МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет міжнародних відносин
Кафедра комп'ютерних мультимедійних технологій

# КОНСПЕКТ ЛЕКЦІЙ

з дисципліни «**Адміністрування операційних систем видавничо-поліграфічного виробництва**»
Освітній ступінь ___бакалавр_____

Спеціальність: 186 Видавництво та поліграфія
Освітньо-професійна програма: Технології електронних мультимедійних видань

Укладач(і): к.т.н., с.н.с. Чаплінський Ю.П.

Конспект лекцій розглянутий та схвалений на засіданні кафедри комп'ютерних мультимедійних технологій

Протокол № ____ від «___»_____20__р.

Завідувач кафедри_____

**Модуль 1. Операційна система, функції та компоненти.**

**Лекція 1.**

**Тема лекції: Основні концепції операційних систем.**

**План лекції**

1. Поняття операційної системи, її призначення та функції..
2. Програма. Термінал. Процес. Ресурси.
3. Функціональні компоненти операційних систем. Типова структура операційної системи.

**Література**

1. Шеховцов В. А. Операційні системи – К.: Видавнича група BHV, 2005. – 576с

2. Чегренець В.М. Операційні системи та системне програмування: навчальний посібник. – К.:Київ. ун-т ім. Б. Грінченка, 2011. – 164 с..

3. Бондаренко М.Ф., Качко О.Г. Операційні системи: навч. посібник. – Х.: Компанія СМІТ, 2008. – 432 с.

**Зміст лекції**

Операційна система надає користувачеві базовий набір інструментів і середовище для зберігання даних, а також засоби управління послідовністю використання інструментів. Інтервал часу, протягом якого користувач вирішує одну або кілька послідовних завдань, користуючись при цьому засобами, наданими ОС, називається сеансом.

На початку будь-якого сеансу користувач ідентифікує себе, в кінці вказує, що сеанс закінчено. Опис послідовності використання програмних інструментів, записане на деякій формальній мові, називається завданням, а сама мова - мовою управління завданнями.

Виконання завдань в більшості операційних систем проводиться командним інтерпретатором, більш докладне визначення якого буде дано далі.

Зазвичай користувачеві надається певний інтерфейс спілкування з командним інтерпретатором, при використанні якого команди вводяться з клавіатури, а результат їх виконання виводиться на екран. Такий інтерфейс асоціюється з логічним поняттям терміналу - сукупності пристрої введення (зазвичай клавіатури) і пристрої виведення (дисплея, що виводить текстову інформацію). В даний час найбільш вживаним є графічний інтерфейс користувача.

**Програма** (в загальному випадку) - набір інструкцій процесора, що зберігається на диску (або іншому накопичувачі інформації). Для того щоб програма могла бути запущена на виконання, операційна система повинна створити середовище виконання - інформаційне оточення розв'язуваної задачі. Після цього операційна система переміщує виконуваний код і дані програми в оперативну пам'ять і ініціює виконання програми.

**Операційна система** виконує функції управління апаратними ресурсами, їх розподілу між виконуваними програмами користувача і формує середовище виконання, яка містить всі дані, необхідні для програми. Таке середовище в подальшому і буде називатися інформаційним оточенням. В інформаційне оточення входять дані і об'єкти, оброблювані операційною системою, які

впливають на виконання програми, тобто на вирішення завдання користувача. В ході подальшого викладу будуть наведені приклади інформаційного оточення різного характеру.

Використовуючи поняття програми, даних і інформаційного оточення, можна визначити поняття завдання в середовищі ОС як сукупність програми і даних, які є частиною інформаційного оточення.

Виконувана програма утворює процес. Процес являє собою сукупність інформаційного оточення і області пам'яті, що містить виконуваний код і дані програми. Зазвичай в пам'яті, контрольованої операційною системою, може одночасно працювати велика кількість процесів.

Цілком природно, що на однопроцесорних комп'ютерах можливе одночасне виконання програмного коду тільки одного процесу, тому частина процесів знаходяться в режимі очікування, а один з процесів - в режимі виконання. Процеси при цьому утворюють чергу, операційна система передає управління першому процесу в черзі, потім наступного і т. д.

Процес, що має потенційну можливість отримати вхідні дані від користувача з клавіатури і вивести результати своєї роботи на екран, називається процесом переднього плану; процес, що виконується без безпосередньої взаємодії з користувачем, - фоновим процесом.

В ході своєї роботи процеси використовують обчислювальну потужність процесора, оперативну пам'ять, звертаються до зовнішніх файлів, внутрішніми даними ядра операційної системи. Всі ці об'єкти входять до інформаційного оточення процесу і називаються ресурсами.

**Ресурсом** може бути як фізичний об'єкт, до якого ОС надає доступ, - процесор, оперативна пам'ять, дискові накопичувачі, так і логічний об'єкт, який існує тільки в межах самої ОС, наприклад таблиця виконуваних процесів або здійснювати підключення до мережі. Необхідність в управлінні ресурсами з боку ОС викликана в першу чергу тим, що ресурси обмежені (за обсягом, часу використання, кількості обслуговуваних користувачів і т.п.). Операційна система в даній ситуації або управляє обмеженнями ресурсів, запобігаючи їх вичерпання, або надає кошти обробки ситуацій, пов'язаних з вичерпанням ресурсів. Ліміти багатьох ресурсів, задані в ОС за замовчуванням, можуть змінюватися потім адміністратором системи. Прикладом такого ресурсу може служити максимальну кількість файлів, одночасно відкритих користувачем.

**Неподільні** ресурси можуть бути використані на заданому відрізку часу тільки одним процесом, при цьому інші процеси не мають доступу до такого ресурсу до повного звільнення ресурсу зайняв його процесом. Прикладом такого ресурсу може служити файл, відкритий на запис у винятковому режимі. Всі спроби використовувати цей файл іншими процесами (навіть на читання) завершуються невдачею.

