МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ ДЕРЖАВНИЙ ПОЛІТЕХНІЧНИЙ КОЛЕДЖ

Для спеціальності:

123 Комп’ютерна інженерія

КОНСПЕКТ ЛЕКЦІЙ

з дисциплін: “Надійність, діагностика та експлуатація комп'ютерних систем та мереж”

Харків 2019

Конспект лекцій з дисципліни **“Надійність, діагностика та експлуатація комп'ютерних систем та мереж”** для студентів освітньо-професійної програми «Обслуговування комп′ютерних систем і мереж» спеціальності 123 «Комп’ютерна інженерія»

Затверджено на засіданні циклової комісії інформаційних технологій
Протокол від _____2019 р. №____

Голова циклової комісії _____М.М. Бочарніков
“____” _____ 2019 р.

Схвалено методичною радою коледжу
Протокол від _____2019 р. №____

Голова методичної ради _____
“____” _____ 2019 р.
року

ЗМІСТ

Лекція 1.

Тема: Поняття надійності ОП та С та шляхи її забезпечення.

Взагалі, надійність ЕОМ і обчислювальних систем (ОС) визначається надійністю її елементів і апаратури, а також надійністю програмного забезпечення. Для досягнення правильних результатів обчислювань протягом заданого часу необхідно, щоб всі складові обчислювального процесу – елементи апаратури та програмне забезпечення мали необхідну надійність.

В ході розвитку елементної бази та технології виготовлення обчислювальної техніки надійність елементів і апаратури значно зросла. Так, ***середній час безвідмовної роботи*** елементної бази першого покоління (електронні лампи) складали порядку 10^3 – 10^4 год.; третього покоління: 10^6 – 10^8 год.; четвертого, п'ятого: 10^8 год. і вище.

Зростанню надійності обчислювальних пристроїв та систем (ОП та С) сприяло як підвищення інтеграції елементів, так і зменшення кількості потенціально ненадійних контактів, а також високій ступінь автоматизації процесу виробництва ВІС та НВІС.

Надійність апаратури збільшилась також за рахунок інших причин, зокрема, більш старанного екранування апаратури, розв'язування полів, фільтрації, забезпечення необхідних теплових режимів, ізоляції від джерел механічної вібрації та ударів. Значно удосконалились периферійні пристрої.Іншим напрямом для підвищення надійності ОС є ***резервування***. Це засіб підвищення надійності за допомогою резервної апаратури, готової в будь- який момент замінити апаратуру, яка відмовила. Але й багатократно резервована апаратура не зберігає свою роботоздатність на необмежений час без прийняття заходів до її ***відновлення***. ***Відновлення*** – це приведення апаратури в роботоздатний стан шляхом заміни окремих елементів, які відмовили.

Поняття надійності ОП та С та шляхи її забезпечення

Забезпечення надійності ОП та С здійснюється в двох напрямах.

В першому випадку вона визначається відсутністю відмов, збоїв, помилок та несправностей. В другому – можливістю швидкого відновлення апаратури та обчислювального процесу.

Надійність можна визначити як властивість об'єкта зберігати в часі у визначених межах значення всіх параметрів, що характеризують спроможність виконувати

потрібні функції в заданих режимах і умовах застосування, технічного обслуговування, ремонтів, зберігання та транспортування.

Надійність сама по собі – складна властивість, яка в залежності від призначення об'єкта та умов його експлуатації складається із сполучень властивостей: безвідмовності, довговічності, ремонтопридатності та збережності.

Безвідмовність – це властивість об'єкта безперервно зберігати роботоздатний стан в проміжку деякого часу або деякого напрацювання.

Напрацювання – об'єм (час) роботи об'єкта. Може визначатися в іншій формі, наприклад, кількості вирішених задач або циклів роботи.

Відмова – подія, яка являє собою порушення роботоздатності об'єкта. Вона в основному викликається фізичним руйнуванням елементів або поступовим погіршенням їх характеристик.

Збій – короткочасне порушення правильної роботи обчислювального пристрою або його елемента, після якого його роботоздатність самовідновлюється або відновлюється оператором без проведення ремонту. Збої можуть бути викликані внутрішніми або зовнішніми перешкодами.

Основні якості обчислювальної техніки, пов'язані з її надійністю, описуються розподіленням відмов в часі, процесами відновлення та організацією обслуговування. Покращання надійності потребує додаткових витрат на розробку, виготовлення та експлуатацію систем.

При встановленні вимог до надійності обчислювальної техніки необхідно зважати на витрати, пов'язані з її розробкою та виготовленням; витрати на персонал, зайнятий ремонтом і обслуговуванням, витрати, які визначаються наслідками відмов та простоїв, зниженням ефективності або продуктивності системи, а також витрати, пов'язані із збільшенням маси або габаритів системи та ін.

Якщо відмова ОП та С може спричинити собою небезпеку для життя людей або аварію, то рівень надійності визначається з такої вимоги, щоб ймовірність відмови відповідала практично неможливій події.

Тема: Основні методи забезпечення надійності на етапах життєвого циклу обчислювальних систем.

Етап складання технічного завдання (ТЗ). Збирають всі дані, які є, про аналогічні та близькі системи; дані про умови застосування ЕОМ або ОС і вимоги, що висуваються до функцій, які виконуються розглянутою системою. За сукупністю цих даних і вимог розробляються основні вимоги до надійності даної системи.

Етап ескізного проектування. На цьому етапі обирається елементна база і визначаються особливості структури, архітектури та організації системи, яка розробляється. За цими даними проводиться попередній розрахунок надійності, виявляються найменш надійні підсистеми, і на цій основі приймається рішення про резервування системи, а також рішення про засоби та організацію технічного обслуговування, тобто профілактичні та ремонтні роботи.

Досліджується питання про доцільність резервування і методи автоматичного відновлення та підвищення відмовостійкості системи.

Етап технічного і робочого проектування. Перевіряються та уточнюються раніше прийняті рішення. Для цього використовують уточнені дані про надійність, отримані на основі розрахунків, зважаючи на режими роботи і точну номенклатуру елементів системи, а також результати експериментів над моделями, макетами, дослідними та промисловими зразками.

Розробляється програмне забезпечення системи, проводиться його перевірка та діагностування за тестами і шляхом імітаційного моделювання на моделі системи, яка проектується.

З метою забезпечення надійності здійснюють виявлення та виправлення всіх помилок в документації, яка розробляється.

Етап виробництва. Основним є технічний контроль, який охоплює всі стадії виробничого процесу, починаючи від вхідного контролю якості матеріалів, які надходять, і комплектуючих виробів, включаючи контроль якості та відповідність технічній документації виготовлених друкованих плат, блоків, пристроїв, схемних з'єднань, конструкції, і закінчуючи випробуваннями готової продукції. Виявляються недоліки в розробці, які впливають на надійність системи, та приймаються заходи з метою їх усунення.

Етап експлуатації. На цьому етапі здійснюється контроль та забезпечення умов навколишнього середовища, які передбачаються проектом, забезпечення достатньої кваліфікації та необхідного складу обслуговуючого персоналу, організація та проведення техобслуговування і ремонтів. Продовжується збирання інформації про відмови апаратури і програмного забезпечення, які передаються розробникам з метою усунення причин відмов.

Лекція 2.

Тема: Надійність елементів ЕОМ.

Самі по собі моделі елементів ЕОМ, які б відображали дуже складні фізико-хімічні процеси виникнення відмови, є занадто складними для практичного застосування.Однією із спрощених моделей є ***метод коефіцієнтів***. Він полягає в тому, що параметр $\boldsymbol{\alpha_i}$ обраної статистичної моделі надійності можна записати як

$$\boldsymbol{\alpha_i = \alpha_{i_1}, \quad \alpha_{i_2} \ldots . \alpha_{i_n} \alpha_{i_0}}$$

де $\alpha_{i1}, \alpha_{i2} \ldots \alpha_{in}$ – емпіричні коефіцієнти, кожний з яких виражає вплив одного з зовнішніх діючих факторів (температури, вологості, радіації та ін.);

$\boldsymbol{\alpha_{i0}}$ – базове значення параметра $\boldsymbol{\alpha_i}$.

Якщо значення всіх діючих факторів невідомі, то використовують коефіцієнти, які залежать від загального характеру умов застосування апаратури. Наприклад, для інтенсивності відмов при експоненціальній моделі надійності існує наступна залежність: $\boldsymbol{\lambda = k\lambda_0}$, де **k** – коефіцієнт, залежний від умов роботи; $\boldsymbol{\lambda 0}$ – деяка базова інтенсивність відмов. Орієнтаційні значення коефіцієнта **k** для лабораторних умов – 0,5; польових умов – 1,5; апаратури, встановленої на автомобілі, – 2,0; борту корабля – 2,5; борту літака – 3,0.

Для наближеної оцінки інтенсивності відмов ВІС, побудованих за біполярною технологією та КМОН

$$\boldsymbol{\lambda}_{\text{ВІС}} = (\frac{\boldsymbol{n}}{\boldsymbol{2100}})\lambda$$

де **n** – число логічних елементів у ВІС; $\boldsymbol{\lambda}$ - значення інтенсивності відмов в залежності від температури корпуса.

Для більш точної оцінки інтенсивності відмов ВІС необхідно користуватися

довідниками, в яких дається залежність інтенсивності їх відмов від ряду факторів, в тому числі, від особливостей конструкції, ступеня освоєння продукції та ін.

Приведемо деякі дані про інтенсивність відмов компонентів обчислювальних систем:

Таблиця 1

Компонент	Інтенсивність відмов
ІС малої степені інтеграції	0,1
діод	0,2 – 0,5
транзистор	0,05 – 0,3
конденсатор	0,002 – 0,04
резистор	0,01 – 0,1
трансформатор	0,1 – 0,2
пайка	0,0001
з'єднання	2,0 - 3,5
вимикач	0,2 – 0,5
вентилятор	3,0
пам'ять на магнітних дисках	250
електромеханічний друкуючий пристрій	420

Приведені цифри необхідно помножити на 10^{-6}, щоб отримати значення інтенсивності відмов у "відмовах за годину".

Наближений метод розрахунку надійності ОП та С

Метод базується на таких припущеннях:

1) час відновлення набагато менше часу безвідмовної роботи;

2) інтенсивності відмов та інтенсивності відновлювань – постійні величини;

3) відмови і відновлення окремих пристроїв ОС є незалежними випадковими подіями.

Для послідовного включення пристроїв системи використовують наступні розрахункові формули:

$$\boldsymbol{\lambda} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{\lambda_i}$$

$$\boldsymbol{K_{\text{ч}} = 1 - n = \sum_{i=1}^{n} K_{\text{ч}i}}$$

$$\boldsymbol{\mu} = \frac{\boldsymbol{\lambda}}{\mathbf{1} - \text{К}_{\text{ч}}}$$

Для паралельного включення:

$$\boldsymbol{\lambda} = \boldsymbol{\mu}(\mathbf{1} - \text{К}_{\text{ч}})$$

$$\boldsymbol{K_{\text{ч}} = 1 - n = \prod_{i=1}^{m}(1 - K_{\text{ч}i}\ )} \quad (3)$$

$$\boldsymbol{\mu = \sum_{i=1}^{m} \mu_i} \quad (4)$$

де **λ** - інтенсивність відмов послідовної (паралельної) групи з **n (m)** підсистем (пристроїв); **Кr** – коефіцієнт готовності послідовної групи з **n (m)** підсистем;

μ- інтенсивність відновлювань послідовної (паралельної) групи з **n (m)** підсистем. Змінні з індексами **i** позначають відповідні показники окремих підсистем.

Якщо в системі застосовується змінне резервування підсистем, то замість формул (9) застосовується формула:

$$\boldsymbol{K_{\text{ч}} = \sum_{i=\text{ч}}^{m} C_m^i\ K_{\text{ч.п.}}^i\ (1 - K_{\text{ч.п.}})^{m-i}}$$

де **ч** - мінімально необхідне, за вимогами продуктивності, число роботоздатних підсистем; $\mathbf{К_{ч.п.}}$ – коефіцієнт готовності підсистеми; $\mathbf{C^i_m}$ –число комбінацій від **m** до **i**.

Інтенсивність відновлення у випадку змінного резервування визначається не за формулою (10), а за формулою:

$$\boldsymbol{\mu = (m - \text{ч} + 1)\mu_n}$$

де $\boldsymbol{\mu_n}$ – інтенсивність відновлення підсистеми.

Інтенсивність відмов l за числом дорівнює параметру потоку відмов **w**.(Табл. 1).

Тема: Надійність програмного забезпечення.

Умовою успішного дослідження чи розробки обчислювальної техніки є системний підхід до задачі, котра вирішується. Це також стосується і програмного забезпечення (ПЗ).

Програмне забезпечення складається з багатьох модулів. Під модулем розуміють елемент (компонент) програми, котрий стандартизований за формою запису і зовнішніми зв'язками. Йому притаманна, як правило, автономність виготовлення. Кожний модуль призначений для вирішення певної задачі, але вирішення всіх задач в межах конкретного ПЗ напрямлене на досягнення єдиної мети. Виходячи з цього, всі модулі ПЗ повинні бути взаємопов'язаними. Взагалі, ПЗ має складну ієрархічну будову, і для його розроблення використовується системний підхід.

Під **системним підходом**, наприклад, до проектування автоматизованих систем, розуміють сумісний аналіз і розроблення всіх елементів системи [1]. Мається на увазі сумісний аналіз і розроблення керуючого і керованого об'єктів, давачів інформації, ліній зв'язку та ін. До складу керуючих і керованих об'єктів можуть входити як апаратні засоби, так і керуючі програми ПЗ. Тому надійність роботи апаратних засобів слід розглядати в комплексі з надійністю програмного забезпечення.

При розробленні керуючих програм аналізують сукупність можливих вихідних даних, можливих і допустимих кінцевих і проміжних результатів та ін.

Надійність програмного забезпечення визначається як властивість програми, котра виражається у виконанні заданих функцій у заданих умовах роботи і на заданій ЕОМ, аналогічно тому, як визначається поняття надійності апаратури (властивість об'єкта зберігати в часі у встановлених межах значення всіх параметрів, які характеризують здатність виконувати потрібні функції у заданих режимах та умовах застосування, технічного обслуговування, зберігання та транспортування – ДСТУ 2668-94).

При імовірнісному підході під **надійністю ПЗ розуміють** ймовірність того, що при функціюванні системи на протязі певного часу не буде виявлено жодної помилки [2]. Але це неточне визначення. В ньому не враховано різницю між помилками різних типів. А помилки бувають далеко не рівнозначними за своїми наслідками.

Краще розглядати надійність ПЗ з точки зору дії помилок на користувача системи.

Надійність ПЗ не обов'язково прямо пов'язана з внутрішньою оцінкою ПЗ. Все залежить від того, на якому етапі (розроблення, експлуатація) і в якому компоненті ПЗ допущена помилка. Інколи помилка, допущена на етапі проектування, приводить до незначних і непомітних наслідків. А помилка, на перший погляд незначна, на етапі експлуатації може привести до катастрофічних наслідків і навпаки. Прикладом цього є випадок, коли запуск космічного корабля на Венеру був невдалим за причиною того, що в операторі DO програми, написаної на Фортрані, була пропущена кома [2].

Надійність ПЗ не варто розглядати як внутрішню властивість програми. Вона дуже пов'язана з тим, як використовується програма. А слово "ймовірність" підкреслює те, що користувач з певною ймовірністю не введе в ПЗ певний набір даних, котрий виведе систему з ладу. З цієї точки зору надійністю програмного забезпечення є ймовірність його роботи без помилок і відмов на протязі певного періоду часу.

Взагалі, надійність є комплексною властивістю, яка в залежності від призначення об'єкта і умов його застосування може відображати такі властивості, як **безвідмовність, довговічність, ремонтопридатність** та **збережуваність** або певні поєднання цих властивостей.

Відмова (ДСТУ 2668-94) – подія, яка полягає у втраті функційним модулем (системою) здатності виконувати потрібну функцію.

Тема: Безвідмовність програмного забезпечення.

Безвідмовність програмного забезпечення оцінюють ймовірністю його роботи без відмов на протязі заданого часу в певних умовах зовнішнього середовища.

