УДК 519.688(045)

ПРИНЦИПИ РОЗРОБКИ ЗАХИЩЕНИХ ОПЕРАЦІЙНИХ СИСТЕМ ДЛЯ ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Іван Четверіков, Микола Чумак, Володимир Шаталюк
Військовий інститут телекомунікацій та інформатизації

Анотація: **Наведені нові принципи побудови операційних систем (ОС) та механізмів захисту інформації в ОС.**
Summary: **This work presents a new prinsipals of creating operating systems for computers and protecting the information in these OS.**
Ключові слова: **Доступ, мікроядро, об'єктно-орієнтований підхід, операційна система.**

I Вступ

Вирішення багатьох спеціальних задач за допомогою обчислювальної техніки, які потребують високого рівня захисту інформації, що обробляється, вимагає застосування захищених операційних систем. Про рівень захищеності ОС можна говорити лише у тому випадку, якщо є повна інформація про її структуру, використані програмні рішення. В даній роботі пропонуються нові підходи до створення операційних систем та реалізації механізмів забезпечення інформаційної безпеки в цих системах.

Відповідно до традиційної архітектури операційна система являє собою монолітне ядро, яке реалізує основні функції з керування апаратними ресурсами, й організуюче середовище для виконання прикладних процесів. Така архітектура є, по-перше, занадто громіздкою, а, по-друге, для реалізації якоїсь нової політики безпеки необхідно змінювати все ядро. Аналіз різних архітектур операційних систем показує, що найперспективнішими є мікроядерні структури, які розподіляють функції ОС між мікроядром і системними сервісами, реалізованими у вигляді процесів, рівноправних з користувальницькими прикладними програмами. При подальшій розробці принципів побудови захищених операційних систем було використано дані про будову ОС Trusted Mach [1].

II Основи мікроядерної архітектури

Визначимо ряд понять, що використовуються у мікроядерній технології побудови ОС.

Задача. У мікроядерних системах це поняття замінює традиційне для ОС поняття процес. Задача є узагальненим поняттям процесу і позначає набір ресурсів, що утворюють середовище для виконання потоків (див. далі). Це середовище містить у собі:

- ізольований від інших задач адресний простір;
- середовище виконання прикладного процесу;
- атрибути безпеки;
- засоби взаємодії з ядром;
- засоби взаємодії з іншими задачами.

Кожна задача має свій власний простір імен портів (див. далі). Задача може виступати в ролі споживача ресурсів (клієнт), або надавати визначені ресурси іншим задачам (сервер). Та сама задача може бути одночасно і сервером і клієнтом, споживаючи ресурси, які контролюються одними задачами, і надаючи свої ресурси іншим задачам. Доступ до ресурсів, як і взаємодія з іншими задачами і ядром, здійснюється тільки за допомогою обміну повідомленнями через порти.

Потік (Thread) – логічно зв**'**язаний потік виконуваних команд. Кожен потік виконується в контексті якої-небудь задачі і може здійснювати безпосередній доступ тільки до її середовища. Потоки є основною одиницею обчислень і єдиним активним елементом в системі. Потік являє собою послідовність команд, що виконуються в рамках задачі. Його єдиним атрибутом є стан процесора. Усі потоки всередині задачі спільно використовують адресний простір і успадковують атрибути безпеки задачі.

Порт – односпрямований комунікаційний канал, за допомогою якого задачі обмінюються інформацією один з одним і з ядром, здійснюючи операції посилки й одержання повідомлень (див. далі). Задачі можуть одержати доступ до портів тільки при наявності в них прав на посилку/прийом повідомлень.

Повідомлення – логічно зв'язаний набір даних, переданий через порт за одне звертання. Для здійснення контролю доступу ядро позначає всі повідомлення спеціальною міткою, що ідентифікує відправник повідомлення.

Простір імен – спосіб ідентифікації ресурсів і об'єктів системи за допомогою унікальних символьних міток-імен, що утворюють ієрархічну деревоподібну структуру типу файлової системи. Функцію

перетворення символьних імен у внутрішні ідентифікатори і навпаки виконує спеціальний компонент мікроядерних ОС, що називається сервером імен.

В основі архітектури мікроядерних ОС покладено наступні базові концепції:

- мінімізація набору функцій, які підтримуються мікроядром, і реалізація традиційних функцій ОС (файлова система, мережна підтримка тощо) поза ним;
- організація синхронної і асинхронної взаємодії між процесами винятково через механізм обміну повідомленнями;
- усі відносини між компонентами будуються на основі моделі клієнт/сервер;
- застосування об'єктно-орієнтованого підходу при розробці архітектури і програмування системи.