**Подільні** ресурси можуть використовуватися кількома процесами. При цьому до таких ресурсів можливий одночасний доступ процесів (наприклад, до годинника, за допомогою яких визначається поточний системний час).

Деякі подільні ресурси не можуть забезпечити одночасний доступ, але дозволяють використовувати їх декількома процесами, не чекаючи моменту повного звільнення ресурсу.

Прикладом ресурсу з доступом з розділенням часу може служити процесорний час в багатозадачних ОС. У кожен квант часу виконується певне число інструкцій одного процесу, після чого управління передається наступному процесу і починається виконання його інструкцій.

Процеси, які очікують надання доступу до ресурсу, організовуються в чергу з пріоритетом. Процеси з однаковим пріоритетом отримують доступ до ресурсу послідовними квантами, при цьому деякі процеси мають більш високий пріоритет і отримують доступ до ресурсу частіше.

Зазвичай у складі операційної системи виділяють два рівня: ядро системи і допоміжні системні програмні засоби, які іноді називають системними утилітами. Ядро виконує всі функції по управлінню ресурсами системи - як фізичними, так і логічними - і розділяє доступ користувачів (програм користувачів) до цих ресурсів. За допомогою системного програмного забезпечення користувач управляє засобами, наданими ядром.

В ядро типовою операційної системи входять наступні компоненти: система управління сеансами користувачів, файлова система, система управління завданнями (процесами), система введення / виведення. Інтерфейс ядра ОС з прикладними програмами здійснюється за допомогою програмного інтерфейсу системних викликів, інтерфейс з апаратним забезпеченням - за допомогою драйверів

**Лекція 2.**

**Тема лекції: Класифікація операційних систем. Функції операційних систем. Операційні системи сімейств Unix і Windows.**

**План лекції**

1. Однокористувальницькі ОС. Багатокористувальницькі ОС.
2. Однозадачні ОС. Багатозадачні ОС.
3. Операційні системи реального часу.
4. Функції операційних систем і етапи їх розвитку.
5.Багатопроцесорні системи. Типи багатопроцесорних систем.
6.Принципи розробки розподілених систем. Сучасні архітектури розподілених систем. Кластерні системи. Grid-системи.

**Література**

1. Шеховцов В. А. Операційні системи – К.: Видавнича група BHV, 2005. – 576с

2. Чегренець В.М. Операційні системи та системне програмування: навчальний посібник. – К.:Київ. ун-т ім. Б. Грінченка, 2011. – 164 с..

3. Бондаренко М.Ф., Качко О.Г. Операційні системи: навч. посібник. – Х.: Компанія СМІТ, 2008. – 432 с.

**Зміст лекції**

Складність складових частин ядра і реалізовані ними функції в першу чергу залежать від числа одночасно обслуговуваних ОС користувачів і від числа одночасно виконуваних процесів. У зв'язку з цим розумно провести класифікацію ОЗ за цими двома параметрами і розглянути особливості компонент ядра в кожному з типів ОС.

За кількістю одночасно обслуговуваних користувачів операційні системи поділяються на однокористувальницькі (одночасно підтримується не більше одного сеансу користувача) і багатокористувальницькі (одночасно підтримується безліч сеансів користувачів). Багатокористувальницькі системи, крім забезпечення захисту даних користувачів від несанкціонованого доступу інших користувачів, надають засоби поділу загальних даних між багатьма користувачами.

Розглянемо особливості цих типів ОС більш докладно.

За кількістю одночасно виконуваних процесів операційні системи діляться на однозадачні (не більше одного працюючого процесу) і багатозадачні (безліч працюючих процесів). Одним з основних відмінностей багатозадачних систем від однозадачних є наявність засобів управління доступом до ресурсів - поділу ресурсів і блокування використовуваних ресурсів.

Крім класифікацій на основі кількості користувачів в системі і на основі кількості одночасно виконуються процесів можна ввести ще один вид класифікації операційних систем: операційні системи загального призначення і операційні системи спеціального призначення.

До класу операційних систем загального призначення відносять операційні системи, які можуть бути як однопроцесні, так і багатопроцесні, як багатокористувальницькі, так і однокористувальницькі. Це операційні системи, які працюють у складі звичайних настільних систем. Їх основне призначення полягає в тому, щоб надати користувачеві системи зручний і зрозумілий механізм управління апаратними засобами обчислювальної системи, обробляти запити користувача, максимально ізолюючи його від низькорівневих операцій і інтерфейсів. Такі операційні системи орієнтуються в першу чергу на простоту застосування, оскільки їх основними користувачами зазвичай не є програмісти, а користувачі середньої або низької кваліфікації. Найчастіше такі користувачі можуть навіть не уявляти, що за красивою картинкою анімованих шпалер і різноманітністю всіляких красот графічних інтерфейсів ховається потужний програмний комплекс, керуючий всім апаратним комплексом комп'ютера. До подібних операційних систем відносяться настільні версії операційних систем сімейства Windows, Linux, Apple iOS і т.д. Іншими словами, це ті операційні системи, з якими користувач може мати справу як на роботі для виконання якихось необхідних завдань, так і вдома для розваги.

На противагу операційним системам загального призначення ОС спеціального призначення відносно рідко використовуються в повсякденному житті. Основні користувачі таких операційних систем - кваліфіковані розробники. Подібні операційні системи призначені для управління ресурсами спеціальних обчислювальних систем. Найчастіше такі системи є вбудованими, тобто системами, які повинні працювати, будучи вбудованими безпосередньо в пристрій, яким вони керують. До них можна віднести такі операційні системи, як Android, iOS, Windows PE і т. д.

**Лекція 3.**

**Тема лекції: Логічна та фізична організація файлових систем.**

**План лекції**

1. Поняття файла і файлової системи.

2. Типи файлів. Принципи організації файлових систем.