Безвідмовність програмного забезпечення з точки зору надійності принципово відрізняється від безвідмовності апаратних засобів обчислювальної техніки. Програми з часом не "зношуються". Характеристики функціювання ПЗ залежать тільки від його якості, яка зумовлена процесом розробки. Не існує поняття виходу з ладу програми. Безвідмовність ПЗ визначається його правильністю (коректністю) і цілком залежить від наявності помилок в ньому, котрі були внесені на етапі проектування. А безвідмовність апаратури визначається в основному випадковими відмовами, які залежать від зміни параметрів апаратури, що сталися на етапі експлуатації.

На функціювання ПЗ в значній мірі впливають ті вхідні дані, які обробляються

обчислювальним пристроєм. З часом роботи програми можуть зустрітися такі дані, котрі програма не може коректно обробити. Це і є причиною прояву помилок програмного забезпечення.

Відмова при функціюванні ПЗ проявляється як наслідок порушення кодів запису програми в пам'яті, витирання чи перекручування даних в оперативній чи довготривалій пам'яті комп'ютера, порушення нормального проходження обчислювального процесу. Відмова може проявитися також у вигляді програмного зупинення, зациклення, систематичного пропускання виконання деякої групи команд, спотворення накопичування даних та ін. Програмні відмови приводять до укорочування виданих абонентам інформації чи керуючих дій. Спотворення, які є результатом відмов, в кінцевому випадку можуть призвести до порушення роботоздатності ПЗ.

Взагалі, першопричиною порушення роботоздатності програмного забезпечення, і конкретних програм зокрема, є конфлікт між реальними вхідними даними, що мають оброблятися, і програмою, що здійснює це оброблення. До вхідних даних відносяться як дані, що вперше надійшли, так і дані, котрі були накопичені за попередній період функціювання ПЗ.

Під **збоєм** в теорії надійності [3] розуміють відмову, що самоусувається. Її усунення не потребує зовнішнього втручання для заміни компонентів, що відмовили. Щодо апаратури, то поняття збою і відмови відрізняються між собою ступенем фізичного руйнування елементів і компонентів і відповідно необхідністю їх заміни. Особливістю при цьому є те, що взагалі неможливо визначити чіткий часовий поріг розподілу збоїв і відмов.

В процесі відпрацювання програми або комплексу програм фізичне руйнування апаратури обчислювального пристрою чи комп'ютера в цілому відсутнє, а необхідність заміни компонентів чи ремонту відпадає.

Важливою характеристикою надійності ПЗ є **відновлюваність**. Вона характеризується витратами часу і праці на усунення відмов за причиною прояву помилок в програмі і його наслідків. При цьому надійна програма повинна забезпечувати низьку ймовірність відмов при її відпрацюванні. Високу надійність програм забезпечує швидке реагування на її спотворення, а також на спотворення даних чи обчислювального процесу,

і відновлення роботоздатності програм за час, що менший, ніж поріг між збоєм і відмовою. Слід зауважити, що при цьому некоректна програма може функціювати абсолютно надійно, тобто невірні результати не приводять до відмов. А формально вірна (коректна) програма може бути ненадійною, в першу чергу, за причини наявності системної помилки при заданій області зміни вихідних даних.

Взагалі, **відновлення** після відмови в програмі може полягати в коректуванні і відновленні тексту програми, виправленні даних, внесенні змін в організацію обчислювального процесу та ін. Відновлюваність ПЗ оцінюється середньою тривалістю усунення помилки (помилок) в програмі і відновлення її роботоздатності.

Відновлюваність ПЗ залежить від багатьох факторів, зокрема, від складності структури комплексу програм, структурованості програм, алгоритмічної мови, на котрій розроблялася програма, якості документації на програму та ін.

У зв'язку з тим, що усунення відмов у програмах не потребує "ремонту", то виникає можливість автоматизації відновлення ПЗ. При цьому головною задачею є відновлення за час, що не перевищує граничного значення часу між збоєм і відмовою. При такому підході до автоматизації відновлення можна перетворити відмови в збої і таким чином покращити показники надійності ПЗ. Для цього в комплексі програм потрібно мати засоби, котрі дозволяють [4]:

1) здійснювати систематичний контроль і оперативно виявляти аномалії стану програм і даних та процесу функціювання ПЗ;

2) діагностувати виявлені спотворення;

3) обирати методи і засоби оперативного відновлення ПЗ;

4) реалізовувати оперативне відновлення роботоздатності програмного комплексу;

5) реєструвати кожний збій, що відбувся, а також кожну відмову з метою встановлення їх систематичності, що повинно бути мотивом для удосконалення і доробки ПЗ чи апаратних засобів.

Інколи ведуть мову про **усталеність функціювання ПЗ** [5]. Під нею розуміють здатність обмежувати наслідки власних помилок і несприятливих впливів зовнішнього середовища (некоректність вхідних даних, помилки оператора та ін.) чи протистояти

їм.

Механізм виникнення відмов програмного забезпечення істотно відрізняється від механізму виникнення відмов апаратури. Відмова апаратури зумовлена руйнуванням певних елементів та компонентів апаратури, а відмова програмного забезпечення – невідповідністю програмного забезпечення поставленим задачам.

Невідповідність може виникнути за двох причин: а) порушення **специфікації** – технічних вимог до програми; б) неточної чи неповної специфікації [6].

Невідповідність за першою причиною зустрічається в першу чергу в складних програмних системах, де окремі помилки програміста важко спостережувані і можуть залишатися незнайденими.

Невідповідність за другою причиною виникає тому, що при складанні специфікації численні фактори, що впливають на роботу програми, невідомі. Вони виясняються поступово, в ході експлуатації програми. Це відноситься особливо до керуючих програм.

Програма, крім того, є коротким записом дуже складних функцій. Тому записати в специфікаціях всі властивості функцій, котрі повинна виконувати програма, не легше, ніж розробити відповідну програму.

Програми **за ступенем точності специфікації** класифікують наступним чином:

1) програми, функції котрих повністю визначаються специфікацією;

2) програми, функції котрих коректуються зіставленням обчислених і виміряних результатів (моделюючі програми, котрі реалізують математичну модель фізичного об’єкта);

3) програми, які діють в постійно змінюваному середовищі (котрі складаються з інших програм, даних, реальних систем, операційні системи, програми керування ресурсами та ін.)

Існують ще такі поняття, як коректність програми, прихованість помилок.

Коректність програми – це її відповідність специфікації. Оскільки специфікація може не відповідати фактичним вимогам до програми, то можливі випадки, коли некоректна програма працює надійно, чи навпаки – коректна програма працює ненадійно.

Відмови програм може зумовлювати **прихованість помилок.** Це означає, що прихована помилка проявляється тільки в рідкісних комбінаціях з великої кількості комбінацій вихідних даних і тому виявляється не відразу, а тільки в процесі тривалої експлуатації. На відміну від таких помилок, помилки, котрі проявляються при будь-яких вихідних даних, не так небезпечні, оскільки виявляються відразу при перших пробних прогонах програми.

Надійність програмного забезпечення $\mathbf{H_{ПЗ}}$ в певній мірі залежить від числа помилок, котрі внесені і не усунуті в процесі розробки. В процесі експлуатації помилки виявляються і усуваються. Якщо при виправленні помилок не вносяться нові, чи, у всякому випадку, нових помилок вноситься менше, ніж усувається, то надійність ПЗ безперервно зростає. Чим інтенсивніше проводиться експлуатація, тим інтенсивніше виявляються помилки і швидше росте надійність ПЗ. Ця закономірність широко підтверджується на практиці. В цьому корінна відмінність надійності ПЗ від надійності апаратних засобів **НАЗ**. Характер цих залежностей показаний на рис.1.

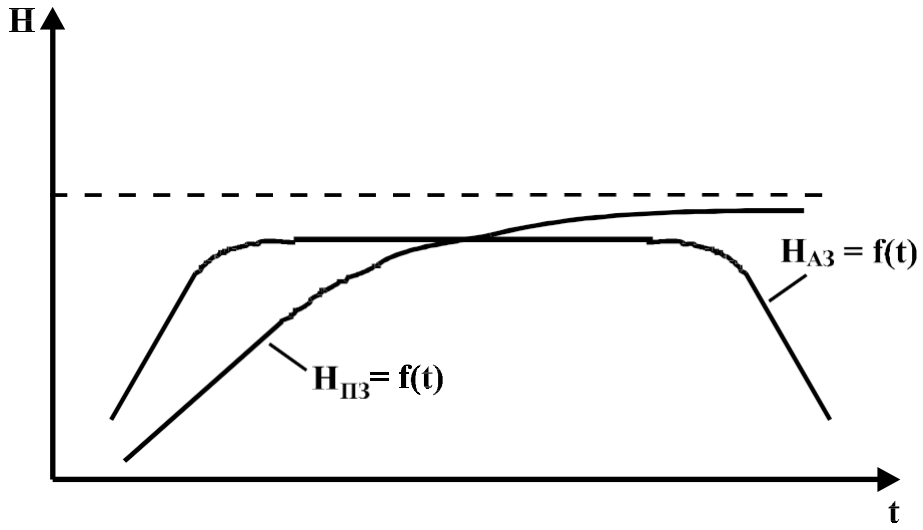

Рис.1. Графіки залежності надійності апаратних засобів $\boldsymbol{H_{АЗ}}$ і програмних засобів $\boldsymbol{H_{ПЗ}}$ від часу ***t***

Зміна частоти відмов типового обчислювального пристрою показана на рис.2.

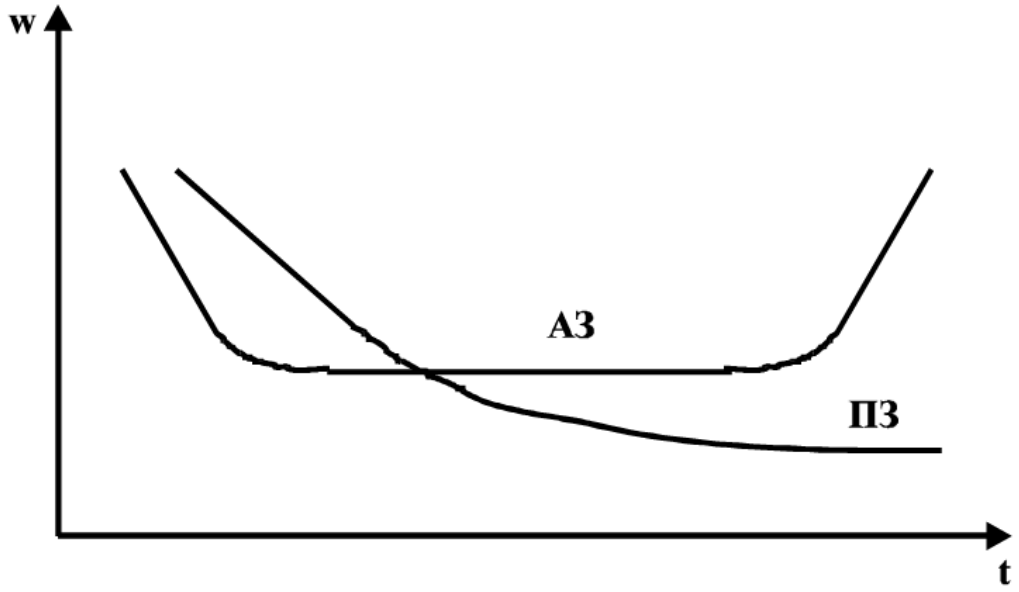

Рис.2. Різниця між надійністю апаратних засобів (АЗ) і програмного забезпечення (ПЗ)

Лекція 3.

Тема: Надійність програмних комплексів.

Надійність програмних комплексів визначається як функція надійності їх складових частин. Такий підхід значно полегшується, якщо програмний комплекс побудований за модульним принципом. Для прогнозу надійності програмного комплексу можна використати марківську модель [5, 10]. Мірою надійності програмного модуля вважають ймовірність того, що модуль коректно виконує задану функцію та коректно передає керування іншому модулю, передбаченому алгоритмом роботи програмного комплексу. При цьому вважають, що надійності модулів є незалежними величинами. Щодо коректності, то загальний результат роботи програми буде некоректним, якщо хоча би в одному з виконаних прогонів і модулів була помилка.

Надійність кожного з модулів вважається визначеною при визначенні надійності програмного комплексу. Тобто, по суті визначається надійність керування модулями. Процес керування модулями програми представимо як марківський процес. Це означає, що вибір наступного модуля для виконання залежить тільки від того модуля, який виконується в даний момент часу і не залежить від передісторії. Ймовірності передачі керування між модулями є величини сталі і повністю характеризують спосіб використання

програми користувачем.

Представимо структуру керування програмами у вигляді напрямленого графу, де кожна вершина N_i $(i=\overline{1,n})$ відповідає певному програмному модулю, а дуга (N_i, N_j) - можливому передаванню керування від модуля N_i до модуля N_j. Співвіднесемо кожній дузі (N_i,N_j) величину ймовірності переходу з вершини N_i по дузі (N_i,N_j), а кожній вершині N_i – ймовірність безвідмовної роботи R_i відповідного модуля. Граф програми має єдину вхідну вершину N_1, вихідну вершину N_n і дві додаткові вершини, котрі відповідають коректному **C** та помилковому **F** вихідним результатам.

Вважатимемо, що при появі помилки по дузі (N_i,F) і з ймовірністю $1-R_i$ незалежно від правильності наступної обробки здійснюється перехід в стан **F**, а при видаванні правильного результату модулем N_i здійснюється перехід до виконання модуля N_j з ймовірністю $R_i P_{ij}$.

Коректному завершенню програми відповідатиме перехід з вихідного стану N_n в стан **C** з ймовірністю R_n.

Тоді матриця перехідних ймовірностей **P**, де її елементи відповідатимуть переходові марківського процесу з стану **i** в стан **j**, матиме такий вигляд:

$$P = \begin{array}{c|cccccccc} & N_1 & N_2 & \dots & N_j & \dots & N_n & C & F \\ \hline N_1 & 0 & R_1P_{12} & \dots & R_1P_{1j} & \dots & R_1P_{1n} & 0 & 1-R_1 \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_i & 0 & R_iP_{i2} & \dots & R_iP_{ij} & \dots & R_iP_{in} & 0 & 1-R_i \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_{n-1} & 0 & R_{n-1}P_{(n-1)2} & \dots & R_{n-1}P_{(n-1)j} & \dots & R_{n-1}P_{(n-1)n} & 0 & 1-R_{n-1} \\ N_n & 0 & 0 & \dots & 0 & \dots & 0 & R_n & 1-R_n \\ C & 0 & 0 & \dots & 0 & \dots & 0 & 1 & 0 \\ F & 0 & 0 & \dots & 0 & \dots & 0 & 0 & 1 \end{array}$$

Надійність усього програмного комплексу в такому разі вираховується як ймовірність досягнення кінцевого стану **C** з початкового стану графу.

Розглянемо матрицю **Q**, котра одержується шляхом викреслювання стовпців і рядків, що відповідають кінцевим станам **C** і **F** матриці **P**. Матриця **Q** матиме наступний вигляд:

$$Q = \begin{array}{c|cccccc} & N_1 & N_2 & \dots & N_j & \dots & N_n \\ N_1 & 0 & R_1P_{12} & \dots & R_1P_{1j} & \dots & R_1P_{1n} \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_i & 0 & R_iP_{i2} & \dots & R_iP_{ij} & \dots & R_iP_{in} \\ \vdots & \dots & \dots & \dots & \dots & \dots & \dots \\ N_{n-1} & 0 & R_{n-1}P_{(n-1)2} & \dots & R_{n-1}P_{(n-1)j} & \dots & R_{n-1}P_{(n-1)n} \\ N_n & 0 & 0 & \dots & 0 & \dots & 0 \end{array}$$

Визначимо P^k для кожного цілого $k > 0$ як k -у степінь $P \cdot P^k (i, j)$ – це ймовірність того, що марківський процес за k кроків перейде зі стану i встан j. Тоді матриця $T = 1 + P + P^2 + \cdots = \sum_{k=0}^{\infty} P^k$ визначає ймовірність переходу з одного стану в інший за довільну кількість кроків.

Якщо S - квадратична матриця розмірності n, то

$$S = 1 + Q + Q^2 \dots = \sum_{k=0}^{\infty} Q^k$$

Якщо матриця $W = I - Q$, згідно теорії ймовірностей [11], матимемо

$$S = W^{-1} = (1 - Q)^{-1}$$

звідки надійність програмного комплексу

$$R = S\overline{(1, n)}\, R_n$$

Тема: Помилки ПЗ.