Відповідно до цього мікроядро має реалізовувати наступний набір функцій:

- керування фізичною апаратурою (оперативна пам'ять, процесори, зовнішні пристрої тощо);
- розподіл ресурсів апаратної платформи між процесами (час процесора, пам'ять тощо);
- ізоляція процесів;
- організація взаємодії між процесами;
- керування процесами (створення, знищення, перемикання).

Мікроядро, таким чином, є своєрідним арбітром, роль якого зводиться до підтримки типового набору "правил гри" всередині операційної системи, всі інші традиційні функції ОС мають бути реалізовані поза ядром.

Мінімізація функцій мікроядра дає можливість сконцентрувати в ньому код, що залежить від апаратної платформи. Це дозволяє підвищити переміщуємість ОС до максимуму. Таким чином, мікроядро реалізує тільки життєво важливі функції, що лежать в основі операційної системи, і є базисом для всіх системних служб, сервісів і прикладних програм.

Використання механізму передачі повідомлень дозволяє встановити єдиний інтерфейс для взаємодії між усіма компонентами системи незалежно від їхнього рівня і призначення, що дає можливість будувати всі інформаційні зв'язки в системі за моделлю клієнт/сервер.

У моделі клієнт/сервер усі компоненти розглядаються або як споживачі (клієнти), або як постачальники (сервери) деяких ресурсів чи сервісів. Стандартизовані протоколи надання сервісу чи ресурсів дозволяють серверу обслуговувати клієнтів незалежно від деталей їхньої реалізації, що відкриває перед розроблювачами широкі можливості для побудови розподілених систем. Ініціатором обміну звичайно є клієнт, який надсилає запит на обслуговування серверу, що знаходиться в стані чекання запиту. Той самий процес може бути клієнтом стосовно одного ресурсу і сервером для інших. Дана модель успішно застосовується не тільки при побудові ОС, але і при створенні програмного забезпечення будь-якого рівня. Застосування моделі клієнт/сервер стосовно ОС полягає в реалізації функцій, що не ввійшли до складу ядра компонентів ОС, у вигляді безлічі серверів, кожний з яких призначений для обслуговування визначеного ресурсу (наприклад, керування пам'яттю, процесами контролю доступу тощо).

Найбільш повно розкрити переваги технології клієнт/сервер дозволяє застосування методів об'єктно-орієнтованого проектування і програмування. Якщо кожен сервер обслуговує тільки один тип ресурсів і представляє його клієнтам у вигляді деякої абстрактної моделі, то такий сервер можна розглядати як об'єкт, тому що він має усі необхідні для цього якості. Згідно з Г. Буч [2] об'єкт повинний мати стан, поводження і індивідуальність. Дійсно, кожен сервер виконується як окремий процес і тому має індивідуальність. Для кожного сервера існує чітко визначена модель станів і переходів між ними. І, нарешті, "поводження" кожного сервера однозначно регламентується протоколом його взаємодії з клієнтами. Відповідно, на цих принципах можна будувати модель ОС, як ієрархію серверів і моделей ресурсів, що представляються ними, а також описувати існуючі між ними взаємозв'язки за допомогою об'єктних відносин спадкування, використання і включення [2].

З погляду створення захищених операційних систем використання об'єктно-орієнтованого підходу в сполученні з мікроядром і технологією клієнт/сервер дозволяє розроблювачу реалізувати взаємодію суб'єктів і об'єктів, а також контроль за інформаційними потоками за допомогою обмеженого числа простих і зрозумілих механізмів, що полегшує адекватність реалізації моделі безпеки і дозволяє застосовувати формальні методи аналізу.

Завдяки принципам, на яких засновані мікроядерні ОС, їхні компоненти функціонують на основі дуже невеликого і порівняно простого набору абстракцій, що складають базис системи і компактно реалізовані у мікроядрі. Можна сказати, що для захищених систем така архітектура є оптимальною, тому що вона дозволяє досить просто й ефективно вирішити цілий ряд питань, що неминуче виникають при реалізації захищених систем.