3. Організація інформації у файловій системі. Розділи. Каталоги. Атрибути файлів. Операції над файлами і каталогами. Розміщення інформації у файлових системах.

4. Фізична організація розділів на диску. Реалізація файлових систем.

**Література**

1. Шеховцов В. А. Операційні системи – К.: Видавнича група BHV, 2005. – 576с

2. Чегренець В.М. Операційні системи та системне програмування: навчальний посібник. – К.:Київ. ун-т ім. Б. Грінченка, 2011. – 164 с..

3. Бондаренко М.Ф., Качко О.Г. Операційні системи: навч. посібник. – Х.: Компанія СМІТ, 2008. – 432 с.

**Зміст лекції**

Файл (іноді його ще називають набором даних) можна розглядати як збережені на диску дані, що мають унікальне ім'я, видиме користувачеві. Дані файлу мають формат, призначений для їх використання прикладними програмами користувача. Так, книга може зберігатися на диску у вигляді файлу з ім'ям book.txt в текстовому форматі 1x1.

Імена файлів є символьні рядки обмеженої довжини (зазвичай до 8 або до 255 символів), які служать для ідентифікації даних, що зберігаються в файлі. Як правило, існує ряд символів, заборонених до використання в іменах файлів.

Операційна система надає прикладним програмам інтерфейс доступу до файлів, але при цьому розглядає дані, з яких складаються файли, по-своєму. Уявімо собі іграшковий конструктор, з якого можна зібрати модель автомобіля. Точно так же операційна система збирає файли з окремих «деталей» - блоків.

При цьому так само, як в конструкторі, до якого прикладена схема зборки, ОС керується правилами складання окремих блоків. Такі сконструйовані набори блоків отримали назву наборів даних. Набір даних - це більш низькорівневе представлення даних, ніж файл. Його організація багато в чому визначається властивостями носія даних.

Набір даних має унікальне ім'я, так само як і файл, містить всі дані, які містить файл, але структура цих даних визначається ядром операційної системи.

Крім того, в набір даних може входити службова інформація, недоступна звичайним програмам.

Весь дисковий простір, що використовується файловою системою, розбивається на окремі блоки - кластери (зазвичай мають розмір 1, 2, 4, 8 або 16 Кбайт). Кожен кластер має свій номер і зберігає або дані користувача, або службову інформацію. Ця службова інформація використовується, в тому числі, і для складання блоків в набори даних. Розмір кластера встановлюється при створенні файлової системи.

Недоліком такого методу організації даних є те, що список номерів блоків у великих файлів може займати більше одного кластера. В результаті розмір файлу виходить обмеженим. В UNIX-системах існують різні методи обходу

цього обмеження, наприклад службові блоки можна організувати в дерево, при цьому кожен службовий блок зберігає послідовність блоків з даними і службові блоки наступного рівня дерева.

Інший підхід до організації блоків даних полягає в тому, що набори даних розміщуються в послідовних кластерах, в цьому випадку в службовому блоці досить зберігати номери першого і останнього кластерів (рис. 2.2). При частих змінах даних цей метод організації незручний. При збільшенні розміру набору даних потрібно або завжди мати достатню кількість вільних блоків після останнього, або кожен раз переміщати набір даних на вільне місце.

Ще один спосіб організації полягає у виділенні в кожному блоці невеликий службової області, в якій зберігається номер наступного блоку набору даних (рис. 2.3). Таким чином, набір даних організовується в лінійний список, а в службовому блоці досить зберігати номер першого блоку з одними даними.

Угоди про структуру наборів даних на носії є частиною файлової системи, яка складається з трьох компонентів:

1) угоди про структуру зберігання даних на носії - визначення типів інформації, яка може бути розміщена на носії (наприклад, призначена для користувача / службова інформація), а також набору правил, згідно з якими дані розміщуються на носії;

2) угоди про правила обробки даних - набір правил, згідно з якими дані обробляються, наприклад «файл повинен бути відкритий до того, як в нього будуть записані дані»;

3) процедури обробки даних, які входять до складу ядра ОС і підкоряються певним вище угодами.

Для зберігання наборів даних на диску в UNIX-системі використовується наступний метод: кожен набір даних, що зберігається на диску, розбивається на блоки (розмір одного блоку зазвичай кратний розміру сектора диска). Для того щоб забезпечити цілісність даних службові блоки містять посилання на блоки, що входять до нього. Ці службові блоки організовані в деревоподібну структуру.

Коренем дерева є індексний блок (i-node), в якому зберігаються атрибути файлів (дата модифікації, дата останнього звернення, права доступу, тип файлу і т.п.) і масив посилань на блоки даних.

Масив посилань має обмежений розмір. У тому випадку, якщо кількість блоків перевищує обсяг масиву, створюється одинарний контрольний блок, на який і починає посилатися один з елементів i-node. Сам одинарний контрольний блок посилається на блоки даних.

Оскільки розмір масиву посилань в посилальному блоці також обмежений, в разі заповнення всіх посилань в індексному блоці і одинарному посилальному блоці створюються подвійні посилальні блоки. Посилання на існуючі одинарні блоки переносяться в подвійні, а індексний блок починає посилатися на подвійні блоки.

Однією з найстаріших і найпопулярніших файлових систем є FAT. Розроблена спочатку для операційної системи MS DOS, дана файлова система отримала широке поширення завдяки простоті, швидкості та ефективності.

Дана файлова структура мала дуже просту структуру. На початку розташовувався завантажувальний сектор, потім - дві таблиці розміщення файлів (File Allocation Table - FAT, які і дали назву файлової системи, причому друга FAT є повною копією першої і зберігається на випадок її пошкодження), кореневий каталог і файли.