В загальному понятті під **помилкою** розуміють неправильність, похибку або ненавмисне спотворення об'єкта чи процесу. При цьому приймають, що правильний стан об'єкта чи процесу відомий. Поняття **помилки в програмі** в літературних джерелах трактується по різному. Але спільним в цих трактуваннях є те, що помилку не можна вважати тільки як відхилення від формалізованого еталону. Не завжди повна визначеність і коректність еталону викликає необхідність включати до складу помилок ПЗ ситуації, коли програми відповідають формалізованим еталонам, але порушуються окремі правила здорового глузду, що не передбачені еталонами. Тоді вважається, що в програмі наявна помилка, якщо вона не виконує того, що користувач від неї розумно очікує. Це помилка як ПЗ, так і неформалізованих сподівань його користувачів.

Відсутність повністю визначеної правильної програми – еталону не дає можливості локалізації помилки шляхом безпосереднього порівняння програми, що діагностується, з програмою без помилок.

Спотворення в тексті програм називають **первинними** помилками. Але помилка, як правило, виявляється за її вторинними проявами. Цим проявом є спотворення вихідних результатів при виконанні програми, а відповідну помилку називають **вторинною**. Її локалізація і усунення вимагають виконання ряду певних операцій.

Крім поняття помилки в програмі існує ще інше – **програмна помилка** [12]. Загального і всеохоплюючого поняття програмної помилки не існує. Їх є кілька. Одне з них стверджує, що програмна помилка виникає тоді, коли програма працює не у відповідності зі своїми специфікаціями. Його недолік в тому, що припускається абсолютна коректність специфікацій. Але це не завжди так. Програма може працювати у відповідності із специфікаціями, але не можна стверджувати, що в ній немає помилок.

Інше визначення стверджує, що помилка в програмі виникає тоді, коли програма працює не у відповідності із специфікаціями при умові, що вона експлуатується у наперед визначених границях. Це визначення є не досить точним, тому що, коли система навіть використовується випадково поза наперед визначених границь, то вона повинна видавати певні розумні результати. У протилежному випадку в самій системі наявна помилка.

Можливе також і наступне визначення. Помилка має місце тоді, коли робота програми не відповідає відповідній їй документації. Але й це визначення має недоліки. Може статися так, що програма працює у відповідності з документацією, але помилка має місце, так як і в документації, і в програмі наявні помилки.

З точки зору замовника могло б підійти таке визначення: помилка – це невідповідність роботи програми початковим вимогам замовника- користувача. Недоліком цього визначення є те, що замовник не завжди може достатньо детально викласти вимоги для опису бажаної поведінки програми при всіх можливих умовах.

Для вирішення згаданих проблем найбільше підходить таке означення: програмна помилка наявна тоді, коли програма працює не так, як передбачав користувач. Це визначення повністю задовольняє користувача, але розробника ПЗ може

задовольнити тільки тоді, коли він точно знає, чого сподівається користувач від системи. Користувач ПЗ є суб'єктом, йому властиві помилки. Задача стає ще складнішою, коли користувачами системи є декілька людей, так як, наприклад, кожен з них може вводити дані в системи іншим чином. Тому останнє визначення помилки добре спрацьовує при виключенні різноманітних людських факторів виникнення помилок в ПЗ.

Основними причинами відмов ПЗ, що приводять до порушення нормального функціювання, є:

а) помилки, приховані в самій програмі;

б) спотворення вхідної інформації, що підлягає обробці;

в) неправильні дії користувача;

г) несправності апаратних засобів, на котрих реалізується обчислювальний процес.

Один з варіантів типів помилок ПЗ та їх відсоток від загальної кількості помилок [6] приведено в таблиці 1.

Таблиця 1.

Причини помилки	**Частота**
Неповне чи помилкове завдання	28
Відхилення від завдання	12
Нехтування правилами програмування	10
Помилкова вибірка даних	10
Помилкова логіка чи послідовність операцій	12
Помилкові арифметичні операції	9
Нестача часу для розв'язку	4
Неправильне оброблення переривань	4
Неправильні сталі чи вихідні дані	3
Неточний запис	8

Взагалі, характеристики помилок ПЗ використовуються для вибору найбільш ефективних методів технології програмування та для супроводжень програм із заданими показниками якості. Для розробників ПЗ статистика прояву помилок може бути орієнтиром на розподіл зусиль на налагодження, а також оберігати від надлишкового оптимізму при оцінці якості розроблених ними програм. В процесі проектування ПЗ характеристики помилок допоможуть оцінити реальний стан розробленого ПЗ, а також планувати трудомісткість і тривалість до завершення його розробки, розраховувати необхідну ефективність засобів оперативного захисту від невиявлених первинних помилок, вираховувати необхідні ресурси комп'ютерів, в тому числі і продуктивності, з

врахуванням витрат на усунення помилок, здійснювати відбір відповідних показників складності компонент ПЗ та інших показників якості.

Первинні помилки ПЗ у порядку ускладнення їх виявлення можна розподілити на такі типи [13]:

а) технологічні помилки підготовки машинних носіїв і документації, а такожведення і виведення програм в комп'ютер і його засоби відображення;

б) алгоритмічні помилки, пов'язані з неповним формуванням необхідних умов розв'язку і некоректною постановкою задачі;

в) програмні помилки, пов'язані з некоректною постановкою задачі;

г) системні помилки, зумовлені відхиленням функціювання ПЗ в реальній системі від очікуваних при проектуванні.

На рис. 4 показана схема зв'язків первинних помилок в ПЗ з методами їх виявлення.

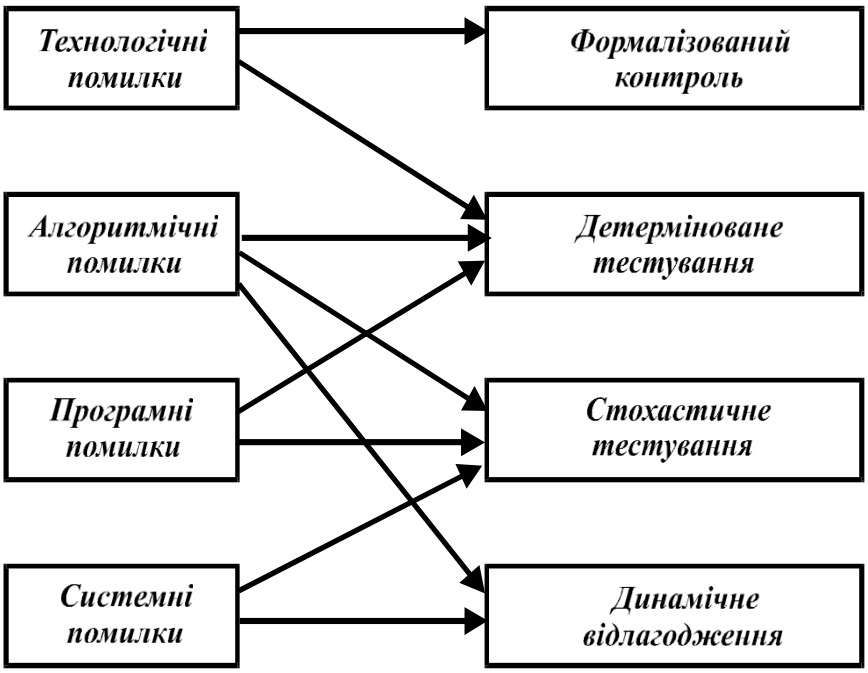

Рис. 4. Первинні помилки ПЗ та методи їх виявлення

Згадані помилки розрізняються за частотою появи і методами їх виявлення на різних етапах проектування ПЗ.

Технологічні помилки документації і запису програм в пам'ять комп'ютера виявляються при налагодженні автоматично формалізованими методами. Вони складають приблизно 5-10% [13] загальної кількості помилок, що виявляються при налагодженні. Навіть при дуже високому рівні функційної налагодженості ПЗ

виявляються окремі технологічні помилки.

Алгоритмічні помилки характеризуються тим, що їх важко виявити за допомогою методів формалізованого автоматичного контролю. Складність їх виявлення і локалізації зумовлена відсутністю повністю формалізованої постановки задачі і точної специфікації. До алгоритмічних помилок, в першу чергу, відносять помилки, котрі зумовлені некоректною постановкою функційних задач, зокрема, коли в специфікаціях не повністю висвітлені всі умови, котрі необхідні для одержання правильного результату. Такі помилки зустрічаються найчастіше і складають до 70-ти відсотків всіх алгоритмічних помилок та біля 30-ти відсотків загальної кількості помилок на початкових етапах проектування. До алгоритмічних помилок відносять також помилки спрягання програмних модулів і функційних груп програм. Їх можна віднести до помилок некоректної постановки задачі.

Частину помилок складають прорахунки у визначенні ресурсів обчислювальних систем для вирішення тієї, чи іншої задачі. Це стосується як нестачі, так і перевищення виділених ресурсів. Такі помилки вимагають коректування з метою задоволення вимог. Найбільш відчутними є алгоритмічні помилки, котрі зумовлені неправильним розрахунком часу реалізації програмних модулів і програм в цілому. Вони знижують достовірність визначення часу обробки інформації за різними технологічними маршрутами.

Неповність інформації про реальні процеси приймання, обробки та видавання даних в комп'ютерних системах приводять до **системних помилок**. Такі помилки зумовлені не завжди наявною можливістю досить точно визначити та описати згадані процеси без попередніх досліджень функціювання системи або її модулів у взаємодії із зовнішніми користувачами. Крім того, на початковій стадії проектування не завжди вдається повно і точно визначити цільову задачу всієї системи, а також задачі основних модулів програми. Як правило, ці задачі уточнюються в процесі подальшого проектування та налагодження.

Відсоток системних помилок зростає на завершальних етапах комплексного налагодження системи, у той час, як при автономному налагодженні він порівняно невеликий – близько 10%. Найбільший відсоток, а саме, близько 80% від загальної

кількості всіх помилок складають системні помилки, котрі виявляються на етапі експлуатації. Виправлення помилок у програмах тягне за собою коректування близько 25 команд на одну помилку, що підтверджує велику трудомісткість процесу виявлення і усунення системних помилок. Важко також і прогнозувати час на їх виявлення та усунення. При інтенсивному виявленні помилок під час налагодження складається враження, що всі помилки виявлені і після нього за причиною різкого зниження інтенсивності виявлення помилок у розробників губиться інтерес до подальшого виявлення помилок і складається ейфорійне уявлення про повну відсутність помилок. Під час серійного випуску комп'ютерних систем на основі розширення умов експлуатації прояв помилок зростає, тому і на протязі певного часу зростає сумарна інтенсивність виявлення помилок. Це є позитивним з тієї сторони, що дозволяє усунути ряд помилок, зумовлених умовами експлуатації систем, і збільшує тривалість між проявами помилок у процесі експлуатації.

Помилки програм можна розділити на помилки, котрі зумовлені обмеженими можливостями програми – неможливість одержання результатів за прийнятний час або при заданих обмеженнях за об'ємом обчислень, та на логічні помилки, результатом яких є одержання неправильних результатів, не дивлячись на час і об'єм обчислень.

Статистика відсотків помилок програмного забезпечення приведена в таблиці 1.

Взагалі, всебічний аналіз помилок, котрі зустрічаються в програмах, можливий тільки при наявності даних про відмови, які зустрічаються в програмах, та їх причини. Вони є основою для побудови математичних моделей надійності ПЗ, оцінки надійності програм з метою її підвищення і забезпечення показників надійності згідно вимог та технічного завдання (ТЗ).

Щодо класифікації помилок ПЗ на різних етапах його життєвого циклу, то їх розподіляють [1] на: помилки при поставі задачі; помилки в проектуванні; помилки програмування; помилки при записі на носії даних; помилки обчислень; логічні помилки; помилки введення-виведення; помилки маніпулювання; помилки спряжень; помилки ініціалізації бази даних; помилки в документації; помилки у визначенні глобальних змінних; помилки порушення технічних вимог; нерозпізнані помилки; помилки оператора.

Постава задачі на розроблення ПЗ оформляється у вигляді ТЗ і технічних умов (ТУ). Правильно поставити задачу, це значить врахувати всі вимоги користувачів. Для цього потрібно добре уяснити задачу, що значить точно встановити очікувані діапазони вихідних даних і одержуваних результатів. Нерозуміння задачі приведе до **помилок у поставі задачі** і відсутності сенсу подальшого проектування ПЗ. Щодо вимог до якості програм, то вони є невід'ємною складовою загальних технічних вимог.

Найбільш часті **помилки ТЗ** до програм представлені в таблиці 2 [6].

Таблиця 2

Причини помилки	**Частота**
Помилки в числових значеннях	12
Недостатні вимоги до точності	4
Помилкові символи чи знаки	2
Помилки оформлення	15
Неправильний опис апаратури	2
Неповні чи неточні основи розроблення	52
Двозначність вимог	13

До **помилок проектування** відносять помилки при виборі параметрів, технічні помилки в проектній документації, недосконаліть обраних математичних методів розв'язку задач, неузгодженість даних у часі, неврахування кореляції між окремими змінними та ін. Більшість із цих помилок виникають за причини неадекватності математичної моделі реальному процесу. Інколи помилки проектування відносять до алгоритмічних помилок.

До **помилок програмування** відносять помилки вибору чисельних методів розв'язку задач, кодування чисел і команд (синтаксичні помилки), розподілу пам'яті, трактування алгоритмічних конструкцій (семантичні помилки), спряження програмних модулів і програм, реалізації умовних пропонувань, у описі даних, в документації. Навіть сучасні засоби налагодження і перевірки програм не завжди достатні для гарантування безпомилкового програмування.

Інколи застосовують класифікацію помилок, що грунтуються на видах робіт по створенню і експлуатації ПЗ. Вона включає в себе такі помилки.

Помилки обчислень, котрі містяться в обраних алгоритмах розв'язку задач. Це може бути невірна рівність чи співвідношення, неадекватність розроблених

математичних моделей реальним процесам, неправильно розрахований логічний чи фізичний номер елементу.

Логічні помилки також в основному носять алгоритмічний характер. Серед них помилки, котрі приводять до зациклення програми, неправильна передача керування, неправильна послідовність дій та ін.

Помилки введення/виведення даних приводять до введення непередбачених або дублюючих елементів даних. Ними є невірні формати даних та ін. Вони в основному містяться в модулях введення/виведення (драйверах) та в операторах інтерфейсу.

Помилки спряжень можуть бути наявні в міжпрограмних інтерфейсах програмних додатків, спряжень прикладних програм з операційною системою та іншими базовими програмами ПЗ, в користувальницьких інтерфейсах, помилки в даних, котрі вводяться оператором вручну, в спряженнях з базами даних за причини невідповідності дійсної структури бази даних з її описом.

Помилки маніпулювання даними виявляються в операціях зчитування, запису, пересилання, зберігання і зміни даних. Ними є: присвоєння вхідним даним неправильних початкових значень; втрата даних; помилки сортування; генерування зайвих елементів інформації, зокрема таблиць, масивів та ін.; пошук даних в неіснуючих таблицях та записах.

Помилки ініціалізації бази даних зустрічаються при описі даних чи в даних, що запитуються, в номінальних, максимальних і мінімальних значеннях, в текстах повідомлень про помилки та ін.

До **помилок в документації** відносять помилки в описах режимів роботи, можливостей програм, описі вихідних форматів, формату карт та ін.

До **інших помилок** відносять помилки у визначенні глобальних змінних, порушення технічних вимог, помилки оператора, нерозпізнані помилки.

Частота прояву згаданих помилок подана в таблиці 3 [1].

Найбільш частими симптомами помилок є [1]: переповнення розрядної сітки (30,4%), невірне передавання керування (16,4%), несумісність з базами даних (14,5%), несумісність програм за даними, що пересилаються (9%), невиконання програмою додаткових функцій (4,9%), несумісність програм (7%), інші симптоми (18,2%).

