначили основні загрози для серверу і що робити для його захисту.
- Розробили програмний продукт, який створювався на мові програмування PHP та програмної платформи PHPStorm.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гафнер, В.В. Информационная безопасность: учебное пособие / В.В. Гафнер. —К. : Феникс плюс, 2010. — 205-210 с.

2. Information security [Electronic resource] / Overview. History – Режим доступа: World Wide Web. — URL: https://en.wikipedia.org/w/index.php?title=Information_security&oldid=660081755 – Загл. з екрану (переглянуто 9 травня 2019).

3. Про захист персональних даних [Електронний ресурс] / Закон України – Режим доступа: World Wide Web. — URL: https://zakon.rada.gov.ua/laws/show/2297-17 – Загл. з екрану (дата звернення: 9 травня 2015).

4. Малюк, А.А. Информационная безопасность. Концептуальные и методологические основы защиты информации / А.А. Малюк. — М. : Горячая Линия - Телеком, 2004. — 105 с.

5. Стакснет [Електронний ресурс]. – Режим доступа: World Wide Web. — URL: https://uk.wikipedia. org/wiki/Стакснет – Загл. з екрану (дата звернення: 8 травня 2019).

6. В Бразилии раскрыта банда сетевых мошенников [Электронный ресурс]. – Режим доступа: World Wide Web. — URL: http://www.securitylab.ru/news/215150.php – Загл. з екрану (дата обращения: 8 травня 2019).

7. В інтернеті з’явилися комерційні секрети “Microsoft” - Korrespondent.net [Електронний ресурс]. – Режим доступа: World Wide Web. — URL: http://ua.korrespondent.net/ tech/247370-v-interneti-zyavilisya-komercijni-sekreti-microsoft – Загл. з екрану (дата звернення: 9 травня 2019).

8. Белокопытов, А.В. Современные информационные технологии: учебное пособие / А.В. Белокопытов. — Смоленск : ФГОУ ВПО «Смоленская сельскохозяйственная академия», 2013 — 13 с

9. Бабичев, С.Г. Основы современной криптографии / С.Г. Бабичев. — М. : ДИАЛОГ-МИФИ, 2011. — 30-35 с.

10. Сингх, С. Книга шифров: тайная история шифров и их расшифровки / С. Сингх. — [Б. м.] : Астрель, 2006. — 76 с.

11. Скляров, Д.В. Искусство защиты и взлома информации / Д.В. Скляров. — СПб. : БХВ-Петербург, 2004. — 36 с.

12. Kruchten, P. The Rational Unified Process: An Introduction / P. Kruchten. : Addison-Wesley Professional, 2003. — №1— P. 336

13. Postel, J. User Datagram Protocol [Електронний ресурс]. – Режим доступа: World Wide Web. — URL: https://tools.ietf.org/html/rfc768 – Загл. з екрану (дата звернення: 9 травня 2019).

14. SQL Ін’єкція – Режим доступа: World Wide Web. – URL:https://uk.wikipedia.org/wiki/SQL_ін%27єкція – Загл. з екрану (дата звернення: 9 травня 2019).

15. IEEE Standard for Software and System Test Documentation // IEEE Std 829-2008. — P. 150.

16. Download Speed by Country | Net Index from Ookla [Electronic resource] — Режим доступа: World Wide Web. — URL: http://www.netindex.com/download/allcountries/ — Загл. з екрану (дата звернення: 9 травня 2019).

17. Хакеры взломали сервер МВД: данные свободно разгуливают по Интернету [Электронный ресурс]. — Режим доступа: World Wide Web. — URL:http://censor.net.ua/news/196307/hakery_vzlomali_server_mvd_dannye_svobodno_razgulivayut_po_internetu – Загл. з екрану (дата звернення: 9 травня 2019).

18. Іванов М. А. ,Зенін О. С., Стандарт криптографічного захисту – AES. Кінцеві поля. М.: КУДИЦ – ОБРАЗ, 2003. 171 с.

19. Столінгс В. криптографія і захист мереж: принципи і практика. М.: видавничий дім «Вильямс», 2001. 677 с.

20. Шнаєр В. Прикладна криптографія. Протоколи, алгоритми, вихідні тексти на мові С. М.: Видавництво «ТРИУМФ», 2002.703 с.

21. Винокуров А. сторінка класичних блочних шифрів. [Электронный ресурс]. — Режим доступа: World Wide Web. — URL:http://www.enlight.ru/crypto/– Загл. з екрану (дата звернення: 9 травня 2019).

22. “On computing multiplicative inverses in GF(2n)”/ Brunner H. Curiger A., Hofstetter// IEEE Transactions on Computers ,1993, – P. 1010 – 1015.

23. New types of cryptanalytic attacks using related keys./ Biham E. // Advances in Criptology, Proceedings Eurocrypt’93, LNCS 765, T. Helleseth, Ed., Springer – Verlag, 1993 – P. 398 – 409.

24. Differential cryptanalysis of DES-like cryptosystems./ Biham E., Sharmir A. // Journal of Cryptology, Vol. 4, No. 1, 1991– P. 3 – 72..

25. Шифрування Blowfish [Электронный ресурс]. — Режим доступа: World Wide Web. — URL:http://www.kryptologia.private.pl/blowfish/– Загл. з екрану (дата звернення: 9 травня 2019).

26. HTML что это такое [Электронный ресурс]. — Режим доступа: World Wide Web. — URL:http://www.mgraphics.com.pl/blog/artykul/jezyk-html-opis. /– Загл. з екрану (дата звернення: 9 травня 2019).

27.CSS стили наше все [Электронный ресурс]. — Режим доступа: World Wide Web. — URL:https://www.w3.org/Style/Examples/007/figures.pl.html/– Загл. з екрану (дата звернення: 9 травня 2019).

28. Evgeny Milanov. The RSA Algorithm, 2009 [Электронный ресурс]. – Режим доступу: World Wide Web . — URL:https://sites.math.washington.edu/~morrow/336_09/papers/Yevgeny. pdf . – Загл. з екрану (дата звернення: 9 травня 2019).

29. G.N. Shinde, H.S. Fadewar. Faster RSA Algorithm for Decryption Using Chinese Remainder Theorem, [Электронный ресурс]. – Режим доступу: World Wide Web — URL:http://www.techscience.com/doi/10.3970/icces.2008.005.255.pdf – Загл. з екрану (дата звернення: 9 травня 2019).

30.PHP – мова програмування[Электронный ресурс]. – Режим доступу: World Wide Web — URL:http://www.mgraphics.com.pl/blog/artykul/php-opis– Загл. з екрану (дата звернення: 9 травня 2019).