Файлова система NTFS розроблялася з урахуванням властивих файловим системам FAT недоліків. Основні властивості, властиві цій файлової системи, такі:

- надійність і відновлюваність. NTFS спроектована таким чином, що операції введення / виводу є транзакції. Ці транзакції неподільні, тобто необхідна операція або завершується повністю, або не виконується взагалі. Це дозволяє безболісно здійснювати операції відкоту стану файлової системи в разі збою;
- безпеку і контроль доступу до даних. Файлова система NTFS трактує файли і каталоги як захищені об'єкти відповідно до загальної архітектурою безпеки Windows. Це дозволяє обмежувати доступ до даних певним учасникам та групам;
- ефективний розподіл дискового простору. Файлова система NTFS підтримує цілий ряд механізмів управління дисковим простором, таких як підтримка стиснення каталогів і дисків, підтримка роботи з розрідженими файлами, наявність спеціалізованого API для дефрагментації даних;
- підтримка POSIX, жорстких посилань, довгих імен, шифрування даних, підтримка роботи з логічними томами розміром більше 4 Гбайт.

**Лекція 4.**

**Тема лекції:Керування оперативною пам'яттю.**

**План лекції**

1. Стратегії розподілу пам'яті між процесами. Алгоритми для реалізації стратегій.
2. Механізми управління пам'яттю.
3. Фізична пам'ять.
4. Віртуальна пам'ять
5. Реалізація керування пам'яттю у Windows 10, Linux та Android.

**Література**

1. Шеховцов В. А. Операційні системи – К.: Видавнича група BHV, 2005. – 576с

2. Чегренець В.М. Операційні системи та системне програмування: навчальний посібник. – К.:Київ. ун-т ім. Б. Грінченка, 2011. – 164 с..

3. Бондаренко М.Ф., Качко О.Г. Операційні системи: навч. посібник. – Х.: Компанія СМІТ, 2008. – 432 с.

**Зміст лекції**

Пам'ять, що частіше називають оперативною пам'яттю, є одним з найважливіших ресурсів будь-якої обчислювальної системи. Тому підсистема управління пам'яттю є найважливішою функцією операційної системи. Для того щоб програми могли бути виконані процесором обчислювальної системи,

необхідно, щоб виконуваний код програми був завантажений в основну пам'ять.

Історично швидкість обробки даних в основній пам'яті істотно швидше за швидкість обробки даних на зовнішніх носіях, на яких зазвичай зберігаються програми. Тому при розробці операційної системи намагаються оптимізувати процес завантаження і виконання програм таким чином, щоб кожен виконує процесу було виділено достатню кількість основної пам'яті. Це дозволяє, по-перше, прискорити виконання поточного процесу і, по-друге, не сповільнювати роботу інших процесів і підсистем операційної системи.

Проблеми виникають тоді, коли операційна система повинна розподілити обмежений обсяг оперативної пам'яті між декількома завданнями (процесами). При цьому кожен отримує тільки таку частину, яка достатня для виконання поточних обчислень, але менше всієї пам'яті, необхідної для роботи програми. Тоді неактивна частина завдання знаходиться у зовнішній пам'яті і підкачуються звідти в міру необхідності.

Звідси випливають основні цілі, яких необхідно досягти при розробці підсистеми управління пам'яттю в операційній системі:

1) скоротити час доступу до основної пам'яті;

2) збільшити розмір доступної процесам основний пам'яті;

3) збільшити ефективність доступу процесів до основної пам'яті.

Основні завдання, які вирішує підсистема управління пам'яттю, такі:

1) виділяти процесам основну пам'ять;

2) відображати адресне простір процесу на виділену область основний пам'яті;

3) мінімізувати час доступу даними, використовуючи доступний обсяг основної пам'яті.

Для розв’язання цих завдань в умовах, коли відбувається динамічне виділення пам'яті, менеджер пам'яті повинен підтримувати список всіх блоків пам'яті, які можуть бути виділені процесам. Цей список може підтримуватися за допомогою практично будь-якої структури даних, що мають в основі пов'язаний список. У найпростішому випадку така структура може являти собою список дескрипторів блоків пам'яті, де кожен дескриптор містить покажчик на відповідний блок пам'яті і його розмір. Менеджер пам'яті підтримує цілісність даної структури і вставляє в список або видаляє з нього елементи в певному порядку, відповідно до прийнятої стратегії управління пам'яттю. Існує кілька стратегій розподілу пам'яті між процесами, що конкурують між собою за право володіти деяким об'ємом пам'яті:

1) розподіл на основі стратегії «найбільш підходящу ділянку»;

2) розподіл на основі стратегії «найменш підходящу ділянку»;

3) розподіл на основі стратегії «першу підходящу ділянку»;

4) розподіл на основі стратегії «наступну підходящу ділянку».

Фізична пам'ять є однією із складових частин апаратних засобів будь-обчислювальної системи, яка використовується для завантаження операційної системи і виконання призначених для користувача і системних процесів. Сьогодні ця пам'ять є декілька модулів пам'яті, в свою чергу, складаються з декількох чіпів RAM. Ця пам'ять є досить швидкою (по швидкості вона

поступається лише реєстрової та кеш-пам'яті процесора), але при цьому досить дорогий, і максимальний обсяг цього виду пам'яті зазвичай сильно обмежений.

Через ці обмеження фізична пам'ять може бути дуже швидко вичерпана, якщо одночасно виконується велика кількість процесів. Для вирішення даної проблеми використовується так звана віртуальна пам'ять. Віртуальна пам'ять дає можливість об'єднати кілька видів пам'яті (пам'ять з довільним доступом RAM, більш повільну дискову пам'ять і т.д.), дозволяючи розробляти досить великі програми. Вся пам'ять для цих програм представляється у вигляді єдиного масиву віртуальної пам'яті, яка поводиться як звичайна фізична пам'ять, але великого обсягу. Таким чином, віртуальна пам'ять дозволяє відокремити логічну структуру пам'яті від її апаратної організації.