Таблиця 3

Причини помилки	**Частота прояву,**
Помилки обчислень	7
Логічні помилки	22
Помилки введення/виведення даних	10
Помилки спряжень	10
Помилки маніпулювання даними	15
Помилки ініціалізації бази даних	6
Помилки в документації	8
Інші помилки	22

Найбільш типовими симптомами помилок в ПЗ є також:

а) передчасне (аварійне) закінчення виконання програми;

б) збільшення часу виконання програми;

в) зациклення на виконання певної послідовності команд однієї з програм;

г) часткова чи повна втрата даних, що накопичувались, і які необхідні для успішного виконання задач, що вирішуються;

д) порушення послідовності виклику окремих програм, що приводить або до пропуску необхідних програм, або до непередбачених викликів програм;

е) спотворення окремих елементів даних, зокрема вхідних, проміжних, вихідних, в результаті спотвореної первинної інформації.

Особливо слід приймати до уваги **несправності апаратури.** Вони виникають при роботі апаратури, котра використовується для реалізації обчислювального процесу. Такі несправності значно впливають на надійність ПЗ. Поява відмови чи збоїв у роботі апаратури приводить до порушення нормального проходження обчислювального процесу, зокрема, до спотворення даних і кодів програм у пам'яті.

По відношенню **виникнення помилок в життєвому циклі ПЗ** їх класифікують у часі за двома найважливішими фазами: перша фаза – специфікація і друга фаза – реалізація.

До фази специфікації відносять помилки: прості "помилки специфікації"; "багатозначність специфікації". Причинами цих помилок є:

а) помилкове чи коректне, але не повне формулювання вимог користувача;

б) неможливість представлення специфічної ситуації даної задачі однозначними і точними формалізмами; в) помилкове чи поверхневе розуміння поставленої задачі.

Помилки специфікації є критичними і важко виявляються і коректуються.

Під час фази реалізації проводяться послідовні кроки доводки. Помилки програмних кодів мають велике значення для застереження помилок реалізації. Ці помилки розподіляють на такі класи:

а) проектні помилки;

б) помилки структури даних;

в) алгоритмічні помилки;

г) логічні помилки;

д) помилки, що зумовлені особливостями мов програмування.

Програмні помилки сучасних персональних комп'ютерів подано у Додатку [12].

Основними принципами досягнення відповідного рівня надійності ПЗ при його розробці щодо помилок є: уникнення, виявлення, виправлення та допущення помилок.

Тема: Уникнення помилок.

Уникнення помилок описує методики, виконання котрих забезпечує мінімізацію помилок, що виникають в процесі створення ПЗ.

Виявлення помилок базується на засобах і методах, котрі забезпечують виявлення помилок в програмах, що розробляються.

Виправлення помилок здійснюється на основі конструювання і методології використання функцій, що коректують виправлені помилки чи усувають їх дії.

Допущення помилок забезпечується засобами і методами, котрі дають можливість виконання заданих функцій при наявності помилок.

До засобів уникнення помилок в процесі проектування ПЗ відносяться засоби, котрі попереджують появу помилок в програмах. Їх розподіляють на такі підгрупи:

а) засоби і методи мінімізації складності ПЗ як основної причини появи помилок трансляції;

б) засоби і методи досягнення точності процесу трансляції;

в) засоби і методи удосконалення інформаційних зв'язків розробників;

г) засоби і методи негайного виявлення і усунення помилок трансляції після здійснення кожного її кроку, а не після її повного закінчення.

Зрозуміло, що уникнення помилок є оптимальним підходом в досягненні

надійності ПЗ, що розробляється. Але практика доводить, що навіть при старанній розробці програм, помилки мають місце. Тому виникає необхідність у трьох наступних заходах.

Якщо розробник вважає, що в розробленому ПЗ є помилки, то наступним заходом повинно бути виявлення помилок. Заходи з метою виявлення помилок можуть бути різноманітними. Але існує деякий їх узагальнений перелік, а саме:

1) перевірка ознак кожного вхідного елемента даних. Якщо вхідні дані є цифровими, то в цьому треба переконатися. Перевірити також, чи число має відповідний розмір і знак;

2) перевірка значень вхідних даних на відомі обмеження;

3) повнота перевірки. Якщо для числа задані границі його значення, як верхня, так і нижня, то слід перевірити дотримання обох цих границь;

4) при наявності явної надлишковості вхідних даних необхідно перевірити їх на сумісність;

5) при відсутності надлишковості вхідних даних доцільно її передбачити;

6) порівняння вхідних даних з повними внутрішніми даними.

Взагалі, розробка заходів для виявлення помилок базується на наступних твердженнях:

1) принципі взаємної підозри, суть котрого полягає у тому, що кожен модуль передбачає, що всі останні модулі, пов’язані з ним, містять помилки. Це потребує перевірки інформації, котра поступає в цей модуль від інших модулів;

2) принципі негайного виявлення помилок – до моменту реалізації функцій модуля;

3) принципі надлишковості, котрий передбачає, що всі заходи щодо виявлення помилок грунтуються на певній явній чи неявній надлишковості.

Відомі методи пасивного і активного виявлення помилок програм. Суть першого полягає у тому, що виявлення помилок здійснюється під час виконання запланованих програмних функцій (робочих алгоритмів). При такому підході помилка виявляється тільки тоді, коли здійснюється контрольна функція.

Але перевірка проявів помилок може здійснюватись і за рахунок спроектованих

програмних функцій, котрі активно використовують програмну систему з метою пошуку помилок. Такі методи відносять до активних. Активні методи пошуку помилок можуть бути реалізовані через монітор виявлення, котрий реалізує паралельний процес періодичного сканування даних. В окремих (екстремальних) ситуаціях використовують монітор, який виконує діагностичні тести системи.

Після виявлення помилок необхідно забезпечити коректування даних чи усунути джерела їх спотворення. Виправлення помилок є одним з основних засобів підвищення надійності ПЗ.

Засоби сучасних ПК виявляють несправні апаратні модулі і компоненти, що дає можливість замінити їх ідентичними резервними. Що стосується ПЗ, то такий підхід не завжди є прийнятним, хоча б із-за того, що, якщо конкретний програмний модуль містить помилку, то вона наявна і в резервному. Тому виправлення наслідків помилок ПЗ має свою специфіку і потребує від розробника передбачення можливих типів руйнувань, а також розробки спеціальних програмних функцій, що виправляють ці наслідки дій помилок. Якщо функції корегування помилок не є достатньо потужними, щоб виправляти наслідки дій помилок, то при проектуванні ПЗ переважають засоби уникнення помилок.

Засоби і методи допуску помилок повинні забезпечувати можливість функціювання ПЗ, коли у ньому наявні помилки. До складу таких засобів і методів відносять динамічну надлишковість, допоміжні методи та ізоляцію помилок.

Суть динамічної надлишковості полягає у тому, що одні і ті ж дані обробляються паралельно кількома ідентичними пристроями (процесорами, комп’ютерами), і одержані результати зіставляються. При збіганні результатів вони вважаються правильними. Найкраще це було б реалізувати на алгоритмічно неідентичних модулях, котрі виконують одну і ту ж функцію. Але такий підхід реалізувати важко, тому що він є трудомістким процесом. Забезпечення динамічної надлишковості полягає у використанні додаткових варіантів програмних модулів, коли результат основного модуля вважають некоректним. При такій ситуації система автоматично викликає запасний модуль. Якщо ситуація повторюється, то викликається наступний модуль-дублер і т. д.

Допоміжні методи допуску помилок застосовуються у тому випадку, коли робота системи повинна бути закінчена будь-яким розумним чином. Це, наприклад, можуть бути функції завантаження і виконання допоміжного модуля після контролю і виявлення помилки з метою гарантованого завершення перевірки всіх контрольованих системою процесів, або функції, котрі необхідні в операційних системах для попередження користувачів про збій і можливе спотворення даних.

Суть ізоляції помилок полягає у спробі ізолювати помилки мінімальною частиною програмної системи з такою метою, щоб система при виникненні помилки не втратила своєї роботоздатності. Наприклад, помилка у прикладному додатку, котрий опрацьовується операційною системою,відіб'ється тільки на роботі цієї програми і не вплине на операційну систему та інші прикладні програми.

Взагалі, тільки останні засоби і методи допуску помилок прийнятні для більшості програмних систем.

Тема: Шляхи забезпечення надійності ПЗ.

Основні фактори, котрі визначають надійність функціювання ПЗ, можна об'єднати в три групи. До першої групи відносяться фактори, які безпосередньо визначають надійність програм, а саме:

а) особливості користувачів результатів виконання програм – вимоги до показників надійності, інерцію користувачів, час, необхідний для відклику на вхідні дані;

б) спотворення початкових даних - спотворення даних, що поступають від людини-оператора, даних, що поступають по телекомунікаційних каналах зв'язку, даних в процесі накопичення і зберігання в комп'ютері;

в) помилки в програмах та їх прояв – статистичні характеристики помилок і спотворень програм, масивів даних та обчислювального процесу.

До другої групи факторів належать методи проектування коректних програм:

а) структурне проектування програм і даних - структурне проектування програмних модулів та взаємодії модулів, структурування даних;

б) тестування програм – детерміноване та статистичне тестування, динамічне тестування і контроль пропускної здатності в реальному часі.

Третя група факторів включає методи контролю і забезпечення надійності

програм:

а) методи використання надлишковості часової, інформаційної та програмної;

б) методи контролю програм, даних і обчислювального процесу – передпусковий та оперативний контроль;

в) методи програмного відновлення – відновлення тестів програм, виправлення даних, коректування обчислювального процесу;

г) методи випробування на надійність - випробування в нормальних умовах експлуатації, форсовані випробування, розрахунково- експериментальні методи визначення надійності;

д) методи забезпечення надійності при супроводженні – забезпечення зберігання програм еталонних версій та забезпечення коректності внесення змін і розвитку версій.

Охарактеризуємо більш детально третю групу факторів забезпечення надійності ПЗ. Для підвищення надійності функціювання ПЗ, захисту інформації програмно-алгоритмічними методами використовують часову, інформаційну та програмну надлишковість.

Лекція 4.

Тема:Вимоги до надійності комп'ютерних мереж.

Відповідність до стандартів є лише однією з багатьох вимог, що пред'являються до сучасних мереж. Важливішим є виконання мережею певного набору послуг, наприклад, надання доступу до файлових архівів або веб-сторінок публічних Internet-сайтів, обмін електронною поштою в межах підприємства або в глобальних масштабах, інтерактивний обмін голосовими повідомленнями IP-телефонії тощо.

Решта вимог — продуктивність, надійність, сумісність, керованість, захищеність, розширюваність і масштабованість — пов'язані з якістю виконання цього основного завдання.

І хоча всі перераховані вище вимоги є важливими, часто поняття «**Якість обслуговування**» (*Quality of Service, QOS*) комп'ютерної мережі трактується вужче: воно містить лише дві важливі характеристики мережі — продуктивність і надійність.

Продуктивність

Потенційно висока продуктивність — це одна з основних переваг розподілених систем, до яких відносяться комп'ютерні мережі. Ця властивість забезпечується принциповою можливістю розподілу робіт.

Основні характеристики продуктивності мережі:

- Час реакції мережі.
- Швидкість передачі трафіку.
- Пропускна здатність.

- Затримка передачі і варіанти затримки передачі.

Час реакції мережі

В загальному випадку час реакції визначається як інтервал між відправленням запиту користувача до мережної служби і отриманням відповіді на нього.

Час реакції мережі є інтегральною характеристикою продуктивності мережі з погляду користувача. Саме цю характеристику має на увазі користувач, коли говорить: «Сьогодні мережа працює поволі».

Значення цього показника залежить від типу служби, до якої звертається користувач, від того, який користувач і до якого сервера звертається, а також від поточного стану елементів мережі — завантаженості сервера та сегментів, комутаторів і маршрутизаторів, через які проходить запит.

Тому, варто усереднювати цей показник по користувачах, серверах і годинах доби (від чого в значній мірі залежить завантаження мережі).

Час реакції мережі зазвичай складається з кількох складових.

- Час підготовки запитів на клієнтському комп'ютері.
- Час передачі запитів між клієнтом і сервером через сегменти мережі і проміжне комунікаційне устаткування.
- Час обробки запитів на сервері.
- Час передачі відповідей від сервера до клієнта.
- Час обробки отриманих від сервера відповідей на клієнтському комп'ютері.

Очевидно, що розкладання часу реакції на складові користувачу не є цікавим, головним є кінцевий результат. Проте, для мережного фахівця дуже важливим є виділення з загального часу реакції складових, що відповідають етапам власне мережної обробки даних, — передачі даних від клієнта до сервера через сегменти мережі і комунікаційне обладнання.

Знання мережних складових часу реакції дозволяє оцінити продуктивність окремих елементів мережі, виявити вузькі місця і за необхідності виконати модернізацію мережі для підвищення її загальної продуктивності.

Швидкість передачі трафіку.

- **Середня швидкість** обчислюється шляхом ділення загального об'єму переданих даних на час їх передачі, зазвичай обирається достатньо тривалий проміжок часу — година, день або тиждень.
- **Миттєва швидкість** відрізняється від середньої тим, що для усереднення вибирається дуже маленький проміжок часу — наприклад, 10 мс або 1 с.
- **Максимальна швидкість** — це найбільша швидкість передачі, що зафіксована протягом періоду спостереження.

Зазвичай, при проектуванні, налаштуванні і оптимізації мережі використовуються такі показники, як середня і максимальна швидкість.

Середня швидкість, з якою обробляє трафік окремий елемент мережі або мережа в цілому, дозволяє оцінити роботу мережі впродовж тривалого часу, протягом якого піки і спади інтенсивності трафіку компенсують один одного.

Максимальна швидкість дозволяє оцінити, як мережа буде долати пікові навантаження, які є характерними для особливих періодів роботи, наприклад в ранкові години, коли співробітники підприємства майже одночасно реєструються в мережі і звертаються до розподілених файлів чи баз даних.

Зазвичай, при визначенні швидкісних характеристик певного сегменту чи пристрою в переданих даних не виділяється трафік певного користувача, застосування або комп'ютера — обчислюється загальний об'єм переданої інформації. Проте, для точної оцінки якості обслуговування така деталізація є бажаною, і системи керування мережами дозволяють її виконувати.

Пропускна здатність

Пропускна здатність - це максимально можлива швидкість обробки трафіку, що визначена стандартом технології, на якій побудована мережа. Пропускна здатність відображає максимально можливий об'єм даних, що передається по мережі або її частині в одиницю часу.

Пропускна здатність не є характеристикою, призначеною для користувача, оскільки вона свідчить про швидкість виконання внутрішніх операцій мережі — передачі пакетів даних між вузлами мережі через різні комунікаційні пристрої. Вона безпосередньо характеризує якість виконання основної функції мережі — транспортування повідомлень, тому її частіше використовують при аналізі продуктивності мережі, ніж час реакції або швидкість.

Пропускна здатність вимірюється або в бітах в секунду, або в пакетах в секунду.

Пропускна здатність мережі залежить як від характеристик фізичного середовища передачі (мідний кабель, оптичне волокно, скручена пара) так і від наявного способу передачі даних (технологія Ethernet, FastEthernet, ATM). Пропускна здатність часто використовується як характеристика не стільки мережі, скільки власне технології, на якій побудована мережа.

На відміну від часу реакції або швидкості передачі трафіку пропускна здатність не залежить від завантаженості мережі і має постійне значення, що визначається використаними в мережі технологіями.

На різних ділянках гетерогенної мережі, де використовується кілька різних технологій, пропускна здатність може бути різною. Для аналізу і налаштування мережі корисно знати дані про пропускну здатність окремих її елементів.

Важливо відзначити, що із-за послідовного характеру передачі даних різними елементами мережі загальна пропускна здатність будь-якого складеного шляху в мережі буде мінімальною з пропускних здатностей елементів маршруту. Для підвищення пропускної здатності складеного шляху необхідно в першу чергу звернути увагу на найповільніші елементи.

Іноді корисно оперувати загальною пропускною здатністю мережі, яка визначається як середня кількість інформації, що передається між всіма вузлами мережі за одиницю часу. Цей показник характеризує якість мережі в цілому, не розкладаючи його по окремих сегментах або пристроях.

Затримка передачі

Затримка передачі визначається як час між моментом надходження даних на вхід мережного пристрою або частини мережі та моментом появи їх на виході цього пристрою.

Цей параметр продуктивності за сенсом є близьким до часу реакції мережі, але відрізняється тим, що завжди характеризує лише мережні етапи обробки даних, без затримок обробки кінцевими вузлами мережі.

