МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КОЛЕДЖ КРЕМЕНЧУЦЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

для самостійної роботи

з дисципліни «ЗАХИСТ ІНФОРМАЦІЇ»
для студентів спеціальності 5.05010201 Обслуговування комп'ютерних систем і мереж

Кременчук 2017 р.

Загальні методичні вказівки

Вивчення предмету “Захист інформації”, включає лекції та самостійну роботу студентів над окремими темами курсу. Самостійна робота дозволяє поглибити знання, виробити вміння працювати з літературою, аналізувати інформацію та коротко, стисло її записувати.

При вивченні окремих тем предмета необхідно:

- Ознайомитись з переліком рекомендованої літератури;
- Ретельно опрацювати матеріал кожної теми;
- Законспектувати стисло опрацьований матеріал в логічній послідовності;
- Відповісти на питання для самоконтролю.

Модуль 1. Інформація, як об’єкт захисту.

Тема 1: Вступ. Інформація як об’єкт захисту. Інформаційна безпека. Загальні загрози інформаційної безпеки.

План

1. Інформація – об’єкт захисту.
2. Об’єкти захисту інформації.
3. Основні організіційні заходи.

Література: 1.с. 5..25

Питання для самоконтролю:

1. Що таке інформація?
2. Що називається загрозою?
3. Які бувають загрози інформації?
4. Що називається сніфер пакетом?
5. Для чого використовується мережна розвідка?
6. Що називається захистом інформації?
7. Що є основними об’єктами захисту інформації?

В загальному комплексі заходів щодо забезпечення інформаційної безпеки важливе місце займають заходи пов’язані із безпосереднім захистом інформації, що спрямована на реалізацію законних інтересів громадян, юридичних осіб, державних органів та здійснення ними своїх завдань і функцій, від загроз, реалізація яких може нанести особі, суспільству, державі політичні, економічні, моральні та інші збитки.

Конфіденційність інформації - це властивість інформації бути недоступною для несанкціонованого ознайомлення.

Загроза – будь-який потенційно можливий негативний вплив, будь-яка обставина або подія, які можуть призвести до порушення політики безпеки.

До можливих загроз відносяться такі:

1) стихійні лиха, техногенні катастрофи. Боротьба: резервне обладнання.
2) відмова обладнання. Боротьба: резервування обладнання та інформації, діагностика, правильний вибір постачальника;
3) наслідки помилок проектування:
    a. „правильний” вибір спеціалістів з побудови КСЗІ (ліцензій);
    b. експертиза;
4) навмисні і ненавмисні дії політики безпеки.

Основний недолік комп’ютерних мереж і телекомунікаційних мереж, які базуються на протоколах TCP або UDP полягає в тому, що інформація по цих каналах передається у відкритому вигляді.

В залежності від типу комп’ютерної мережі змінюється і характер основної загрози. Для локальних мереж основою загрози є порушення політик безпеки персоналом, з чим необхідно боротися адміністративними заходами.

Якщо система має вихід в глобальну мережу, до порушень персоналу додаються ще навмисні атаки порушників з боку глобальної мережі. Таким чином, основні загрози IP – мережам можна класифікувати наступним чином:

1) підслуховування (Sniffing). Packed Sniffer

Сніфер пакет – це прикладна програма, яка перехоплює всі пакети, що передаються мережам. Підслуховування використовується для подальших парольних атак, а також для мережної розвідки, коли необхідно визначити структуру мережі та діапазон доступних IP-адрес. Боротьба: антисніфер, криптографія, одноразові паролі

2) парольні атаки;

3) мережна розвідка – використовується для збирання інформації про структуру мережі. Використовуються сніфери або сканери мереж. Боротьба: а) IDS (Intrusion Detection Systems) – програмне забезпечення – системи виявлення вторгнень;

4)„маскарад" (IP -spoofing) – підміна IP-мережі, Боротьба: одноразові паролі, не зловживати довірчивими відносинами між серверами;

5) маскарад часто застосовується для підготовки та створення умов для інших небезпечних хакерських атак. Класичним прикладом такої атаки є відмова в обслуговуванні. DOS (Denial of Service) – відмова в обслуговуванні. Якщо атака відбувається з однієї IP- адреси, то це DOS-атака. Якщо ж з багатьох – DDOS – Distributed DOS – розподілена DOS - атака. Відмова в обслуговуванні – найбільш небезпечна хакерська атака. Суть її полягає в наступному. До сервера посилається велика кількість запитів через завчасно відкритий порт або через легальний порт, які він не може обробити. Таким чином, робота сервера блокується і легальні користувачі не отримують доступу до інформаційних ресурсів;

6) посередництво – (Man-in-middle). Посередництво – атака напрямлена на викрадення криптографічних ключів в асиметричних системах шифрування. Або ще називається „підміна відкритого ключа";

7) атаки на рівні прикладного ПЗ.

Захист інформації - це сукупність організаційних, технічних та правових заходів, спрямованих на запобігання нанесенню збитків інтересам власника інформації.

Основними об'єктами захисту інформації є:

1. Інформація з обмеженим доступом (ІзОД), тобто інформаційні ресурси, зокрема ті, що містять відомості, які належать або до державної таємниці, або до конфіденційної інформації.
2. Технічні засоби приймання, обробки, зберігання та передання інформації (ТЗПІ), а саме: системи та засоби інформатизації (обчислювальна техніка, інформаційно-обчислювальні комплекси, мережі та системи); програмні засоби (операційні системи, системи керування базами даних та інше загальносистемне і прикладне програмне забезпечення); автоматизовані системи керування; системи зв'язку; технічні засоби отримання, передання та обробки ІзОД (звукозапис, звукопідсилення, звукосупроводження, переговорні та телевізійні пристрої; засоби тиражування і виготовлення документів та інші технічні засоби обробки графічної, алфавітно-цифрової та текстової інформації), їх інформативні фізичні поля.
3. Допоміжні технічні засоби і системи (ДТЗС), тобто технічні засоби і системи, які не належать до ТЗПІ, але розташовані в приміщеннях, де оброблюється ІзОД. До них відносять технічні засоби відкритого телефонного або

гучномовного зв'язку, системи пожежної та охоронної сигналізації, система енергопостачання, радіотрансляційна мережа, система часофікації, енергопобутові прилади тощо, а також самі приміщення, де циркулює ІзОД.

Захист інформації від витоку по технічних каналах забезпечують проектно-архітектурними рішеннями, проведенням організаційних і технічних заходів, а також виявленням портативних закладних пристроїв.

Організаційні заходи - це спрямовані на захист інформації заходи, проведення яких не потребує спеціально розроблених технічних засобів. *До основних організаційних заходів відносять:*

- залучення до робіт для захисту інформації організацій, що мають ліцензії відповідних органів на діяльність в області ТЗІ;
- використання на об'єкті сертифікованих ТЗПІ та ДТЗС;
- встановлення КЗ навколо об'єкта;
- залучення до робіт із монтування апаратури, будування чи реконструкції об'єктів ТЗПІ організацій з відповідними ліцензіями;
- організацію контролю та обмеження доступу на об'єкти ТЗПІ та у виділені приміщення;
- введення територіальних, частотних, енергетичних, просторових і часових обмежень у режимах використання технічних засобів, що підлягають захисту;
- відключення технічних засобів, що мають елементи з якостями електроакустичних перетворювачів, від ліній зв'язку на період проведення секретних заходів.

Технічні заходи - це спрямовані на захист інформації заходи, проведення яких передбачає використання спеціальних технічних засобів, а також реалізацію технічних рішень.

Тема 2: Помаранчева книга США.

План

1. Шість базових вимог безпеки.
2. Чотири групи критеріїв ступеня захищеності.

Література: З.с. 230..442

Питання для самоконтролю:

1. Що називається політикою безпеки?
2. В чому полягають вимоги 1 та 2?
3. В чому полягають вимоги 3, 4 та 5?
4. Вимога 6.
5. Чим відрізняються системи захисту D і C?
6. Порівняйте системи захисту B і A.

“Критерії безпеки комп'ютерних систем” (“Trusted Computer System Evfluation Criteria”), які одержали неформальну, але назву , що міцно закріпилася - “Помаранчева книга”, були розроблені Міністерством оборони США в 1983 році з метою визначення вимог безпеки, запропонованих до апаратному, програмному і спеціальному забезпеченню комп'ютерних систем і виробітки відповідної методології аналізу політики безпеки, реалізованої в комп'ютерних системах військового призначення.

У даному стандарті були вперше нормативно визначені шість базових вимог безпеки. Перші чотири вимоги спрямовані безпосередньо на забезпечення безпеки інформації, а два останніх - на якість самих засобів захисту.

Вимога 1. Політика безпеки.

Система повинна підтримувати точно визначену політику безпеки. Можливість здійснення суб'єктами доступу до об'єктів повинна визначатися на підставі їхньої ідентифікації і набору правил керування доступом. Там, де необхідно, повинна використовуватися політика нормативного керування доступом, що дозволяє ефективно реалізувати розмежування доступу до категорірованої інформації (інформації, відзначеної грифом таємності, типу “секретно”, “абсолютно секретно” і т.д.)

Вимога 2. Мітки.

З об'єктами повинні бути асоційовані мітки безпеки, використовувані в якості атрибутів контрольного доступу. Для реалізації нормативного керування доступом система повинна забезпечувати можливість привласнювати кожному об'єкту мітку або набір атрибутів, що визначають ступінь конфіденційності (гриф таємності) об'єкта та режими доступу до цього об'єкта.

Вимога 3. Ідентифікація і аутентифікація.

Всі суб'єкти повинні мати унікальні ідентифікатори. Контроль доступу повинний здійснюватися на підставі результатів ідентифікації суб'єкта й об'єкта доступу, підтвердження істинності їхніх ідентифікаторів (аутентифікації) і правил розмежування доступу. Дані, використовувані для ідентифікації і аутентифікації, повинні бути захищені від несанкціонованого доступу, модифікації і знищення і

повинні бути асоційовані з всіма активними компонентами комп'ютерної системи, функціонування котрих критично з погляду безпеки.

Вимога 4. Реєстрація й урахування.

Для визначення ступеня відповідальності користувачів за дії в системі, усієї події , що відбуваються в ній, що мають значення з погляду безпеки, повинні відсліжуватися і реєструватися в захищеному протоколі. Система реєстрації повинна здійснювати аналіз загального потоку подій і виділяти з нього тільки ті події, що впливають на безпеку для скорочення обсягу протоколу і підвищення ефективності його аналізу. Протокол подій повинний бути надійно захищений від несанкціонованого доступу, модифікації або знищення.

Вимога 5. Контроль коректності функціонування засобів захисту.

Засоби захисту повинні містити незалежні апаратні та програмні компоненти, що забезпечують працездатність функції захисту. Це означає, що всі засоби захисту, що забезпечують політику безпеки, керування атрибутами і мітками безпеки, ідентифікацію і аутентифікацію, реєстрацію й урахування, повинні знаходиться під контролем засобів, що перевіряють коректність їхній функціонування. Основний принцип контролю коректності складається в тому, що засоби контролю повинні бути цілком незалежні від засобів захисту.

Вимога 6. Безперервність захисту.

Всі засоби захисту повинні бути захищені від несанкціонованого втручання і відключення, причому цей захист повинний бути постійної і безупинної в будь-якому режимі функціонування системи захисту і комп'ютерної системи в цілому. Дана вимога поширюється на весь життєвий цикл комп'ютерної системи. Крім того, його виконання є одним із ключових аспектів формального доказу безпеки системи.

“Помаранчева книга” передбачає чотири групи критеріїв, що відповідають різноманітного ступеня захищеності: від мінімальної (група D) до формально доведеної (група А). Кожна група включає один або декілька класів. Групи D і А містять по однім класі (класи D1 і А1 відповідно), група С - класи С1, С2, а група У - В1, В2, В3, що характеризуються різноманітними наборами вимог безпеки. Рівень безпеки зростає при прямуванні від групи D до групи А, а усередині групи - із зростанням класу.

Клас D1. До цього класу ставляться всі системи, що не задовольняють вимогам інших класів.

Класи С1 і С2 характеризуються наявністю довільного керування доступом і реєстрацією дій суб'єктів.

Клас С1. Системи цього класу задовольняють вимогам забезпечення поділу користувачів і інформації і включає засоби контролю і керування доступом, що дозволяють задавати обмеження для індивідуальних користувачів, що дає їм можливість захищати свою приватну інформацію від інших користувачів. Клас С1 розрахований на багато споживаєми системи, у яких здійснюється спільне опрацювання даних одного рівня таємності.

Клас С2. Системи цього класу здійснюють більш виборче керування доступом, чим системи класу С1. за допомогою застосування засобів індивідуального контролю за діями користувачів, реєстрацією, урахуванням подій і виділенням ресурсів.

Класи B1, B2 і B3 характеризуються нормативним доступом із використанням міток безпеки, підтримка моделі і політики безпеки, а також наявність специфікації на функції ТСВ. Для систем цієї групи монітор взаємодій повинний контролювати всі події в системі.

Клас B1. Системи класу B1 повинні відповідати усім вимогам, запропонованим до систем класу C2, і, крім того, повинні підтримувати визначену неформально модель безпеки, маркірування даних і нормативне керування доступом. При експорті із системи інформація повинна піддаватися маркіруванню. Виявлені в процесі тестування помилки повинні бути виправлені.

Клас B2. Для відповідності класу B2 ТСВ системи повинно підтримувати формально визначену і чітко документовану модель безпеки, що передбачає довільне і нормативне керування доступом, що поширюється в порівнянні із системами класу B1 на всі суб'єкти. Крім того, повинний здійснюватися контроль схованих каналів витоку інформації. У структурі ТСВ повинні бути виділені елементи, критичні з погляду безпеки. Інтерфейс ТСВ повинний бути чітко визначений, а його архітектура і реалізація повинні бути виконані з урахуванням можливості проведення тестових іспитів. У порівнянні з класом B1 повинні бути посилені засоби аутентифікації. Керування безпекою здійснюється адміністраторами системи. Повинні бути передбачені засоби керування конфігурацією.

Клас B3. Для відповідності цьому класу ТСВ системи повинні підтримувати монітор взаємодій, що контролює всі типи доступу суб'єктів до об'єктів і котрий неможливо обминути. Крім того, ТСВ повинно бути структуроване з метою винятку з нього підсистем, не відповідальних за реалізацію функції захисту, і бути достатньо компактно для ефективного тестування й аналізу. У ході розробки і реалізації ТСВ повинні застосовуватися методи і засоби, спрямовані на мінімізацію його складності. Засоби аудиту повинні включати механізми оповіщення адміністратора при виникненні подій, що мають значення для безпеки системи. Потрібно наявність засобів відновлення працездатності системи.

Клас A1. Системи класу A1 функціонально еквівалентні системам класу B3, і до них не подається ніяких додаткових функціональних вимог. У відмінності від систем класу B3 у ході розробки повинні застосовуватися формальні методи верифікації, що дозволяє з високою впевненістю одержати коректну реалізацію функцій захисту. Процес доказу адекватності реалізації починається на ранній стадії розробки з побудови формальної моделі політики безпеки і специфікації високого рівня. Для забезпечення методів верифікації системи класу A1 повинні містити більш потужні засоби керування конфігурацією і захищеною процедурою дистрибуції.

Опублікування “Помаранчевої книги” стало важливим етапом і зіграло значну роль у розвитку технологій забезпечення безпеки комп'ютерних систем. Проте в ході застосування її положень з'ясувалося, що частина практично важливих питань залишилася за рамками даного стандарту і, крім того, із часом (із моменту опублікування пройшло більш п'ятнадцятьох років) ряд положень застарів і зажадав перегляду.

Тема 3: Атакуємі мережні компоненти. Рівні мережевих атак згідно моделі OSI.

План

1. Класифікація атак.
2. Рівні моделі OSI.

Література: 2.с. 290..300

Питання для самоконтролю:

1. За якими ознаками можна класифікуати атаки?
2. Перерахувати рівні моделі OSI і охарактеризувати їх.
3. Як взаємодіють між собою рівні моделі OSI?
4. В чому полягає недоробка OSI?

Віддалені атаки можна класифікувати за такими ознаками:

1. За характером впливу – пасивні та активні. Пасивним впливом на розподілену обчислювальну систему, назвемо вплив, який надає безпосереднього впливу роботі системи, а може порушувати її політику безпеки. Під активним впливом на розподілену ОС, усвідомимо вплив, що надає безпосередній вплив роботі системи (зміна конфігурації РВР, порушення працездатності й т. буд.) і порушує прийняту у ній політику безпеки. Практично всі типи віддалених атак є активними впливами.

2. По меті впливу – нарушення конфіденційності інформації або ресурсів системи, порушення цілісності інформації, порушення працездатності (доступності) системи.

3. За умовою початку здійснення впливу – атака на запит від атакуемого об'єкта та атака по наступові очікуваної події на атакуемом об'єкті. У першому випадку атакуючий очікує передачі від потенційної мети атаки запиту певного типу, що й буде умовою початку здійснення впливу. А до другого випадку – атакуючий здійснює постійний нагляд станом ОС віддаленної мети атаки і за виникнення певної події з цією системою починає вплив.

4. За наявністю зворотного зв'язку з атакуемим об'єктом – зі зворотнім зв’язком та без зворотного зв’язку (односпрямована атака). Віддалена атака, здійснювана за наявності зворотного зв'язку з атакуемим об'єктом, характеризується тим, що у деякі запити, передані на атакований об'єкт, атакуючому потрібно одержати відповідь, отже, між атакуючим і метою атаки існує зворотний, що дозволяє атакуючому адекватно реагувати попри всі зміни, що відбуваються на атакуемому об'єкті. На відміну від атак із другого зв'язку, віддаленим атакам без зворотного зв'язку непотрібено реагувати на певні зміни, що відбуваються на атакуемому об'єкті.

5. По розташуванню суб'єкта атаки щодо атакуемого об’єкта – внутрішньосигментні та між сегментні.

6. За рівнем еталонної моделі ISO/OSI, у якій здійснюється вплив – фізичний, канальний, мережевий, транспортний, сеансовий, представницький, прикладний.

Модель OSI (англ. Open Systems Interconnection Reference Model - модель взаємодії відкритих систем) - абстрактна модель для мережних комунікацій і розробки мережевих протоколів. Представляє рівневий підхід до мережі. Кожен рівень обслуговує свою частину процесу взаємодії.

Історія

В 1978 році Міжнародний комітет зі стандартизації (ISO) розробив стандарт архітектури ISO 7498, для об'єднання різних мереж. У розробці брало участь 7 комітетів, кожному з них був відведений свій рівень. В 1980 році IEEE опублікував специфікацію 802, що детально описала механізми взаємодії фізичних пристроїв на канальному й фізичному рівнях моделі OSI. В 1984 році специфікація моделі OSI переглянули й прийняли як міжнародний стандарт для мережних комунікацій.

Рівні моделі OSI

Модель складається з 7-ми рівнів, розташованих вертикально один над іншим. Кожен рівень може взаємодіяти тільки зі своїми сусідами й виконувати відведені тільки йому функції.

1 рівень – Фізичний рівень (Physical layer). Найнижчий рівень моделі, призначений безпосередньо для передачі потоку даних. Здійснює передачу електричних або оптичних сигналів у кабель і відповідно їхній прийом і перетворення в біти даних відповідно до методів кодування цифрових сигналів. Інакше кажучи, здійснює інтерфейс між мережним носієм і мережним пристроєм. На цьому рівні працюють концентратори й повторювачі (ретранслятори) сигналу. Цей рівень приймає кадр даних від канального рівня, кодує його в послідовність сигналів, які потім передаються у лінію зв'язку. Передача кадру даних через лінію зв'язку вимагає від фізичного рівня визначення наступних елементів: тип середовища передавання (дротовий або бездротовий, мідний кабель або оптичне волокно) і відповідних конекторів; як повинні бути представлені біти даних у середовищі передавання; як кодувати дані; якими повинні бути схеми приймача і передавача. В сучасних мережах використовуються 3 основних типа середовища передавання: мідний кабель (copper), оптичне волокно (fiber) та бездротове середовище передавання (wireless). Тип сигналу, за допомогою якого здійснюється передача даних, залежить від типу середовища передавання. Для мідного кабелю сигнали, що представляють біти даних є електричними імпульсами, для оптичного волокна - імпульсами світла. У випадку використання бездротових з'єднань сигнали є радіохвилями (електромагнітними хвилями). Основними функціями фізичного рівня є: фізичні компоненти, кодування даних, передача даних. Фізичні компоненти - електронне обладнання, середовище передавання і конектори, через які передаються сигнали, що представляють біти даних. Кодування є процесом, за допомогою якого потік бітів даних перетворюється у певний код. Кодування здійснюється над групою бітів. Це необхідно для того, щоб забезпечити створення передбачуваної комбінації кодів, яка буде правильно розпізнаватися як передавачем, так і приймачем.

2 рівень – Канальний рівень (Data Link layer). Цей рівень призначений для забезпечення взаємодії мереж на фізичному рівні й контролю за помилками, які можуть виникнути. Отримані з фізичного рівня дані він упаковує в кадри даних, перевіряє на цілісність, якщо потрібно виправляє помилки й відправляє на мережний рівень. Канальний рівень може взаємодіяти з одним або декількома

фізичними рівнями, контролюючи й управляючи цією взаємодією. Специфікація IEEE 802 розділяє цей рівень на 2 підрівня - MAC (Media Access Control) регулює доступ до поділюваного фізичного середовища, LLC (Logical Link Control) забезпечує обслуговування мережного рівня. На цьому рівні працюють комутатори, мости й мережні адаптери.

3 рівень – Мережевий рівень (Network layer). Призначений для визначення шляху передачі даних. Відповідає за трансляцію логічних адрес й імен у фізичні, визначення найкоротших маршрутів, комутацію й маршрутизацію пакетів, відстеження неполадок і заторів у мережі. На цьому рівні працює такий мережний пристрій, як маршрутизатор.

4 рівень – Транспортний рівень (Transport layer). Призначений для доставлення даних без помилок, втрат і дублювання в тій послідовності, у якій вони були передані. При цьому не має значення, які дані передаються, звідки й куди, тобто він визначає сам механізм передачі. Блоки даних він розділяє на фрагменти, розмір яких залежить від протоколу, короткі об’єднує в один, довгі розбиває. Протоколи цього рівня призначені для взаємодії типу точка-точка.

5 рівень – Сеансовий рівень (Session layer). Відповідає за підтримку сеансу зв'язку, дозволяючи додаткам взаємодіяти між собою тривалий час. Рівень управляє створенням/завершенням сеансу, обміном інформацією, синхронізацією завдань, визначенням права на передачу даних і підтримкою сеансу в періоди неактивності додатків. Синхронізація передачі забезпечується розміщенням у потік даних контрольних точок, починаючи з яких відновляється процес при порушенні взаємодії.

6 рівень – Рівень відображення(Presentation layer). Цей рівень відповідає за перетворення протоколів і кодування/декодування даних. Запити додатків, отримані з прикладного рівня, він перетворить у формат для передачі по мережі, а отримані з мережі дані перетворить у формат, зрозумілий додаткам. На цьому рівні може здійснюватися стиснення/розпакування або кодування/декодування даних, а також перенапрямок запитів іншому мережному ресурсу, якщо вони не можуть бути оброблені локально.

7 рівень – Прикладний рівень (Application layer). Верхній (7-й) рівень моделі, забезпечує взаємодію мережі й користувача. Рівень дозволяє додаткам користувача доступ до мережних служб, таким як обробник запитів до баз даних, доступ до файлів, пересиланню електронної пошти. Також відповідає за передачу службової інформації, надає додаткам інформацію про помилки й формує запити до рівня подання.