**Модуль 2. Процеси та взаємодії між процесами. Взаємодія з користувачем. Захист інформації.**

**Лекція 1.**

**Тема лекції: Процеси. Завдання. Взаємодії між процесами.**

**План лекції**

1. Поняття Процеси, Завдання.
2. Моделі процесів і потоків. Складові елементи процесів і потоків..
3. Багатопотоковість та її реалізація. Планування процесів і потоків. Взаємодія потоків.
4. Міжпроцесова взаємодія.
5. Керування процесами і потоками в Windows 10, Linux та Android.

**Література**

1. Шеховцов В. А. Операційні системи – К.: Видавнича група BHV, 2005. – 576с

2. Чегренець В.М. Операційні системи та системне програмування: навчальний посібник. – К.:Київ. ун-т ім. Б. Грінченка, 2011. – 164 с..

3. Бондаренко М.Ф., Качко О.Г. Операційні системи: навч. посібник. – Х.: Компанія СМІТ, 2008. – 432 с.

**Зміст лекції**

Програма в загальному випадку - набір інструкцій процесора, представлений у вигляді файлу. Для того щоб програма могла бути запущена на виконання, ОС повинна спочатку створити оточення або середу виконання завдання, що включає в себе ресурси пам'яті, можливість доступу до системи введення / виводу і т. п. Сукупність оточення і області пам'яті, що містить код і дані виконуваної програми , називається процесом. Процес в ході своєї роботи може перебувати в різних станах, в кожному з яких він особливим чином використовує ресурси, що надаються йому операційною системою.

Два основні стани: виконання процесу в режимі завдання, коли виконується власний програмний код, і виконання в режимі ядра, коли процес виконує системні програми, що знаходяться в адресному просторі ядра операційної системи.

Для управління процесами операційна система використовує системні дані, які існують протягом усього часу виконання процесу. Вся сукупність цих

даних утворює контекст процесу. Контекст визначає стан процесу в заданий момент часу.

З точки зору структур, підтримуваних ядром ОС, контекст процесу включає в себе наступні складові:

■ контекст, що призначений для користувача - вміст пам'яті коду процесу, даних, стека, що розділяється пам'яті, буферів введення / виведення;

■ регістровий контекст - вміст апаратних регістрів (реєстр лічильника команд, регістр стану процесора, регістр покажчика стека і регістри загального призначення);

■ контекст системного рівня - структури даних ядра, що характеризують процес.

Контекст системного рівня складається з статичної та динамічної частин.

В статичну частину входять дескриптор процесу и область користувача (U-область).

Дескриптор процесу включає в себе системні дані, які використовуються операційною системою для ідентифікації процесу. Ці дані застосовуються при побудові таблиці процесів, що містить інформацію про всі виконуваних в поточний момент часу процесах. Дескриптор процесу включає в себе наступну інформацію:

■ розташування і займаний процесом обсяг пам'яті - зазвичай вказується у вигляді базового адреси і розміру при безперервному розподілі процесу в пам'яті або списку початкових адрес і розмірів блоків пам'яті, якщо процес розташовується в пам'яті декількома фрагментами;

■ ідентифікатор процесу PID (Process IDentifier) - унікальне ціле число, що знаходиться зазвичай в межах від 1 до 65 535, яке присвоюється процесу в момент його створення;

■ ідентифікатор батьківського процесу PPID (Parent Process IDentifier) - ідентифікатор процесу, який породив даний. Всі процеси в UNIX-системах породжуються іншими процесами (наприклад, при запуску програми на виконання з командного інтерпретатора її процес вважається породженим від процесу командного інтерпретатора);

■ пріоритет процесу - число, що визначає відносну кількість процесорного часу, яке може використовувати процес. Процесам з більш високим пріоритетом управління передається частіше;

■ U-область містить наступну інформацію:

■ покажчик на дескриптор процесу;

■ лічильник часу, протягом якого процес виконувався (тобто використовував процесорний час) в режимі користувача і режимі ядра;

■ параметри останнього системного виклику;

■ результати останнього системного виклику;

■ таблиця дескрипторів відкритих файлів;

■ максимальні розміри адресного простору, займаного процесом;

■ максимальні розміри файлів, які може створювати процес.

Динамічна частина контексту системного рівня - це один або кілька стеків, які використовуються процесом при його виконанні в режимі ядра.

Для перегляду таблиці процесів може застосовуватися команда ps. Будучи виконаної без параметрів командного рядка, вона виведе всі процеси, запущені

поточним користувачем. Досить повну для практичного використання інформацію про таблиці процесів можна отримати, викликавши ps з параметрами aux; при цьому буде виведена інформація про процеси всіх користувачів (a), частина даних, що входять в дескриптор і контекст процесу (i), а також будуть виведені процеси, для яких не визначено термінал (x):

При роботі процесу надається доступ до ресурсів ОС - оперативної пам'яті, файлів, процесорного часу.

Для розподілу ресурсів між процесами і управління доступом до ресурсів використовується планувальник завдань, що входить до складу ядра операційної системи, а також застосовуються механізми захисту пам'яті і блокування файлів і пристроїв.

Основна функція планувальника завдань - балансування навантаження на систему між процесами, розподіл процесорного часу відповідно до пріоритету процесів.

Механізм захисту пам'яті забороняє доступ процесу до області оперативної пам'яті, зайнятої іншими процесами (за винятком випадку взаємодії між процесами з використанням загальної пам'яті).

Механізм блокування файлів і пристроїв працює за принципом унікального доступу - якщо який-небудь процес відкриває файл на запис, то на цей файл ставиться блокування, що унеможливлює запис в цей файл даних іншим процесом.

У значної частини складних програм в даний час присутня та чи інша форма взаємодії між процесами. Це обумовлено природною еволюцією підходів до проектування програмних систем, яка послідовно пройшла наступні етапи:

1) монолітні програми, що містять в своєму коді всі необхідні для своєї роботи інструкції.

2) модульні програми, що складаються з окремих програмних модулів з чітко визначеними інтерфейсами викликів.