Звичайну якість мережі характеризують величинами максимальної затримки передачі і варіантом затримки. Не всі типи трафіку є чутливими до затримок передачі, оскільки, зазвичай затримки не перевищують сотень мілісекунд, рідше — кількох секунд.

Затримки пакетів, що зумовлені файловою службою, службою електронної пошти або службою друку, мало впливають на якість цих служб з погляду користувача мережі.

З іншого боку, затримки пакетів, що містять голосові або відеодані, можуть призводити до значного зниження якості наданої користувачу інформації — виникає ефект «відлуння», неможливість розібрати деякі слова, вібрації зображення тощо.

Всі вказані характеристики продуктивності мережі є достатньо незалежними. Тоді як пропускна здатність мережі є постійною величиною, швидкість передачі трафіку може коливатися в залежності від завантаження мережі, не перевищуючи встановлених меж пропускної здатності.

Надійність і безпека

Однією з первинних цілей створення розподілених систем, до яких відносяться і комп’ютерні мережі, було досягнення більшої надійності у порівнянні з окремими обчислювальними машинами. Важливо розрізняти кілька аспектів надійності.

Для простих технічних пристроїв використовуються наступні показники надійності:

- Середній час напрацювання на відмову.
- Вірогідність відмови.
- Інтенсивність відмов.

Проте, ці показники є придатними лише для оцінки надійності простих елементів і пристроїв, які можуть знаходитися лише в двох станах, — працездатному або непрацездатному. Складні системи, що складаються з багатьох елементів, окрім станів працездатності і непрацездатності, можуть мати інші проміжні стани, які ці характеристики не враховують.

Для оцінки надійності складних систем застосовується інший набір характеристик:

- Готовність або коефіцієнт готовності.
- Збереження даних.
- Узгодженість (несуперечність) даних.
- Вірогідність доставки даних.
- Безпека.
- Відмовостійкість.

Готовність або коефіцієнт готовності (*availability*) означає період часу, протягом якого система є готовою до використання. Готовність може бути підвищена шляхом введення надлишковості до структури системи: ключові елементи системи повинні існувати в кількох екземплярах, щоб при відмові одного з них функціонування системи забезпечували інші елементи.

Високонадійна комп'ютерна система повинна як мінімум мати високу готовність, але цього недостатньо. Необхідно забезпечити **збереження даних** і захист їх від спотворень. Крім того, повинна підтримуватися **узгодженість** (несуперечність) даних, наприклад якщо для підвищення надійності на кількох файлових серверах зберігається кілька копій даних, то потрібно постійно забезпечувати їх ідентичність.

Оскільки мережа працює на основі механізму передачі пакетів між кінцевими вузлами, однією з характеристик надійності є **вірогідність доставки пакету** до вузла призначення без спотворень. Разом з цією характеристикою можуть використовуватися і інші показники: вірогідність втрати пакету (із-за переповнення буфера маршрутизатора, не збіганні контрольної суми, відсутності працездатного шляху до вузла призначення тощо), вірогідність спотворення окремого біта переданих даних, співвідношення кількості втрачених і доставлених пакетів.

Іншим аспектом загальної надійності є **безпека** (*security*), тобто здатність системи захистити дані від несанкціонованого доступу. В розподіленій системі це зробити набагато складніше, ніж в централізованій. В мережах повідомлення передаються по лініях зв'язку, що часто проходять через загальнодоступні приміщення, в яких можуть бути встановлені засоби прослуховування ліній. Іншим вразливим місцем можуть стати залишені без нагляду персональні комп'ютери. Крім того, завжди є потенційна загроза злому захисту мережі від неавторизованих користувачів, якщо мережа має виходи в глобальні загальнодоступні мережі.

Ще однією характеристикою надійності є **відмовостійкість** (*fault tolerance*). В мережах під відмовостійкістю розуміють здатність системи приховати від користувача відмову окремих її елементів. Наприклад, якщо копії таблиці бази даних зберігаються одночасно на кількох файлових серверах, користувачі можуть просто не помітити відмови однієї з них. У відмовостійкій системі вихід з ладу одного з її елементів призводить до певного зниження якості її роботи (деградації), а не до повного останову. Так, при відмові одного з файлових серверів збільшується лише час доступу до бази даних із-за зменшення ступеня розпаралелювання запитів, але в цілому система буде продовжувати виконувати свої функції.

Розширюваність і масштабованість

Розширюваність (*extensibility*) означає можливість порівняно легкого додавання окремих елементів мережі (користувачів, комп'ютерів, застосувань, служб), нарощування довжини сегментів мережі і заміни існуючої

апаратури на більш потужну. При цьому є важливим, що легкість розширення системи іноді може забезпечуватися в обмеженому діапазоні.

Наприклад, локальна мережа Ethernet, яка побудована на основі одного сегменту товстого коаксіального кабелю, має хорошу розширюваність, що дозволяє без зусиль під'єднувати нові станції. Проте, така мережа має обмеження на число станцій — воно не повинне перевищувати 30–40. Хоча мережа допускає фізичне під'єднання до сегменту і більшого числа станцій (до 100), але при цьому, зазвичай, різко знижується продуктивність мережі. Наявність такого обмеження і є ознакою поганої масштабованості системи при хорошій розширюваності.

Масштабованість (*scalability*) означає можливість нарощувати кількість вузлів і протяжність зв'язків мережі в дуже широких межах, при цьому її продуктивність не погіршується. Для забезпечення масштабованості мережі доводиться застосовувати додаткове комунікаційне обладнання і спеціальним чином структурувати мережу. Наприклад, хорошу масштабованість має багатосегментна мережа, що побудована з використанням комутаторів і маршрутизаторів і має ієрархічну структуру зв'язків. Така мережа може містити кілька тисяч комп'ютерів і при цьому забезпечувати кожному користувачеві мережі потрібну якість обслуговування.

Прозорість

Прозорість (*transparency*) мережі досягається, коли мережа представлена користувачам не як множина окремих комп'ютерів, що зв'язані між собою складною системою кабелів, а як єдина традиційна обчислювальна машина з системою розділення часу.

Прозорість може бути досягнута на двох різних рівнях — на рівні користувача і на рівні адміністратора. На рівні користувача прозорість означає, що для роботи з віддаленими ресурсами він використовує ті ж команди і звичні процедури, що і для роботи з локальними ресурсами. На адміністративному рівні прозорість полягає в тому, що застосуванню для доступу до віддалених ресурсів потрібні ті ж виклики, що і для доступу до локальних ресурсів. Прозорості на рівні користувача досягти простіше, оскільки всі особливості процедур, що пов'язані з розподіленим характером системи, є прихованими від користувача програмістом, який створює застосування. Прозорість на рівні застосування вимагає приховування всіх деталей розподілення засобами мережної операційної системи.

Прозорість — це властивість мережі приховувати від користувача деталі свого внутрішнього устрою, що спрощує роботу в мережі.

Мережа повинна приховувати всі особливості операційних систем і відмінності в типах комп'ютерів. Користувач комп'ютера Macintosh повинен мати можливість звертатися до ресурсів, що підтримуються UNIX-системою, а користувач UNIX — розділяти інформацію з користувачами Windows.

Концепція прозорості застосовується до різних аспектів мережі. Наприклад, прозорість розташування означає, що користувачу не потрібно знати місцезнаходження програмних і апаратних ресурсів, таких як процесори, принтери, файли і бази даних. Ім'я ресурсу не повинне містити інформацію про місце його розташування.

Аналогічно, прозорість переміщення означає, що ресурси можуть вільно переміщатися з одного комп'ютера до іншого без зміни імен. Ще одним з можливих аспектів прозорості є прозорість паралелізму, яка полягає в тому, що процес розпаралелювання обчислень відбувається автоматично, без участі адміністратора, при цьому система сама розподіляє паралельні гілки застосування по процесорах та комп'ютерах мережі.

Підтримка різних видів трафіку

Комп'ютерні мережі спочатку призначалися для сумісного доступу до ресурсів комп'ютерів: файлів, принтерів тощо. Трафік, що створюється традиційними службами комп'ютерних мереж, має свої особливості і істотно відрізняється від трафіку повідомлень в телефонних мережах або, наприклад, в мережах кабельного телебачення. Проте з 90-их років в комп'ютерних мережах з'являється трафік мультимедійних даних (звук і відео в цифровій формі).

Комп'ютерні мережі стали використовуватися для організації відеоконференцій, навчання на основі відеофільмів, прямих теле- та радіотрансляцій тощо. Природно, що для динамічної передачі мультимедійного трафіку потрібні інші алгоритми та протоколи, і, відповідно, інше устаткування.

Головною особливістю трафіку, що утворюється при динамічній передачі голосу або зображення, є наявність жорстких вимог до синхронності переданих повідомлень. Для якісного відтворення безперервних процесів, якими є звукові коливання чи зміни інтенсивності світла у відеозображенні, необхідним є отримання виміряних і закодованих амплітуд сигналів з тією ж частотою, з якою вони були виміряні на стороні відправника. При запізненні повідомлень будуть спостерігатися спотворення.

В той же час трафік комп'ютерних даних характеризується вкрай нерівномірною інтенсивністю надходження повідомлень до мережі за відсутності жорстких вимог до синхронності доставки цих повідомлень. Наприклад, доступ користувача, що працює з текстом на віддаленому диску, породжує випадковий потік повідомлень між віддаленим і локальним комп'ютерами, що залежить від дій користувача, причому затримки при доставці в певних (достатньо широких з комп'ютерної точки зору) межах мало впливають на якість обслуговування користувача мережі.

Всі алгоритми комп'ютерного зв'язку, відповідні протоколи і комунікаційне устаткування були розраховані саме на такий «пульсуючий» характер трафіку, тому необхідність передавати мультимедійний трафік вимагає внесення принципових змін, як до протоколів, так і до устаткування. Сьогодні всі нові протоколи надають підтримку мультимедійного трафіку.

Особливу складність представляє поєднання в одній мережі традиційного комп'ютерного і мультимедійного трафіку. Передача виключно мультимедійного трафіку по комп'ютерній мережі хоч і пов'язана з певними складнощами, але доставляє менше клопоту. А ось співіснування двох типів трафіку з протилежними вимогами до якості обслуговування є набагато складнішим завданням.

Керованість

В ідеалі засоби керування мережами є системою, що здійснює спостереження, контроль і управління кожним елементом мережі, — від простих до найскладніших пристроїв, при цьому така система розглядає мережу як єдине ціле, а не як розрізнений набір окремих пристроїв.

Керованість мережі передбачає можливість централізованого контролю стану основних елементів мережі, виявлення і вирішення проблем, що виникають при роботі мережі, виконання аналізу продуктивності і планування розвитку мережі.

Хороша система керування:

- Спостерігає за мережею і при виявленні проблеми, активізує певну дію, виправляє ситуацію і повідомляє адміністратора про те, що відбулося і які кроки зроблено.
- Повинна накопичувати дані, на підставі яких можна планувати розвиток мережі.
- Повинна бути незалежною від виробника і мати зручний інтерфейс, що дозволяє виконувати всі дії з однієї консолі.

Вирішуючи тактичні завдання, адміністратори і технічний персонал стикаються з щоденними проблемами забезпечення працездатності мережі. Ці завдання вимагають швидкого рішення, обслуговуючий персонал мережі повинен оперативно реагувати на повідомлення про несправності, що поступають від користувачів або автоматичних засобів управління мережею. Поступово стають помітними загальні проблеми продуктивності, конфігурації мережі, обробки збоїв і безпеки даних, що вимагають стратегічного підходу, тобто планування мережі. Планування, також містить прогноз змін вимог користувачів до мережі, питання застосування нових додатків та мережних технологій.

Необхідність в системі керування виявляється у великих мережах: корпоративних або глобальних. Без системи керування в таких мережах потрібна присутність кваліфікованих фахівців з експлуатації в кожній будівлі кожного міста, де встановлено устаткування мережі, що в результаті приводить до необхідності формування великого штату обслуговуючого персоналу.

Сумісність

Сумісність або інтегрованість означає, що мережа може містити різноманітне програмне і апаратне забезпечення, тобто в ній можуть співіснувати різні операційні системи, що підтримують різні стеки комунікаційних протоколів, а також апаратні засоби та застосування від різних виробників.

Мережа, що складається з різнотипних елементів, називається неоднорідною або гетерогенною. Якщо гетерогенна мережа працює без проблем, то вона є інтегрованою. Основним шляхом побудови інтегрованих мереж є використання модулів, які виконані відповідно до відкритих стандартів і специфікацій.

Якість обслуговування

Якість обслуговування (*QOS, Quality of Service*) визначає кількісні оцінки вірогідності того, що мережа буде передавати певний потік даних між двома вузлами відповідно до потреб застосування або користувача.

Важливі параметри для оцінювання якості обслуговування:

- Пропускна здатність.
- Затримки передачі пакетів.
- Рівень втрат і спотворень пакетів.

Якість обслуговування гарантується для любого потоку даних, тобто для послідовності пакетів, що мають певні загальні ознаки, наприклад адресу вузла-джерела, інформацію, що ідентифікує тип застосування тощо.

До потоків застосовують поняття агрегації та диференціювання. Так, потік даних від одного комп'ютера може бути представлений як сукупність потоків від різних застосувань, а потоки від комп'ютерів одного підприємства агреговані до одного потоку даних абонента певного провайдера послуг.

Механізми підтримки якості обслуговування самі по собі не створюють пропускної здатності. Фактична пропускна здатність каналів зв'язку і транзитного комунікаційного устаткування — це ресурси мережі, що є відправною точкою для роботи механізмів QOS. Механізми QOS лише керують розподілом наявної пропускної здатності відповідно до вимог застосувань і налаштувань мережі. Найочевиднішим способом перерозподілу пропускної здатності мережі є керування чергами пакетів.

Оскільки дані, якими обмінюються два кінцеві вузли, проходять через певну кількість проміжних мережних пристроїв, таких як концентратори, комутатори і маршрутизатори, то підтримка QOS вимагає взаємодії всіх мережних елементів на шляху трафіку, тобто «з-кінця-в-кінець» («*end-to-end*», «e2e»). Будь-які гарантії QOS настільки відповідають дійсності, наскільки їх забезпечує найбільш «слабкий» елемент в ланцюжку між відправником і одержувачем. Тому, слід чітко розуміти, що підтримка QOS лише в одному мережному пристрої, нехай навіть і магістральному, може лише дещо покращити якість обслуговування або ж зовсім не мати впливу на параметри QOS.

Лекція 5.

Тема: Кібербезпека, атаки, поняття та методи. Захист інформації.

Кібербезпека - це постійні зусилля спрямовані на захист мережних систем та усіх видів даних від несанкціонованого використання або пошкодження. На особистому рівні вам необхідно захищати свій обліковий запис, дані та електронні пристрої.

Захист мережних систем та даних від неавторизованого використання та пошкодження

Ваша онлайн (online) та офлайн (offline)-ідентифікація

Офлайн ідентичність

Ваша особа, з якою щодня взаємодіють ваші друзі та сім'я вдома, в школі, на роботі.

• Онлайн ідентичність

- Ваша ідентичність в кіберпросторі.
- Має розкривати мінімум інформації про вас.
- Ім'я користувача або псевдонім
- Не має містити ніякої персональної інформації.
- Має бути доречним та прийнятним.
- Не має привертати небажану увагу.

▪ Ваші дані

• Медичні записи

- Електронна медична картка (EHR) - містить інформацію про фізичне, психічне здоров'я та іншу особисту інформацію.
- рекомендації лікарів.

• Дані про освіту

- Оцінки, результати тестування, нагороди та присвоєні ступені.
- Відвідування занять.
- Звіти про дисциплінарні порушення.

• Записи про працевлаштування та фінансова документація

- Доходи та витрати
- Податкові записи - інформація про заробітну плату, виписки з кредитної

картки, кредитний рейтинг та інша банківська інформація.

- Інформація про попередні місця роботи та результати діяльності.

Інтернет речей та великі дані

- IoT - це велика мережа фізичних об'єктів, таких як сенсори.
- Big Data - це дані з IoT.

▪ Конфіденційність, цілісність та доступність

• Конфіденційність – приватність інформації.

• Цілісність - точність і достовірність інформації.

• Доступність - інформація є доступною.