Рівні взаємодіють зверху вниз і знизу нагору за допомогою інтерфейсів і можуть ще взаємодіяти з таким же рівнем іншої системи за допомогою протоколів.

Основна недоробка OSI - непродуманий транспортний рівень. На ньому OSI дозволяє обмін даними між додатками (вводячи поняття порту — ідентифікатора додатка), однак, можливість обміну простими датаграмами в OSI не передбачена — транспортний рівень повинен утворювати з'єднання, забезпечувати доставку, управляти потоком і т.п. Реальні ж протоколи реалізують таку можливість.

Тема 4. Прості криптосистеми

План

1. Шифрування методом заміни (підстановки).
2. Шифрування методом перестановки.
3. Шифрування методом гаммирования.
4. Шифрування за допомогою аналітичних перетворень.
5. Комбіновані методи шифрування.

Література: 1.с. 93..95

Питання для самоконтролю:

1. Що є найбільш ефективними засобами захисту інформації в автоматизованих системах?
2. Якими показниками характеризується будь-який криптографічний метод?
3. Які основні вимоги до криптографічного закриття інформації в АС?
4. Як класифікуються основні методи криптографічного закриття інформації?
5. В чому полягає шифрування методом заміни?
6. В чому полягає шифрування методом перестановки?
7. Які основні правила криптозахисту?
8. Що є основними правилами механізму розподілу ключів?

Криптографічні методи є найбільш ефективними засобами захисту інформації в автоматизованих системах (АС). А при передачі інформації з протяжних ліній зв'язку вони є єдиним реальним засобом запобігання несанкціонованого доступу.

Будь-який криптографічний метод характеризується такими показниками, як стійкість і трудомісткість:

- стійкість методу - це той мінімальний обсяг зашифрованого тексту, статистичним аналізом якого можна розкрити вихідний текст. У такий спосіб стійкість шифру визначає припустимий обсяг інформації, зашифровуваний при використанні одного ключа;
- трудомісткість методу визначається числом елементарних операцій, необхідних для шифрування одного символу вихідного тексту.

Основні вимоги до криптографічного закриття інформації в АС

1. Складність і стійкість криптографічного закриття даних повинні вибиратися в залежності від обсягу і ступеня таємності даних.
2. Надійність закриття повинна бути такий, щоб таємність не порушувалася навіть у тому випадку, коли зловмиснику стає відомий метод шифрування.
3. Метод закриття, набір використовуваних ключів і механізм їхнього розподілу не повинні бути занадто складними.
4. Виконання процедур прямого і зворотного перетворень повинне бути формальним. Ці процедури не повинні залежати від довжини повідомлень.
5. Помилки, що виникають у процесі перетворення не повинні поширюватися по системі.
6. Внесена процедурами захисту надмірність повинна бути мінімальною.

Класифікація основних методів криптографічного закриття інформації

I. Шифрування

1. Підстановка (заміна)
a. Одноалфавітна
b. Багатоалфавітна одноконтурна звичайна
c. Багатоалфавітна одноконтурна монофонічна
d. Багатоалфавітна багатоконтурна

2. Перестановка

a. Проста
b. Ускладнена по таблиці
c. Ускладнена по маршрутах

3. Гаммування

a. З кінцевою короткою гамою
b. З кінцевою довгою гамою
c. З нескінченною гамою

4. Аналітичні перетворення

a. Матричні
b. По особливих залежностях

5. Комбіновані

a. Підстановка+перестановка
b. Підстановка+гаммування
c. Перестановка+ гаммування
d. Гаммування+ гаммування

II. Кодування

1. Значеннєве
a. По спеціальних таблицях
2. Символьне
a. По кодовому алфавіту

III. Інші види

1. Рознесення-розсічення-рознесення
a. Значеннєве
b. Механічне
2. Розширення-поширення-стиск-розширення

Починаючи розмову про шифрування, визначимося з термінологією на прикладі фільму "Сімнадцять митей весни": Юстас - зашифровує, Алекс - розшифровує, а старовина Мюллер - дешифрує повідомлення.

Шифрування методом заміни (підстановки)

Найбільш простий метод шифрування. Символи зашифрованного тексту заміняються іншими символами, узятими з одного алфавіту (одноалфавітна заміна) чи декількох алфавітів (багатоалфавітна підстановка).

Одноалфавітна підстановка

Найпростіша підстановка - пряма заміна символів шифруємого повідомлення іншими буквами того ж самого чи іншого алфавіту.

Приклади таблиць заміни:

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я
М Л Д О Т В А Ч К Е Ж Х Щ Ф Ц Э Г Б Я Ъ Ш Ы З И Ь Н Ю У П С Р Й

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я
Q W E R T Y U I O P [ ] A S D F G H J K L Z X C V B N M < > @ %

Стійкість методу простої заміни низька. Зашифрований текст має ті ж самі статистичні характеристики, що і вихідний, тому знаючи стандартні частоти появи символів у тій мові, на якому написане повідомлення, і підбираючи по частотах появи символи в зашифрованому повідомленні, можна відновити таблицю заміни. Для цього потрібно лише досить довгий зашифрований текст, для того, щоб одержати достовірні оцінки частот появи символів. Тому просту заміну використовують лише в тому випадку, коли зашифроване повідомлення достатньо коротке!

Стійкість методу дорівнює 20 - 30, трудомісткість визначається пошуком символу в таблиці заміни. Для зниження трудомісткості при шифруванні таблиця заміни сортується по зашифрованих символах, а для розшифровки формується таблиця дешифрування, що виходить з таблиці заміни сортуванням по символах, що заміняють.

Багатоалфавітна заміна підвищує стійкість шифру.

Багатоалфавітна одноконтурна звичайна підстановка

Для заміни символів використовуються кілька алфавітів, причому зміна алфавітів проводиться послідовно і циклічно: перший символ заміняється на відповідний символ першого алфавіту, другий - із другого алфавіту, і т.д. поки не будуть вичерпані всі алфавіти. Після цього використання алфавітів повторюється.

Розглянемо шифрування за допомогою таблиці Віжинера - квадратної матриці з n^2 елементами, де n - число символів використовуваного алфавіту. У першому рядку матриці міститься вихідний алфавіт, кожна наступна рядок виходить з попередньої циклічним зрушенням уліво на один символ.

Таблиця Віжинера для російського алфавіту:

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я
Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А
В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б
Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В
Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г
Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д
Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е
З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж
И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З
Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И
К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й
Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К
М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л
Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М
О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н
П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О
Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П
С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р
Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С
У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т

Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У
Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф
Ц Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х
Ч Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц
Ш Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч
Щ Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш
Ь Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ
Ы Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь
Ъ Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы
Э Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ
Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э
Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю

Для шифрування необхідно задати ключ - слово з неповторюваними символами. Таблицю заміни одержують у такий спосіб: рядок "Символи шифруємого тексту" формують з першого рядка матриці Вижинера, а рядка з розділу "символи, що заміняють", утворяться з рядків матриці Вижинера, перші символи яких збігаються із символами ключового слова.

При шифруванні і дешифруванні немає необхідності тримати в пам'яті всю матрицю Вижинера, оскільки використовуючи властивості циклічного зрушення, можна легко обчислити будь-який рядок матриці по її номері і першому рядку.

При шифруванні символи з першого рядка заміняються символами інших рядків за правилом

a(1,i) -> a(k,i),

де k - номер використовуваної для шифрування рядка.

Використовуючи властивості циклічного зрушення вліво елементи k-ої рядка можна виразити через елементи першого рядка

a(1,i+k-1), якщо i<=n-k+1
a(k,i)=
a(1,i-n+k-1), якщо i>n-k+1

При дешифруванні виробляється зворотна заміна

a(k,i) -> a(1,i).

Тому необхідно вирішити наступну задачу: нехай черговий дешифрований символ у тексті - a(1,j) і для дешифрування використовується k-я рядок матриці Віжинера. Необхідно знайти в k-ої рядку номер елемента, рівного a(1,j). Очевидно,

a(k,j-k+1), якщо j>=k
a(1,j)=
a(k,n-k+j+1), якщо j<k

У такий спосіб при дешифруванні по k-ої рядку матриці Віжинера символу з зашифрованого тексту, значення якого дорівнює a(1,j), проводиться зворотна підстановка

a(1,j-k+1), якщо j>=k
a(1,j) ->
a(1,n-k+j+1), якщо j<k

Стійкість методу дорівнює стійкості методу підстановки, помноженої на кількість використовуваних при шифруванні алфавітів, тобто на довжину ключового слова і дорівнює 20*L, де L - довжина ключового слова.

З метою підвищення стійкості шифрування пропонуються наступні удосконалення таблиці Віжинера:

1. В усіх (крім першої) рядках таблиці букви розташовуються в довільному порядку.
2. Як ключ використовуються випадкові послідовності чисел, що задають номера використовуваних рядків матриці Вижинера для шифрування.

Багатоалфавітна одноконтурна монофонічна підстановка

У монофонічній підстановці кількість і склад алфавітів вибирається таким чином, щоб частоти появи всіх символів у зашифрованому тексті були однаковими. При такому положенні ускладнюється криптоаналіз зашифрованого тексту за допомогою його статистичної обробки. Вирівнювання частот появи символів досягається за рахунок того, що для часто зустрічаються символів вихідного тексту передбачається більше число символів, що заміняють, ніж для тих, що рідко зустрічаються.

Приклад таблиці монофонічної заміни:

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ы Ъ Э Ю Я _
Ф Н (Щ И Г Е R А Д Ы ~ @ S Л Я Ж ^ С Ш М Б Q П Т Х Ю Ъ Р } \ _ #
* Н У Щ D + Е R = Д Ц Й Ч [У Ь) О & { М Б Q П Т Х Ю Ъ Р } \ _ <
Л Н (Щ И] Е R % Д Ы ~ @ G / Я Э З " Ш М Б Q П Т Х Ю Ъ Р } \ _ W
Ф Н У Щ D К Е R А Д Ц Й Ч S + Ь Ж ^ С { М Б Q П Т Х Ю Ъ Р } \ _ V

Шифрування проводиться так само, як і при простій підстановці, з тією лише різницею, що після шифрування кожного символу відповідний йому стовпець алфавітів циклічно зрушується нагору на одну позицію. Таким чином, стовпці алфавітів як би утворять незалежні друг від друга кільця, що повертаються нагору на один знак щораз після шифрування відповідного знака вихідного тексту.

Багатоалфавітна багатоконтурна підстановка

Багатоконтурна підстановка полягає в тому, що для шифрування використовуються кілька наборів (контурів) алфавітів, використовуваних циклічно, причому кожен контур у загальному випадку має свій індивідуальний період застосування. Часто багатоконтурну поліалфавітну підстановку здійснюють заміною по таблиці Віжинера, якщо для шифрування використовується кілька ключів, кожний з який має свій період застосування.

Загальна модель шифрування підстановкою може бути представлена в наступному виді:

$tш = to + w \bmod (k-1)$,

де tш - символ зашифрованого тексту,

to - символ вихідного тексту,

w - ціле число в діапазоні 0 - (k-1),

k - число символів використовуваного алфавіту.

Якщо w фіксовано, то формула описує одноалфавітну підстановку, якщо w вибирається з послідовності w1 ,w2 ,...,wn, то виходить багатоалфавітна підстановка з періодом n.

Якщо в багатоалфавітній підстановці n > m (де m - число знаків шифруємого тексту) і будь-яка послідовність wi, i=1,2,...,n використовується тільки один раз, то такий шифр є не розкриваємим теоретично. Такий шифр одержав назву шифру Вермена.

Стійкість простої багатоалфавітної підстановки оцінюється величиною 20*n, де n - число різних алфавітів, використовуваних для заміни. Ускладнення багатоалфавітної підстановки істотно підвищує її стійкість. Монофонічна підстановка може бути дуже стійкою (і навіть нерозкриваємою теоретично), однак суворо монофонічну підстановку реалізувати на практиці важко, а будь-які відхилення від монофонічності знижують реальну стійкість шифру.

Шифрування методом перестановки

При шифруванні перестановкою символи шифруємого тексту переставляються за визначеними правилами усередині шифруємого блоку цього тексту.

Проста перестановка

Вибирається розмір блоку шифрування в n стовпців і m рядків і ключова послідовність, що формується з натурального ряду чисел 1,2,...,n випадковою перестановкою.

Шифрування проводиться в наступному порядку:

1. Шифруємий текст записується послідовними рядками під числами ключової послідовності, утворити блок шифрування розміром n*m.
2. Зашифрований текст виписується колонками в порядку зростання номерів колонок, що задаються ключовою послідовністю.
3. Заповнюється новий блок і т.д.

Наприклад, зашифруємо текст

ГРУЗИТЕ_АПЕЛЬСИНЫ_БОЧКАХ

блоком розміром 8*3 і ключем 5-8-1-3-7-4-6-2.

Таблиця простої перестановки буде мати вид:

Ключ						
5 8	1	3	7	4	6	2
Г Р	У	З	И	Т	Е	_
АП	Е	Л	Ь	С	И	Н
Ы	_	Б	О	Ч	К	А

Зашифроване повідомлення:

УЕБ_НХЗЛОЕСЛГАЫЕИАИЬЧРП_

Розшифрування виконується в наступному порядку:

1. З зашифрованого тексту виділяється блок символів розміром n*m.
2. Цей блок розбивається на n груп по m символів.
3. Символи записуються в ті стовпці таблиці перестановки, номера яких збігаються з номерами груп у блоці. Розшифрований текст читається по рядках таблиці перестановки.
4. Виділяється новий блок символів і т.д.

Перестановка, ускладнена по таблиці

При ускладненні перестановки по таблицях для підвищення стійкості шифру в таблицю перестановки вводяться невикористані клітки таблиці. Кількість і розташування невикористаних елементів є додатковим ключем шифрування.

При шифруванні тексту, в невикористані елементи не заносяться символи тексту й у зашифрований текст із них не записуються ніякі символи - вони просто

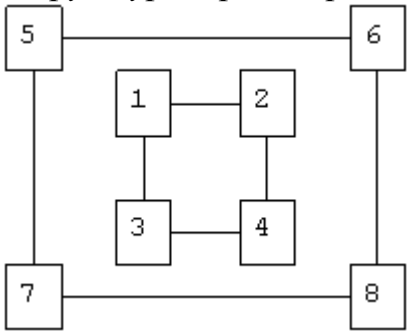
пропускаються. При розшифровці символи зашифрованого тексту також не заносяться в невикористані елементи.

Для подальшого збільшення криптостійкості шифру можна в процесі шифрування змінювати ключі, розміри таблиці перестановки, кількість і розташування невикористаних елементів по деякому алгоритмі, причому цей алгоритм стає додатковим ключем шифру.

Перестановка, ускладнена по маршрутах

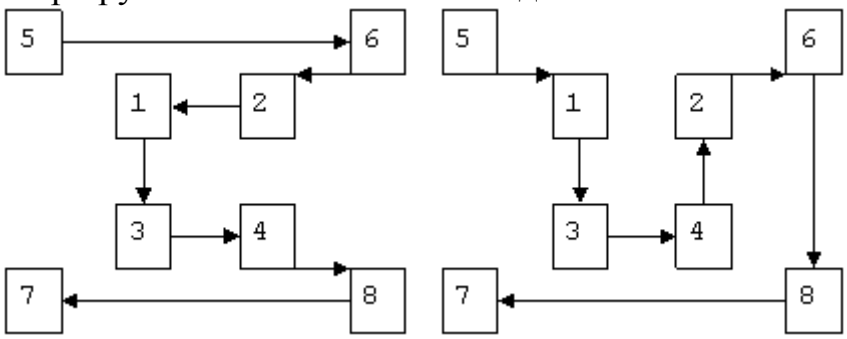
Високу стійкість шифрування можна забезпечити ускладненням перестановок по маршрутах типу гамильтоновських. При цьому для запису символів шифруємого тексту використовуються вершини деякого гіперкуба, а знаки зашифрованого тексту зчитуються по маршрутах Гамильтона, причому використовуються кілька різних маршрутів. Для приклада розглянемо шифрування по маршрутах Гамільтона при n=3.

Структура тривимірного гіперкуба:



Номера вершин куба визначають послідовність його заповнення символами шифруємого тексту при формуванні блоку. У загальному випадку n-мірний гіперкуб має n^2 вершин.

Маршрути Гамільтона мають вид:



Послідовність перестановок символів у шифруємому блоці для першої схеми 5-6-2-1-3-4-8-7, а для другий 5-1-3-4-2-6-8-7. Аналогічно можна одержати послідовність перестановок для інших маршрутів: 5-7-3-1-2-6-8-4, 5-6-8-7-3-1-2-4, 5-1-2-4-3-7-8-6 і т.д.

Розмірність гіперкуба, кількість вид обираних маршрутів Гамільтона складають секретний ключ методу.

Стійкість простої перестановки однозначно визначається розмірами використовуваної матриці перестановки. Наприклад, при використанні матриці 16*16 число можливих перестановок досягає 1.4E26. Таке число варіантів неможливо перебрати навіть з використанням ЕОМ. Стійкість ускладнених перестановок ще вище. Однак варто мати на увазі, що при шифруванні

перестановкою цілком зберігаються ймовірні характеристики вихідного тексту, що полегшує криптоаналіз.

Шифрування методом гаммування

Суть методу полягає в тому, що символи шифруємого тексту послідовно складаються із символами деякої спеціальної послідовності, називаною гамою. Іноді такий метод представляють як накладення гами на вихідний текст, тому він одержав назву " гаммування ".

Накладення гами можна здійснити декількома способами, наприклад по формулі

tш = to XOR tг,

де tш,to,tг - ASCII коди відповідно зашифрованого символу, вихідного символу і гами. XOR - побітова операція " що виключає чи".

Розшифрування тексту проводиться по тій же формулі:

to = tш XOR tг,

Послідовність гами зручно формувати за допомогою датчика псевдослучайних чисел (ПСЧ).

Стійкість гаммирования однозначно визначається довжиною періоду гами. При використанні сучасних ПСЧ реальним стає використання нескінченної гами, що приводить до нескінченної теоретичної стійкості зашифрованого тексту.

Шифрування за допомогою аналітичних перетворень

Досить надійне закриття інформації може забезпечити використання при шифруванні деяких аналітичних перетворень. Наприклад, можна використовувати методи алгебри матриць - зокрема множення матриці на вектор.

Як ключ задається квадратна матриця ||a|| розміру n*n. Вихідний текст розбивається на блоки довжиною n символів. Кожен блок розглядається як n-мірний вектор. А процес шифрування блоку полягає в одержанні нового n-мірного вектора (зашифрованого блоку) як результату множення матриці ||a|| на вихідний вектор.

Розшифрування тексту відбувається за допомогою такого ж перетворення, тільки за допомогою матриці, зворотної ||a||. Очевидно, що ключова матриця ||a|| повинна бути не виродженою.

Комбіновані методи шифрування

Досить ефективним засобом підвищення стійкості шифрування є комбіноване використання декількох різних способів шифрування, тобто послідовне шифрування вихідного тексту за допомогою двох чи більш методів.

Стійкість комбінованого шифрування S не нижче добутку стійкості використовуваних способів

*S >= S1*S2*...*Sk*

Якщо який-небудь спосіб шифрування при незалежному застосуванні може забезпечити стійкість не нижче S, то комбінувати його з іншими способами доцільно лише при виконанні умови

R > R1+R2+ ... +Rk

де Ri - трудомісткість і-го способу, використовуваного при комбінованому шифруванні, R - трудомісткість того способу, що забезпечує стійкість не нижче S.

Організаційні проблеми криптозахисту

Розглянуті значення стійкості шифрів є потенційними величинами. Вони можуть бути реалізовані при строгому дотриманні правил використання криптографічних засобів захисту.

Основні правила криптозахисту:

1. Збереження в таємниці ключів.
2. Виключення дублювання.
3. Досить часта зміна ключів.

Під дублюванням тут розуміється повторне шифрування того самого уривка тексту з використанням тих же ключів (наприклад, якщо при першому шифруванні відбувся збій). Порушення цього правила різко знижує надійність шифрування, тому що вихідний текст може бути відновлений за допомогою статистичного аналізу двох варіантів зашифрованого тексту.

Найважливішим правилом криптозахисту є досить часта зміна ключів. Причому частота може визначатися виходячи з тривалості використання чи ключа виходячи з обсягу зашифрованого тексту. При цьому зміна ключів за графіком є захисною мірою проти можливого їхнього розкрадання, зміна після шифрування визначеного обсягу тексту - від розкриття шифру статистичними методами.

Не можна допускати зловмиснику можливості направити в систему ряд спеціально підібраних повідомлень і одержувати їх у зашифрованому виді. Такого злому не може витримати жодна криптосистема!

Важливими аспектами організації криптозахисту є вибір способу закриття, розподіл ключів і доставка їхній у місця користування (механізм розподілу ключів).

Вибір способу захисту тісно зв'язаний із трудомісткістю методу шифрування, ступенем таємності закриваємих даних, стійкістю методу й обсягом шифруємої інформації.

Один із принципів криптографії є припущення про нетаємність методу закриття інформації. Передбачається, що необхідна надійність закриття забезпечується тільки за рахунок збереження в таємниці ключів. Звідси випливає принципова важливість формування ключів, розподілу їх і доставка в пункти призначення. Основними правилами механізму розподілу ключів є:

1. Ключі повинні вибиратися випадково.
2. Обрані ключі повинні розподілятися таким чином, щоб не було закономірностей у зміні ключів від користувача до користувача.
3. Повинна бути забезпечена таємниця ключів на всіх етапах функціонування системи. Ключі повинні передаватися по лініях зв'язку, чи пошті кур'єрами в зашифрованому виді за допомогою іншого ключа. На практиці часто утвориться ієрархія ключів шифрування, у якій ключі нижнього рівня при пересиланні шифруються за допомогою ключів верхнього рівня. Ключ у вершині ієрархії не шифрується, а задається і зберігається в довіреного обличчя, розсилається користувачам кур'єрами. Чим нижче рівень ключа, тим частіше він міняється і розсилається по лініях зв'язку. Подібна схема шифрування ключів часто використовується в мережах.

Тема 5: Стандарт шифрування даних DES.

План

1. Поняття стандарту шифрування DES.
2. Осноні положення.
3. Структура DES.
4. Фунції DES.

Література: 1.с. 215..217

Питання для самоконтролю:

1. Що таке блоковий шифр DES?
2. Яка історія виникнення блокового шифру DES?
3. Які існують основні положення блокового шифру DES?
4. Структура DES.
5. Як відбувається шифрування DES?
6. Що називається функцією DES?
7. 16 раундів блокового шифру.

Стандарт шифрування даних (DES) — блоковий шифр із симетричними ключами, розроблений Національним Інститутом Стандартів і Технології (NIST – National Institute of Standards and Technology). У l973 року NIST видав запит для розробки пропозиції національної криптографічної системи із симетричними ключами. Запропонована IBM модифікація проекту, названа Lucifer, була прийнята як DES. DES були видані в ескізному виді у Федеральному Регістрі в березні 1975 року як Федеральний Стандарт Обробки Інформації (FIPS – Federal Information Processing Standard). Після публікації ескіз строго критикувався із двох причин. Перша: критикувалася сумнівно маленька довжина ключа (тільки 56 бітів), що могло зробити шифр уразливим до атаки "грубої силою". Друга причина: критики були стурбовані деякою схованою побудовою внутрішньої структури DES. Вони підозрювали, що деяка частина структури (S-Блоки) може мати сховану лазівку, яка дозволить розшифровувати повідомлення без ключа. Згодом проектувальники IBM повідомили, що внутрішня структура була дороблена, щоб запобігти криптоаналіз. DES був нарешті виданий як FIPS 46 у Федеральному Регістрі в січні 1977 року. Однак FIPS оголосив DES як стандарт для використання в неофіційних додатках. DES був найбільше широко використовуваним блоковим шифром із симетричними ключами, починаючи з його публікації. Пізніше NIST запропонував новий стандарт (FIPS 46-3), який рекомендує використання потрійного DES (трикратно повторений шифр DES) для майбутніх додатків.