3) програми, що використовують межпроцесну взаємодію.

Для реалізації процесів, які вирішують загальну задачу, операційна система повинна бути забезпечена засобами взаємодії між ними. Надання коштів взаємодії - завдання операційної системи, а не прикладних програм, оскільки необхідно виключити вплив прикладних програм на самі механізми обміну, а крім того, підтримка механізмів обміну може зажадати доступу до ресурсів, недоступним звичайним процесам користувача.

Типові механізми взаємодії між процесами призначені для вирішення наступних завдань:

- передача даних від одного процесу до іншого;
- спільне використання одних і тих же даних кількома процесами;
- повідомлення про зміну стану процесів.

Сучасні операційні системи реалізують сім основних механізмів взаємодії між процесами:

Переривання. Спочатку механізм переривань використовувався в операційних системах для оповіщення. У певні моменти часу апаратний пристрій сповіщає про настання деякої події (готовності до дії, збої, завершення передачі блоку даних). Кількість варіантів таких подій може бути досить

великим, і всі вони повинні бути помітні операційною системою. Таке повідомлення про готовність отримало назву переривання, оскільки в момент отримання переривання операційна система повинна призупинити виконання поточних завдань і зреагувати на яке надійшло переривання. Реакція на переривання, як правило, полягає у виконанні програмного коду, який знаходиться в пам'яті, що адресується операційною системою. Для кожного системного виклику ОС також активує переривання. Наприклад, відповідне переривання активується при виведенні тексту на екран терміналу.

Сигнали. Механізм сигналів має багато спільних рис з механізмом програмних переривань. Він також призначений для того, щоб процеси могли бути сповіщені про деякі події. Основна відмінність сигналів від переривань полягає в тому, що за допомогою сигналів один процес оповіщає інші процеси, а не операційну систему. На відміну від переривання сигнал повинен мати точку призначення - процес-одержувач.

Повідомлення. Механізм повідомлень призначений для обміну даними між процесами. Для цього створюється спеціальна чергу повідомлень, підтримувана операційною системою. У черзі накопичуються дані, які записуються в неї процесами. Накопичені в черзі дані можуть бути «прочитані» іншими процесами, таким чином відбудеться передача даних від одного процесу до іншого.

Іменовані канали. Іменовані канали також призначені для обміну даними між процесами. Однак в якості черзі використовується не область оперативної пам'яті, а файл спеціального виду. Процеси можуть записувати в цей файл і зчитувати з нього дані. При цьому операційна система підтримує рівноправний доступ до іменовані канали для всіх процесів, що використовують його для обміну даними.

Гнізда (сокети). Механізм взаємодії між процесами за допомогою сокетів в першу чергу забезпечує взаємодію процесів, які виконуються на різних комп'ютерах. Кожен процес створює гніздо, в яке може записувати дані і зчитувати їх з нього. Гнізда пов'язані один з одним через мережеві протоколи.

Процеси при взаємодії один з одним обмінюються даними через пов'язані гнізда, а ядро системи передає дані процесів через мережу, як правило, використовуючи стек протоколів TCP/IP і драйвери мережевих адаптерів.

Спільна пам'ять. Механізм загальної пам'яті полягає в тому, що в адресний простір процесів відображається один і той же ділянку фізичної пам'яті, що адресується операційною системою. Використовуючи цей загальний ділянку пам'яті, процеси можуть обмінюватися даними без участі ядра операційної системи, що значно збільшує швидкість роботи.

Семафори. Одна з основних проблем при використанні спільної пам'яті полягає в тому, що необхідно запобігати конфліктним ситуаціям одночасного доступу. Така ситуація може виникати при одночасному записі даних в одну ділянку пам'яті декількома процесами. У цьому випадку пам'ять буде містити дані останнього записуючого процесу, інші будуть втрачені. Інша конфліктна ситуація виникає в момент читання одним процесом даних з області пам'яті, що модифікується іншим процесом. В цьому випадку лічені дані можуть містити як стару, так і частково оновлену інформацію. Для запобігання конфліктним

ситуаціям застосовується механізм семафорів - спеціальних прапорів, що вказують на можливість використання тієї чи іншої ділянки пам'яті.

**Лекція 2.**

**Тема лекції:Взаємодія з користувачем в операційних системах.**

**План лекції**

1. Поняття Користувач.
2. Опис користувача системи.
3. Групи користувачів.
4. Графічний інтерфейс користувача.
5. Інтерфейс віконної та графічної підсистеми Windows 10, Linux та Android.

**Література**

1. Шеховцов В. А. Операційні системи – К.: Видавнича група BHV, 2005. – 576с

2. Чегренець В.М. Операційні системи та системне програмування: навчальний посібник. – К.:Київ. ун-т ім. Б. Грінченка, 2011. – 164 с..

3. Бондаренко М.Ф., Качко О.Г. Операційні системи: навч. посібник. – Х.: Компанія СМІТ, 2008. – 432 с.

**Зміст лекції**

Як правило користувачем системи розглядається людина. Однак це вірно далеко не завжди - в якості користувача системи може виступати будь-який об'єкт, що володіє правами на використання об'єктів, які надає операційна система. Прикладом таких прав можуть служити права доступу до певних файлів, запуску на виконання програм, доступу до пристрою друку.

У ролі користувачів можуть виступати запускаються програми з певним набором прав. Такі програми запускаються від імені конкретного користувача, який в такому випадку іменується псевдокористувачем.

Крім звичайних користувачів і псевдокористувачів в системі ще існує як мінімум один привілейований користувач, що виконує функцію системного адміністратора. Цей користувач має виключне право доступу до всіх ресурсів, що надаються операційною системою.

Кожен користувач системи має унікальне в межах даної системи облікове ім'я (логін). Однак ім'я присвоюється користувачу тільки для його зручності - операційна система розрізняє користувачів по їх унікальним ідентифікаторів - UID (User IDentifier). Ідентифікатор UID є цілим числом, більше або рівне нулю. UID унікальний, як і облікове ім'я користувача.