Конфіденційність, цілісність та доступність, відомі як тріада CI, є керівним принципом безпеки інформації для організації. Конфіденційність гарантує нерозголошення даних, обмежуючи доступ до них через механізми аутентифікації і шифрування.

▪ Цілісність гарантує точність і достовірність інформації.

▪ Доступність гарантує, що інформація доступна для авторизованих користувачів.

▪ **Конфіденційність (Confidentiality)**

Синонімом до терміну конфіденційність є приватність. Політики компанії мають обмежувати доступ до інформації колом авторизованих користувачів та гарантувати, що тільки ці особи бачать ці дані. Доступ до даних надається відповідно до політики безпеки або рівня секретності інформації.

Методи забезпечення конфіденційності передбачають шифрування даних, ідентифікацію по імені користувача та паролю, двофакторну аутентифікацію та мінімізацію розголошення конфіденційної інформації.

▪ **Цілісність (Integrity)**

Цілісність - точність, узгодженість і достовірність даних впродовж всього їх життєвого циклу. Дані повинні залишатися незмінними в процесі передачі, а також недоступними для змін неавторизованими суб'єктами. Дозволи на використання файлів та контроль доступу користувачів можуть запобігти неавторизованому доступу. Контроль версій може використовуватися для запобігання випадковим змінам авторизованими користувачами. Резервні копії повинні бути доступні для відновлення будь-якихі пошкоджених даних, а хеш (контрольна сума) може бути використаний для перевірки цілісності даних під час передавання.

▪ Контрольна сума використовується для перевірки цілісності файлів або рядків символів після перенесення їх з одного пристрою на інший через локальну мережу або Інтернет. Контрольна сума розраховується за допомогою хеш-функцій. Деякі популярні функції обчислення контрольних сум: MD5, SHA-1, SHA-256 та SHA-512. Хеш-функція використовує математичний алгоритм для перетворення даних у значення фіксованої довжини, що представляє ці дані, як зображено на рис. 2.

▪ **Доступність (Availability)**

Обслуговування обладнання, проведення ремонту апаратного забезпечення, оновлення операційних систем та програмного забезпечення і створення резервних копій забезпечують доступність мережі та даних для авторизованих користувачів. Має бути план швидкого відновлення систем після природних катаклізмів або техногенних катастроф.

Апаратні та програмні засоби безпеки, як наприклад, фаєрволи, захищають систему від простоїв через такі атаки, як Відмова в обслуговуванні (Denial of Service, DoS). Відмова в обслуговуванні виникає, коли зловмисник намагається надмірно завантажити ресурси системи, щоб унеможливити доступ користувачів до потрібних сервісів.

▪ Наслідки порушення безпеки

• Неможливо запобігти кожній атаці.

• Зловмисники завжди будуть шукати нові шляхи для здійснення атак.

• Зрунована репутація, вандалізм, витік конфіденційних документів, втрата доходу, порушення прав інтелектуальної власності.

▪ Приклад порушення безпеки - LastPass

• Це онлайн-менеджер паролів

• Було вкрадено адреси електронної пошти, нагадування про паролі та хеші аутентифікації.

• Вимагає підтвердження через електронну пошту або використовує багатофакторну аутентифікацію під час нового входу з невідомого пристрою.

• Користувачі мають використовувати складні майстер-паролі та регулярно їх змінювати, пам'ятати про фішінгові атаки.

▪ Приклад порушення безпеки - Vtech

• Vtech - виробник високотехнологічних іграшок для дітей,розкрито таку конфіденційну інформацію, як імена клієнтів, адреси електронної пошти, паролі, фотографії та журнали чату.

• Vtech не захищав інформацію належним чином.

• Хакери могли створювати облікові записи електронної пошти, подавати заявки на отримання кредитів та вчиняти інші злочини, використовуючи здобуту інформацію про дітей.

• Хакери також могли дістатися облікових записів батьків в Інтернеті.

▪ Приклад порушення безпеки - Equifax

• Equifax - одна з національних агенцій зі звітності про споживачів кредитів у США.

• Нападники використали вразливість у програмному коді веб-застосунку.

• Equifax створив виділений веб-сайт з новим доменним іменем, що дозволило неналежним сторонам створювати неавторизовані веб-сайти для реалізації фішингової схеми.

Зловмисники та експерти з кібербезпеки

Аматори

Скріпт-діти, які майже не мають навичок.

Використовують існуючі інструменти або вказівки з Інтернету.

Хакери - зламують комп'ютери або мережі з метою отримання доступу

Білі капелюхи - втручаються в системи за попереднім дозволом, щоб виявити слабкі місця та покращити безпеку цих систем.

Сірі капелюхи - компрометують системи без дозволу.

Чорні капелюхи - використовують будь-яку вразливість для отримання незаконної особистої, фінансової або політичної користі.

Організовані хакери - організації кіберзлочинців, хактивісти, терористи та хакери, що фінансуються державою.

Нападники - це окремі особи або групи, які намагаються використати кібер-вразливість для особистої або фінансової вигоди. Вони зацікавлені в усьому: від кредитних карток до дизайну продукту, та будь-чого, що має цінність.

Аматори (Amateurs), таких людей іноді називають скріпт-дітьми (Script Kiddies). Зазвичай це зловмисники, які практично не мають навичок і часто для запуску атак використовують існуючі інструменти або вказівки, знайдені в Інтернеті. Деякі з них роблять це просто з цікавості, а інші намагаються продемонструвати свою майстерність і заподіяти шкоду. Навіть якщо вони використовують прості інструменти, наслідки таких атак можуть бути руйнівними.

Хакери (Hackers) - це група зловмисників, яка зламує комп'ютери або мережі з метою отримання доступу. Залежно від наміру вторгнення ці зловмисники класифікуються як Білі, Сірі або Чорні капелюхи. Білі капелюхи втручаються в мережі або комп'ютерні системи, щоб виявити слабкі місця та покращити безпеку цих систем.

Такі вторгнення виконуються за попереднім дозволом і всі результати повідомляються власнику. З іншого боку, Чорні капелюхи використовують будь-яку вразливість для отримання незаконної особистої, фінансової або політичної користі. Сірі капелюхи знаходяться десь посередині між Білими та Чорними. Сірі капелюхи можуть виявити вразливість у системі та повідомити про неї власників системи, якщо це збігається з їхніми планами. Деякі Сірі капелюхи публікують факти про вразливість в Інтернеті, щоб інші нападники могли нею скористатися.

Організовані хакери (Organized Hackers) – до таких хакерів належать організації кіберзлочинців, хактивісти, терористи та хакери, що фінансуються державою. Кіберзлочинці - це, зазвичай, групи професійних злочинців, орієнтовані на контроль, владу та збагачення. Ці злочинці висококваліфіковані та організовані і навіть можуть надавати кіберзлочинність, як сервіс іншим злочинцям. Хактивісти роблять політичні заяви, щоб привернути увагу до важливих для них проблем. Нападники, які фінансуються державою, здійснюють розвідку або диверсії від імені свого уряду. Ці нападники зазвичай якісно підготовлені та добре фінансуються, а їх атаки зосереджені на корисних для їхнього уряду конкретних цілях.

- Внутрішні загрози безпеці
- Може нашколити працівник або бізнес-партнер.
- Неправильне поводження з конфіденційною інформацією.
- Загроза роботі внутрішніх серверів або пристроїв мережної інфраструктури.
- Сприяння зовнішнім атакам, за рахунок підключення USB-носія до корпоративної комп'ютерної системи.
- Випадкове завантаження шкідливого ПЗ до мережі через зловмисні електронні листи або веб-сайти.
- Можуть спричинити великі збитки через наявність прямогодоступу.
- Зовнішні загрози безпеці використовують вразливості в мережах та комп'ютерних пристроях;
- застосовують соціальну інженерію для отримання доступу.

Кібервійни. Що таке кібервійна?

Кіберпростір став ще одним важливим аспектом війни, де країни можуть конфліктувати без зіткнення традиційних військ та техніки. Це дає змогу країнам з мінімальними військовими силами бути такими ж сильними, як і інші країни в кіберпросторі. Кібервійна - це Інтернет-конфлікт, який передбачає проникнення у комп'ютерні системи та мережі інших країн. Нападники мають ресурси та спеціальні знання, щоб започатковувати масштабні інтернет-атаки проти інших країн, завдаючи збитків або порушуючи роботу сервісів, наприклад припинення роботи енергосистеми.

- Конфлікт з використанням кіберпростору
- Зловмисне ПЗ Stuxnet
- Розроблене для пошкодження фабрики зі збагачення урану в Ірані.
- Використання модульного кодування.
- Використовує вкрадені цифрові сертифікати.
- Отримати перевагу над супротивниками, націями, урядами
- Диверсії на інфраструктурі інших країн.
- Надає атакуючим можливості шантажувати членів уряду.

• Громадяни можуть втрати довіру до здатності уряду захищати їх взагалі.

• Впливає на довіру громадян до уряду, навіть коли немає фізичного вторгнення до країни-жертви.

Атаки, поняття та методи.

Аналіз кібератаки

Розгляд характеристик і функціонування кібератаки.

Пояснення, як використовується вразливість системи безпеки.

Визначення прикладів вразливостей систем безпеки.

Опис типів шкідливого ПЗ та їх симптомів.

Опис методів проникнення.

Опис методів, які використовуються для реалізації атак відмови в обслуговуванні.

Ландшафт кібербезпеки

Пояснення тенденцій ландшафту кіберзагроз.

Опис змішаних атак.

Опис важливості зменшення наслідків атаки.

Аналіз кібератаки*Експлойт* - це термін для опису програми, написаної для використання відомої вразливості.

Атака - це дія, яка полягає у застосуванні експлойта проти вразливості.

Вразливість програмного забезпечення

Помилки в ОС або коді програми SYNful Knock – Вразливість у Cisco IOS дає можливість нападникові отримати контроль над маршрутизаторами відстежувати мережне з'єднання заражати інші мережні пристрої.

Project Zero – Google сформував постійну команду для виявлення вразливостей програмного забезпечення.

Вразливість обладнання

Недоліки конструкції обладнання

Rowhammer - експлойт оперативної пам'яті, якій дозволяє отримувати дані з суміжних комірок пам'яті.

Пошук вразливостей в системі безпеки

Вразливості системи безпеки - це будь-які дефекти програмного або апаратного забезпечення. Після отримання інформації про вразливість, зловмисники намагаються її використати.

Експлойт - це термін, який вживають для опису програми, що написана для використання відомої вразливості. Використання експлойта для вразливості називається атакою.

Мета атаки - отримати доступ до системи та розміщених на ній даних або до певного ресурсу.

▪ **Вразливості програмного забезпечення**

▪ Вразливості програмного забезпечення зазвичай є наслідками помилок в коді операційної системи або коді застосунку. Незважаючи на всі зусилля компаній з пошуку та виправлення вразливих місць ПЗ, регулярно виявляються нові вразливості.

▪ **Вразливості апаратного забезпечення**

▪ Вразливості апаратного забезпечення часто обумовлені недоліками його проектування. Наприклад оперативна пам'ять (RAM), по своїй суті, є набором дуже

близько розташованих один до одного конденсаторів. Було виявлено, що через конструктивні особливості постійні зміни в одному з цих конденсаторів можуть впливати на сусідні. На основі цього конструктивного недоліку було створено експлойт Rowhammer. Постійно перезаписуючи пам'ять за тими самими адресами, експлойт Rowhammer дозволяє змінювати значення даних у сусідніх комірках адресної пам'яті, навіть якщо вони захищені.

- Вразливості апаратного забезпечення характерні для конкретних моделей пристроїв і зазвичай не використовуються під час випадкових спроб компрометації. Оскільки експлойти апаратного забезпечення характерні для цілеспрямованих атак, традиційний засоби захисту від зловмисного ПЗ та фізична безпека є достатніми для безпеки пересічного користувача.
- Переповнення буфера
- Дані записуються за межі буфера
- Неперевірені вхідні дані
- Викликає непередбачувану поведінку програми
- Стан гонитви
- Невідповідне впорядкування подій за послідовністю і часом
- Недоліки реалізації системи безпеки
- Захист важливих даних за допомогою автентифікації,авторизації та шифрування
- Проблеми контролю доступу
- Контроль доступу до фізичного обладнання та ресурсів
- Практики безпеки

- **Переповнення буфера (Buffer overflow) –** ця вразливість виникає, коли дані записуються за межами буфера. Буфери - це області пам'яті, виділені застосунку. Змінюючи дані за межами буфера, застосунок отримує доступ до пам'яті, яка була виділена для інших процесів. Це може спричинити крах системи, компрометацію даних або отримання повноважень більш високого рівня.
- **Неперевірені вхідні дані (Non-validated input) –** програми часто працюють з вхідними даними. Ці дані, що надходять до програми, можуть мати зловмисний вміст, який здатен спричинити непередбачену поведінку програми. Розглянемо програму, яка отримує зображення для обробки. Зловмисник може створити файл зображення із некоректними розмірами. Некорректні розміри можуть примусити програму виділити буфери неправильних та неочікуваних розмірів.
- **Стан гонитви (Race conditions) –** ця вразливість полягає в тому, що результат події залежить від того, в якій послідовності, або ж з якою тривалістю виконуються деякі окремі події. Стан гонитви стає джерелом вразливості, коли події, які вимагають впорядкування або часової синхронізації, не відбуваються у правильному порядку або не вкладаються у належні часові межі.
- **Недоліки реалізації системи безпеки (Weaknesses in security practices) –** системи та конфіденційні дані можуть бути захищені за допомогою таких методів, як аутентифікація, авторизація та шифрування. Розробники не повинні намагатися створювати власні алгоритми безпеки, оскільки це, найпевніше, додасть вразливостей. Настійно рекомендується використовувати готові бібліотеки безпеки, які вже протестовані та перевірені.

▪ **Проблеми контролю доступу (Access-control problems) –** Контроль доступу – це процес, який визначає дозволи на певні дії. Він охоплює питання від управління фізичним доступом до обладнання, до визначення, хто має доступ до ресурсу, наприклад, файлу, і що він з ним може робити, читати чи змінювати. Багато вразливостей безпеки з'являється через неправильне використання засобів контролю доступу.

▪ Майже всі засоби контролю доступу та заходи безпеки можна обійти, якщо зловмисник має фізичний доступ до цільового обладнання. Наприклад, не має значення які дозволи ви встановили для файлу, ОС не може заборонити зловмиснику зчитати дані безпосередньо з диску в обхід операційної системи. Для захисту пристроїв та розміщених на них даних потрібно застосовувати обмеження фізичного доступу. Для захисту даних від крадіжки або пошкодження слід використовувати методи шифрування.

▪ Зловмисне програмне забезпечення (Malicious Software) - це будь-який код, який може використовуватися для викрадання даних, обходу системи керування доступом, пошкодження або компрометації системи.

Типи зловмисного ПЗ

- **Шпигунське ПЗ** - відстежує дії та шпигує за користувачем.
- **Рекламне ПЗ** - поширює рекламу, зазвичай постачається разом зі шпигунськими програмами.
- **Бот** - автоматично виконує дію.
- **Програма-вимагач** - блокує комп'ютерну систему або розміщені на ній дані до отримання викупу.
- **Псевдовірус** - переконує користувача у необхідності виконання певних дій, часто шляхом залякування.

▪ **Шпигунські програми (Spyware) –** це зловмисне програмне забезпечення, призначене для стеження та шпигування за користувачем. Шпигунські програми часто містять засоби відстеження активності, зчитування з клавіатури та перехоплення даних. Намагаючись подолати заходи безпеки шпигунські програми часто змінюють налаштування безпеки. Шпигунські програми часто прив'язуються до легального програмного забезпечення або до троянських коней.

▪ **Рекламне ПЗ (Adware) –** призначене для автоматичного поширення реклами. Рекламне ПЗ часто встановлюється разом з деякими версіями програмного забезпечення. Іноді рекламне ПЗ призначене лише для поширення реклами, але досить часто з ним поширюється шпигунське ПЗ.

▪ **Бот (Bot) –** походить від слова robot, бот – це шкідлива програма, призначена для автоматичного виконання дій, зазвичай в Інтернеті. Хоча більшість ботів безпечні, зростання кількості зловмисних ботів призводить до створення бот-мереж (botnet). Певна кількість комп'ютерів інфікується ботами, які запрограмовані спокійно очікувати команди від зловмисника.