Основні положення

Як показано на рисунку 1, DES — блоковий шифр.

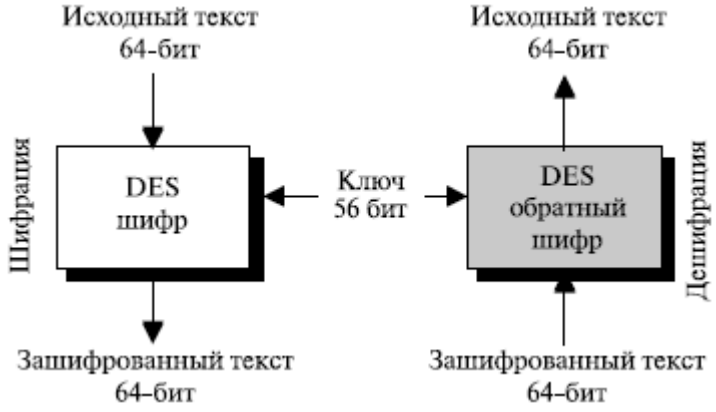


Рисунок 1 – Шифрування й дешифрування в DES

На стороні шифрування DES ухвалює 64-бітовий вихідний текст і породжує 64-бітовий зашифрований текст; на стороні дешифрування DES ухвалює 64-бітовий зашифрований текст і породжує 64-бітовий вихідний текст. На обох сторонах для шифрування й дешифрування застосовується той самий 56-бітовий ключ.

Структура DES

Розглянемо спочатку шифрування, а потім дешифрування. Процес шифрування складається із двох перестановок (Р-Блоки) — вони називаються початкові й кінцеві перестановки, — і шістнадцяти раундів Файстеля.

Кожна з перестановок ухвалює 64-бітовий вхід і переставляє його елементи за заданим правилом. Ці перестановки — прямі перестановки без ключів, які інверсні один одному. Наприклад, у початковій перестановці 58-й біт на вході переходить у перший біт на виході. Аналогічно, у кінцевій перестановці перший вхідний біт переходить в 58-й біт на виході. Інакше кажучи, якщо між цими двома перестановками не існує раунду, 58-й біт, що зробив на вхід обладнання початкової перестановки, буде доставлений на 58-й вихід фінальною перестановкою.

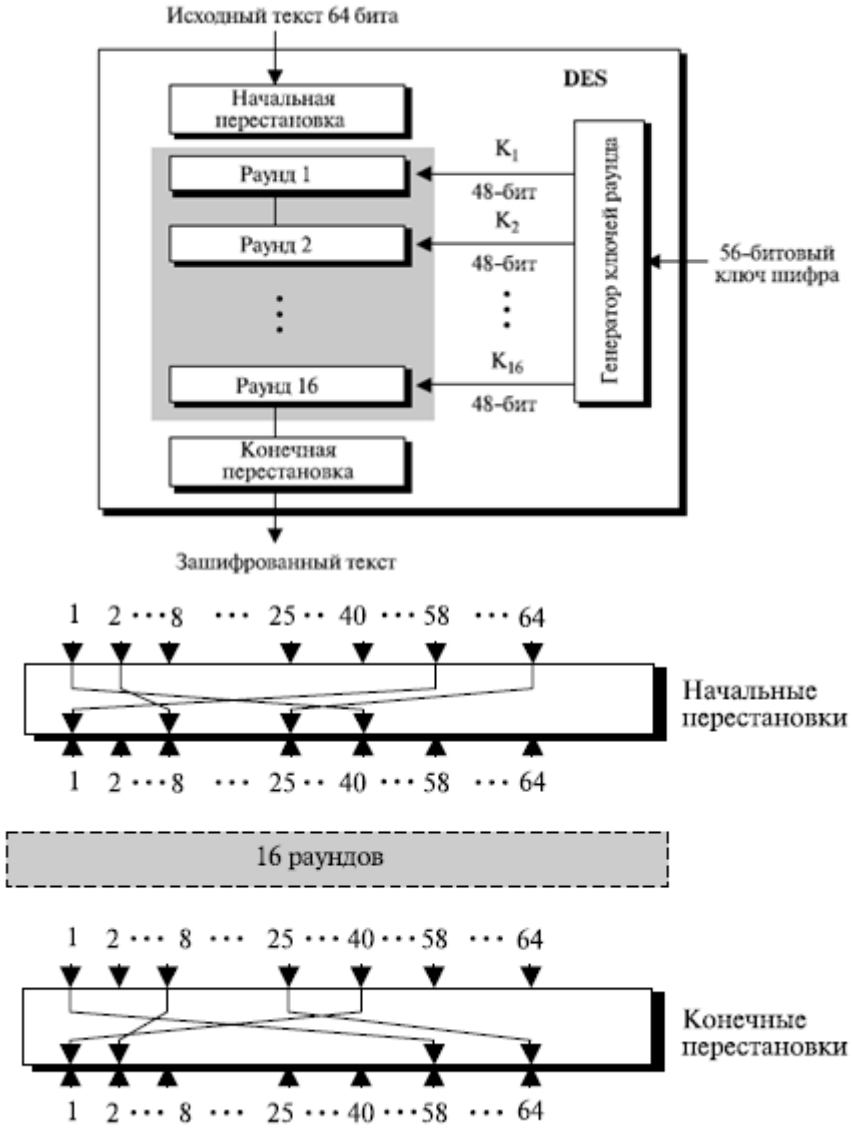


Рисунок 2 – Структура DES

Початкові й кінцеві перестановки – це прямі Р-Блоки, які інверсні один одному. Вони не мають значення для криптографії DES.

DES використовує 16 раундів. Кожний раунд DES застосовує шифр Файстеля, як це показане на рисунку 2.

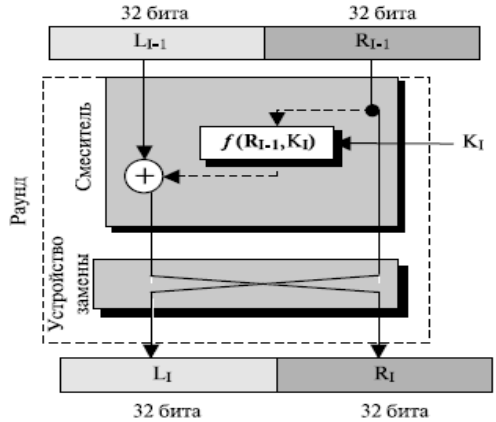


Рисунок 3 – Раунд в DES (сторона шифрування).

Раунд ухвалює LI-1 RI-1 від попереднього раунду (або початкового блоку перестановки) і створює для наступного раунду LI I RI, які надходять на наступний раунд (або кінцевий блок перестановки). Можна прийняти, що кожний раунд має

два елементи шифру (змішувач і обладнання заміни). Кожний із цих елементів є оборотним. Обладнання заміни — очевидно оборотне, воно міняє місцями ліву половину тексту із правою половиною. Змішувач є оборотним, тому що операція, ЩО ВИКЛЮЧАЄ АБО оборотна. Усі необоротні елементи зосереджені у функції f (RI-1,KI).

Функція DES

Основний блок DES — функція DES. Функція DES за допомогою 48-бітового ключа зашифровує 32 самих правих біт RI-1, щоб одержати на виході 32-бітове слово. Ця функція містить, як це показане на Рисунок 4, чотири секції: відбілювач (whitener), P-Блок розширення, групу S-Блоків і прямій P-Блок.

P-Блок розширення. Тому що вхід RI-1 має довжину 32 біта, а ключ KI — довжину 48 бітів, ми спочатку повинні розширити RI-1 до 48 біт. RI-1 розділяється на 9 секцій по 4 біта. Кожна секція на 4 біта розширюється до 6 біт. Ця перестановка розширення випливає по заздалегідь певних правилах. Для секції значення вхідних біт 1, 2, 3 і 4 привласнюються бітам 2, 3, 4 і 5 відповідно на виході. Вихідний біт 1 формується на основі вхідного біта 4 з попередньої секції; біт виходу 6 формується з біта 1 у наступній секції. Якщо секції 1 і 8 розглядати як сусідні секції, то ті ж самі правила застосовуються до бітів 1 і 32. Рисунок 4 показує входи й виходи в перестановці розширення.

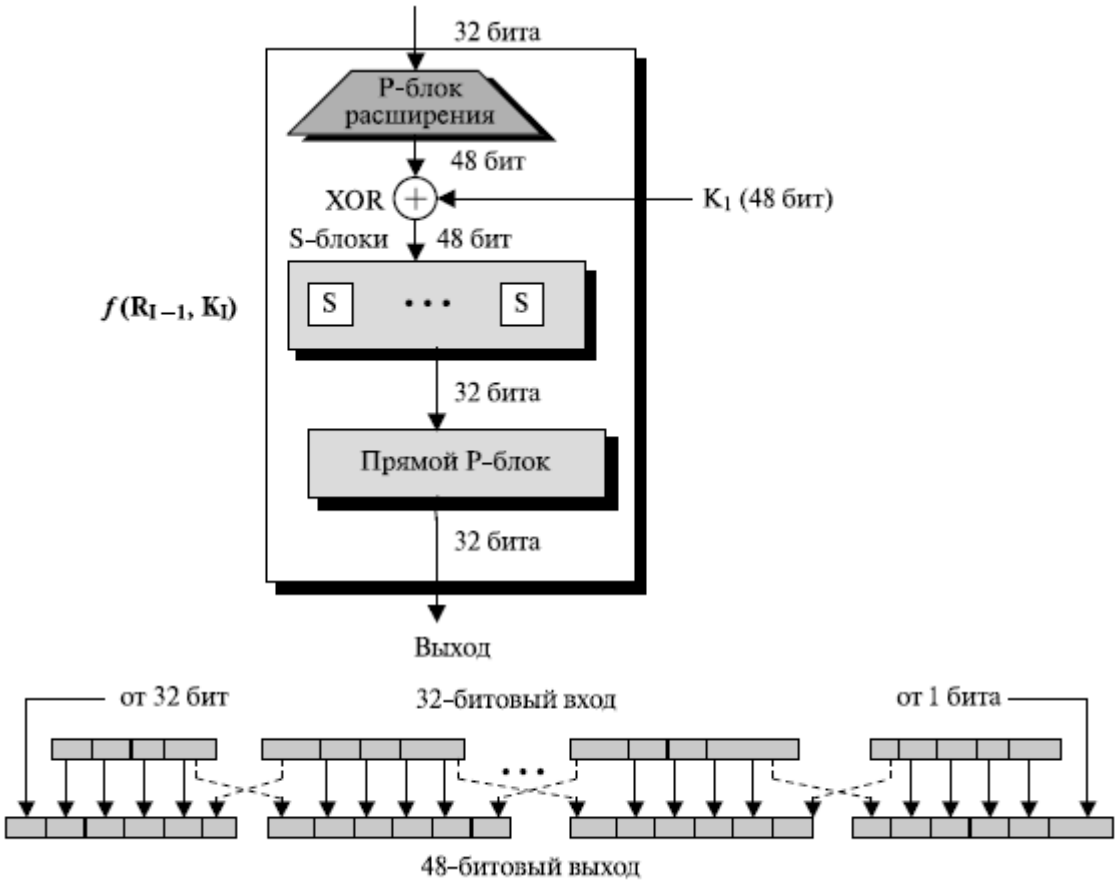


Рисунок 4 – Функція DES

S-Блоки. S-Блоки змішують інформацію (операція перемішування). DES використовує S-Блоки, кожний з 6-ю вхідними бітами й 4-мя вихідними.

Тема 6: Алгоритм шифрування даних IDEA.

План

1. IDEA – симетричний блочний алгоритм даних.
2. Історія створення.
3. Загальний опис.

Література: 8.с. 251..270, 7.с 136..140

Питання для самоконтролю:

1. Яка історія виникнення алгоритму шифрування IDEA?
2. Перша версія алгоритму.
3. Опис роботи шифрування.
4. Що є суттєвою відмінністю алгоритму IDEA від попередніх алгоритмів?
5. Як генеруються субключі?

IDEA (англ. International Data Encryption Algorithm – Міжнародний алгоритм шифрування даних) - симетричний блочний алгоритм шифрування даних, запатентований швейцарською фірмою Ascom. Відомий тим, що застосовувався в пакеті програм шифрування PGP. У листопаді 2000 IDEA був представлений в якості кандидата в проекті NESSIE в рамках програми Європейської комісії IST (англ. Information Societes Technology – Інформаційні суспільні технології).

Історія

Першу версію алгоритму розробили в 1990 Лай Сюецзя і Джеймс Мессі зі Швейцарського інституту ETH Zrich (за контрактом з Hasler Foundation, яка пізніше влилася в Ascom-Tech AG) в якості заміни DES і назвали її PES (англ. Proposed Encryption Standard – Запропонований стандарт шифрування). Потім, після публікації робіт Біхама і Шаміра по диференціальному криптоаналізу PES, алгоритм був поліпшений з метою посилення криптостійкості і названий IPES (англ. Improved Proposed Encryption Standard – Покращений запропонований стандарт шифрування). Через рік його перейменували в IDEA (англ. International Data Encryption Algorythm).

Опис

Так як IDEA використовує 128-бітний ключ і 64-бітний розмір блоку, відкритий текст розбивається на блоки по 64 біт. Якщо таке розбиття неможливо, останній блок доповнюється різними способами певною послідовністю біт. Для уникнення витоку інформації про кожному окремому блоці використовуються різні режими шифрування. Кожен вихідний незашифрований 64 - бітний блок ділиться на чотири подблока по 16 біт кожен, так як всі алгебраїчні операції, що використовуються в процесі шифрування, вчиняються над 16-бітними числами. Для шифрування й розшифрування IDEA використовує один і той же алгоритм.

Фундаментальним нововведенням в алгоритмі є використання операцій з різних алгебраїчних груп, а саме:

• додавання за модулем 2^{16}

• множення по модулю $2^{16}+1$

• побітове виключає АБО (XOR).

Ці три операції несумісні в тому сенсі, що:

• ніякі дві з них не задовольняють дистрибутивного закону, тобто: $a*(b+c) <> (a*b)+(a*c)$;

• ніякі дві з них не задовольняють асоціативному закону, тобто: $a+(b \oplus c) <> (a+b) \oplus c$.

Застосування цих трьох операцій ускладнює криптоаналіз IDEA в порівнянні з DES, який заснований виключно на операції виключає АБО, а також дозволяє відмовитися від використання S-блоків і таблиць заміни. IDEA є модифікацією мережі Фейстеля.

Нехай чотири четверті вихідного тексту мають імена A, B, C і D, а 52 субключі – K(1), K(2),…, K(52). Перед реалізацією алгоритма виконуються операції:

A = A*K(1); B = B + K(2); C = C + K(3); D = D * K(4);

Перший цикл обчислення додає до себе:

E = A XOR C; F = B XOR D

E = E * K(5)

F = F + E

F = F * K(6)

E = E + F

A = A XOR F

C = C XOR F

B = B XOR E

D = D XOR E

Змінюємо місцями B і C.

Повторюємо це все 8 разів, використовуючи K(7) – K(12) для другого разу і, відповідно, K(43) – K(48) - для восьмого. Після восьмого раза B і C місцями не міняються. Виконуємо після цього операції:

A = A * K(49)

B = B + K(50)

C = C + K(51)

D = D * K(52)

В результаті закодований текст має ту ж саму довжину, що і вихідний. Схема цього алгоритма може бути пояснена на рисунку 5. По характеру алгоритм нагадує процедури обчислення хеш-функції.

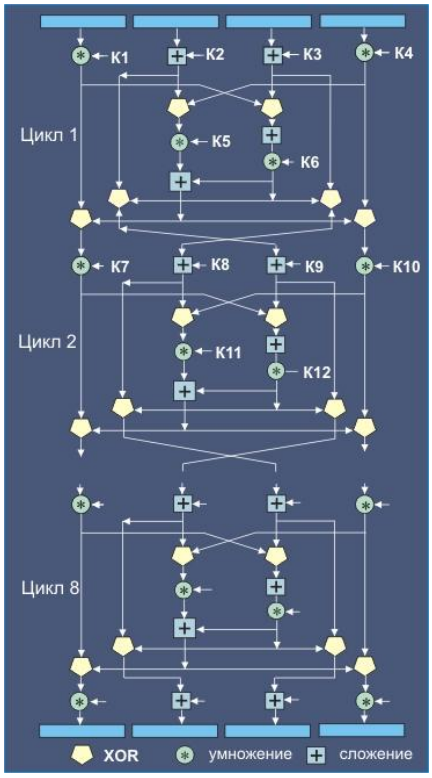


Рисунок 5 – Схема реалізації алгоритма шифрування IDEA.

При дешифровці використовується той факт, що A XOR B не змінюється, якщо C A і B буде виконана операція XOR C з використанням якогось числа. Це твердження справедливе для інших значень A і B. Операції додавання (доданки замінюються їх доповненням по модулю 2) и множення (множники замінюються з обратними величинами по модулю 65537) також допускают інверсію. Перші чотири ключі дешифровки (KD) визначаються по іншому, ніж інші.

KD(1) = 1/K(49);
KD(2) = -K(50);
KD(3) = -K(51);
KD(4) = 1/K(52);

Наступні операції проводяться 8 разів з додаванням 6 до індексу ключів дешифровки і відніманням 6 із індекса ключів шифрування.

KD(5) = K(47)
KD(6) = K(48)
KD(7) = 1/K(43)
KD(8) = -K(45)
KD(9) = -K(44)
KD(10) = 1/K(46)

Субключі IDEA генерируются наступним чином. 128-бітовий ключ IDEA визначає перші вісім субключів (128=8*16). Наступні ключі (44) отримуються шляхом послідовності циклічних зрушень вліво цього коду на 25 двійкових розрядів.

Тема 7: Концепція криптосистеми з відкритим ключем.

План

1. Поняття асиметричних криптосистем.
2. Історія виникнення.
3. Принцип роботи криптоситеми з відкритим ключем.
4. Ідея створення.

Література: 5.с 45..48

Питання для самоконтролю:

1. Розкрити поняття асиметричні криптосистеми.
2. Що називається криптографією з відкритим ключем?
3. Головне досягнення асиметричного шифрування.
4. Характеристика I періоду історії криптосистем.
5. Характеристика II періоду історії криптосистем.
6. Як працює асиметричний шифр.
7. Основна ідея створення.

Асиметричні криптосистеми — ефективні системи криптографічного захисту даних, які також називають криптосистемами з відкритим ключем. В таких системах для зашифровування даних використовується один ключ, а для розшифровування — інший ключ (звідси і назва — асиметричні). Перший ключ є відкритим і може бути опублікованим для використання усіма користувачами системи, які шифрують дані. Розшифровування даних за допомогою відкритого ключа неможливе. Для розшифровування даних отримувач зашифрованої інформації використовує другий ключ, який є секретним. Зрозуміло, що ключ розшифровування не може бути визначеним з ключа зашифровування.

Головне досягнення асиметричного шифрування в тому, що воно дозволяє людям, що не мають існуючої домовленості про безпеку, обмінюватися секретними повідомленнями. Необхідність відправникові й одержувачеві погоджувати таємний ключ по спеціальному захищеному каналі цілком відпала. Прикладами криптосистем з відкритим ключем є Elgamal (названа на честь автора, Тахіра Ельгамаля), RSA (названа на честь винахідників: Рона Рівеста, Аді Шаміра і Леонарда Адлмана), Diffie-Hellman і DSA, Digital Signature Algorithm (винайдений Девідом Кравіцом).

Історія

Історія криптографії налічує близько 4 тисяч років. В якості основного критерію періодизації криптографії можливо використовувати технологічні характеристики використовуваних методів шифрування.

Перший період (приблизно з третього тисячоліття до н. е..) Характеризується пануванням багатоалфавітних шифрів (основний принцип — заміна алфавіту вихідного тексту іншим алфавітом через заміну літер іншими літерами або символами). Другий період (хронологічні рамки — з IX століття на Близькому Сході (Ал-Кінді) і з XV століття в Європі (Леон Баттіста Альберті) — до початку XX століття) ознаменувався введенням в обіг поліалфавітних шифрів. Третій період (з

початку і до середини XX століття) характеризується впровадженням електромеханічних пристроїв в роботу шифрувальників. При цьому продовжувалося використання поліалфавітних шифрів.

Принцип роботи

Початок асиметричним шифру було покладено в роботі «Нові напрямки в сучасній криптографії» Уітфілд Діффі та Мартіна Хеллмана, опублікованій в 1976 році. Перебуваючи під впливом роботи Ральфа Меркле (Ralph Merkle) про поширення відкритого ключа, вони запропонували метод отримання секретних ключів, використовуючи відкритий канал. Цей метод експоненціального обміну ключів, який став відомий як обмін ключами Діффі-Хеллмана, був першим опублікованим практичним методом для встановлення поділу секретного ключа між завіреними користувачами каналу. У 2002 році Хеллмана запропонував називати даний алгоритм «Діффі — Хеллмана — Меркле», визнаючи внесок Меркле в винахід криптографії з відкритим ключем. Ця ж схема була розроблена Малькольмом Вільямсоном в 1970-х, але трималася в секреті до 1997 року. Метод Меркле з розповсюдження відкритого ключа був винайдений в 1974 році і опублікований в 1978, його також називають загадкою Меркле.

У 1977 році вченими Рональдом Рівестом (Ronald Linn Rivest), Аді Шамір (Adi Shamir) і Леонардом Адлеманом (Leonard Adleman) з Массачусетського Технологічного Інституту (MIT) був розроблений алгоритм шифрування, заснований на проблемі про розкладанні на множники. Система була названа за першими літерами їхніх прізвищ. Ця ж система була винайдена Клиффордом Коксом (Clifford Cocks) в 1973 році, що працював в центрі урядового зв'язку (GCHQ). Але ця робота зберігалася лише у внутрішніх документах центру, тому про її існування було не відомо до 1977 року. RSA став першим алгоритмом, придатним і для шифрування, і для цифрового підпису.

Криптографія з відкритим ключем

Проблема керування ключами була вирішена криптографією з відкритим, або асиметричним, ключем, концепція якої була запропонована Уітфілдом Діффі і Мартіном Хеллманом у 1975 році. Криптографія з відкритим ключем — це асиметрична схема, у якій застосовуються пари ключів: відкритий (public key), що зашифровує дані, і відповідний йому закритий (private key), що їх розшифровує. Ви поширюєте свій відкритий ключ по усьому світу, у той час як закритий тримаєте в таємниці. Будь-яка людина з копією вашого відкритого ключа може зашифрувати інформацію, що тільки ви зможете прочитати. Хто завгодно. Навіть люди, з якими ви ніколи не зустрічалися.

Хоча ключова пара математично зв'язана, обчислення закритого ключа з відкритого в практичному плані нездійсненна. Кожний, у кого є відкритий ключ, зможе зашифрувати дані, але не зможе їх розшифрувати. Тільки людина, яка володіє відповідним закритим ключем може розшифрувати інформацію.