Крім UID і логіна для кожного користувача визначається набір атрибутів, що містять системну і довідкову інформацію: пароль, паспортне ім'я, шлях до домашнього каталогу, повне ім'я виконуваного файлу командного інтерпретатора за замовчуванням. Ця інформація зберігається в файлі / etc / passwd. Інформація про кожного користувача знаходиться в окремому рядку, атрибути розділені двокрапкою «:». Послідовність атрибутів наступна: облікове ім'я, пароль (в зашифрованому вигляді), ідентифікатор користувача (UID),

ідентифікатор групи (GID), паспортне ім'я, шлях до домашнього каталогу і повне ім'я командного інтерпретатора.

Користувач завжди є членом однієї або декількох груп користувачів. Навіть якщо користувач є єдиною людиною, яка має доступ до системи, він є членом, як мінімум групи «Користувачі системи» (зазвичай з ім'ям users) або групи «Адміністратори» (зазвичай з ім'ям root).

Група користувачів - це множина користувачів, що задається у вигляді списку. Об'єднання користувачів в групи зазвичай відбувається за принципом розмежування завдань, що виконуються користувачами. Так, в окрему групу зазвичай виділяються адміністратори системи.

Кожна група має унікальне облікове ім'я групи і унікальний ідентифікатор групи GID (Group IDentifier).

Інформація про групи, визначених у системі, зберігається в файлі / etc / group. Інформація про кожну групу знаходиться в окремому рядку, атрибути групи розділені символами двокрапки «:». Послідовність атрибутів наступна: облікове ім'я групи, прапор стану групи (зазвичай символ «*»), ідентифікатор групи (GID), список користувачів, що входять в групу, перерахованих через кому.

Графі́чний інтерфе́йс кори́стувача — інтерфейс між комп'ютером і його користувачем, що використовує піктограми, меню, і вказівний засіб для вибору функцій та виконання команд. Зазвичай, можливе відкриття більше, ніж одного вікна на одному екрані. ГІК — система засобів для взаємодії користувача з комп'ютером, заснована на представленні всіх доступних користувачеві системних об'єктів і функцій у вигляді графічних компонентів екрану (вікон, значків, меню, кнопок, списків і т. п.). При цьому, на відміну від інтерфейса командного рядка, користувач має довільний доступ (за допомогою клавіатури або пристрою координатного введення типу «миша») до всіх видимих екранних об'єктів. Вперше концепція ГІК була запропонована вченими з дослідницької лабораторії Xerox PARC в 1970-х, але отримала комерційне втілення лише в продуктах корпорації Apple Computer. В даний час ГІК є стандартною складовою більшості доступних на ринку операційних систем і застосунків. Більшість сучасних операційних систем мають графічний інтерфейс користувача (ГІК, англ. Graphical User Interfaces, GUIs). Багато операційних систем дозволяють користувачеві встановити будь-який графічний інтерфейс на власний вибір.

**Лекція 3.**

**Тема лекції: Захист інформації в операційних системах.**

**План лекції**

1.Основні завдання забезпечення безпеки.

2. Базові поняття криптографії. Поняття криптографічного алгоритму і протоколу.

3. Аутентифікація. Принципи аутентифікації і керування доступом. Основи аутентифікації.

4. Керування доступом. Локальна безпека даних. Мережна безпека даних.

5. Аутентифікація та керування доступом у Windows 10 та Linux.

**Література**

1. Шеховцов В. А. Операційні системи – К.: Видавнича група BHV, 2005. – 576с

2. Чегренець В.М. Операційні системи та системне програмування: навчальний посібник. – К.:Київ. ун-т ім. Б. Грінченка, 2011. – 164 с..

3. Бондаренко М.Ф., Качко О.Г. Операційні системи: навч. посібник. – Х.: Компанія СМІТ, 2008. – 432 с.

**Зміст лекції**

Захист в операційних системах виконує такі основні функції:

1. Ідентифікація та аутентифікація. Жоден користувач не може почати роботу з ОС, не ідентифікувавши себе і не надавши системі інформації, що підтверджує, що користувач дійсно є тим, ким він себе заявляє.

2. Розмежування доступу. Кожен користувач системи має доступ тільки до тих об'єктів ОС, до яких йому надано доступ відповідно до поточної політики безпеки.

3. Аудит ОС реєструє в спеціальному журналі події, потенційно небезпечні для підтримки безпеки системи.

4.Управління політикою безпеки. Політика безпеки повинна постійно підтримуватися в адекватному стані, тобто повинна гнучко реагувати на зміни умов функціонування ОС. Управління політикою безпеки здійснюється адміністраторами системи з використанням відповідних засобів, вбудованих в ОС.

5. Криптографічні функції. Захист інформації немислима без використання криптографічних засобів захисту. Шифрування використовується в ОС при зберіганні і передачі по каналах зв'язку паролів користувачів і деяких інших даних, критичних для безпеки системи.

6. Мережеві функції. Сучасні ОС, як правило, працюють не ізольовано, а в складі локальних і / або глобальних комп'ютерних мереж. ОС комп'ютерів, що входять в одну мережу, взаємодіють між собою для вирішення різних завдань, у тому числі і завдань, що мають пряме відношення до захисту інформації.

Підсистема захисту зазвичай не представляє собою єдиний програмний модуль. Як правило, кожна з перерахованих функцій підсистеми захисту вирішується одним або декількома програмними модулями. Деякі функції вбудовуються безпосередньо в ядро ОС. Між різними модулями підсистеми захисту повинен існувати чітко визначений йнтерфейс , використовуваний при взаємодії модулів для вирішення спільних завдань.