▪ **Програми-вимагачі (Ransomware) –** це шкідливе ПЗ призначене для блокування комп'ютерної системи або розміщених на ній даних до моменту здійснення викупу. Такі програми зазвичай шифрують дані на комп'ютері за допомогою невідомого користувачу ключа. Деякі інші версії програм-вимагачів можуть для блокування системи використати її певні вразливості. Ransomware розповсюджується через завантажений файл або певну вразливість програмного забезпечення.

▪ **Псевдоантивірус (Scareware) –** це тип шкідливого ПЗ, що переконує користувача виконати конкретну дію, використовуючи його страх. Scareware створює спливаючі вікна, схожі на діалогові вікна операційної системи. Ці вікна відображають підроблені повідомлення про те, що система знаходиться під загрозою або необхідне виконання відповідної програми для повернення до нормальної роботи. Насправді жодних проблем на комп'ютері немає, і якщо користувач погоджується та дозволяє запустити зазначену програму, його система буде заражена шкідливим ПЗ.

Типи зловмисного ПЗ

Руткіт - вносить зміни до операційної системи для створення чорного ходу.

- **Вірус** - шкідливий виконуваний код, прикріплений до інших файлів запуску.
- **Троянський кінь** - виконує зловмисні дії під виглядом бажаної операції.
- **Хробаки** - самостійно розмножуються, використовуючи вразливості мереж.
- **Людина посередині (Man-in-The-Middle** або **Man-in-The-Mobile)** – бере під контроль пристрій без відома користувача.

▪ **Руткіт (Rootkit) –** це зловмисне програмне забезпечення, призначене для змін в операційній системі з метою створення чорного ходу (backdoor). Після чого нападники використовують цей чорний хід для віддаленого доступу до комп'ютера. Більшість руткітів використовують вразливості ПЗ для підвищення привілеїв та модифікації системних файлів. Руткітам також притаманні зміни в системних інструментах експертизи та моніторингу, що дуже ускладнює виявлення цього зловмисного ПЗ. Часто для знищення руткіту необхідноповністю очистити ПК та перевстановити операційну систему.

▪ **Вірус (Virus) -** це шкідливий виконуваний код, який прикріплюється до інших виконуваних файлів, часто легітимних програм. Більшість вірусів вимагає активації з боку кінцевих користувачів і можуть справцьовувати у певний день або час. Віруси можуть бути нешкідливими і просто відображати рисунок або можуть мати руйнівні наслідки, такі, як зміна або видалення даних. Віруси також можуть бути запрограмовані на мутацію для запобігання виявленню. На сьогодні більшість вірусів поширюється через USB-накопичувачі, оптичні диски, мережні ресурси або електронною поштою.

▪ **Троянський кінь (Trojan horse) -** це зловмисне ПЗ, яке здійснює шкідливі дії під виглядом бажаної операції. Цей шкідливий код використовує привілеї користувача, який його запускає. Часто трояни містяться в файлах зображень, аудіофайлах або іграх. Троянський кінь відрізняється від вірусу тим, що він приєднується до невиконуваних файлів.

▪ **Черв'яки або хробаки (Worms) –** це шкідливий код, який клонує себе, самостійно використовуючи вразливості в мережах. Хробаки зазвичай уповільнюють роботу мереж. Якщо для активації вірусу необхідно запустити на виконання програму-носій, то хробаки можуть працювати самостійно. Участь користувача потрібна тільки під час первинного зараження. Після того, як хост заражений, хробак може швидко поширюватися мережею. Хробаки мають схожі шаблони поведінки. А саме, використовують вразливість системи, мають здатність до самостійного розповсюдження і виконання дій на користь зловмисника.

▪ **Людина посередині (Man-In-The-Middle або MitM) –** дає змогу зловмиснику контролювати потік даних між користувачами без їх відома. За допомогою

такого рівня доступу зловмисник може перехоплювати, збирати та підмінювати інформацію користувача перш ніж предати її одержувачу. Атаки MitM широко використовуються для викрадення фінансової інформації. Існує безліч зловмисних програм та методик, які наділяють атакуючих можливостями MitM.

- **Посередник в мобільному телефоні (Man-In-The-Mobile або MitMo)** – варіант атаки "Людина посередині", який використовується для отримання контролю над мобільним пристроєм. Заражений мобільний пристрій може отримати наказ зібрати конфіденційну інформацію користувача і відправити її нападникам. ZeuS, як приклад експлойта з можливостями MitMo, дає змогу атакуючим непомітно перехоплювати SMS-повідомлення, які надсилаються користувачам під час проведення 2-етапної перевірки.
- Збільшується навантаження на центральний процесор.
- Знижується швидкодія комп'ютера.
- Комп'ютер часто зависає або дає збої.
- Знижується швидкість перегляду веб-сторінок.
- Виникають незрозумілі проблеми з мережними з'єднаннями.
- Модифікуються файли.
- Видаляються файли.
- З'являються невідомі файли, програми або піктограми на робочому столі.
- Запускаються невідомі процеси.
- Програми самостійно припиняють виконання або змінюють свої налаштування.
- Електронні листи надсилаються без відома та згоди користувача.
- Соціальна інженерія – це маніпулювання особою задля спонукання її до виконання певних дій або розголошення конфіденційної інформації,це психологічна атака доступу, яка намагається маніпулювати особами з метою спонукання іх до виконання певних дій або розголошення конфіденційної інформації.

• **Претекстинг** - нападник телефонує особі і обманом намагається отримати доступ до привілейованих даних.

• **Тейлгейтінг** - зловмисник намагається швидко прослизнути за уповноваженою особою до захищених приміщень.

• **Послуга за послугу (Quid pro quo)** - нападник запитує в учасника деяку особисту інформацію в обмін на щось.

- Злам пароля Wi-Fi - Виявлення пароля

• **Соціальна інженерія** - Нападник маніпулює людиною, яка знає пароль, змушуючи розкрити його.

• **Атаки грубої сили** - Нападник перебирає велику кількість можливих комбінацій у спробі вгадати пароль.

• **Прослуховування мережі** - Пароль може бути виявлений під час прослуховування та захоплення пакетів, які проходять мережею.

- Фішинг

• зловмисники надсилають шахрайсткого електронного листа, який виглядає як повідомлення від легального надійного джерела.

• змушує одержувача встановити шкідливе ПЗ на своєму пристрої або розкрити ососбисту чи фінансову інформацію.

- Спрямований фішинг

- цілеспрямована фішингова атака.
- Експлуатація вразливості - сканування з ціллю знайти вразливість для використання.
- **Крок 1** - Збір інформації про систему жертви з використанням сканування портів та методів соціальної інженерії.
- **Крок 2** - Визначення інформації, отриманої на Кроці 1.
- **Крок 3** - Пошук вразливості.
- **Крок 4** - Використання існуючого експлойта або написання нового.
- Вдосконалені постійні загрози (***advanced persistent threats - APT***) – багатофазні, довготривалі, непомітні та складні дії, спрямовані на конкретну жертву.
- зазвичай добре фінансуються
- розгортання спеціалізованого ПЗ
- Відмова в обслуговуванні DoS
-

Атака "Відмова в обслуговуванні" (DoS) - це тип мережної атаки. Результатом DoS-атаки є переривання роботи мережних сервісів для користувачів, пристроїв або застосунків.

Розрізняють два основні типи DoS-атак: DoS призводить до переривання роботи мережних сервісів.

Надмірна кількість даних - коли до мережі, хоста або програми надсилається величезна кількість даних з такою швидкістю, що вони не спроможні їх обробити. **(Overwhelming Quantity of Traffic)** Це спричиняє затримку передачі чи відгуку, або відмову пристрою, аварійне завершення роботи сервісу.

Пакети зловмисного формату - коли пакет зловмисного формату надсилається до хоста або програми і одержувач не здатен його обробити. **(Maliciously Formatted Packets)** - це Наприклад, зловмисник пересилає пакети, що містять помилки, які не можуть бути ідентифіковані програмою, або передає неправильно відформатовані пакети. Це може викликати сповільнення роботи або відмову пристрою-отримувача.

Подібна до DoS атаки, але походить від декількох скоординованих джерел.

Ботнет - мережа заражених хостів.

Зомбі - інфіковані хости.

Зомбі контролюються центрами керування.

Зомбі постійно заражають нові хости, збільшують кількість зомбі.

SEO

Пошукова оптимізація (Search Engine Optimization)

Набір методів, що використовуються для покращення рейтингу веб-сайту в пошуковій системі.

Отруєння SEO

Збільшення трафіку на шкідливі веб-сайти.

Оптимізує зловмисні сайти, щоб підвищити їх рейтинг.

Пошукова оптимізація (Search Engine Optimization, SEO) - це набір методів, що використовуються для покращення рейтингу веб-сайту в пошуковій системі. В той час як багато легальних компаній спеціалізуються на оптимізації веб-сайтів з метою їх кращого позиціонування, зловмисник може використовувати SEO для того, щоб його шкідливий

веб-сайт відображався вище у результатах пошуку. Ця техніка називається "отруєння SEO".

2.2 Ландшафт кібербезпекиВикористовує декілька методів для компрометації жертви.

Є гібридом хробаків, троянських коней, шпигунських програм, клавіатурних шпигунів, спаму та фішингових схем.

Приклади поширених змішаних атак

спам-повідомлення електронною поштою, миттєві повідомлення або легітимні сайти для розповсюдження посилань.

DDoS комбінований з фішинговими електронними листами.

Приклади: Nimbda, CodeRed, BugBear, Klez,

Slammer, Zeus/LICAT, та Conficker

Повідомте про проблему.

Будьте щирими та визнайте свою відповідальність.

Надайте детальну інформацію.

Зрозумійте, що стало причиною і сприяло зламу.

Виконайте певні кроки, щоб уникнути подібного порушення в майбутньому.

Переконайтеся, що всі системи є чистими.

Навчіть співробітників, партнерів та клієнтів як запобігти зламу у майбутньому.

Захист даних і конфіденційність.

Захист ваших даних

Пояснити, як захистити пристрої від загроз.

Пояснити, як захистити Ваші пристрої та мережу від загроз.

Описати безпечні процедури технічного обслуговування даних.

Захист конфіденційності в Інтернеті

Пояснити, як захистити свою конфіденційність.

Описати методи надійної автентифікації.

Описати основи безпечної онлайн-поведінки.

Захист ваших даних.Тримайте брандмауер увімкненим

Попереджайте неавторизований доступ до ваших даних та комп'ютерних пристроїв

Тримайте брандмауер оновленим

Використовуйте антивірусне та антишпигунське програмне забезпечення

Попереджайте неавторизований доступ до ваших даних та комп'ютерних пристроїв

Завантажуйте програмне забезпечення тільки з довірених сайтів

Регулярно оновлюйте програмне забезпечення.

Контролюйте Вашу операційну систему та браузер

Встановіть налаштування захисту на середній або вищий рівень

Регулярно оновлюйте ОС та браузер, встановлені на вашому комп'ютері

Завантажуйте та встановлюйте останні виправлення ПЗ та оновлення безпеки

Захистіть всі ваші пристрої

Захист паролем

Шифруйте дані

Зберігайте тільки необхідну інформацію

IoT пристрої

Домашня бездротова мережа

Змініть попередньо заданий SSID та пароль за замовчуванням для доступу до адміністративного інтерфейсу Wi-Fi маршрутизатора.

Вимкніть функцію широкомовного поширення SSID

Використовуйте шифрування WPA2.

Знайте про прогалину в захисті WPA2 – KRACK

Вона дозволяє зловмиснику зламувати шифрування між бездротовим маршрутизатором та клієнтами

Будьте обережні, використовуючи публічні Wi-Fi точки доступу

Уникайте доступу до або пересилання конфіденційної інформації

Використовуйте VPN тунель для запобігання підслуховування

Вимкніть Bluetooth, коли не користуєтесь ним

Щоб запобігти вторгненню до вашої домашньої бездротової мережі, потрібно змінити попередньо заданий SSID та пароль за замовчуванням для доступу до адміністративного веб-інтерфейсу. Хакери в змозі знайти інформацію для доступу, задану за замовчуванням. Як варіант, можна вимкнути на бездротовому маршрутизаторі функцію широкомовного поширення SSID, що створить додатковий бар'єр для виявлення мережі.

Крім того, слід шифрувати бездротове з'єднання, активувавши бездротову безпеку та функцію шифрування WPA2 на бездротовому маршрутизаторі. Хоча при використанні шифрування WPA2, бездротова мережа може все ще залишатися вразливою.

Щоб запобігти перехопленню вашої інформації (так званого "прослуховування") під час використання публічної бездротової мережі, використовуйте VPN-тунелі та сервіси з послугою шифрування. VPN-сервіс надає безпечний доступ до Інтернету, використовуючи зашифрований зв'язок між вашим комп'ютером та VPN-сервісом на VPN-сервері провайдера. При використанні зашифрованого VPN-тунелю, навіть в разі перехоплення, передані дані не підлягають дешифруванню.

Запобігайте тому, щоб надати злочинцям доступ до всіх ваших облікових записів за допомогою одного скомпрометованого облікового запису

- Використовуйте менеджер паролів щоб спростити запам'ятовування паролів
- Поради щодо правильного вибору пароля:
- Не використовуйте слова зі словника або імена будь-якою мовою.
- Не використовуйте поширені орфографічні помилки в словах зі словника.
- Не використовуйте імена комп'ютерів або імена облікових записів.
- Якщо це можливо, використовуйте спеціальні символи, такі як ! @ # $ % ^ & * ( )
- Використовуйте пароль з десяти або більше символів.
- Поради для вибору надійної парольної фрази:
- Вибирайте значуще для вас речення.
- Додайте спеціальні символи, такі як ! @ # $ % ^ & * ( )
- Чим фраза довша, тим краще.
- Уникайте загальних або відомих висловлювань, наприклад, текстів із популярної пісні.

Зведення нових рекомендацій NIST:

Довжина не менше 8 і не більше 64 символів.

Не використовувати поширені паролі, які легко вгадати, такі як password, abc123.

Немає вимог щодо складу, зокрема, необхідності використання великих чи малих літер або цифр.

Відсутність автентифікації на основі знань, таких як секретне запитання, маркетингові дані, історія транзакцій.

Покращити правильність введення, дозволивши користувачу бачити пароль під час набору.

Дозволяється використовувати усі друковані символи та пробіли.

Жодних підказок щодо паролю.

Відсутність періодичного або випадкового терміну дії

пароля.

Зашифровані дані можна прочитати лише за допомогою секретного ключа або пароля

Запобігайте доступу неавторизованих користувачів до вмісту даних

Що таке шифрування.

Процес перетворення інформації у таку форму, що неавторизована сторона не зможе її прочитати.

Шифрована файлова система (Encrypting File System - EFS) - це функція Windows, яка може шифрувати дані. EFS безпосередньо пов'язана з певним обліковим записом користувача. Лише користувач, який зашифрував дані, матиме доступ до них після того, як вони будуть зашифровані EFS.

Запобігайте втраті унікальних даних

Вам знадобиться додаткове сховище даних

Резервні копії мають створюватися регулярно та автоматично

Локальна резервна копія

NAS, зовнійшій жорсткий диск, CD/DVD, флеш-накопичувачі, або стрічки

Повний контроль та відповідальність за витрати на придбання та технічне обслуговування пристрою зберігання

Сервіс хмарного сховища, такий як AWS

Доступ до резервних копій зберігається доки у вас є доступ до вашого облікового запису

Необхідно ретельніше вибирати дані для яких необхідне резервне копіювання

Використовуйте доступні інструменти для незворотного видалення даних: наприклад, SDelete та Secure Empty Trash

Пристрій зберігання необхідно зруйнувати, щоб бути впевненим що дані неможливо відновити

Видаляйте онлайн версії даних

Захист конфіденційності в ІнтернетіПопулярні онлайн-сервіси використовують двофакторну автентифікацію

Необхідні ім'я користувача /пароль або PIN та додатковий токен для доступу:

Фізичний об'єкт - кредитна картка, АТМ-картка, телефон або ключ-брелок.

Біометричне сканування - відбитки пальців, відбиток долоні, розпізнавання обличчя або голосу.