Поява шифрування з відкритим ключем стала технологічною революцією, яка зробила стійку криптографію доступною масам.

Ідея створення

Ідея криптографії з відкритим ключем дуже тісно пов'язана з ідеєю односторонніх функцій, тобто таких функцій f (x), що за відомим x досить просто знайти значення f (x), тоді як визначення x з f (x) складно в сенсі теорії.

Але сама одностороння функція марна в застосуванні: нею можна зашифрувати повідомлення, але розшифрувати не можна. Тому криптографія з відкритим ключем використовує односторонні функції з лазівкою. Лазівка — це якийсь секрет, який допомагає розшифрувати. Тобто існує такий y, що знаючи f (x), можна обчислити x. Приміром, якщо розібрати годинник на безліч складових частин, то дуже складно зібрати знову працюючий годинник. Але якщо є інструкція по зборці (лазівка), то можна легко вирішити цю проблему.

Тема 8: Електронний цифровий підпис.

План

1. Що таке електронний цифровий підпис.
2. Перевірка електронного цифрового підпису.

Література: 1.с. 152..159

Питання для самоконтролю:

1. Що таке електронний цифровий підпис?
2. Що необхідно для функціонування ЕЦП?
3. Операція перевірки електронного цифрового підпису.
4. Як відбувається генерація таємного та відкритого ключів ЕЦП?
5. Виконання сертифікації відкритого ключа.
6. Які основні правила сертифікації?

Електронний цифровий підпис (ЕЦП) – це блок інформації, який додається до файлу даних автором (підписувачем) та захищає файл від несанкціонованої модифікації і вказує на підписувача (власника підпису). Для функціонування ЕЦП використовуються 2 ключі захисту (які зберігаються в різних файлах): таємний ключ, який зберігається у підписувача (наприклад, на дискеті, пристрої Touch Memory, Smart-карті і т.і.); відкритий ключ, який, як правило, публікується в загальнодоступному або спеціалізованому довіднику.

Для накладання ЕЦП використовується таємний (особистий) ключ, а для його для перевірки – відкритий (загальновідомий) ключ. Накладання електронного цифрового підпису (підписування) - це операція, яка здійснюється відправником (підписувачем) документу із використанням його таємного ключа. При виконанні цієї операції на вхід відповідної програми подаються данні, які треба підписати, та таємний ключ підписувача. Програма створює із даних за допомогою таємного ключа унікальний блок даних фіксованого розміру (власне ЕЦП), який може бути справжнім тільки для цього таємного ключа та саме для цих вхідних даних. Тобто, ЕЦП – це своєрідний "цифровий відбиток таємного ключа і документа".

Перевірка електронного цифрового підпису – це операція, яка виконується отримувачем захищеного електронного документу з використанням відкритого ключа підписувача (відправника). Для виконання цієї операції необхідно отримати відкритий ключ відправника (наприклад, із довідника) та захищеного документа (тобто даних документа та даних ЕЦП). Відповідний програмний модуль перевіряє, чи дійсно цифровий підпис відповідає документу та відкритому ключу. Якщо в документ або у відкритий ключ внесено будь-які зміни, перевірка закінчиться із негативним результатом.

Центри сертифікації ключів перевіряють дані власника відкритого ключа та видають захищені електронні документи спеціального зразка – сертифікати відкритих ключів, в яких міститься відкритий ключ та перевірена центром сертифікації інформація про власника ключа. Сертифікат відкритого ключа підписується електронним цифровим підписом центру сертифікації ключів.

Генерація таємного та відкритого ключів ЕЦП - початкова процедура, яка виконується користувачем до виконання процедури сертифікації відкритого ключа. Генерацію виконує спеціалізоване програмне забезпечення – генератор ключів, який надається центром сертифікації ключів.

Для виконання сертифікації відкритого ключа до центру сертифікації подається:

1. Електронний документ спеціального зразка – заявка на сертифікацію відкритого ключа, яка містить відкритий ключ та електронну картку із реквізитами власника ключа. Заявка генерується спеціальною програмою – генератором ключів.

2. Комплект документів, що засвідчує особу власника ключа (для посадових ключів юридичних осіб додатково додаються документи, що засвідчують правомочність власника ключа діяти від імені юридичної особи).

Правила сертифікації – документ, в якому описані індивідуальні для кожного центру сертифікації правила, за якими перевіряються відомості, що знаходяться в сертифікаті. Тобто, не всі центри сертифікації ключів діють за однаковими правилами та вимогами при перевірці документів, що встановлюють особу-власника ключа (ці правила також називаються "політикою сертифікації").

Кількість перевірок та їх ретельність може бути різною в різних центрах сертифікації ключів. Значна кількість центрів сертифікації використовують декілька політик сертифікації одночасно, тобто видають сертифікати різних рівнів довіри (ці рівні довіри іноді називаються «класами сертифікатів»). Існування різних класів сертифікатів обумовлено різницею у їх вартості (більшій рівень довіри потребує більше перевірок, більше перевірок – більше роботи, більше роботи – більша вартість).

Правила сертифікації, якими користується центр сертифікації, та клас сертифікату безумовно безпосередньо впливають на рівень довіри до сертифікатів ключів, які видані такими центрами сертифікації.

Тобто, довіра до сертифікату відкритого ключа залежить не тільки від того факту, що відкритий ключ сертифіковано, а також від того, ким (яким центром сертифікації) сертифіковано та за якими правилами (політика та клас сертифікату).

Модуль 2. Захист компютера від несанкціонованого доступу.

Тема 1: Захисти від несанкціонованого доступу.

План

1. Захисні засоби ПК.
2. Використання криптографії.
3. Захист ПК за допомогою апаратних засобів.

Література: 3.с. 275..276

Питання для самоконтролю:

1. Механічний засіб захисту.
2. Апаратні засоби захисту.
3. Програмні засоби захисту.
4. Що називається несанкціонованим доступом?
5. Три основні напрямки захисту інформації ПК від НСД.
6. Які існують стандартні захисні засоби?
7. Що називається парольною ідентифікацією?

Методи захисту даних на персональних комп'ютерах надзвичайно різноманітні як по кінцевій меті, так і по технічному втіленню; їх можна розділити на механічні, апаратні і програмні.

До механічних засобів захисту ставляться різноманітні кришки і чохли з замками (що замикають, наприклад, дисковод гнучких дисків або мережний вимикач), клейкі пластини для приклеювання термінала до комп'ютера, а комп'ютера до столу, помешкання що замикаються із сигналізацією і багато інших.

Апаратні засоби реалізуються у вигляді спеціальних електронних модулів, що підключаються до системного каналу комп'ютера або портів вводу-виводу, і здійснюють обмін кодовими послідовностями програмами, що захищається.

Найбільш різноманітні програмні засоби. Сюди відносяться програми шифрації даних по заданому користувачем ключу, адміністратори дисків, що дозволяють обмежити доступ користувачів до окремих логічних дисків, методи встановлення програмного продукту з дистрибутивних дискет, що дозволяють виконати установку не більше вказаного числа запуску програм, що захищаються за допомогою некопійованих ключових дискет, спеціальні захисні програмні оболонки, куди поміщаються програми що захищуються.

Особливості захисту персональних комп'ютерів (ПК) обумовлені специфікою їх використання. Як правило, ПК користується обмежене число користувачів. ПК можуть працювати як в автономному режимі, так і в складі локальних мереж (сполученими з іншими ПК) і можуть бути залучені до віддаленого ПК або локальної мережі за допомогою модему по телефонній лінії.

Несанкціонованим доступом (НСД) до інформації ПК будемо називати незаплановане ознайомлення, опрацювання, копіювання, застосування різноманітних вірусів, у тому числі руйнуючі програмні продукти, а також

модифікацію або знищення інформації та порушення встановлених правил розмежування доступу.

У захисті інформації ПК від НСД можна виділити три основні напрямки:

1 – орієнтується на недопущення порушника до обчислювального середовища і ґрунтується на спеціальних технічних засобах упізнання користувача;

2 – пов'язано з захистом обчислювального середовища і ґрунтується на створенні спеціального програмного забезпечення по захисту інформації;

3 – пов'язаний із використанням спеціальних засобів захисту інформації ПК від несанкціонованого доступу.

Для захисту персональних комп'ютерів використовуються різноманітні програмні методи, що значно розширюють можливості по забезпеченню безпеки інформації, що зберігається. Серед стандартних захисних засобів персонального комп'ютера найбільше поширення одержали:

1 – засоби захисту обчислювальних ресурсів, що використовують парольну ідентифікацію й обмежують доступ несанкціонованого користувача;

2 – застосування різноманітних методів шифрування, що не залежать від контексту інформації;

3 – засоби захисту від копіювання комерційних програмних продуктів;

4 – захист від комп'ютерних вірусів і створення архівів.

Захист вмонтованого накопичувача на жорсткому магнітному диску складає одну з головних задач захисту ПК від стороннього вторгнення. Існує декілька типів програмних засобів, спроможних вирішити проблему захисту:

• захист від будь-якого доступу до жорсткого диска;

• захист диска від запису/читання;

• контроль за зверненням до диска;

• засоби видалення залишків секретної інформації.

Захист умонтованого жорсткого диска звичайно здійснюється шляхом застосування спеціальних паролів для ідентифікації користувача (так звана парольна ідентифікація). У даному випадку доступ до жорсткого диска можна одержати при правильному введенні пароля при завантаженні операційної системи. У противному випадку завантаження системи не відбудеться, а при спробі завантаження з гнучкого диска, жорсткий диск стає "невидимим" для користувача. Ефект захисту жорсткого диска в системі досягається зміною завантажувального сектора диска, із якого видалиться інформація про структуру диска.

Самим надійним захистом від несанкціонованого доступу до переданої інформації і програмних продуктів ПК є застосування різноманітних методів шифрування (криптографічних методів захисту інформації).

Використання криптографії

Криптографічні методи захисту інформації - це спеціальні методи шифрування, кодування або іншого перетворення інформації, у результаті якого її утримання стає недоступним без пред'явлення ключа криптограми й оберненого перетворення. Даний метод захисту реалізується у виді програм або пакетів програм, що розширюють можливості стандартної операційної системи.

Захист ПК за допомогою апаратних засобів

Призначення даного пристрою - видалення інформації при спробі вилучення накопичувача, при викраденні комп'ютера, при проникненні в зону обслуговування комп'ютера (серверу) або при натисканні визначеної кнопки.

Принцип дії даного пристрою - форматування накопичувача. Відомо, що на початку кожного накопичувача розташовані таблиці розділів, таблиці розміщення файлів, каталоги - тому знищення інформації починається саме з них і навіть після декількох секунд роботи даного пристрою на накопичувачі залишену інформацію, дуже важко відновити. Якщо ж пристрій відпрацює декілька хвилин, то вся інформація буде знищена. Після першого циклу знищення починається другий і так далі (до розряду автономного джерела живлення), тому інформацію не можна відновити навіть по залишковій намагніченості. Подібний метод стирання є в утилітах WipeFile і WipeDisk, проте для їхнього запуску комп'ютер повинний бути включений, та й багато операційних систем не мають безпосередній доступ до диска і відповідно ці утиліти можуть не спрацювати.

Для ідентифікації адміністратора, що має доступ до комп'ютера, застосовуються електронні ключі з довжиною коду 48 бітів. Невеличкий час, відведений для пред'явлення даного ключа - 10 секунд, цілком виключає можливість його підбору. Датчики, при спрацьовуванні яких відбувається знищення інформації, користувачі вибирають самі. Найбільше часто використовуються вимикачі (вимикання комп'ютера, відкриття кімнати, тривожна кнопка), дзвоник телефону (пейджера), датчики що спрацьовують при зміні обсягу.

Якщо потрібно виключити копіювання програми з жорсткого диска на інший жорсткий диск, її можна прив'язати до номера кластера або сектора, із якого починається файл програми на диску. Прив'язка здійснюється в такий спосіб. Спеціально підготовлена установча програма відкриває файл із робочою програмою і по таблиці відкритих файлів знаходить початковий номер кластера. Це число, що є своєрідним ключем, записується установчою програмою у визначене місце файла робочої програми (у поле даних). Робоча ж програма після запуску насамперед виконує ту ж операцію - визначає свою початкову адресу, а потім порівнює його з ключем. Якщо числа збігаються, програма приступає до виконання своєї змістовної частини; якщо не збігаються - аварійно завершується. При копіюванні програми на інший диск (або навіть на той же самий) вона виявиться розташованою в іншому місці і номер кластера, записаний установчою програмою вже не буде відповідати реальній адресі файла.

Тема 2. Захист вихідних текстів і двійкового коду. Засоби відладки та злому програм.

План

1. Динамічне гілкування.
2. Контекстна залежність.
3. Хукі.
4. Протидія аналізу двійкового коду.
5. Три ключі.
6. Відладчики реального режиму.
7. Відладчики захищеного режиму.
8. Емулятори.
9. Автоматичні розпаковщики.

Література: 2.с. 163..255

Питання для самоконтролю:

1. Що називається динамічним гілкуванням?
2. Що таке контекстна залежність?
3. Що таке хукі?
4. Як відбувається протидія аналізу двійкового коду?
5. Що таке три ключі?
6. Які ви знаєте групи засобів відладки і злому програм?
7. Опишіть відладчики реального режиму.
8. Опишіть відладики захищеного режиму.
9. Що таке емулятори?
10. Для чого потрібні автоматичні розпаковщики?

Великі виробники тиражного програмного забезпечення відмовилися від захисту своїх продуктів, зробивши ставку на їх масове поширення. Нехай ліцензійні копії придбає лише кілька відсотків користувачів, але, якщо число цих копій вимірюється мільйонами, навіть лічені відсотки виливаються в солідну суму, з лишком окупаються розробку. Однак для невеликих колективів та індивідуальних програмістів така тактика є неприйнятною. Їм необхідно запобігти "піратське" копіювання свого продукту.

Надання вихідних текстів часто є обов'язковою умовою замовника і закріплюється в контракті. Тій же меті домагаються прихильники популярного руху відкритих вихідних текстів, які ратують за їх вільне поширення. Крім того, вихідні тексти можуть банальним чином вкрасти. Тому, варто уважно поставитися до організації технологічного захисту інтелектуальної власності.

Протидія вивченню вихідних текстів.

Взагалі кажучи, відкритість вихідних текстів - поняття розпливчасте. Питання: "Чи є бінарна програма в 1 Кбайт більш відкритою, ніж мільйон рядків «исходников» без адекватної інфраструктури та документації?" Мало мати вихідний текст - у ньому ще належить розібратися. Досить видалити всі або, принаймні, більшу частину коментарів, дати змінним і функціям безглузді, нічого не значащі

імена, як в програмі не розбереться і сам автор. А наявність і повнота коментарів до вихідного тексту в контракті, як правило, не обумовлюється. Виходить, що контракт може бути формально виконаний, а наданий замовнику вихідний текст практично даремний. "Практично" не означає "повністю": такий простий прийом не дозволить забезпечити абсолютний захист.

Перешкодити аналізу можна відділенням, абстрагуванням алгоритму від мови реалізації. Наприклад, реалізувати критичні для розкриття компоненти на машині Тьюринга, а її підтримку забезпечити на цільовій мові. Рівнів абстракції може бути декілька - чим їх більше, тим важче здійснювати аналіз. Крім машини Тьюринга для цієї мети підходять стрілка Пірса, мережі Петрі і т.д. Такий підхід дає чудовий результат, але вимагає глибоких математичних знань та значних накладних витрат - програмувати на машині Тьюринга набагато складніше, ніж на асемблері. Окреме питання - ефективність отриманої програми. Для багатьох проектів це неприйнятно, тому доводиться використовувати інші прийоми.

Динамічне розгалуження

Програму, що складається з декількох сотень тисяч рядків, неможливо розглядати як просту сукупність команд. На такому рівні деталізації "за деревами лісу не видно". Спочатку необхідно проаналізувати взаємозв'язок окремих функцій один з одним, виділити цікавлять фрагменти і лише потім вивчати реалізацію самих функцій. Чим вище ступінь дроблення програми, тим важче аналіз. Елементарні функції, що складаються з десятка рядків, самі по собі дають мало інформації. Для формування цілісної картини необхідно розглянути викликає їх код, піднімаючись по ієрархії викликів до тих пір, поки не вдасться реконструювати весь алгоритм цілком або, принаймні, охопити його ключовий фрагмент. Щоб побудувати дерево викликів, потрібно вміти відслідковувати перехресні посилання в обох напрямках: визначати, який функції передається керування даними викликом, і, навпаки, знаходити всі виклики, передають управління даної функції. Цьому легко перешкодити, скориставшись динамічним галуженням - тобто обчисленням адреси переходу безпосередньо перед передачею управління. Якщо функція повертає не результат своєї роботи, а покажчик на наступну виконувану функцію (результат роботи можна повернути через аргументи, передані по посиланню), то статичний аналіз не дозволить визначити порядок виконання програми і побудувати дерево викликів стане неможливо. Попутно зникне маса дубльованого коду, зокрема, стане непотрібною перевірка результату завершення на коректність - у разі виникнення помилки функція сама передасть управління потрібної гілки програми.

Тут варто зробити одне застереження. Мова Сі не дозволяє оголосити функцію, що повертає покажчик на функції - це оголошення рекурсивне. Доводиться оголошувати функцію, що повертає безтиповою покажчик void *. Аналогічно - якщо функція в ході своєї роботи викликає якісь інші функції, краще робити це безпосередньо, а передавати покажчики на викликувані функції як аргументи. Такий підхід не тільки перешкоджає аналізу, але ще і збільшує гнучкість програми, полегшуючи повторне використання старого коду в нових проектах - при належній культурі програмування кожна функція представляє "річ в собі" і не прив'язана до всіх інших.

Контекстна залежність

Розглянемо тепер інший прийом, коли алгоритм роботи більшості функцій залежить від прапора - глобальної змінної в межах одного модуля. Якщо прапори змінюються в залежності від результату, що повертається функцією, автоматично виникає контекстна залежність (не надто сильна, але це краще, ніж нічого). Робота однієї функції стає залежною від інших, викликаних до неї. Аналіз однієї, окремо взятої функції, стає багатоваріантним. Щоб визначити, що конкретно вона робить, необхідно знати значення прапорів, а значить - мати уявлення про те, які функції виконувалися до цього і які саме дані вони обробляли.

Знову зауваження: в многопоточной середовищі глобальний прапор можна використовувати лише в тому випадку, якщо всі потоки явно синхронізовані, в іншому випадку необхідно надати кожному потоку свій власний екземпляр прапора. Глобальний прапор вимагає особливої уваги, найменша недбалість призводить до важковловимий помилок. Розібратися в програмі з безліччю одночасно виконуваних потоків, що маніпулюють з однією змінною, неймовірно важко; найменша неуважність або зайва квапливість призводять до помилок аналізу, ускладнює розуміння суті алгоритму.

Хуки

Хуки ("гаки") - витончений, але нині практично забутий прийом програмування. Його суть полягає в суміщенні декількох різнотипних даних в одному аргументі. Наприклад, якщо значення аргументу по модулю менше 0x400000, функція вважає його безпосереднім значенням, в іншому випадку - покажчиком на функцію, результат виконання якої слід поставити на його місце. Це збільшує гнучкість програми, але і ускладнює її аналіз, не дозволяючи швидко визначити, чи відбувається передача змінної за значенням або за вказівником. Покажчики на змінну, в свою чергу, стають не відрізняються від покажчиків на функцію. Зрозуміло, відмова від строгої типізації може призводити до помилок, але вірогідність їх появи в ретельно продуманою програмою невелика. Знову зауваження: хуки негативно позначаються на переносимості програм, оскільки подання покажчиків має свої особливості на кожній апаратній платформі. Тому використовувати їх слід лише в тих випадках, коли переносимість не потрібно, або коли на всіх обраних платформах уявлення покажчиків уніфіковано. Існує ще безліч інших способів заплутати того, хто намагається проаналізувати вихідний текст. Наприклад, можна використовувати абсолютно коректні з точки зору мови, але незвичайні конструкції, що ставлять у глухий кут незнайомого з ними людини. Класичний приклад - перестановка індексу та імені масиву в Сі. З точки зору мови, вирази "buff [666]" і "666 [buf]" абсолютно рівнозначні, але всякий Чи про це пам'ятає?

Препроцесор Сі також має свої тонкощі. Можливо, найпопулярніша з них полягає в чутливості до прогалин при оголошенні макросу: "# define x (a, b) a + b" створить макрос x (a, b), замінюється сумою своїх аргументів, але "# define x (a, b) a + b "створить макрос x, замінюється послідовністю" (a, b) a + b ". Якщо не звернути увагу на зайвий пробіл, можна отримати зовсім не той результат. В тій чи іншій мірі ці, а також інші прийоми використовуються в більшості вільно розповсюджуваних "відкритих текстів". Останнє словосполучення укладено в лапки для додання йому іронічного відтінку: одна лише наявність вихідного тексту ще не забезпечує відкритості, - потрібно, щонайменше, грамотно продумана і якісно

складена документація, а ще краще - досвід роботи з цим текстом. Розібратися з вихідними текстами редактора emacs або операційної системи Linux не набагато простіше, ніж написати їх "з нуля", - надто вже скупі їх розробники на коментарі, а документація часто і зовсім відсутня.

У більшості випадків витрати на аналіз чужих вихідних текстів порівнянні або навіть перевищують вартість закладених у них алгоритмів (якщо стоять алгоритми там взагалі є), а їх модифікація просто вбивче заняття: внесені зміни здатні породжувати помилки в непередбачуваних місцях і в непередбаченому кількості. Словом, розумніше було б говорити про закриті вихідних текстах.

Протидія аналізу двійкового коду.

Відсутність вихідних текстів аж ніяк не є непереборною перешкодою для вивчення і модифікації коду програми. Методики зворотного проектування дозволяють автоматично розпізнавати бібліотечні функції, локальні змінні, стекові аргументи, типи даних, розгалуження, цикли і т.д. У недалекому майбутньому дізассемблер, ймовірно, навчаться генерувати листинги, близькі за зовнішнім виглядом до мов високого рівня.

Сьогодні трудомісткість аналізу двійкового коду не настільки велика, щоб надовго зупинити зловмисників. Величезне число постійно здійснюваних зломів - найкраще тому підтвердження. В ідеальному випадку знання алгоритму захисту не повинно впливати на її стійкість, але це досяжно далеко не завжди. Наприклад, якщо виробник серверної програми вирішить встановити в демонстраційній версії обмеження на кількість одночасно оброблюваних з'єднань, зловмисникові достатньо знайти інструкцію процесора, яка здійснює таку перевірку і видалити її. Модифікації програми можна перешкодити постійною перевіркою контрольної суми, але знову-таки код, який обчислює цю контрольну суму і звіряє її з еталоном, можна знайти і видалити.

Скільки б рівнів захисту не передбачити, один або мільйон, програма може бути зламана - це тільки питання часу та витрачених зусиль. Але в відсутність дієвих правових регуляторів захисту інтелектуальної власності розробникам доводиться більше покладатися на стійкість свого захисту, ніж на закон. Існує думка, що якщо витрати на нейтралізацію захисного механізму будуть не нижче вартості легальної копії, її ніхто не буде зламувати. Це невірно. Матеріальний стимул - не єдине, що рухає хакером. Набагато більш сильною мотивацією виявляється інтелектуальна боротьба з автором захисту, спортивний азарт, цікавість, підвищення свого професіоналізму, та й просто цікаве проведення часу. Багато молодих людей можуть тижнями сидіти над відладчиком, знімаючи захист з програми вартістю в кілька доларів, а то й зовсім поширюваної безкоштовно (наприклад, диспетчер файлів FAR для жителів Росії абсолютно безкоштовний, але це не рятує його від злому).