У таких ОС, як Windows, підсистема захисту чітко виділяється в загальній архітектурі ОС, в інших, як UNIX, захисні функції розподілені практично по всіх елементах ОС. Зазвичай підсистема захисту ОС допускає розширення додатковими програмними модулями.

Головні захисні механізми ОС Windows

− можливість захищеного входу в систему з ідентифікацією користувачів;

− контроль доступу до ресурсів по категоріях користувачів;

− аудит.

Головні захисні механізми ОС сімейства UNIX

− ідентифікації й аутентифікації користувача при вході в систему;

− розмежуванні прав доступу до файлової системи, у базі якого лежить реалізація дискреційної моделі доступу;

− аудит, іншими словами реєстрація подій.

Для різних клонів ОС сімейства UNIX можливості пристроїв захисту можуть некардинально різнитися, але будемо розглядати ОС UNIX у загальному випадку, без врахування якихось незначущих особливостей окремих ОС цього сімейства.

Основу існуючих принципів контролю доступу до ресурсів, становить завдання прав і реалізація розмежувань на підставі певних правил доступу суб'єктів до об'єктів, зокрема, до файлових об'єктів. Сам процес, і може нести в собі уразливість по наступних причинах:

Несанкціоновані (сторонні) процеси. Це процеси, які не потрібні користувачеві для виконання своїх службових обов'язків і можуть несанкціоновано встановлюватися на комп'ютер (локально, або віддалено) з різними цілями (наприклад, так звані, шпигунські програми), у тому числі, і з метою здійснення НСД до інформації.

Критичні процеси - ті процеси, які запускаються в системі із привілейованими правами, наприклад, під обліковим записом System або root, а також процеси, які найбільше ймовірно можуть бути піддані атакам, у першу чергу, це мережеві служби. Атаки на подібні процеси ( як правило, мережні атаки) найбільш критичні, що пов'язане з можливістю розширення привілеїв, у межі – одержання повного керування системою (тому що ОС не забезпечує можливості в необхідному обсязі встановлювати розмежування прав доступу для користувача System або root). У якості зауваження відзначимо, що на сьогоднішній день суттєво зросла частка атак, спрямованих на уразливості не властиво в ОС, а на уразливості застосувань. Однак застосування користуються сервісами ОС, які повинні захищатися ОС, тобто уразливість застосування при коректній реалізації механізмів захисту ОС, не повинні приводити до можливості несанкціонованого доступу до інформації;

Скомпрометовані процеси – процеси, що містять помилки (уразливості), що стали відомими, використання яких дозволяє здійснити НСД до інформації.

Віднесення даних процесів в окрему групу обумовлене тим, що з моменту виявлення уразливості й до моменту усунення її розроблювачем системи або застосування, може пройти кілька місяців. Протягом цього часу в системі перебуває відома уразливість, тому система не захищена;

Процеси, що апріорі володіють недекларованими (документально не описаними) властивостями. До цієї групи ми віднесемо процеси, що є середовищем виконання (насамперед, це віртуальні машини, що є середовищем виконання для скриптів і аплетів, і офісні застосування, що є середовищем виконання для макросів).

Мережевий фільтр (екран) призначений для заборони чи дозволу користування певними сервісами зовнішньої мережі комп’ютерові чи їх групі, а також навпаки. Перш за все мережеві фільтри поділяються на персональні та міжмережні. Персональний мережевий фільтр призначений для захисту одного комп’ютера, в той час коли міжмережні використовують для цілої робочої групи чи підмережі. Мережеві фільтри бувають програмні, а також апаратні

(тільки міжмережні). Апаратні мережеві фільтри підключають у вигляді окремого пристрою до мережі.

У багатьох операційних системах існують засоби обмеження доступу до файлів. В одного користувача ОС зазвичай обмежується доступ до файлів ядра операційної системи.

У багатокористувацьких ОС доступ обмежується за допомогою визначення прав доступу того чи іншого користувача до файлів. Права доступу визначають можливість виконання тієї чи іншої операції над файлом.

Взагалі кажучи, термін «права доступу до файлу» не зовсім коректний. Права доступу визначаються для наборів даних, що зберігаються на диску. Для всіх файлів, пов'язаних з цим набором даних, визначені ті ж самі права доступу, що і для набору даних. Для всіх файлів, пов'язаних з одним набором даних, ці права однакові.

Доступ користувачів до файлів визначається елементами трійки суб'єкт-об'єкт-право доступу. При цьому під суб'єктом розуміється користувач, який звертається до об'єкта, або певне безліч користувачів, під об'єктом - файл або каталог, а під правом доступу - дозвіл або заборона на виконання суб'єктом тієї чи іншої операції над об'єктом.

Права доступу до файлів і каталогів можуть змінюватися з плином часу. Це може бути пов'язано, наприклад, з тим, що старий власник файлу передав права на його зміну новому власнику, або з тим, що файл, який раніше був доступний тільки одному користувачеві, тепер доступний всім.

Права доступу до файлів і каталогів можуть бути змінені за допомогою команди chmod. Змінювати права доступу може тільки власник файлу або адміністратор системи. В якості аргументів команді chmod передаються специфікатор доступу і імена файлів, для яких визначається доступ.

Специфікатор доступу - рядок символів, що визначає права доступу до файлу. Специфікатор складається з трьох наступних один за одним елементів: суб'єкта, права доступу та операції над правом доступу.

Обмеження прав доступу до файлів і каталогів значно змінюють інформаційне оточення, в якому виконуються завдання користувача. Якщо користувач свідомо має повний доступ до всіх файлів, що впливає на роботу завдання. Однак може виникнути ситуація, коли необхідні файли або каталоги недоступні в необхідному режимі доступу. Тому необхідно передбачати в завданні обробку таких виняткових ситуацій. Як правило, обробники виняткових ситуацій поміщаються на початку тексту завдання і перевіряють можливість використання інформаційного оточення завданням. В даному випадку під можливістю використання розуміється наявність необхідних прав доступу.