1. Виявлення потоків інформації в системі. Оскільки всі взаємодії здійснюються винятково за допомогою механізму передачі повідомлень, то, контролюючи потоки повідомлень, можна бути впевненим у тому, що контролюються всі інформаційні потоки в системі.
2. Визначення суб'єктів і об'єктів взаємодії. Як уже згадувалося, всі задачі в мікроядерних системах пов'язані між собою відносинами клієнт-сервер. Відповідно, суб'єктом взаємодії завжди є задача-клієнт, а об'єктом – ресурс, що обслуговується задачею-сервером.
3. Розміщення підсистеми контролю доступу. Оскільки єдиним механізмом взаємодії є передача повідомлень, то функція контролю за інформаційними потоками має бути покладена на ту частину ядра системи, яка реалізує цей механізм. Контроль і керування доступом до ресурсів і об'єктів можуть бути реалізовані як у складі серверів, що відповідають за обслуговування цих ресурсів і об'єктів, так і на рівні всієї системи в цілому. Через те, що багато серверів обслуговують однотипні ресурси й об'єкти (файли, пристрої тощо), контроль доступу до них з метою уніфікації реалізується на рівні глобального простору імен системи.
4. Мінімізація обсягу програмного коду, що відповідає за контроль доступу. Мікроядерна ОС має всього дві процедури, що реалізують контроль за здійсненням доступу: на рівні передачі повідомлень і глобальна система на рівні іменованих об'єктів. Таким чином, обсяг програм, коректність функціонування яких критична для безпеки всієї системи, скорочений до мінімуму.
5. Використання об'єктно-орієнтованих технологій програмування. Контроль за потоками повідомлень і доступом процесів до ресурсів із глобального простору імен можна здійснювати на основі уніфікованого набору властивостей повідомлень (джерело, приймач) і ресурсів (ідентифікатор процесу, ім'я ресурсу). За рахунок цього досягається абстрагування системи захисту від специфіки інформаційних взаємодій, але в той же час зберігається її гнучкість за рахунок можливості використання в задачах-серверах спеціалізованих механізмів захисту, адаптованих і конкретизованих відносно ресурсів, що обслуговують ці сервери.
6. Верифікація й аналіз захисту. Досягнута за допомогою застосування описаних рішень простота і компактність засобів контролю за здійсненням доступу за рахунок структуризації системи сприяє застосуванню формальних методів верифікації й аналізу програмного коду засобів захисту.

III Реалізація політики безпеки в мікроядерних ОС

У мікроядрі можна виключити засоби, що безпосередньо відповідають за реалізацію політики безпеки і високорівневого керування доступом до інформаційних ресурсів системи. Однак мікроядро повинно мати два необхідних для реалізації цих функцій механізми, а саме: ізоляцію задач і контроль за передачею повідомлень. Всі інші механізми захисту, що спираються на цей сервіс, можуть бути реалізовані в складі серверів.

Керування доступом, контроль за його здійсненням є основним механізмом реалізації діючої в системі політики безпеки. Керування доступом в операційних системах базується на традиційній концепції доступу до активних компонентів системи (суб'єктів) і до пасивних (об'єктів). Основним механізмом контролю доступу є монітор взаємодій, в обхід якого здійснити доступ неможливо.

Розглянемо застосування деяких основних понять політики безпеки відносно мікроядерних ОС.

Користувачами будемо називати власників інформації, що мають право доступу до системи. Для реалізації нормативного керування і контролю доступу з кожним користувачем зв'язаний визначений діапазон рівнів безпеки. Щоб користувач міг отримати доступ до системи, його діапазон рівнів безпеки не може бути порожнім. Після успішної ідентифікації й автентифікації користувач може створювати суб'єкти, що діють від його імені. При цьому користувач вказує рівень безпеки цих суб'єктів, але тільки в рамках відведеного йому діапазону рівнів.

Суб'єкт являє собою сукупність задач (звичайно запущених самим користувачем) з однаковими атрибутами безпеки. Кожній задачі привласнюється спеціальний ідентифікатор безпеки, що є визначником атрибутів безпеки суб'єкта, якого представляє ця задача. Іншими словами, усі задачі з тим самим ідентифікатором безпеки виступають як один суб'єкт і не розрізняються монітором взаємодій у ході здійснення контролю доступу.

Суб'єкти є єдиними активними елементами системи. Кожному суб'єкту в момент створення привласнюється ідентифікатор безпеки. Ідентифікатор безпеки кожного суб'єкта залишається незмінним протягом усього часу існування суб'єкта і посилається на атрибути безпеки суб'єкта. Атрибути безпеки суб'єктів включають максимальний рівень безпеки об'єктів, до яких суб'єкту дозволено мати доступ, ідентифікатор користувача, від імені якого діє суб'єкт, і список груп, членом яких є цей користувач.

Таким чином для мікроядерних ОС об'єкт – це поділювана область пам'яті, що може спільно використовуватися декількома суб'єктами. Вважається, що пам'ять використовується спільно у тому

випадку, коли дії однієї задачі з цією областю пам'яті можуть бути помічені іншою задачею, тобто через цю область пам'яті може відбуватися обмін інформацією між двома задачами. Сервер імен зв'язує з кожним іменованим об'єктом унікальне ім'я у вигляді рядка символів, включає його в простір імен і створює список прав доступу і мітку. Наприклад, доцільно організувати наступні різновиди об'єктів: файли, каталоги, посилання, з'єднання, пристрої.