Тим не менш, є досвід створення захистів, зламати які майже неможливо. (Точніше, їх злом вимагав би багатьох тисяч, а то і мільйонів років на типовому побутовому комп'ютері.)

Гарантовано перешкодити аналізу коду дозволяє тільки шифрування програми. Але сам процесор не може безпосередньо виконувати зашифрований код, тому перед передачею управління його необхідно розшифрувати. Якщо ключ міститься всередині програми, стійкість такого захисту близька до нуля. Все, чого

може досягти розробник, - ускладнити пошук і отримання цього ключа, тим або іншим способом перешкоджаючи налагодженні і дизасемблювання програми. Інша справа, якщо ключ міститься поза програмою. Тоді стійкість захисту визначається стійкістю використовуваного криптоалгоритму (звичайно, за умови, що ключ перехопити неможливо). Опубліковані і детально описані багато криптостійкі шифри, злом яких свідомо недоступний рядовим зловмисникам.
У загальних рисах ідея захисту полягає в описі алгоритму за допомогою якоїсь математичної моделі, одночасно з цим використовуваної для генерації ключа. Різні гілки програми зашифровані різними ключами, і щоб обчислити цей ключ, необхідно знати стан моделі на момент передачі управління відповідної гілки програми. Код динамічно розшифровується у процесі виконання, а щоб розшифрувати його цілком, потрібно послідовно перебрати всі можливі стани моделі. Якщо їх число буде дуже велике (цього неважко досягти), відновити весь код стане практично неможливо.

Для реалізації цієї ідеї був створений спеціальний подієво-орієнтована мова програмування, де події є єдиним засобом виклику підпрограми. Кожна подія має свій код, один або кілька аргументів і яке завгодно кількість обробників, а може не мати жодного (в останньому випадку викликається коду повертається помилка). На основі коду події та значення аргументів диспетчер подій генерує три ключі - перший тільки на основі коду події, другий - тільки на основі аргументів, і третій на основі коду та аргументів. Потім він намагається отриманими ключами послідовно розшифрувати всіх обробників подій. Якщо розшифрування відбувається успішно, це означає, що даний обробник готовий обробити дану подію, і тоді йому передається управління.

Алгоритм шифрування повинен бути вибраний так, щоб зворотна операція була неможлива. При цьому встановити, яка подія даний обробник обробляє, можна лише повним перебором. Для блокування можливості перебору в мову була введена контекстна залежність - генерація додаткової серії ключів, що враховують деяку кількість попередніх подій. Це дозволило встановлювати обробники на будь послідовності дій користувача, наприклад, на відкриття файлу з ім'ям "Мій файл", запис у нього рядки "Моя рядок" та перейменування його у "Не мій файл". Очевидно, перебір комбінацій всіх подій з усіма можливими аргументами займе нескінченне час, а відновити вихідний код програми, захищеної таким чином, вдасться не раніше, ніж усі її гілки хоча б одноразово отримають управління. Однак частота виклику різних гілок не однакова, а в деяких і зовсім дуже мала. Наприклад, можна встановити на слово "сосна", введене в текстовому редакторі, свій оброблювач, що виконує деякі додаткові перевірки на цілісність коду програми або на ліцензійну чистоту використовуваного ПЗ. Зломщик не зможе швидко з'ясувати, чи до кінця зламана програма чи ні. Йому доведеться провести ретельне і копітке тестування, але навіть після цього він не буде в цьому впевнений.
Таким же точно чином здійснюється обмеження терміну служби демонстраційних версій. Зрозуміло, звертатися до годинника реального часу даремно, їх дуже легко перевести назад, вводячи захист в оману. Набагато надійніше спиратися на дати відкриваються файлів: навіть якщо годинник переведені, створені іншими користувачами файли в більшості випадків мають правильний час. Але зломщик не зможе дізнатися ні алгоритм визначення дати, ні саму дату закінчення використання

продукту. Втім, дату в принципі можна знайти і повним перебором, але що це дає? Модифікації коду перешкодити дуже легко: достатньо, щоб довжина зашифрованого тексту була чутлива до будь-яких змін початкового. У цьому випадку зломщик не зможе підправити "потрібний" байт в захисному обробнику і знову зашифрувати його. Доведеться розшифровувати і вносити зміни в усі інші обробники (за умови, що вони контролюють зсув, по якому розташовані), а це неможливо, оскільки відповідні їм ключі заздалегідь невідомі. Суттєвими недоліками пропонованого рішення є низька продуктивність і висока складність реалізації. Якщо зі складністю реалізації можна змиритися, то швидкість накладає серйозні обмеження на сферу застосування. Втім, можна значно оптимізувати алгоритм або залишити всі критичні до швидкодії модулі незашифрованими (або розшифровувати кожен обробник тільки один раз). Цікаво інше: чи справді ця технологія дозволяє створювати принципово неізучаемие додатка або в наведені міркування вкралася помилка?

Доцільність захисту вихідних текстів обмежується конкурентною боротьбою - за інших рівних умов клієнт завжди вибирає незахищений продукт, навіть якщо захист не ущемляє його прав. Сьогодні попит на програмістів перевищує пропозицію, але у віддаленому майбутньому тенденція зміниться і розробникам доведеться або змовитися, або повністю відмовитися від захистів, а фахівці із захисту будуть змушені шукати собі іншу роботу.

Засоби відладки і злому програм.

Наступні групи засобів відладки і злому програм:

- відладчики реального режиму
- відладчики V86
- емулятори
- автоматичні розпаковщики
- дизасемблери
- інші програми

У свою чергу перші дві групи знову розділяються на ті, що використовують стек відлагоджувальної програми і ті, що не використовують його.

Відладчики реального режиму.

Самі відомі:

1. **TurboDebugger Borland International**

Створений в 1988 році двома братами Chris'ом і Rich'ем Williams Містить безліч помилок, що активно використовуються захистами, таких як:

a. використання стека відлагоджувальної програми
b. використання int 1, int 3 для трасування
c. перехоплення переривань int 0, int 1, int 3
d. некоректна робота з відеобуфером
e. некоректне виставляння початкових значень регістрів

API відсутній.

Володіє надзвичайно розвинутим віконним інтерфейсом, надає можливості по перегляду коду і вихідного тексту програми, шестнадцатеричного дампу, змінних (за наявності налагоджувальної інформації), створенню макросів, відрізняється завидною повільністю, пов'язаною з підкачкою оверлею.

2. **CodeView Microsoft**

По своїх помилках нічим не відрізняється від TurboDebugger'a. Підтримує власний формат налагоджувальної інформації. Через пристрій самого ядра відладчика не пристосований як середовище для злому.

3. **AFD**

Створений в 1988 році H.Puttkamer'ом відладчик надає наступні можливості: покроковий режим виконання інструкцій, покрокове виконання підпрограм, збереження точок останову в призначеному для користувача файлі, пошук даних в пам'яті, створення макросів і запис їх у файл. Для використовування як середовище для злому не призначений.

4. **Debug**

Одним з найперших відладчиків, що існували для IBM PC, є відладчик DEBUG, що поставляються з операційною системою MsDos. Містить всі помилки відладчиків реального режиму. В даний час ніде не використовується.

Відладчики захищеного режиму.

1. **TurboDebugger/386 Borland International**

Надбудова над TD, представляюча device-driver TDH386.Sys (низькорівневий інтерфейс співпроцесору) і запускаючу програму TD386, що вводить процесор в режим V86. Повністю підтримує помилки свого попередника. Надає можливість установки апаратних точок останову: по звертанню на читання/запис байта в пам'яті, перекриття звернення до портів (не завжди коректно оброблюване).

2. **Soft-Ice by Nu-Mega Techologies**

Найбільш портужний відладчик. Підтримка VCPI. Є різновиди під Win95/WinNT. Містить також деякі помилки:

a. Не є повністю stealth-відладчиком, оскільки залишає частину свого коду в conventional memory V86 машини.
b. Існує API між програмою і відладчиком
c. S-Ice можна знайти по пристрою SOFTICE1
d. Завантажувач LDR неправильно виставляє значення SP
e. некоректне виставлення початкових значень регістрів

3. **Soft-Ice/W by Nu-Mega Techologies**

Відладчик під Windows 3.xx. Виявляється по присутності VxD пристрою WINICE. Відстежує конструкції виду cs:pushf.

4. **Deglucker S.Gorokhov & A.Ilyushin**

Помилки:

a. Перемикання в нестандартний відеорежим
b. Неможливість перехоплення портів вводу-виводу
c. Запирання клавіатури через i/o портів 60h/64h

Емулятори

1. **EDB Serge Pachkovsky**

Емулятор 80286 процесора. Украй убогий інтерфейс, на рівні DEBUG. Є можливість перегляду/зміни пам'яті, декілька режимів емуляції.

2. **Soft Debugger**

Повноцінний емулятор 80386, без підтримки функцій захищеного режиму. Підтримує налагоджувальну інформацію компіляторів Borland International. Відстежує зміну байтів в конвейєрі, є декілька режимів емуляції: з викликом власного int 1/int 3, режим Full Tracing, Auto Tracing і інші.

3. **SD Dmitry Groshev**

Зручний і потужний сервіс. Гнучкі можливості для роботи з найрізноманітнішими структурами даних. Може підвантажувати сервісні модулі.

Автоматичні розпаковщики.

До автоматичних розпаковщиків відносяться програми, що запускають в автоматичному або напівавтоматичному режимі програму і відстежують типові ділянки startup-коду, що захищається, і відповідно налагоджувальні relocations.

Функція автоматичних розпаковщиків - здирання захисту з файлу і отримання працездатного EXE файлу. Фізика даного процесу така: перехоплюючи перше програмне переривання, викликане програмою після відпрацювання захисту, розпаковщик знімає дамп пам'яті з вже розшифрованим кодом захищеної програми. Перший етап роботи по зняттю - знаходження цього найпершого переривання. Це робиться за допомогою будь-якого відладчика.

1. **Autohack BCP group**

Надає три варіанти запуску:

1. Розпаковування трасуванням. В даному режимі працюють майже всі розпаковщики програм. В даному режимі можливе розпаковування програм, не захищених від трасування. Режим працює таким чином: програма завантажується в пам'ять, перехоплюється перше переривання, встановлюється прапорець покрокового трасування, управління передається завантаженій програмі, далі обробник першого переривання аналізує сегмент коду трасованої програми і чекає зміни регістра CS. Після цього скидаються дампи пам'яті, і операція повторюється із завантаженням програми з іншої початкової адреси.
2. Стандартний режим злому. Режим роботи програми заснований на режимі перехоплення певних моментів після відпрацювання механізму захисту і скидання дампів пам'яті.
3. Режим злому з підтримкою таблиць компіляторів. Ідентичний другому режиму, але націлений на певні компілятори, тому зламує більш коректно.

Дизасемблери

Дизасемблер переводить виконуваний код в лістинг на асемблері.

Інші програми

До інших програм можна віднести програми орієнтовані на спеціальні мови, наприклад:

- Clipper

Valkyrie Declipper 5, Version 1.0, Revision K

"+" Є можливість аналізувати низькорівневий код, декомпілювання до вихідних текстів

"-" Може працювати тільки з відомими йому лінкувальниками, якщо лінкувальник йому не відомий, то він відмовляється працювати.

Hackers Declipper v1.3 KrK //UCL

"+" Дозволяє аналізувати низькорівневий код, можна самостійно задати початок псевдокоду і таблиці імен змінних.

"-" Повністю ручна робота при декомпілюванні, не розпізнає початку процедур, не створює початкового тексту, і т.д і т.п.

Rescue5 v1.0 CA-Clipper decompiller

"+" Декомпілювання до вихідних текстів.

"-" Розуміє дуже мало лінкерів, немає можливості аналізувати псевдокод

- FoxPro

ReFox

Програми типу Hiew (Hacker's view SEN), що дозволяють проглянути код, змінити його, дизасемблювати ділянки коду.

Тема 3: Захист від відладчиків.

План

1. Методи захисту ПЗ від дослідження.
2. Способи захисту ПЗ від дослідження.
3. Способи вбудовування захисних механізмів в програмне забезпечення.

Література: 2.с. 563..755

Питання для самоконтролю:

1. Основа методу захисту ПЗ від дослідження.
2. Які компоненти повинна включати захищувана від дослідження програма?
3. Динамічне дешифрування виконуваного коду.
4. Функції безпеки спрямовані на захист.
5. Способи захисту від дослідження.
6. Робота механізму захисту від дослідження.
7. Спосіб динамічного перетворення.
8. Вбудовування захисних механізмів.\

Методи захисту ПЗ від дослідження

Для захисту програм від дослідження необхідно застосовувати методи захисту від дослідження файлу з виконуваним кодом програми, що зберігається на зовнішньому носії, а також методи захисту виконуваного коду, який завантажується в оперативну пам'ять для виконання цієї програми. У першому випадку захист може бути заснований на шифруванні конфіденційної частини програми, а в другому - на блокуванні доступу до виконуваного коду програми в оперативній пам'яті з боку відгадчика. Крім того, перед завершенням роботи програми повинен обнулятися весь її код в оперативній пам'яті. Це запобігає можливості несанкціонованого копіювання з оперативної пам'яті дешифрованого виконуваного коду після виконання програми.

Таким чином, захищувана від дослідження програма повинна включати наступні компоненти:

1. ініціалізатор;
2. зашифровану конфіденційну частину;
3. деструктор (деіцініалізатор).
4. Ініціалізатор повинен забезпечувати виконання таких функцій:
5. збереження параметрів операційного середовища функціонування (векторів переривань, вмісту регістрів процесора і т.д.);
6. заборона всіх внутрішніх і зовнішніх переривань, обробка яких не може бути записана в програмі;
7. завантаження в оперативну пам'ять і дешифрування коду конфіденційної частини програми;
8. передача керування конфіденційної частини програми.

Для більшої надійності ініціалізатор може бути частково зашифрованим, по мірі виконання може дешифрувати сам себе. Дешифруватися по мірі виконання

може і конфіденційна частина програми. Таке дешифрування називається *динамічним дешифруванням виконуваного коду*. У цьому випадку чергові ділянки програм перед безпосереднім виконанням розшифровуються, а після виконання відразу знищуються. Для підвищення ефективності захисту програм від дослідження необхідно внесення в програму додаткових функцій безпеки, спрямованих на захист від трасування. *До таких функцій можна віднести:*

1. періодичний підрахунок контрольної суми області оперативної пам'яті, займаної вихідним кодом; порівняння поточної контрольної суми з попередньо сформованою еталонною і прийняття необхідних заходів у випадку розбіжності;
2. перевірку кількості займаної програмою оперативної пам'яті;
3. порівняння з обсягом, до якого програма адаптована, і прийняття необхідних заходів у разі невідповідності;
4. контроль часу виконання окремих частин програми;
5. блокування клавіатури на час відпрацювання особливо критичних алгоритмів.

Способи захисту ПЗ від дослідження

Способи захисту від дослідження можна розділити на чотири класи.

1. Спосіб, сутність якого полягає у наданні впливу на процес функціонування налагоджувальному кошти через спільні програмні або апаратні ресурси. В даному випадку найбільш відомі:

• використання апаратних особливостей мікропроцесора (особливості черговості вибірки команд, особливості виконання команд і т.д.);

• використання загального програмного ресурсу (наприклад, загального стека) і руйнування даних або коду відладчика, що належать загальному ресурсу, або перевірка використання загального ресурсу тільки захищуваною програмою (наприклад, визначення стека в області, критичній для виконання захищуваної програми);

• переадресація обробників налагоджувальних подій (переривань) від налагоджувальних засобів до захищуваної програми.

2. Вплив на роботу налагоджувальних засобів шляхом використання особливостей його апаратного або програмного середовища. Наприклад:

• переміщення фрагментів коду або даних за допомогою контролера прямого доступу до пам'яті;

• впливу на процес регенерації оперативної пам'яті (на деякій ділянці коду регенерація пам'яті відключається, а потім знову включається, - при нормальній роботі ніяких змін немає, при повільному виконанні програми відладчиком вона «зависає»);

• переходу мікропроцесора в захищений режим.

3. Вплив на роботу відладчика через органи управління або / та пристрої відображення інформації. Видавана налагоджувальними засобами інформація аналізується людиною. Отже, додатковий спосіб захисту від налагодження це порушення процесу спілкування оператора і відладчика, а саме спотворення або блокування вводу з клавіатури і виводу на термінал інформації.

4. Використання принципових особливостей роботи керованого людиною відладчика. У даному випадку захист від дослідження полягає в нав'язуванні для аналізу надмірно великого обсягу коду (як правило, за рахунок циклічного виконання деякої його ділянки).

Робота механізму захисту від дослідження виглядає таким чином. Програма повноциклового перетворення починає роботу з детермінованого або випадкового значення. На встановленому значенні проводиться дешифрування зашифрованої ділянки коду. Правильність дешифрування перевіряється підрахунком значення хеш-коду розшифрованої ділянки програмного коду з використанням елементів, пов'язаних з налагодженням (стек, налагоджувальні переривання та ін.) Хеш-функція повинна з імовірністю 1 визначати правильність дешифрування (для цього значення хеш-коду повинно бути не менше k).

Спосіб динамічного перетворення полягає в наступному: спочатку в оперативну пам'ять завантажується фрагмент коду, зміст частини команд якого не відповідає тим командам, які цей фрагмент у дійсності виконує; потім цей фрагмент по деякому закону перетворюється, перетворюючись на виконавчі команди, які потім і виконуються.

Перетворення коду програми під час її виконання може переслідувати три основні мети:

• протидія файловому дизасемблюванню програми;

• протидія роботі відладчика;

• протидія зчитуванню коду програми у файл з оперативної пам'яті.

Способи вбудовування захисних механізмів в програмне забезпечення

Вбудовування захисних механізмів можна виконати наступними основними способами:

• ставкою фрагмента перевірочного коду у виконуваний файл;

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

• вставкою перевірочного механізму у вихідний код на етапі розробки та налагодження програмного продукту;

• комбінуванням зазначених методів.

Тема 4. Програми з потенційно небезпечними наслідками.

План

1. Вірус.
2. Люк.
3. Троянський кінь.
4. Логічна бомба.
5. Програмні закладки.
6. Атака салямі.

Література: 3.с. 163..255

Питання для самоконтролю:

1. Що називається програмою з потенційно небезпечними наслідками?
2. Які дії здатна виконати програма з потенційно небезпечними наслідками?
3. Як можна розділити програми з потенційно небезпечними наслідками?
4. Як класифікуються програми з потенційно небезпечними наслідками по методу і місцю їх впровадження і застосування?
5. Що називається вірусом?
6. За якими ознаками поділяються віруси на класи?
7. Які віруси розрізняють по середовищу існування?
8. Які бувають способи зараження?
9. Як можна поділити віруси по деструктивним можливостям?
10. Що називається люком?
11. Що таке троянський кінь?
12. Що таке логічна бомба?
13. Що називається програмними закладками?
14. Три основні групи деструктивних функцій, які можуть виконуватися закладками.
15. Що таке атака салямі?

Програмою з потенційно небезпечними наслідками називається програма або частина програми, яка здатна виконати одну з наступних дій:

1. приховати ознаки своєї присутності в програмному середовищі ПЕОМ;
2. самодублюватися, асоціювати себе з іншими програмами і/або переносити свої фрагменти в які-небудь області оперативної або зовнішньої пам'яті, що не належать програмі;
3. змінювати код програм в оперативній або зовнішній пам'яті;
4. зберігати фрагменти інформації з оперативної пам'яті в деяких областях зовнішньої пам'яті (локальних або видалених);
5. спотворювати довільним чином, блокувати і/або підміняти той масив інформації, що виводиться в зовнішню пам'ять або канал зв'язку, що утворився в результаті роботи прикладних програм, або вже що знаходиться в зовнішній пам'яті масиву даних.

Програми з потенційно небезпечними наслідками можна умовно підрозділити на:

1. класичні програми-"віруси";
2. програми типу "програмний черв'як" або "троянський кінь" і фрагменти програм типу "логічний люк";
3. програми типу "логічна бомба";
4. програмні закладки - узагальнений клас програм з потенційно небезпечними наслідками.

Крім того, такі програми можна класифікувати по методу і місцю їх впровадження і застосування (тобто за "способом доставки" в систему):

1. закладки, пов'язані з програмно-апаратним середовищем (BIOS);
2. закладки, пов'язані з програмами первинного завантаження;
3. закладки, пов'язані з драйвером DOS, командним інтерпретатором, мережними драйверами, тобто із завантаженням і роботою операційного середовища;
4. закладки, пов'язані з прикладним програмним забезпеченням загального призначення (вбудовані в клавіатурні і екранні драйвери, програми тестування ПЕОМ, утиліти, файлові оболонки);
5. виконувані модулі, що містять тільки код закладки (як правило, впроваджувані в пакетні файли типу BAT);
6. модулі-імітатори, співпадаючі на вигляд з легальними програмами, що вимагають введення конфіденційної інформації;
7. закладки, масковані під програмні засоби оптимізаційного призначення (архіватори, прискорювачі і т.д.);
8. закладки, масковані під програмні засоби ігрового і розважального призначення (як правило, використовуються для первинного впровадження інших закладок; умовна назва - "дослідник").

Вірус.

Комп'ютерним вірусом називається програма, яка може створювати свої копії (не обов'язково співпадаючі з оригіналом) і впроваджувати їх у файли, системні області комп'ютера, мережі і так далі. При цьому копії зберігають здатність подальшого розповсюдження.

Віруси можна розділити на класи по наступних ознаках:

1. по середовищу існування вірусу;
2. за способом зараження середовища існування;
3. по деструктивних можливостях.

По середовищу існування розрізняють віруси мережні, файлові, завантажувальні і спеціальні. Мережні віруси розповсюджуються по комп'ютерній мережі, файлові впроваджуються у виконувані файли, завантажувальні в завантажувальний сектор диска (Boot) або сектор, що містить системний завантажувач вінчестера (Master Boot Record). Спеціальні орієнтовані на конкретні особливості ПЗ, наприклад вірус, що заражає документи редактора Word. Існують поєднання - наприклад, файлово-завантажувальні віруси, що заражають і файли і завантажувальні сектори дисків. Крім того, по мережі можуть розповсюджуватися віруси будь-яких типів.

Способи зараження діляться на резидентний і нерезидентний. Резидентний вірус при інфікуванні комп'ютера залишає в оперативній пам'яті свою резидентну

частину, яка потім перехоплює звернення операційної системи до об'єктів зараження і впроваджується в них. Резидентні віруси знаходяться в пам'яті і залишаються активними аж до виключення або перезавантаження комп'ютера. Нерезидентні віруси не заражують пам'ять комп'ютера і є активними обмежений час. Деякі віруси залишають в оперативній пам'яті невеликі резидентні програми, які не поширюють вірус. Такі віруси вважаються нерезидентними.

По деструктивних можливостях віруси можна розділити на:

1. нешкідливі, ніяк не впливаючі на роботу комп'ютера (окрім зменшення вільної пам'яті на диску в результаті свого розповсюдження);
2. безпечні, вплив яких обмежується зменшенням вільної пам'яті на диску і графічними, звуковими і іншими ефектами;
3. небезпечні віруси, які можуть привести до серйозних збоїв в роботі комп'ютера;
4. дуже небезпечні віруси, які можуть привести до втрати програм, знищити дані, сприяти прискореному зносу або пошкодженню частин механізмів (наприклад, головок вінчестерів).