Популярні онлайн-сервіси, такі як Google, Facebook, Twitter, LinkedIn, Apple та Microsoft, використовують двофакторну аутентифікацію для забезпечення додаткового рівеня захисту при вході до облікових записів.

Це протокол відкритого стандарту, який дає змогу використовувати облікові дані кінцевого користувача для отримання доступу до сторонніх програм, не розкриваючи пароль користувача.

Діє як посередник при вирішенні питання, чи дозволяти кінцевим користувачам доступ до сторонніх програм.

Надавайте в соціальних мережах тільки мінімально необхідну інформацію

Не поширюйте таку інформацію як:

Дата народження

Адреса електронної пошти

Номер телефону

Перевірте ваші налаштування у соцмережах

Email подібний до поштової листівки.

Копії електронного листа можуть бути прочитані будь ким, хто має до нього доступ.

Електронний лист передається між різними серверами.

Використовуйте режим приватного (анонимного) перегляду в браузері щоб попередити збір інформації про вашу активність в Інтернеті.

Назви анонімного режиму в популярних браузерах

Microsoft Internet Explorer: InPrivate

Google Chrome: Incognito

Mozilla Firefox: Private tab/private window

Safari: Private: Private browsing

Підсумки розділуБуло розглянуто, як захистити ваші пристрої та мережу від загроз.

Описані безпечні процедури технічного обслуговування даних.

Описано, як захистити свою конфіденційність за допомогою надійних методів аутентифікації та безпечної поведінки в Інтернеті.

Захист організації

МіжмережніекраниКонтроль або фільтрація вхідних або вихідних повідомлень в мережі або на пристрої.

Найбільш поширені типи міжмережних екранів

Міжмережний екран мережного рівня - фільрування за IP-адресами джерела та отримувача.

Міжмережний екран транспортного рівня - фільтрування на основі портів відправника та отримувача, стану з'єднання.

Міжмережний екран прикладного рівня - фільрація на основі застосунку, програми або сервісу.

Міжмережний екран на основі контексту – фільтрування за користувачем, пристроєм, роллю, типом застосунку та профілем загроз.

Проксі-сервер - фільтрування вмісту веб-запитів.

Зворотній проксі-сервер - захищає, приховує, розвантажує та розподіляє доступ до серверів.

NAT - міжмережний екран - приховує або маскує приватні адреси хостів у мережі**.**

Міжмережний екран на базі хоста - фільтрація портів та викликів системних служб на одному комп'ютері.

Фаєрвол може встановлюватися на одному комп'ютері з метою його захисту

(фаєрвол на базі хоста - host-based firewall), або він може бути автономним пристроєм, який захищає комп'ютери та всі інші пристрої у мережі (мережний фаєрвол - network-based firewall).

Процес зондування комп'ютера, сервера або іншого мережного пристрою для виявлення відкритих портів.

Номери портів призначається кожному запущеному застосунку на пристрої.

Може використовуватися як інструмент розвідки для ідентифікації операційної системи і служб.

Nmap – інструмент для сканування портів.

Загальні відповіді:

Open or Accepted - служба прослуховує порт.

Closed, Denied, або Not Listening – з'єднання з портом будуть відхилені.

Filtered, Dropped, або Blocked – відповідь від хоста відсутня.

Сканування портів - це процес зондування комп'ютера, сервера або іншого мережного пристрою для виявлення відкритих портів.

В мережах кожному застосунку, запущеному на пристрої, призначається ідентифікатор, який називається номером порта. Саме завдяки використанню цього номера на обох кінцях сеансу зв'язку необхідні дані передаються до потрібного застосунку.

Сканування портів може використовуватися зі зловмисною метою як розвідувальний інструмент для ідентифікації операційної системи та служб, що працюють на комп'ютері або іншому пристрої. А також цей підхід застосовується мережним адміністратором для перевірки політик безпеки в мережі.

Пристрої безпеки поділяють на такі основні категорії:

Маршрутизатори - можуть виконувати багато функцій міжмережного екрану, включно з фільтрацією трафіку, IPS, шифруванням та VPN.

Міжмережні екрани - можуть також підтримувати
можливості маршрутизатора, розширені функції
керування та аналітики в мережі.

IPS - виділений пристрій для запобігання вторгненням.

VPN - розроблений для безпечного зашифрованого тунелювання.

Malware/Antivirus - Cisco Advanced Malware Protection (AMP) постачається на маршрутизаторах Cisco нового покоління, міжмережних екранах, пристроях IPS, пристроях безпеки для Web та електронної пошти, а також може бути встановлений як ПЗ на комп'ютерах-хостах.

Інші пристрої безпеки - до цієї категорії належать засоби захисту веб та електронної пошти, пристрої дешифрування, сервери керування доступом клієнтів та системи контролю безпеки.

Атака нульового дня

Зловмисник намагається використовувати вразливості в частині коду програми, перш ніж їх виправить розробник.

Сканування у реальному часі від периметру до кінцевих вузлів мережі.

Активне зондування на наявність атак за допомогою міжмережного екрану та мережних пристроїв IDS/IPS.

виявлення з під'єднанням до глобальних онлайн-центрів аналізу загроз.

виявлення аномалій в мережі, використовуючи аналіз з врахуванням контексту на основі поведінки.

DDoS атаки та реагування в режимі реального часу

DDoS - це одна з найбільших загроз, яка може порушити роботу Інтернет-серверів та негативно впливає на доступність мережі.

DDoS породжується з сотен або тисяч зомбі-хостів і атака виглядає як дозволений трафік.

Деякі організації опублікували рекомендації щодо передових практик безпеки:

Оцінка ризиків – Розуміння цінності того, що ви захищаєте, допоможе вам виправдати витрати на безпеку.

Створення політики безпеки – Розробіть політику, яка чітко окреслює корпоративні правила, обов'язки та очікування.

Заходи фізичної безпеки – Обмежте доступ до комунікаційних шаф, серверних кімнат, а також забезпечте дотримання норм протипожеженої безпеки.

Ретельна перевірка персоналу – При прийомі на роботу необхідно належним чином вивчати анкетні дані та минуле співробітників.

Створення і перевірка резервних копій – Регулярно виконуйте резервне копіювання та перевіряйте можливість відновлення даних з них.

Підтримка виправлень і оновлень системи безпеки – Регулярно оновлюйте операційні системи і програмне забезпечення серверів, клієнтів і мережних пристроїв.

Використання засобів контролю доступу – Налаштуйте ролі і рівні привілеїв користувачів, а також надійну аутентифікацію.

Регулярна перевірка реагування на інциденти – Сформуйте команду реагування на інциденти і випробовуйте сценарії реагування на надзвичайні ситуації.

Впровадження інструменту моніторингу, аналізу та керування мережею – Виберіть рішення для моніторингу безпеки, яке інтегрується з іншими технологіями.

Впровадження мережних пристроїв безпеки – Використовуйте нове покоління маршрутизаторів, міжмережних екранів та інших пристроїв безпеки.

Впровадження комплексного рішення для захисту кінцевих вузлів – Використовуйте корпоративні версії антивірусів та захисних програм від усіх видів шкідливого ПЗ.

Навчання користувачів – Навчайте користувачів та співробітників процедурам безпеки.

Шифрування даних – Шифруйте всі конфіденційні дані компанії, включно з електронною поштою.

Підхід до кібербезпеки на основі поведінки Ботнет

Група ботів, з'єднаних через Інтернет

Контролюється одним зловмисником або групою

Бот

Зазвичай заражається внаслідок відвідування веб-сайту, відкривання вкладення електронної пошти або через заражений медіафайл.

Ботнет може налічувати десятки або навіть сотні тисяч ботів. Ці боти можуть активуватися для розповсюдження шкідливого програмного забезпечення, запуску DDoS-атак, розповсюдження спаму через електронну пошту або виконання атаки підбору пароля методом "грубої сили". Ботнети зазвичай контролюються командами з центру

керування.

Ланцюг кібервбивства - це атака, яка складається з наступних етапів.

1. Розвідка - Збір інформації.

2. Озброєння - Створення експлойту та засобів виконання зловмисної дії.

3. Доставка - Відправка експлойта та шкідливого навантаження на ціль.

4. Експлуатація - Виконання експлойта.

5. Встановлення - Встановлюється зловмисне ПЗ та чорні ходи.

6. Керування та контроль - віддалене керування ціллю через канал керування та контролю або сервер.

7. Дія - Виконання шкідливих дій або запуск додаткових атак на інші пристрої.

Безпека на основі аналізу поведінки (Behavior-based security) - це форма виявлення загроз, яка не покладається на відомі сигнатури загроз, а замість цього використовує інформаційний контекст для виявлення аномалій в мережі. Виявлення загроз на основі аналізу поведінки передбачає перехоплення та аналіз потоку трафіку між користувачем у локальній мережі та локальним або віддаленим пунктом призначення.

Пастки для хакерів (Honeypots)

Приманюють атакуючого, використовуючи передбачувані моделі поведінки нападника.

Захоплює, реєструє і аналізує поведінку зловмисників.

Адміністратор може отримати більше знань і покращити захист.

Архітектура рішення Cisco’s Cyber Threat Defense

Використовує виявлення на основі аналізу поведінки та індикатори атаки.

Забезпечує більшу прозорість, контекст і контроль.

Технологія NetFlow використовується для збору інформації про дані, що проходять через мережу. Інформацію, що надає NetFlow, можна порівняти з телефонним рахунком за ваш мережний трафік. Вона показує вам, які користувачі та пристрої є у вашій мережі, а також коли і як ці користувачі та пристрої отримували доступ до неї. NetFlow - важлива технологія для виявлення кібератак, яка базується на аналізі поведінки. Комутатори, маршрутизатори і міжмережні екрани, оснащені NetFlow, можуть повідомляти інформацію щодо входу, виходу та транзиту даних у мережі. Інформація надсилається до колекторів NetFlow, які збирають, зберігають та аналізують записи NetFlow.

Використовується для збору інформації про дані, що проходять через мережу.

Важливий компонент виявлення атак, що базується на аналізі поведінки.

Може визначити еталонні шаблони поведінки.

4.3 Підхід Cisco до кібербезпекиГрупа реагування на інциденти з комп'ютерної безпеки

допомагає забезпечити недоторканість даних, систем і самої компанії, шляхом проведення комплексних розслідувань інцидентів з комп'ютерної безпеки

надає проактивну оцінку загроз, планування пом'якшення наслідків, аналіз тенденцій інцидентів та огляд архітектури безпеки

Набір повторюваних запитів (звітів) щодо джерел даних інцидентів безпеки, які дають можливість їх виявити та відреагувати на них.

Що потрібно зробити.

Виявити пристрої, заражені зловмисним ПЗ.

Виявити підозрілу мережну активність.

Виявити невдалі спроби аутентифікації.

Описати і зрозуміти вхідний і вихідний трафік.

Надати зведену інформацію, включно з тенденціями, статистикою та метриками.

Забезпечити зручний та швидкий доступ до статистики та метрик.

Корелювати події з усіх відповідних джерел даних.

SIEM – система керування інформаційною безпекою та подіями безпеки (Security Information and Event Management)

Програмне забезпечення, яке збирає з пристроїв безпеки в мережі та аналізує попередження безпеки (alerts), журнали та інші дані, отримані в реальному часі та в минулому.

DLP – програмне забезпечення для запобігання втраті даних (Data Loss Prevention Software)

Зупиняє крадіжку конфіденційних даних, не дозволяє передати їх за межі мережі.

Розроблено для моніторингу та захисту даних в трьох різних країнах.

Cisco Identity Services Engine (Cisco ISE) та **TrustSec**

Використовують політику контролю доступу на основі ролей.

IDS – система виявлення вторгнень (Intrusion Detection System)

Зазвичай розміщується для роботи в режимі офлайн.

Не запобігає атакам.

Просто виявляє, реєструє і створює попередження.

IPS – система запобігання вторгненням (Intrusion Prevention System)

Блокує або відхиляє трафік на основі спрацювання правила або відповідної сигнатури.

Системи IDS/IPS

Snort

Sourcefire (Cisco)

Література

1. Кулаков А. Ф. Оценка качества программ ЭВМ. – К.: Техніка, 1984. – 167с.

2. Майерс Г. Надежность программного обеспечения / Пер. с англ. – М.: Мир, 1980. – 360 с.

3. Надежность автоматизированных систем управления / Аповмян И. О., Вайрадян А. С., Руднев Ю. П. и др. Под ред. Я. А. Хетагурова. – М.: Высшая школа, 1979. – 288 с.

4. Локазюк В. М. Контроль і діагностування обчислювальних пристроїв та систем: Навч. посібник для вузів. – Хмельницький: ТУП, 1996. – 175 с.: іл.

5. Каган Б. М., Мкртумян И. Б. Основы эксплуатации ЭВМ: Учебн. пособие для вузов / Под ред. Б. М. Кагана. – М.: Энергоатомиздат, 1983. – 376 с.

6. Иыду К. А. Надежность, контроль и диагностика вычислительных машин и систем: Учебн. пособие для вузов. – М.: Высш. шк., 1989. – 216 с.

7. Тейер Т., Липов М., Нельсон Э. Надежность программного обеспечения. – М.: Мир, 1981.

8. Mills H. D. On the Statistical Validation of Computer Programs, FSC – 72 – 6015, IBM Federal Systems Div., Gaithersburg, Md., 1972.

9. Richards F. R. Computer Software: Testing, Reliability, Models, and Quality Assurance, NRS–55RH74071A, Naval Postgraduate School, Monterey, Ca., 1974.

10. Cheung R. C. A User-Oriented Software Reliable Model. – IEEE Transactions on Software Engineering, March 1980, vol. SE-6, №2, p. 118 – 125.

11. Вентцель Е. С. Теория вероятностей. – М.: Наука, 1969. –576 с.

12. Канер Сэм и др. Тестирование программного обеспечения: Пер. с англ. / Сэм Канер, Джек Фолк, Енг Кек Нгуен. – К.: Издательство «Диа Софт», 2000. – 544 с.

13. Липаев В. В. Качество программного обеспечения. – М.: Финансы и статистика, 1983. – 263 с.

14. Зинглер К. Методы проектирования программных систем: Пер. с англ. – М.: Мир, 1985. – 328 с.

15. Кулаков А. Ф. Оценка качества программ ЭВМ. – К.: Техніка, 1984. – 167с.

16.Майерс Г. Надежность программного обеспечения / Пер. с англ. – М.: Мир, 1980. – 360 с.

17.Надежность автоматизированных систем управления / Аповмян И. О., Вайрадян А. С., Руднев Ю. П. и др. Под ред. Я. А. Хетагурова. – М.: Высшая школа, 1979. – 288 с.

18.Локазюк В. М. Контроль і діагностування обчислювальних пристроїв та систем: Навч. посібник для вузів. – Хмельницький: ТУП, 1996. – 175 с.: іл.

19.Каган Б. М., Мкртумян И. Б. Основы эксплуатации ЭВМ: Учебн. пособие для вузов / Под ред. Б. М. Кагана. – М.: Энергоатомиздат, 1983. – 376 с.

20.Иыду К. А. Надежность, контроль и диагностика вычислительных машин и систем: Учебн. пособие для вузов. – М.: Высш. шк., 1989. – 216 с.

21.Тейер Т., Липов М., Нельсон Э. Надежность программного обеспечения. – М.: Мир, 1981.

22.Mills H. D. On the Statistical Validation of Computer Programs, FSC – 72 – 6015, IBM Federal Systems Div., Gaithersburg, Md., 1972.

23.Richards F. R. Computer Software: Testing, Reliability, Models, and Quality Assurance, NRS–55RH74071A, Naval Postgraduate School, Monterey, Ca.,1974.

24.Cheung R. C. A User-Oriented Software Reliable Model. – IEEE Transactions on Software Engineering, March 1980, vol. SE-6, №2, p. 118 – 125.

25.Вентцель Е. С. Теория вероятностей. – М.: Наука, 1969. –576 с.

26.Канер Сэм и др. Тестирование программного обеспечения: Пер. с англ. / Сэм Канер, Джек Фолк, Енг Кек Нгуен. – К.: Издательство «Диа Софт»,2000. – 544 с.

27.Липаев В. В. Качество программного обеспечения. – М.: Финансы и статистика, 1983. – 263 с.

28.Зинглер К. Методы проектирования программных систем: Пер. с англ. – М.: Мир, 1985. – 328 с.