Люк.

Люком називається не описана в документації на програмний продукт можливість роботи з цим програмним продуктом. Сутність використання люків полягає в тому, що при виконанні користувачем деяких не описаних в документації дій він дістає доступ до можливостей і даних, які в звичайних умовах для нього закриті (зокрема - вихід в привілейований режим).

Люк (або люки) може бути присутній в програмі з огляду на те, що програміст:

1. забув видалити його;
2. умисно залишив його в програмі для забезпечення тестування або виконання частини відладки, що залишилася;
3. умисно залишив його в програмі на користь полегшення остаточної збірки кінцевого програмного продукту;
4. умисно залишив його в програмі з тим, щоб мати прихований засіб доступу до програми вже після того, як вона ввійшла до складу кінцевого продукту.

Троянський кінь

Існують програми, що реалізовують, крім функцій, описаних в документації, і деякі інші функції, в документацію не описану. Такі програми називаються "троянськими кіньми".

Логічна бомба.

"Логічною бомбою" звичайно називають програму або навіть ділянку коду в програмі, реалізовуючий деяку функцію при виконанні певної умови.

Програмні закладки.

Для того, щоб закладка змогла виконати які-небудь функції по відношенню до іншої прикладної програми, вона повинна отримати управління на себе, тобто процесор повинен почати виконувати інструкції, що відносяться до коду закладки.

Це можливо тільки при одночасному виконанні двох умов:

1. закладка повинна знаходитися в оперативній пам'яті до початку роботи програми, на яку направлена її дія;

2. закладка повинна активізуватися по деякій загальній для закладки і для прикладної програми події.

Несанкціонований запис закладкою може відбуватися:

1. в масив даних, не співпадаючий з призначеною для користувача інформацією - збереження інформації;
2. в масив даних, співпадаючий з призначеною для користувача інформацією або її підмножиною - спотворення, знищення або нав'язування інформації закладкою.

Три основні групи деструктивних функцій, які можуть виконуватися закладками:

1. збереження фрагментів інформації, що виникає при роботі користувачів, прикладних програм, введення-виведення даних, в зовнішній пам'яті мережі (локальної або видаленої) або виділеної ПЕОМ, у тому числі різних паролів, ключів і кодів доступу, власне конфіденційних документів в електронному вигляді;
2. зміна алгоритмів функціонування прикладних програм (тобто цілеспрямована дія в зовнішній або оперативній пам'яті), наприклад, програма розмежування доступу стане пропускати користувачів по будь-якому паролю;
3. нав'язування деякого режиму роботи (наприклад, при знищенні інформації - блокування запису на диск, при цьому інформація, природно, не знищується), або заміна записуваної інформації інформацією, нав'язаною закладкою (наприклад, при виводу на екран слово "невірно" замінюється словом "вірно", а "рубель" - "долар" і т.д.).

Комплекс організаційно-технічних заходів захисту від вірусів і програмних закладок.

Заходи захисту можна підрозділити на дві основні групи:

- Заходи захисту на етапі розробки програмного забезпечення (ПЗ) системи.
- Заходи захисту на етапі експлуатації.

Атака салямі

При розробці банківських систем встановлюється правило округлення (або усікання), що використовується при виконанні всіх операцій.

Тема 5: Брандмауер.

План

1. Опис брандмауера.
2. Класифікація брандмауерів.
3. Пакетний рівень.
4. Прикладний рівень.
5. Рівень з'єднання.
6. Функції брандмауерів.

Література: 2.с. 376..403

Питання для самоконтролю:

1. Що таке брандмауер?
2. По якому критерію функціонують брандмауери?
3. Робота брандмауера на пакетному рівні.
4. Прикладний рівень роботи брандмауера.
5. Робота на рівні з'єднання.
6. Функції брандмауерів.
7. Недоліки.

Загальний опис

Брандмауер (firewall, міжмережевий екран) – це система, яка дозволяє розділити мережу на дві або більше частин і реалізувати набір правил, визначаючих умову проходження інформації із однієї частини в другу. Брандмауери уявляють собою цілий клас систем, іноді порівнянних по складності з ОС. Класифікувати їх можна по виконанню: програмні, апаратні та апаратні і змішаного типу (апаратно-програмного); по компонентній моделі: локальні (працюючі на одному хості) і розподілені (distributed firewall). Проте найбільш "корисною" буде класифікація з точки зору рівня, на якому функціонують брандауери: пакетний рівень, прикладний, рівень зв'язку. Розбивка на рівні є умовною, що має на увазі можливість роботи окремо взятого брандауера більше ніж на одному рівні одночасно. Можна сказати, що практично всі сучасні брандауери функціонують одразу на кількох рівнях, прагнучи розширити функціональність і максимально використати переваги роботи по тій або іншій схемі. Така технологія отримала назву – Stateful Inspection, а брандмауери, працюючі по змішаній схемі, називаються – Stateful Inspection Firewall.

Пакетний рівень

Робота на пакетному рівні полягає у фільтрації пакетів. Рішення про те пропустити даний пакет чи ні, приймається на основі наступної інформації: IP-адреса, номера портів відправника і отримувача, флаги. По суті, задача адміністратора зводиться до складання простої таблиці, на основі якої здійснюється фільтрація.

Переваги пакетного рівня:

1. низька вартість;
2. висока продуктивність.

Недоліки:

1. складність конфігурування і підтримки;
2. відсутність додаткових можливостей;
3. зруйнована мережа залишається відкритою (не захищеною);
4. не захищені від фальсифікації IP- і DNS-адреса

Прикладний рівень

Фактично брандмауери даного рівня уявляють собою кілька окремих підсистем (так званих (application gateways – серверів прикладного рівня), по числу обслуговуємих сервісов. Між користувальним процесом і потрібним сервісом виникає посередник, пропускаючий через себе трафік і приймаючий в рамках встановленої політики безпеки рішення про його легітимість.

Як правило, сучасні брандмауери підтримують практично увесь спектр існуючих сервісов – HTTP, FTP, SMTP, POP3, IMAP, Telnet, Gopher, Wais, Finger, News, - дозволяючи або повністю блокувати той чи інший сервіс, або ввести якісь обмеження. Такий принцип роботи забезпечує ряд додаткових можливостей в області безпеки: по-перше, маскується структура захищеної мережі; по-друге, з'являється можливість більш гнучно керувати доступом – наприклад, через надання різних прав окремим категоріям користувачів і т.д.

Переваги прикладного рівня:

1. маскування захищаемої мережі;
2. широкі можливості (посилена аутентифікація);
3. зруйнована мережа залишається заблокованою (захищеною)

Недоліки:

1. висока вартість;
2. низька продуктивність.

Рівень з'єднання

Шлюз на рівні з'єднання уявляє собою систему, транслюючу з'єднання зовні. При встановленні доступу процес користувача з'єднується з брандмауером, який, в свою чергу, самостійно встановлює з'єднання з зовнішнім вузлом. Під час роботи брандмауер просто копіює вхідну/вихідну інформацію. Данний шлюз потрібно розглядати не як самостійний механізм, а як специфічне рішення деяких задач (наприклад, для роботи з нестандартними протоколами, якщо необхідно створити систему збору статистики для якогось незвичного сервіса, надати доступ тільки певним зовнішнім адресам, здійснити базовий моніторинг і т.д.).

Функції брандмауерів

Ідеальний персональний брандмауер повинен виконувати шість функцій:

1 функція – Блокування зовнішніх атак. В ідеалі брандмауер повинен блокувати всі відомі типи атак, включаючи сканування портів, IP-спуффинг, DoS і DdoS, підбір поролів та інші.

2 функція – Блокування витоку інформації. Навіть коли шкідливий код проник в комп'ютер (не обов'язково через мережу, а, наприклад у вигляді вірусу на купленому піратському CD), брандмауер повинен зупинити витік інформації, заблокувавши вірусу вхід у мережу.

3 функція – Контроль додатків. Неминуча наявність відкритих дверей (тобто відкритих портів) є одним із самих слизьких місць в блокуванні витіку інформації, а

один із надійних способів перешкодити проникненню вірусів через ці двері – контроль додатків, запитуючих дозвіл на доступ.

4 функція – Підтримка зонального захисту. Робота у локальній мережі часто має на увазі практично повну довіру до локального контенту. Це відкриває унікалльні можливості по використанню новітніх (і, як правило, потенційно небезпечних) технологій. В цей же час рівень довіри к Інтернет-контенту значно нижчий, а отже значить, необхідним диференціюємим підхідом до аналізу небезпеки того чи іншого змісту.

5 функція – Протоколювання и попередження. Брандмауер повинен збирати строго необхідний обсяг інформації. Надлишок (рівно як і недолік) відомостей неприпустим.

6 функція – Максимально прозора робота. Єфективність і вживання системи часто обернено пропорційні складності її налаштування, супроводу і адміністрування.

Недоліки брандмауерів:

1. Розореність систем захисту – в більшості брандмауерів відсутній захист від ”саботажа” зі сторони авторизованих користувачів. Брандмаунри не взмозі заборонити авторизованому користувачу вкрасти важливу інформацію.
2. Відсутність захисту для нестандартних або нових мережевих сервісів – рішенням в цьому можуть бути шлюзи на рівні з’єднання або пакетні фільтри, але, як вже говорилося, їм невистачає гнучкості.
3. Зниження продуктивності – апаратні рішення демонструють відмінну продуктивність і масштабованість, але ціна (іноді обрахована десятками тисяч доларів) перекладає питання їх призначення в іншу площину, так що деколи максимум можливого – це виділення спеціального сервера, ”заточеного” виключно під обслуговування брандмауера.

Тема 6: Комп’ютерні атаки і їхнє виявлення.

План

1. Система аномальної поведінки.
2. Система виявлення зловживань.

Література: 7.с. 55..99

Питання для самоконтролю:

1. Що таке комп’ютерний вірус?
2. На які дві категорії можна розділити системи атак?
3. В чому полягає перевага системних мереж?
4. Які групи аспектів потрібні при виборі систем виявлення атак?

Комп'ютерні «віруси» – різновид комп'ютерних програм, які розповсюджуються та самовідтворюються, упроваджуючи себе у виконуваний код інших програм або в документи спеціального формату, що містять макрокоманди, такі, як MS Word. і Excel. Багато вірусів шкодять даним на заражених комп'ютерах, хоча іноді їх єдиною метою є лише зараження якомога більшої кількості комп'ютерів.

Історично технології, по яких будуються системи виявлення атак (intrusion detection systems) прийнято умовно поділяти на дві категорії: виявлення аномальної поведінки (anomaly detection) і виявлення зловживань (misuse detection). Однак у практичній діяльності застосовується інша класифікація, що враховує принципи практичної реалізації таких систем - виявлення атак на рівні мережі (network-based) і на рівні хосту (host-based). Перші системи аналізують мережний трафік, у той час як другі - реєстраційні журнали операційної системи чи додатка.

Системи виявлення аномальної поведінки – технологія, по якій побудовані дані системи, заснована на гіпотезі, що атака чи якась ворожа дія користувача часто виявляється як відхилення від нормальної поведінки. Прикладом аномальної поведінки може служити велике число з'єднань за короткий проміжок часу, високе завантаження центрального процесора чи використання периферійних пристроїв, що звичайно не задіються користувачем. Саме цій технології була присвячена перша публікація (1980 р. J. P. Anderson), де описувалася модель виявлення аномальної поведінки на основі даних у реєстраційних журналах. Робота була в основному присвячена розробці алгоритмів автономного (offline) аналізу журналів реєстрації операційної системи.

Системи виявлення зловживань – інший підхід до виявлення атак - виявлення зловживань, що полягає в описі атаки у вигляді шаблона (pattern) чи сигнатури (signature) і пошуку даного шаблона в контрольованому просторі (мережному трафіку чи журналі реєстрації). Ця технологія дуже схожа на виявлення вірусів (антивірусні системи являють яскравий приклад системи виявлення атак), тобто система може знайти усі відомі атаки, але вона мало пристосована для виявлення нових, ще невідомих атак.

Міжмережний екран забезпечує попереднє фільтрування, а кожен модуль спостереження системи виявлення атак - це фільтр біля кожного конкретного додатка.

Як уже було відзначено вище, існують два класи систем, що виявляють атаки на мережному й операційному рівні. *Принципова перевага мережних (network-based) систем* виявлення атак у тім, що напади ідентифікуються перш, ніж вони досягнуть вузла, що атакується. Ці системи простіше для розгортання у великих мережах, тому що вони не вимагають установки на різні платформи, використовувані в організації. Крім того, системи виявлення атак на рівні мережі практично не знижують продуктивності мережі.

Системи виявлення атак на рівні хосту були розроблені для роботи під управлінням конкретної операційної системи, що накладає на них певні обмеження. Такі системи сильно завантажують процесор і вимагають великих обсягів дискового простору для збереження журналів реєстрації й у принципі не застосовні для критичних систем, що працюють у режимі реального часу (наприклад, система "Операційний день банку" чи система диспетчерського управління). Ідеальним рішенням була б система виявлення атак, що поєднує в собі обидва ці підходи. Але на початок 1999 р. існує тільки одна система, що задовольняє цій вимозі. Це розроблена компанією Internet Security Systems система RealSecure, до складу якої входять мережний і системний агенти, що виявляють атаки на рівні мережі і хосту відповідно.

Існує і ще одна класифікація систем виявлення атак. Вона поділяє системи по тому, коли аналізуються дані - у реальному масштабі часу чи після здійснення події. Як правило, *системи виявлення атак на рівні мережі працюють* у реальному режимі часу, у той час як системи, що функціонують на рівні хосту, забезпечують автономний аналіз реєстраційних журналів операційної системи чи додатків. Однак і автономний аналіз має чимале значення. Він дозволяє докладно досліджувати, коли і як зловмисники проникнули у вашу систему, для того щоб виробити ефективні заходи протидії нападнику. Реалізовано такий аналіз може бути по-різному, починаючи від простої генерації звіту з інформацією про усіх чи обраних минулих подіях, і закінчуючи відтворенням (playback) у реальному часі всіх дій, зроблених при атаці (як це реалізовано в системі RealSecure).

Не розглядаючи докладно алгоритми виявлення атак, більш докладно зупинимося на деяких моментах, на яких варто звертати увагу при виборі системи для своєї організації. По-перше, це варіанти реагування на виявлену атаку. Як мінімум, система виявлення атак повинна видавати сигнал про тривогу на консоль адміністратора чи повідомляти його по електронній пошті. Більшість представлених на ринку систем має ці можливості. Найбільш просунуті системи пропонують і інші варіанти повідомлення. Наприклад, система виявлення атак RealSecure дозволяє посилати повідомлення про атаку на консоль деяких міжмережних екранів (наприклад, CheckPoint Firewall-1 і Lucent Managed Firewall), систем мережного управління (наприклад, HP OpenView чи CA Unicenter), а також генерувати сигнал тривоги і передавати його на пейджер, факс чи телефон. Системи виявлення атак пропонують і інші варіанти реагування - реконфигурація міжмережних екранів і маршрутизаторів для блокування доступу з вузла, що атакує, автоматичне

завершення з'єднання з вузлом, що атакує, блокування облікового запису користувача, що атакує, і т.д.

Друге, на що варто звернути свій погляд - архітектура системи. Існує два розповсюджених варіанти: архітектура "автономний агент" і "агент-менеджер". В обох випадках ви встановлюєте компоненти системи виявлення атак на виділені комп'ютери, але в другому випадку ви зможете централізовано керувати модулями спостереження з єдиної консолі (менеджера) та віддалено збирати реєстраційні дані, не відвідуючи кожен вузол, на якому встановлена система виявлення атак. У невеликих мережах це не має великого значення, однак, для великих організацій, у яких філії рознесені по різних територіях і навіть містах, це принциповий момент.

У цілому аспекти, які необхідно враховувати при виборі систем виявлення атак, можна умовно розділити на кілька груп:

а) Інсталяція і розгортання системи.
б) Безпека самої системи.
в) Виявлення атак.
г) Реагування на атаки.
д) Конфігурація системи.
е) Контроль подій.
є) Управління даними системи.
ж) Продуктивність системи.
з) Архітектура системи.
к) Технічна підтримка системи.

Таблиця 1 – Список відомих систем виявлення атак

Назва	Виробник	Протокол	Інтерфейс	Web-сервер	Тип виявлення атак	ОС	Приміт ка
RealSecure	Internet Security Systems	TCP/IP	Ethernet, Fast Ethernet, FDDDL, Token Ring	www.iss.net www.infosec.ru	На рівні мережі на рівні хосту	Windows NT, Unix	Перша система, що одержала поширення в Росії
OmniGuard Intruder Alert	Axent Technologies	Не застосовано	Не застосовано	www.axent.com	На рівні хосту	Windows NT, Unix, Netware	
NetRanger	Cisco Systems	TCP/IP	Ethernet, Fast Ethernet, FDDDL, Token Ring	www.cisco.com	На рівні мережі	Solanis	Система була розроблена компанією

							Wheel Group, що придбала корпорація Cisco.
Session Wall-3	MEMCO Software	TCP/IP	Ethernet, FDDDL, Token Ring	www.abimet.com	На рівні мережі	Windows NT, Windows 9x	
Kane Security Monitor	Security Dynamics	Не застосовно	Не застосовно	www.securid.com	На рівні хосту	Windows NT, Netware	Система була розроблена компанією Intrusion Detection, що придбала корпорація Security Dynamics
Network Flight Recorder	NFR	TCP/IP	Ethernet, Fast Ethernet, FDDI	www.nfr.com	На рівні мережі	Unix	

Тема 7: Безпека електронної комерції.

План

1. Інформаційна безпека.
2. Методи забезпечення інформаційної безпеки.

Література: 1.с. 269..365, 2.с. 316..359

Питання для самоконтролю:

1. Що таке захист інформації?
2. Які бувають три рівні політики безпеки?
3. Що таке мережевий екран?
4. Які вимоги до екрануючої системи?
5. Як відбувається шифрування інформації?
6. Які ви знаєте особливості SKIP?

Під інформаційною безпекою розуміється "стан захищеності інформації, оброблюваної засобами обчислювальної техніки або автоматизованої системи від внутрішніх або зовнішніх загроз". Захист інформації - це комплекс заходів, направлених на забезпечення інформаційної безпеки. На практиці під цим розуміється підтримка цілісності, доступності і, якщо необхідно, конфіденційності інформації і ресурсів, що використовуються для введення, зберігання, обробки і передачі даних. Комплексний характер, проблеми захисту говорить про те, що для її вирішення необхідне поєднання законодавчих, організаційних і програмно-технічних заходів.

Найчастішими і самими небезпечними (з погляду розміру збитку) є ненавмисні помилки користувачів, операторів, системних адміністраторів і інших осіб, обслуговуючих інформаційні системи. Іноді такі помилки приводять до прямого збитку (неправильно введені дані, помилка в програмі, що викликала зупинку або руйнування системи). Іноді вони створюють слабкі місця, якими можуть скористатися зловмисники (такі звичайно помилки адміністрування). Згідно даним Національного Інституту Стандартів і Технологій США (NIST), 65% випадків порушення безпеки ІС - слідство ненавмисних помилок.

На другому місці за розмірами збитку розташовуються крадіжки і фальсифікації. В більшості розслідуваних випадків винуватцями виявлялися штатні співробітники організацій, відмінно знайомі з режимом роботи і захисними заходами. Ключовим етапом для побудови надійної інформаційної системи є вироблення політики безпеки. Під політикою безпеки розуміють сукупність документованих управлінських рішень, направлених на захист інформації і пов'язаних з нею ресурсів.

З практичної точки зору політику безпеки доцільно розділити на три рівні:

- Рішення, що зачіпають організацію в цілому. Вони носять вельми загальний характер і, як правило, виходять від керівництва організації.
- Питання, що стосуються окремих аспектів інформаційної безпеки, але важливі для різних систем, експлуатованих організацією.
- Конкретні сервіси інформаційної системи.

Третій рівень включає два аспекти - мета (політики безпеки) і правило їх досягнення.

Методи забезпечення інформаційної безпеки

Міжмережеве екранування

Міжмережевий екран (firewall) - це засіб розмежування доступу клієнтів з однієї безлічі мережі до серверів з іншої безлічі мережі. Екран виконує свої функції, контролюючи всі інформаційні потоки між двома безліччю систем.

Вимоги до реальної системи, що здійснює міжмережеве екранування. В більшості випадків екрануюча система повинна:

- Забезпечувати безпеку внутрішньої (що захищається) мережі і повний контроль над зовнішніми підключеннями і сеансами зв'язку;
- Володіти могутніми і гнучкими засобами управління для повного і, наскільки можливо, простого втілення в життя політики безпеки організації;
- Працювати непомітно для користувачів локальної мережі і не утрудняти виконання ними легальних дій;
- Працювати достатньо ефективно і встигати обробляти весь вхідний і витікаючий трафік;
- Володіти властивостями самозахисту від будь-яких несанкціонованих дій, оскільки міжмережевий екран є ключем до конфіденційної інформації в організації;
- Якщо у організації є декілька зовнішніх підключень, у тому числі і у видалених філіалах, система управління екранами повинна мати нагоду централізований забезпечувати для них проведення єдиної політики безпеки;
- Мати засобу авторизації доступу користувачів через зовнішні підключення.

Екранування дозволяє підтримувати доступність сервісів усередині інформаційної системи, зменшуючи або взагалі ліквідовуючи навантаження, ініційоване зовнішньою активністю. Екранування дає можливість контролювати інформаційні потоки, направлені в зовнішню область, забезпечуючи режим конфіденційності.

Шифрування інформації

Технології інформаційної безпеки протоколу TCP/IP дозволяють з регульованим ступенем надійності захищати трафік всіх без виключення користувачів і прикладних систем при повній прозорості (невидимості для додатків) засобів захисту.

Протокол, що управляє шифруванням трафіку SKIP (Simple Key management for Internet Protocol) і створений на його основі ряд продуктів захисту інформації (ряд програмних реалізацій протоколу SKIP для базових апаратно-програмних платформ, пристрій колективного захисту локальної мережі SKIPBridge, пристрій забезпечення регульованої політики безпеки SunScreen).

SKIP (Simple Key mamagement for Internet Protocol - Простий протокол управління криптоключами в інтермережі) розроблений компанією Sun Microsystems в 1994 році і запропонований як стандарт Internet.

У основі SKIP лежить криптографія відкритих ключів Діффі-Хеллмана. SKIP має, в порівнянні з існуючими системами шифрування трафіку ряд унікальних особливостей:

- SKIP універсальний: він шифрує IP-пакеты, не знаючи нічого про додатки, користувачів або процеси, їх формуючих; він обробляє весь трафік, не накладаючи ніяких обмежень ні на вищерозміщене програмне забезпечення, ні на фізичні канали, на яких він використовується;
- SKIP сеансонезалежний: для організації захищеної взаємодії не вимагається додаткового інформаційного обміну;
- SKIP незалежний від системи шифрування - користувач може вибирати будь-який з пропонованих постачальником або використовувати свій алгоритм шифрування інформації.

IP-пакети, що приймаються із зовнішньої мережі, обробляються протоколом SKIP. Пакети, що пройшли фільтрацію SKIP, за допомогою протоколу IP передаються програмному забезпеченню SKIPBridge, вирішальному задачі адміністративної безпеки (забезпечуючому пакетну фільтрацію), і потім - операційній системі.

SunScreen - це спеціалізована система захисту, розроблена компанією Sun Microsystems, вирішальна задачі фільтрації пакетів, аутентифікації і забезпечення конфіденційності трафіку. Пристрій SunScreen виконаний на основі апаратного модуля SPF-100. SPF-100 містить SPARC-процессор, що працює під управлінням ОС Solaris. SunScreen не має IP-адреса, тому він "невидимий" із зовнішньої мережі і несхильний до атаки.

Пристрій SunScreen, містить п'ять Ethernet-адаптеров, до яких можуть під'єднуватися чотири незалежні сегменти локальної мережі і комунікаційний провайдер. Для кожного сегменту забезпечується настройка індивідуальної політики безпеки шляхом завдання складного набору правил фільтрації пакетів.

System Security Scanner

Програма System Security Scanner призначена для проведення перевірки стану безпеки на окремих UNIX-компьютерах і пошуку уразливості ОС як зовні (по мережі з використанням Internet Scanner), так і зсередини (з самої ОС комп'ютера). При цьому проводиться перевірка прав доступу і прав власності файлів, конфігурацій мережних сервісів, установок ресурсів користувачів, програм аутентифікації і інших, пов'язаних з користувачами слабких місць (наприклад, паролів).

Web Secure Scanner

Призначений для пошуку слабких місць безпеки на web-серверах. Забезпечує аудит ОС, під управлінням якої працює web-сервер, програм-додатків, встановлених на web-сервере, і CSI scripts в web-приложениях. Проводить тестування конфігурації web-сервера, оцінює рівень безпеки основної файлової системи.

Firewall Scanner

Забезпечує пошук слабких місць в міжмережевих екранах, перш за все в їх конфігурації, і надає рекомендації по їх корекції. Проводить тестування реакції міжмережевих екранів на різні типи спроб порушення безпеки. Виконує сканування сервісів - ідентифікацію всіх мережних сервісів, доступ до яких здійснюється через міжмережевий екран.

Intranet Scanner

Призначений для автоматичного виявлення потенційних слабких місць усередині мереж з використанням різних тестів для перевірки реакції на

несанкціоновані проникнення. Забезпечує перевірку різних мережних пристроїв, включаючи UNIX hosts, системи, що працюють під Microsoft NT/Windows 95, маршрутизатори, web-серверы і X-терминалы.

Сімейство продуктів SAFEsuite, розроблених американською компанією Internet Security Systems (ISS) - комплект засобів призначених для оцінки захищеності інформаційних систем.

Складається з системи аналізу захищеності на рівні мережі Internet Scanner, засобу аналізу захищеності на рівні хосту System Scanner і виявлення атак на рівні мережі RealSecure Network Engine, Database Scanner.

Internet Scanner

Система аналізу захищеності — Internet Scanner — призначена для проведення регулярних всесторонніх або вибіркових тестів мережних служб, операційних систем, прикладного, що використовується, ПЗ, маршрутизаторів, міжмережевих екранів, Web-серверов і т.п. Результатом тестування є звіти, що містять докладний опис кожної знайденої уразливості, її дислокації в корпоративній мережі, а також рекомендації по корекції або усуненню «слабкого місця».

Система аналізу захищеності — *Database Scanner* - призначена для виявлення проблем, пов'язаних з безпекою баз даних. В цьому ПО реалізовані перевірки підсистем аутентифікації, авторизації і контролю цілісності. Вбудована база знань, містить перелік коректуючих дій, що рекомендуються, для усунення знайдених уязвімостей.

Database Scanner підтримує СУБД Microsoft SQL Server, Sybase Adaptive Server, Oracle, Informix.

Тема 8: Безпека електронних платіжних систем.

План

1. Електронні платіжні системи та їх призначення.
2. Безпека електронних платіжних систем.

Література: 7.с 204..222

Питання для самоконтролю:

1. Що називаєтья електронною платіжною системою?
2. На які види поділяються електронні гроші?
3. Які існують групи платіжних систем?
4. Що називається пластиковими картами?
5. Чим забезпечується безпека електронних платіжних систем?
6. Вимоги до безпеки електронних платіжних систем.

Електронні платіжні системи (англ. Electronic Payment Systems) — призначені для здійснення платіжних операцій у всесвітній мережі Інтернет. За допомогою платіжної системи можна здійснювати розрахунок за товари та послуги різних проектів і сервісів. Наприклад, оплачувати мобільний зв'язок, комунальні послуги, кабельне або супутникове телебачення, послуги Інтернет-провайдерів, а також різноманітні покупки в Інтернет- магазинах.

Електронні гроші поділяють на два види по носію: на базі карток (card - based) і на базі мереж (network - based). У свою чергу, і перша, і друга група поділяються на анонімні системи, що дозволяють проводити операції без ідентифікації користувача, дані системи близькі по суті до традиційного поняття готівкових грошей. Не анонімні системи, вимагають обов'язкової ідентифікації учасників системи. Електронних платіжних засобів багато і споживач, часто втрачається у виборі .

Існуючі на даний момент електронні платіжні системи за типом доступу до електронного рахунку можна *розділити на 2 великі групи*:

- Вимагають установки на комп'ютер користувача додаткового програмного забезпечення;
- Платіжні системи мають веб-інтерфейс.

Пластикові карти - це персоніфікований платіжний інструмент, що дає котра має картою особі можливість безготівкової оплати товарів або послуг, а також отримання готівкових коштів у відділеннях банків або банківських автоматах (банкоматах). Приймаючі карту підприємства торгівлі та відділення банків утворюють мережу точок обслуговування картки.

Безпека електронних платіжних систем забезпечується тим, що:

• вся інформація про банківський рахунок і коди платіжних карт (номер карти, строк дії, CVV2), а також доступ до банківського рахунку здійснюється тільки на сайтах банку-партнера та не зберігається на порталі електронної платіжної системи;

• передача даних здійснюється по захищеному протоколу HTTPS (Hypertext Transfer Protocol Secure) з використанням 256-bit SSL (Secure Sockets Layer) сертифіката, що включає в себе EV (Extended Validation). Стандарт SSL гарантує безпеку передачі конфіденційних даних між порталом електронної платіжної

системи і браузером користувача за допомогою технології аутентифікації, шифрування і цифрового підпису;

• всі транзакції здійснюються в захищеній зоні фінансового порталу банку-партнера. Інформація про дані платіжного доручення надходить у банк за захищеними каналами і не має доступу до рахунку користувача. Платіж здійснюється банком тільки після особистого підтвердження користувача;

• для входу на портал електронної платіжної системи використовуються логін і пароль, які вибирає сам користувач. Якщо він сумнівається в конфіденційності пароля, він можете його змінити в своєму електронному кабінеті;

• для додаткового захисту від вірусних програм - шпигунів клавіатури, рекомендується використовувати віртуальну клавіатуру для введення логіна і пароля;

• платіжний пароль дає можливість управління акантом користувача, створюючи додатковий ступінь захисту від несанкціонованого зовнішнього доступу. Платіжний пароль генерується системою один раз після реєстрації і не зберігається на порталі електронної платіжної системи (його необхідно записати і зберегти зручним способом). Пароль так само необхідний для підтвердження переказу коштів з використанням електронного еквівалента валют;

• портал електронної платіжної системи блокує можливість підбору пароля трьома спробами входу в систему. Якщо тричі був введений невірний пароль, система не дозволить знову до неї увійти протягом 40 хвилин;

• невірне введення платіжного пароля три рази поспіль приводить до автоматичного блокування акаунта. Повторний доступ і розблокування можливі тільки після звернення до адміністратора системи;

• для активації акаунта користувача на вказаний ним e-mail відправляється лист, що містить посилання, для підтвердження реєстрації.

Адміністрація порталу електронної платіжної системи забезпечує і супроводжує процес розслідування всіх проблемних фактів платежу і гарантує повернення коштів потерпілій стороні.

Тема 9: Ідеальна служба інформаційної безпеки.

План

1. Ідеальні умови для реалізації інформаційної безпеки.

Література: 1.с 219..322

Питання для самоконтролю:

1. Які умови для реалізації інформаційної безпеки є ідеальними?
2. Перерахуйте пункти для реалізації ідеальних умов інформаційної безпеки.
3. Дайте повний опис пунктів реалізації ідеальних умов.

Щоб служба інформаційної безпеки в організації працювала в деяких ідеальних умовах (необмежений бюджет, достатня кількість кваліфікованих фахівців і т.п.) повинно бути реалізовано:

1. Вся робота служби інформаційної безпеки (як і всього підприємства) організована в суворій відповідності до законодавства держави.
2. Всі кандидати для прийому на роботу проходять співбесіду і перевірку службою інформаційної безпеки. Знову прийняті працівники і сторонні фахівці за контрактом проходять навчання основам безпеки, ознайомлення з нормативними документами з безпеки, тестіроване на рівень кваліфікації для роботи з інформаційними системами та підтверджують підписом своє обязательчтво слідувати нормативам організації (в тому числі і по безпеці). Звільняються працівники проходять співбесіду з представником служби безпеки, звільнення працівника відразу відбивається в інформаційних системах (блокування і видалення облікових записів, вилучення карток доступу і т.д.). Співробітники організації проходять періодичні семінари і навчання з інформаційної безпеки.
3. Ведеться аналіз моральної і психологічної обстановки в організації, облік порушень безпеки конкретними співробітниками для виявлення можливих тенденцій. Заохочується тісний інформаційний контакт користувачів зі службою безпеки. Вибірково або постійно аналізується електронна пошта співробітників з відповідним нормативним оформленням процедури. Регулярно проводиться авторизований моніторинг активності користувача на робочій станції.
4. Всі дані, об'єкти та інформаційні системи прокласифіковані. Суб'єкти розподілені за ролями. Визначена надійна структура та запроваджений механізм доступу суб'єктів до об'єктів з урахуванням найменших привілеїв і поділу обов'язків. Кожен новий об'єкт своєчасно класифікується і встановлюється в загальну схему інформаційного простору.
5. Забезпечено нормативний простір. Для всіх інформаційних систем існують політики, для функціональних обов'язків користувачів - правила, процедури та методики. Зміни в роботі організації та співробітників адекватно відображаються у відповідних документах.

6. Способи аутентифікації користувачів знаходяться під контролем служби інформаційної безпеки, тобто проводиться періодичний аналіз відсутності слабких паролів, фактів передачі паролів іншим особам, залишення токенів без нагляду і т.п. Ідентифікація користувачів стандартизована, є чітка таблиця відповідності користувач - мережеві адреси, дозволені для роботи даного користувача (username, IP-адресу, MAC-адресу і т.д.).
7. Зовнішній віддалений доступ в інформаційну мережу підприємства (вихід у зовнішнє інформаційний простір) обмежений єдиним центральним шлюзом плюс існує резервний канал зв'язку, неактивний в штатному режимі. Обидва канали захищені міжмережевим екранами, які надійно функціонують, коректно налаштовані і регулярно піддаються перевірці службою інформаційної безпеки. Всі модеми й інші пристрої віддаленого доступу враховані і також зведені до зазначеної єдиної точки входу/виходу. Зовні по відношенню до точки входу встановлено агент мережевої системи виявлення атак (СОА).
8. Агент мережевий СОА присутній на кожному сегменті в мережі організації. Для СОА сигнатурного типу бази даних сигнатур регулярно оновлюються, є фахівець зі створення власних сигнатур. Для СОА з виявленням аномалій визначені всі необхідні профілі, які регулярно переглядаються. Крім цього проводиться періодичний аналіз мережевої активності засобами мережевого моніторингу (перехоплювачами мережевих пакетів - сніффер).
9. Агенти СОА-хоста встановлені на кожному вузлі мережі, база даних атак (або профілі) регулярно оновлюються. Крім цього проводиться періодичний вибірковий аналіз реєстраційних журналів (які налаштовані на фіксацію всіх подій), в тому числі особисто уповноваженим персоналом. СОА інтегрована з міжмережевим екраном і іншими пристроями захисту.
10. Налаштовується докладне ведення реєстраційних журналів по всіх інформаційних системах. Ведеться регулярна автоматизована обробка цих журналів, а також періодичний вибірковий ручний їх аналіз на предмет виявлення підозрілих або злочинних подій. Система аналізу інформаційної активності інтегрована з системою фізичного доступу співробітників в будівлю і до робочих місць. Всі реєстраційні журнали зберігаються поряд з резервними копіями та архівами бізнес-даних.
11. На кожен комп'ютер в мережі встановлений антивірусний пакет, крім того антивіруси встановлені на поштовий сервер, міжмережевий екран і інші ключові вузли. Режим роботи антивіруса - перехоплення на льоту (autoprotect). Антивірусні бази оновлюються щодня, а також при надходженні інформації про новий вірус.
12. Ведеться суворий облік руху будь-якої одиниці апаратного забезпечення або її складової, а також суворий облік апаратної або програмної конфігурації кожного об'єкта або системи. При зміні конфігурації негайно проводиться збереження відповідних налаштувань. Ведеться регулярний моніторинг продуктивності роботи апаратного забезпечення, операційних систем, інформаційної системи в цілому.
13. Здійснена підписка на відповідні бюлетені з інформаційної безпеки, вивчаються повідомлення про нові атаки, віруси, уязвимостях і т.п., нових

рішеннях і механізмах з безпеки. Налагоджена процедура регулярного отримання і установки програмних латок (patches, hotfixes, updates і т.п.) і їхнє тестування. Виробляється аналітична робота по визначенню тенденцій розвитку атак, появі вразливостей, нових продуктів на ринку безпеки, нових технологій.

14. Всі канали обміну даними в інформаційній мережі криптографічно захищені, всередині локальної мережі організовані віртуальні мережі. Будь обмін даними реєструється в електронних журналах і забезпечений засобом контролю цілісності. Забезпечено контроль цілісності даних, системних файлів і т.п. на серверах і робочих станціях. Обмін даними між користувачами здійснюється з використанням електронного цифрового підпису. Організовано надійне управління криптографічними ключами.
15. Забезпечено контроль вхідної та вихідної інформації на зовнішніх носіях. Встановлені надійні захищені (можливо, віртуальні) шлюзи обміну інформацією із зовнішніми інформаційними системами (забезпечені відповідні інтерфейси, встановлено зв'язок з центром сертифікатів, отриманий кореневий сертифікат організації і т.д.).
16. Регулярно проводиться процедура тестування всієї мережі або окремих систем на злом. У розпорядженні служби інформаційної безпеки є команда програмістів для створення власних програм або утиліт для аналізу захищеності або, навпаки, для захисту об'єктів і систем.
17. Після зміни будь-якої одиниці даних забезпечується її резервне копіювання або організовано надлишкове збереження даних (RAID). Забезпечується географічно рознесене захищене збереження архівів і резервних копій. Всі мережеві пристрої мають можливість швидкої заміни (продубльовані), всі канали передачі даних мають альтернативні лінії. Будівля організації має гарячий резерв (hotside). Забезпечено безперебійне електроживлення (і контроль його роботи) основного та резервного будівель. Ключові працівники організації можуть виконувати функції один одного або мають функціональних дублерів. Мається регулярно оновлюваний аварійний план.
18. Контроль за будь-якою системою чи об'єктом децентрализован. Служба інформаційної безпеки бере участь в будь-якому проекті організації (в тому числі і в розробці програмного забезпечення) з правом внесення серйозних змін і заборон в рамках своїх функцій. Служба має повноваження до прийняття та реалізації рішень, пов'язаних з інформаційною безпекою.
19. Встановлена система відволікання уваги зловмисника (honeypot), сформована власна команда реагування на інциденти, встановлено зв'язок з аналогічними командами інших організацій, зовнішніми експертами, силовими структурами.
20. Забезпечено єдине узгоджене неротіворечівое управління всіма автоматизованими засобами інформаційної безпеки.

Слід звернути увагу на те, що одні роботи носять одноразовий характер по типу "зробили і забули". Другі - це разове виконання з подальшим мінімальним контролем відповідності поточного стану деяким умовам. Треті - періодичні роботи після закінчення проміжку часу або настання якоїсь події. І четверті - рутинні постійні роботи типу моніторингу. Природно, що в реальному житті не вдасться реалізувати цю ідеальну програму повністю. Однак, по-перше, вона позначає деяку

мету, до якої варто прагнути, а по-друге, багато з позначених пунктів можна реалізувати в усіченому варіанті.

Модуль 3. Криптографи.
Тема 1: Класифікація криптоалгоритмів. Скремблери.

План

1. Визначення криптографічного алгоритму.
2. Скремблери. Їх принцип роботи.

Література: 1.с. 475..476

Питання для самоконтролю:

1. Що називається криптоалгоритмом?
2. Способи захисту інформації КА.
3. Що називаєься тайнописом?
4. Які бувають криптоалгоритми з ключем?
5. Що таке скремблер?
6. Як працює скремблер?

Криптографічний алгоритм, або шифр - це математична формула, що описує процеси шифрування і розшифрування. Щоб зашифрувати відкритий текст, криптоалгоритм працює в сполученні з ключем - словом, числом або фразою.

Криптографічних алгоритмів існує безліч. Їх призначення в загальних рисах зрозуміло: захист інформації. Захищати ж інформацію потрібно від різних загроз і різними способами. Щоб правильно задіяти криптоалгоритм (КА), тобто забезпечити надійний і адекватний захист, потрібно розуміти, які бувають КА і який тип алгоритму краще пристосований для вирішення конкретного завдання.

Рисунок 6 – Класифікація крипографічних алгоритмів.

1. Тайнопис. Відправник і одержувач роблять над повідомленням перетворення, відомі лише їм двом. Стороннім особам невідомий сам алгоритм шифрування. Деякі фахівці вважають, що тайнопис не є криптографією взагалі.

2. Криптографія з ключем. Алгоритм впливу на передані дані відомий усім стороннім особам, але він залежить від деякого параметра - "ключа", яким володіють лише відправник і одержувач.

• *Симетричні криптоалгоритми.* Для зашифровки і розшифровки повідомлення використовується один і той же блок інформації (ключ);

• *Асиметричні криптоалгоритми.* Алгоритм такий, що для зашифровки повідомлення використовується один ("відкритий") ключ, відомий усім бажаючим, а для розшифровки - інший ("закритий"), який існує тільки в одержувача.

В залежності від кількості ключів, які застосовуються у конкретному алгоритмі:

• *Безключові КА* – не використовують в обчисленнях ніяких ключів;

• *Одноключові КА* – працюють з одним додатковим ключовим параметром (якимсь таємним ключем);

• *Двухключові КА* – на різних стадіях роботи в них застосовуються два ключових параметри: секретний та відкритий ключі.

В залежності від характеру впливів, що виробляються над даними, алгоритми підрозділяються на:

• *Перестановочні* - Блоки інформації (байти, біти, більші одиниці) не змінюються самі по собі, але змінюється їх порядок проходження, що робить інформацію недоступною сторонньому спостерігачеві.

• *Підстановочні* - Самі блоки інформації змінюються за законами криптоалгоритму. Переважна більшість сучасних алгоритмів належить цій групі.

Залежно від розміру блоку інформації криптоалгоритми поділяються на:

• *Потокові шифри* - Одиницею кодування є один біт. Результат кодування не залежить від минулого раніше вхідного потоку. Схема застосовується в системах передачі потоків інформації, тобто в тих випадках, коли передача інформації починається і закінчується в довільні моменти часу і може випадково перериватися. Найбільш поширеними представниками потокових шифрів являються скремблери.

• *Блочні шифри* - Одиницею кодування є блок з декількох байтів (в даний час 4-32). Результат кодування залежить від усіх вихідних байтів цього блоку. Схема застосовується при пакетній передачі інформації та кодування файлів.

Скремблер (англ. scrambled - зашифрований) - програмне або апаратне пристрій (алгоритм), що виконує скремблювання. *Скремблювання* - це обернене перетворення цифрового потоку без зміни швидкості передачі з метою отримання властивостей випадкової послідовності. Після скремблювання поява «1» і «0» у вихідний послідовності рівноймовірні. Скремблювання - оборотний процес, тобто вихідне повідомлення можна відновити застосувавши зворотний алгоритм.

Принцип роботи

Генерується псевдовипадкова послідовність (однакова для скремблера і дескремблера) біт. Знову надходить у скремблер біт підсумовується по модулю два з бітом псевдовипадковою послідовності. Після чого біт відправляється на вихід, скремблер бере наступний вхідний біт псевдовипадковою і вхідної послідовності і повторює операцію. Таким чином, в простому випадку скремблер може бути зібраний на двох тактируемого регістрах зсуву з зворотними зв'язками і 2-х входові елемента «виключає або». Зворотне перетворення здійснюється в зворотному порядку. Псевдослучайная послідовність використовується циклічно. Скремблювання застосовується в багатьох сучасних системах цифрового зв'язку (SDH).

Тема 2: Загальні відомості про блокові шифри. Мережа Фейштеля.

План
1. Загальні відомості про блокові шифри.
2. Мережа Фейштеля.

Література: 4.с. 22..45

Питання для самоконтролю:
1. Робота блокових шифрів.
2. Вимоги до блокових шифрів.
3. Умова функції стійкого блокового шифру.
4. Що таке мережа Фейштеля і її особливість.

Характерною особливістю блокових криптоалгоритмів є той факт, що в ході своєї роботи вони виробляють перетворення блоку вхідної інформації фіксованої довжини і отримують результуючий блок того ж об'єму, але недоступний для прочитання стороннім особам, які не володіють ключем. Таким чином, схему роботи блочного шифру можна описати функціями Z = EnCrypt (X, Key) і X = DeCrypt (Z, Key).

Ключ Key є параметром блокового криптоалгоритму і представляє собою деякий блок двійкової інформації фіксованого розміру. Вихідний (X) і зашифрований (Z) блоки даних також мають фіксовану розрядність, рівну між собою, але необов'язково рівну довжині ключа.

Блокові шифри є основою, на якій реалізовані практично всі криптосистеми. Методика створення ланцюжків із зашифрованих блоковими алгоритмами байт дозволяє шифрувати ними пакети інформації необмеженої довжини. Така властивість блокових шифрів, як швидкість роботи, використовується асиметричними криптоалгоритмами, повільними за своєю природою. Відсутність статистичної кореляції між бітами вихідного потоку блочного шифру використовується для обчислення контрольних сум пакетів даних і в хешуванні паролів.

Криптоалгоритм іменується ідеально стійким, якщо прочитати зашифрований блок даних можна тільки перебравши всі можливі ключі, до тих пір, поки повідомлення не виявиться осмисленим. Так як по теорії ймовірності шуканий ключ буде знайдено з ймовірністю 1/2 після перебору половини всіх ключів, то на злом ідеально стійкого криптоалгоритму з ключем довжини N буде потрібно в середньому 2N-1 перевірок. Таким чином, в загальному випадку стійкість блочного шифру залежить тільки від довжини ключа і зростає експоненціально з її зростанням. Навіть припустивши, що перебір ключів проводиться на спеціально створеній багатопроцесорної системі, в якій завдяки діагональному паралелізму на перевірку 1 ключа йде тільки 1 такт, то на злом 128 бітного ключа сучасній техніці буде потрібно не менше 1021 років. Природно, все сказане стосується лише ідеально стійким шифрам.

Крім цієї умови до ідеально стійким криптоалгоритмів застосовується ще одна дуже важлива вимога, якій вони повинні обов'язково відповідати. При відомих

вихідному і зашифрованому значеннях блоку ключ, яким вироблено це перетворення, можна дізнатися також тільки повним перебором. Ситуації, в яких сторонньому спостерігачеві відома частина вихідного тексту зустрічаються повсюдно. Це можуть бути стандартні написи в електронних бланках, фіксовані заголовки форматів файлів, досить часто зустрічаються в тексті довгі слова або послідовності байт. У світлі цієї проблеми описане вище вимога не є нічим надмірним і також строго виконується стійкими криптоалгоритмами, як і перше.

Таким чином, на функцію стійкого блочного шифру Z = EnCrypt (X, Key) накладаються наступні умови: Функція EnCrypt повинна бути оборотної. Не повинно існувати інших методів прочитання повідомлення X за відомим блоку Z, окрім як повним перебором ключів Key. Не повинно існувати інших методів визначення яким ключем Key було вироблено перетворення відомого повідомлення X у повідомлення Z, окрім як повним перебором ключів.

Мережа Фейштеля

Мережею Фейштеля називається метод оборотних перетворень тексту, при якому значення, обчислені від однієї з частин тексту, накладається на інші частини. Часто структура мережі виконується таким чином, що для шифрування і дешифрування використовується один і той же алгоритм - різниця полягає лише в порядку використання матеріалу ключа.

Особливістю модифікованої мережі Фейштеля в повторному використанні даних ключа в зворотньому порядку в другій половині циклу. Треба відзначити, що саме через таку специфіку схеми (потенційною можливістю ослаблення зашифрованого тексту зворотніми перетвореннями) її використовують в криптоалгоритмах з обережністю.

Модифікацію мережі Фейштеля для більшого числа віток використовують частіше. Це пов’язано з тим, що при великих розмірах кодуючи блоків (128 біт і більше) стає незручно працювати з математичними функціями по модулю 64 і вище. В блочних криптоалгоритмах найчастіше зустрічається мережа Фейштеля з 4-ма вітками.

Мережа Фейштеля зарекомендувала себе, як криптостійка схема і її можна знайти практично в кожному блочному шифрі.

Тема 3: Симетричні криптосистеми

План

1. Визначення симетричних криптосистем.
2. Принцип симетричних криптосистем.

Література: 4.с. 13..16

Питання для самоконтролю:

1. Що таке симетричні криптосистеми.
2. Приклади алгоритмів криптосистем.

Симетричні криптосистеми (також симетричне шифрування, симетричні шифри) - спосіб шифрування, в якому для шифрування і розшифрування застосовується один і той же криптографічний ключ. До винаходу схеми асиметричного шифрування єдиним існуючим способом було симетричне шифрування. Ключ алгоритму повинен зберігатися в секреті обома сторонами. Алгоритм шифрування обирається сторонами до початку обміну повідомленнями.

Алгоритми шифрування і дешифрування даних широко застосовуються в комп'ютерній техніці в системах приховування конфіденційної і комерційної інформації від не коректного використання сторонніми особами. *Головним принципом* у них є умова, що передавач і приймач заздалегідь знають алгоритм шифрування, а також ключ до повідомлення, без яких інформація являє собою всього лише набір символів, що не мають сенсу.

Класичним прикладом таких алгоритмів є симетричні криптографічні алгоритми, перераховані нижче:

- Проста перестановка;
- Одиночна перестановка по ключу;
- Подвійна перестановка;
- Перестановка "Магічний квадрат".

Проста перестановка

Проста перестановка без ключа - один з найпростіших методів шифрування. Повідомлення записується в таблицю по стовпцях. Після того, як відкритий текст записаний колонками, для освіти шифровки він зчитується по рядках. Для використання цього шифру відправнику і одержувачу потрібно домовитися про загальному ключі у вигляді розміру таблиці. Об'єднання букв в групи не входить у ключ шифру і використовується лише для зручності запису несмислового тексту.

Одиночна перестановка по ключу

Більш практичний метод шифрування, званий одиночній перестановкою по ключу дуже схожий на попередній. Він відрізняється лише тим, що колонки таблиці переставляються за ключовим словом, фразі або набору чисел довжиною в рядок таблиці.

Подвійна перестановка

Для додаткової скритності можна повторно шифрувати повідомлення, яке вже було зашифровано. Цей спосіб відомий під назвою подвійна перестановка. Для цього розмір другій таблиці підбирають так, щоб довжини її рядків і стовпців були

інші, ніж в першій таблиці. Найкраще, якщо вони будуть взаємно простими. Крім того, в першій таблиці можна переставляти стовпці, а в другій рядки. Нарешті, можна заповнювати таблицю зигзагом, змійкою, по спіралі або якимось іншим способом. Такі способи заповнення таблиці якщо і не посилюють стійкість шифру, то роблять процес шифрування набагато більш цікавим.

Перестановка "Магічний квадрат"

Магічними квадратами називаються квадратні таблиці зі вписаними в їх клітини послідовними натуральними числами від 1, які дають в сумі по кожному стовпцю, кожному рядку і кожній діагоналі одне і те ж число. Подібні квадрати широко застосовувалися для вписування тексту що шифрується за наведеною в них нумерації. Якщо потім виписати вміст таблиці по рядках, то виходила шифровка перестановкою літер. На перший погляд здається, ніби магічних квадратів дуже мало. Тим не менш, їх число дуже швидко зростає із збільшенням розміру квадрата. Так, існує лише один магічний квадрат розміром 3 х 3, якщо не брати до уваги його повороти. Магічних квадратів 4 х 4 налічується вже 880, а число магічних квадратів розміром 5 х 5 близько 250000. Тому магічні квадрати великих розмірів могли бути хорошою основою для надійної системи шифрування того часу, тому що ручний перебір всіх варіантів ключа для цього шифру був немислимий.

Тема 4: Хешування паролів.

План

1. Опис хешування паролів.
2. Властивості хешування.

Література: 4.с. 42..45

Питання для самоконтролю:

1. Метод хешування паролів.
2. Що таке пароль?
3. Які ви знаєте функції хешування?
4. Для чого призначені функції хешування?
5. Що таке хеш-функція?
6. Які вимоги до хеш-функції?

Хешування паролів – метод, що дозволяє користувачам запам`ятовувати, наприклад, не 128 байт, тобто 256 шестнадцятковий шифр ключа, а деякий осмислений вираз, слово чи послідовність символів, що називається паролем. Дійсно, при розробці будь-якого криптоалгоритмами слід враховувати, що в половині випадків кінцевим користувачем системи є людина, а не автоматична система. Це ставить питання про те, зручно, і взагалі чи реально людині запам'ятати 128-бітний ключ. Насправді межа запам'ятовуваності лежить на кордоні 8-12 подібних символів, а, отже, якщо ми будемо примушувати користувача оперувати саме ключем, тим самим ми практично змусимо його до запису ключа на будь-якому листку паперу або електронному носії, наприклад, в текстовому файлі. Це, звичайно, різко знижує захищеність системи. Для вирішення цієї проблеми були розроблені методи, які перетворюють осмислений рядок довільної довжини – пароль, у вказаний ключ заздалегідь заданої довжини. У переважній більшості випадків для цієї операції використовуються так звані хеш-функції. Хеш-функцією в даному випадку називається таке математичне або алгоритмічне перетворення заданого блоку даних, яке володіє наступними властивостями:

• хеш-функція має нескінченну область визначення;

• хеш-функція має кінцеву область значень;

• вона незворотня;

• зміна вхідного потоку інформації на один біт змінює близько половини всіх біт вихідного потоку, тобто результату хеш-функції.

Ці властивості дозволяють подавати на вхід хеш-функції паролі, тобто текстові рядки довільної довжини на будь-якою національною мовою і, обмеживши область значень функції діапазоном 0 .. 2N-1, де N - довжина ключа в бітах, отримувати на виході досить рівномірно розподілені по області значення блоки інформації – ключі.

Хеш функція – це деяка функція h(K), яка бере ключ K і повертає адресу, по якому проводиться пошук в хеш-таблиці, щоб отримати інформацію, пов'язану з K. *Колізія* — це ситуація, коли h(K1) = h(K2). У цьому випадку, очевидно, необхідно

знайти нове місце для зберігання даних. Очевидно, що кількість колізій необхідно мінімізувати. Хеш-функція повинна задовольняти двом вимогам:

• її обчислення повинно виконуватися дуже швидко;

• вона повинна мінімізувати число колізій.

Отже, перша властивість хеш-функції залежить від комп'ютера, а друга — від даних. Якщо б всі дані були випадковими, то хеш-функції були б дуже прості (кілька бітів ключа, наприклад). Однак на практиці випадкові дані зустрічаються вкрай рідко, і доводиться створювати функцію, яка залежала б від усього ключа. Теоретично неможливо визначити хеш-функцію так, щоб вона створювала випадкові дані з реальних невипадкових файлів. Однак на практиці реально створити достатньо хорошу імітацію за допомогою простих арифметичних дій. Більш того, часто можна використовувати особливості даних для створення хеш-функцій з мінімальним числом колізій (меншим, ніж при істинно випадкових даних). Можливо, одним з найбільш очевидних і простих способів хешування є метод середини квадрата, коли ключ піднімається до квадрату і береться декілька цифр у середині. Тут і далі передбачається, що ключ спочатку заокруглюється до цілого числа, для здійснення з ним арифметичних операцій. Однак такий спосіб добре працює до моменту, коли немає великої кількості нулів зліва або справа. Численні тести показали хорошу роботу двох основних типів хешування, один з яких заснований на розподілі, а інший на множенні. Втім, це не єдині методи, які існують, більш того, вони не завжди є оптимальними.

Тема 5: Транспортне кодування. Асиметричні криптоалгоритми.

План

1. Транспортне кодування.
2. Асиметричні криптоалгоритми.

Література: 1.с. 275..301

Питання для самоконтролю:

1. Що таке транспортне кодування?
2. В чому полягає мета транспортного кодування?
3. Недолік транспортного кодування.
4. Асиметрична криптографія її робота.
5. Умови асиметричної криптографії.

Транспортне кодування - в криптографії використовується додаткове кодування для сумісності з протоколами передачі даних. Мета кодування - виключити появу у вихідному потоці перших 32 символів ASCII набору та інших службових символів. Способи реалізації можуть бути різними. Часто використовується система Base64 (стандарт - RFC1251), розроблена для мережі Інтернет.

Симетричні криптосистеми, розглянуті нами в попередніх главах, незважаючи на множину переваг, мають одним серйозним недоліком, про який Ви, напевно, ще не замислювалися. Зв'язаний він із ситуацією, коли спілкування між собою роблять не три-чотири людини, а сотні й тисячі людей. У цьому випадку для кожної пари, що листується між собою, необхідно створювати свій секретний симетричний ключ. Це в підсумку приводить до існування в системі з N користувачів N2/2 ключів. А це вже дуже "пристойне" число. Крім того, при порушенні конфіденційності якої-небудь робочої станції зловмисник одержує доступ до всіх ключів цього користувача й може відправляти, нібито від його імені, повідомлення всім абонентам, з якими "жертва" вела переписку.

Своєрідним розв'язком цієї проблеми з'явилася поява асиметричної криптографії. Ця область криптографії дуже молода в порівнянні з іншими представниками. Перша схема, що мала прикладну значимість, була запропонована всього близько 20 років тому. Але за цей час асиметрична криптографія перетворилася в одне з основних напрямків криптології, і використовується в сучасному світі також часто, як і симетричні схеми.

Асиметрична криптографія споконвічно задумана як засіб передачі повідомлень від одного об'єкта до іншого (а не для конфіденційного зберігання інформації, яке забезпечують тільки симетричні алгоритми). Тому подальше пояснення ми будемо вести в термінах "відправник" – особа, яка шифрує, а потім відправляє інформацію з незахищеного каналу й "одержувач" – особа, що ухвалює, що й відновлює інформацію в її вихідному виді. Основна ідея асиметричних криптоалгоритмів полягає в тому, що для шифрування повідомлення використовується один ключ, а при дешифруванні – іншої.

Крім того, процедура шифрування обрана так, що вона необоротна навіть по відомому ключу шифрування – це друга необхідна умова асиметричної криптографії. Тобто, знаючи ключ шифрування й зашифрований текст, неможливо відновити вихідне повідомлення – прочитати його можна тільки за допомогою другого ключа – ключа дешифрування. А раз так, то ключ шифрування для відправлення листів якій-небудь особі можна взагалі не приховувати – знаючи його однаково неможливо прочитати зашифроване повідомлення. Тому, ключ шифрування називають в асиметричних системах "відкритим ключем", а от ключ дешифрування одержувачеві повідомлень необхідно тримати в секреті – він називається "закритим ключем". Напрошується питання : "Чому, знаючи відкритий ключ, не можна обчислити закритий ключ ?" – це третя необхідна умова асиметричної криптографії – алгоритми шифрування й дешифрування створюються так, щоб знаючи відкритий ключ, неможливо обчислити закритий ключ.

У цілому система переписки при використанні асиметричного шифрування виглядає в такий спосіб. Для кожного з N абонентів, ведучих переписку, обрана своя пара ключів : "відкритий" Ej і "закритий" Dj, де j – номер абонента. Усі відкриті ключі відомі всім користувачам мережі, кожний закритий ключ, навпаки, зберігається тільки в того абонента, якому він належить. Якщо абонент, скажемо під номером 7, збирається передати інформацію абонентові під номером 9, він шифрує дані ключем шифрування E9 і відправляє її абонентові 9. Незважаючи на те, що всі користувачі мережі знають ключ E9 і, можливо, мають доступ до каналу, по якім іде зашифроване послання, вони не можуть прочитати вихідний текст, тому що процедура шифрування необоротна по відкритому ключу. І тільки абонент №9, одержавши послання, робить над ним перетворення за допомогою відомого тільки йому ключа D9 і відновлює текст послання. Помітьте, що якщо повідомлення потрібно відправити в протилежному напрямку (від абонента 9 до абонента 7), те потрібно буде використовувати вже іншу пару ключів (для шифрування ключ E7, а для дешифрування – ключ D7).

Як ми бачимо, по-перше, в асиметричних системах кількість існуючих ключів пов'язане з кількістю абонентів лінійно (у системі з N користувачів використовуються 2*N ключів), а не квадратичне, як у симетричних системах. По-друге, при порушенні конфіденційності k-ой робочої станції зловмисник довідається тільки ключ Dk : це дозволяє йому читати всі повідомлення, що приходять абонентові k, але не дозволяє видавати себе за нього при відправленні листів. Крім цього, асиметричні криптосистеми мають ще декількома дуже цікавими можливостями, які ми розглянемо через кілька розділів.

Алгоритм RSA коштує в джерел асиметричної криптографії. Він був запропоновано трьома дослідниками-математиками Рональдом Рівестом (R.Rivest), Аді Шаміром (A.Shamir) і Леонардом Адльманом (L.Adleman) в 1977-78 роках.

Першим етапом будь-якого асиметричного алгоритму є створення пари ключів: відкритого й закритого й поширення відкритого ключа по усьому світу. Для алгоритму RSA етап створення ключів складається з наступних операцій :

• Вибираються два прості (!) числа p і q.

• Обчислюється їхній добуток n(=p*q).

• Вибирається довільне число e (e<n), таке, що НСД(e,(p-1)(q-1))=1, тобто e повинне бути взаємно простим із числом (p-1)(q-1).

• методом Евкліда вирішується в цілих числах (!) рівняння e*d+(p-1)(q-1)*y=1. Тут невідомими є змінні d і y – метод Евкліда саме й знаходить множину пар (d,y), кожна з яких є розв'язком рівняння в цілих числах.

• Два числа (e,n) – публікуються як відкритий ключ.

• Число d зберігається в найсуворішому секреті – це і є закритий ключ, який дозволить читати всі послання, зашифровані за допомогою пари чисел (e,n).

Проводиться властиво шифрування за допомогою цих чисел :

• відправник розбиває своє повідомлення на блоки, рівні k=[log2(n)] біт, де квадратні дужки позначають узяття цілої частини від дробового числа.

• подібний блок, може бути інтерпретований як число з діапазону (0;2k-1). Для кожного такого числа (наприклад mi) обчислюється вираження ci=((mi)e)mod n. Блоки ci і є зашифроване повідомлення Їх можна спокійно передавати по відкритому каналу, оскільки, операція піднесення в ступінь по модулю простого числа, є необоротною математичною задачею. Зворотна їй задача зветься "логарифмування в кінцевім полі" і є на кілька порядків більш складною задачею. Тобто навіть якщо зловмисник знає числа e і n, то по ci прочитати вихідні повідомлення mi він не може ніяк, крім як повним перебором mi.

А от на прийомній стороні процес дешифрування все-таки можливий, і допоможе у цьому збережене в секреті число d. Досить давно була доведена теорема Ейлера, окремий випадок якої затверджує, що якщо число n представимо у вигляді двох простих чисел p і q, те для будь-якого x має місце рівність (x(p-1)(q-1))mod n = 1. Для дешифрування Rsa-повідомлень скористаємося цією формулою. Зведемо обидві її частини в ступінь (-y) : (x(-y)(p-1)(q-1))mod n = 1(-y) = 1. Тепер помножимо обидві її частини на x : (x(-y)(p-1)(q-1)+1)mod n = 1*x = x.

А тепер згадаємо як ми створювали відкритий і закритий ключі. Ми підбирали за допомогою алгоритму Евкліда d таке, що e*d+(p-1)(q-1)*y=1, тобто e*d=(-y)(p-1)(q-1)+1. А отже в останньому вираженні попереднього абзацу ми можемо замінити показник ступеня на число (e*d). Одержуємо (xe*d)mod n = x. Тобто для того щоб прочитати повідомлення ci=((mi)e)mod n досить піднести його до степеня d по модулю m : ((ci)d)mod n = ((mi)e*d)mod n = mi.

Насправді операції піднесення в ступінь більших чисел досить трудомісткі для сучасних процесорів, навіть якщо вони проводяться по оптимізованих за часом алгоритмах. Тому звичайно весь текст повідомлення кодується звичайним блоковим шифром (набагато більш швидким), але з використанням ключа сеансу, а от сам ключ сеансу шифрують саме асиметричним алгоритмом за допомогою відкритого ключа одержувача й міститься в початок файлу.

Тема 6. Технології цифрових підписів.

План

1. Цифровий підпис.

Література: 1.с. 304..322

Питання для самоконтролю:

1. Що називається хеш-функцією?
2. Що називається електронним підписом?
3. Що таке асиметричне шифрування навпаки?

Як виявилось, теорія асиметричного шифрування дозволяє дуже красиво вирішувати ще одну проблему інформаційної безпеки – перевірку достовірності автора повідомлення. Для вирішення цієї проблеми за допомогою симетричної криптографії була розроблена дуже трудомістка і складна схема. В той же час за допомогою, наприклад, того ж алгоритму RSA створити алгоритм перевірки достовірності автора і незмінності повідомлення надзвичайно просто.

Припустимо, що нам потрібно передати який-небудь текст, не обов'язково секретний, але важливе те, щоб в нього при передачі по незахищеному каналу не були внесені зміни. До таких текстів зазвичай відносяться різні розпорядження, довідки, і тому подібна документація, що не представляє секрету. Обчислимо від нашого тексту яку-небудь хеш-функцію – це буде число, яке більш менш унікально характеризує даний текст.

В принципі, можна знайти інший текст, який дає те ж саме значення хеш-функції, але змінити в нашому тексті десять-двадцять байт так, щоб текст залишився повністю осмисленим, та ще і змінився у вигідну нам сторону (наприклад, зменшив суму до оплати в два рази) – надзвичайно складно. Саме для усунення цієї можливості хеш-функції створюють такими ж складними як і криптоалгоритми – якщо текст з таким же значенням хеш-функції можна буде підібрати тільки методом повного перебору, а безліч значень складатиме як і для блокових шифрів 232–2128 можливих варіантів, то для пошуку подібного тексту зловмисникові буде "потрібно" ті ж самі мільйони років.

Таким чином, якщо ми зможемо передати одержувачеві захищеним від зміни методом хеш-сумму від тексту, що пересилається, то у нього завжди буде можливість самостійно обчислити хеш-функцию від тексту вже на приймальній стороні і звірити її з присланою нами. Якщо хоч би один біт в обчисленій їм самостійно контрольній сумі тексту не співпаде з відповідним бітом в одержаному від нас хеш-значенні, значить, текст по ходу пересилки піддався несанкціонованій зміні.

Представимо тепер готову до передачі хеш-сумму у вигляді декількох к-бітних блоків hi. Обчислимо над кожним блоком значення $si=((hi)^d) \bmod n$, де d – це той самий закритий ключ відправника. Тепер повідомлення, що складається з блоків si можна "спокійно" передавати по мережі. Ніякої небезпеки по відомих hi і si знайти Ваш секретний ключ немає – це настільки ж складне завдання, як і завдання "логарифмування в кінцевому полі". А ось будь-який одержувач повідомлення може

легко прочитати початкове значення hi, виконавши операцію $((si)^{e})$ mod n = $((hi)^{d*e})$mod n = hi – Ваш відкритий ключ (e,n) є у всіх, а то, що піднесення будь-якого числа до ступеня (e*d) по модулю n дає початкове число, ми довели в минулій темі. При цьому ніхто інший, окрім Вас, не знаючи Вашого закритого ключа d не може, змінивши текст, а отже, і хеш-суму, обчислити такі s'i, щоб при їх піднесенні до ступеня e вийшла хеш-сума h'i, співпадаюча з хеш-сумою фальсифікованого тексту.

Таким чином, маніпуляції з хеш-сумою тексту вдають із себе "асиметричне шифрування навпаки": при відправці використовується закритий ключ відправника, а для перевірки повідомлення – відкритий ключ відправника. Подібна технологія одержала назву "Електронний підпис". Інформацією, яка унікально ідентифікує відправника (його віртуальним підписом), є закритий ключ d. Жодна людина, що не володіє цією інформацією, не може створити таку пару (текст,si), що описаний вище алгоритм перевірки дав би позитивний результат.

Подібний обмін місцями відкритого і закритого ключів для створення з процедури асиметричного шифрування алгоритму електронного підпису можливий тільки в тих системах, де виконується властивість комутативності ключів. Для інших асиметричних систем алгоритм електронного підпису або значно відрізняється від базового, або взагалі не реалізовується.

Перелік
навчально-методичної літератури

№ п/п	Назва, автори, рік видання	Кількість примірників
	ОСНОВНА	
1.	Мельников В. П. Информационная безопасность и защита информации: учебное пособие для студ. высш. учеб. заведений/В. П. Мельников, С. А. Клейменов, А. М. Петраков; под. ред. С. А. Клейменова. — 3-е изд., стер. — М.: Издательский центр «Академия», 2008. — 336 с.	
2.	Глушаков С.В., Хачиров Т.С., Соболев Р.О. Секреты хакера. Защита и атака/Худож. оформитель А.С. Юхтман Харьков: Фолио, 2006. 414 с.	
3.	Закер К. Компьютерные сети. Модернизация и поиск неисправностей. «БХВ - Петербург». Санкт – Петербург, 2005, 988 с.	
4.	«Информационная безопасность офиса». Научно – практический сборник. Выпуск первый «Технические средства защиты информации». К.: ООО «ТИД ДС», 2003. – 216 с.	
5.	Левин Максим. Библия хакера 3. «Майор», 2006, 576 с.	
6.	Комер Д. Принципы функционирования Интернета. Учебный курс. «Питер», 2002, 384 с.	
7.	Гришина Н.В. Организация комплексной системы защиты информации. М.: Гелиос АРВ, 2007. — 256 с., ил.	
	ДОПОМІЖНА	
1.	Панасенко Сергей. Алгоритмы шифрования. Специальный справочник. БХВ-Петербург, 2009, 578 с.	
2.	Шаньгин В. Ф. Информационная безопасность компьютерных систем и сетей. Инфра-М, 2008, 416 с	