Монітор взаємодій визначає атрибути безпеки суб'єкта й об'єкта і на підставі правил керування доступом, заданих політикою безпеки, вирішує – дозволяти доступ, чи ні.

В операційній системі має сенс підтримувати п'ять режимів (прав) доступу до об'єктів: читання, запис, додавання, створення об'єктів і керування доступом. Ці режими мають наступний сенс:

- право на читання має на увазі можливість переглядати, але не модифікувати інформацію, що міститься в об'єкті;
- право на запис означає можливість переглядати і модифікувати об'єкт;
- право на додавання має на увазі можливість модифікувати об'єкт, але без можливості читання (так званий "сліпий запис");
- право створення об'єкту означає можливість створювати об'єкт;
- право керування доступом має на увазі можливість знищувати об'єкт і змінювати права доступу до нього відповідно до політики довільного керування доступом.

IV Механізми контролю та здійснення доступу

Основана задача засобів контролю доступом – забезпечити виконання правил політики безпеки при обробці конфіденційної інформації. У традиційних ОС контроль доступом здійснювався на двох рівнях – апаратному (кільця захисту й ізоляція адресних просторів) і загальносистемному (контроль доступу до об'єктів файлової системи на рівні системних викликів ядра). Переваги технологій, що застосовуються при побудові мікроядерних ОС, дозволяють здійснювати багатоступінчастий контроль за здійсненням доступу на декількох рівнях взаємодії. Для мікроядерних ОС в ієрархії засобів контролю доступом можна виділити чотири рівні: апаратний контроль, контроль на рівні мікроядра, загальносистемний і прикладний.

При цьому високорівневі засоби у своїй роботі спираються на базові, що приводить до спрощення всієї системи в цілому і підвищує гнучкість керування доступом. Контроль доступу на декількох рівнях дає можливість підвищити гнучкість і ефективність контролю доступу, тому що дозволяє:

- здійснювати контроль над інформаційними потоками між суб'єктами системи окремо від контролю і керування доступом до об'єктів;
- реалізувати в рамках однієї ОС різні політики безпеки, що описують керування доступом до об'єктів різної природи, без конфліктів і протиріч.

Цілі, які переслідуються на кожнім рівні контролю доступу, методи їхнього досягнення і компоненти ОС, у яких доцільно розмістити відповідні засоби, показані в таблиці.

Таблиця – Опис рівнів контролю доступу

Рівень	Цілі	Методи	Засоби
апаратний	Ізоляція суб'єктів і підтримка контролю за взаємодіями	Керування адресними просторами задач і режимами захисту процесора	Процесор і окремі компоненти мікроядра
мікроядра	Контроль взаємодій у системі	Керування потоками повідомлень за допомогою доступу до портів	Мікроядро
системний	Реалізація загальної політики безпеки ОС	Керування атрибутами безпеки суб'єктів і об'єктів. Керування і контроль доступу суб'єктів до об'єктів відповідно до правил політики безпеки системи	Сервер імен
прикладний	Реалізація спеціалізованих політик безпеки для керування доступу до ресурсів прикладних програм	Керування і контроль доступу до ресурсів відповідно до правил політики безпеки	Прикладні сервери, що обслуговують ресурси

V Висновки

Мікроядро реалізує базові функції операційної системи, на які спираються системні сервіси і прикладні програми. В результаті такі важливі компоненти ОС як файлова система, підтримка мережі тощо перетворюються в незалежні модулі, що функціонують як окремі процеси і взаємодіють з ядром і один з одним на загальних підставах. Це означає, що чіткий поділ програмного забезпечення на системні і прикладні програми розмивається, тому що фактично між процесами, що реалізують функції ОС, і прикладними процесами, що виконують програми користувача, немає ніяких відмінностей. Усі компоненти системи використовують засоби мікроядра для обміну повідомленнями, але взаємодіють безпосередньо. Мікроядро лише перевіряє законність повідомлень, пересилає їх між компонентами і забезпечує доступ до апаратури.

Керуючись запропонованими принципами можливо створити захищену операційну систему відповідно до тих вимог, які будуть ставитися до неї на етапі її розробки, та необхідного варіанту політики безпеки в цій системі. При цьому необхідно мати на увазі, що викладені теоретичні принципи побудови захищених операційних систем є лише узагальненою моделлю, на основі якої можна практично реалізувати спеціалізовані ОС.

Література: 1. MK++ Kernel High Level Design. Open Software Foundation, Inc. Revision 2.2 January 12 1996. 2. Буч Г. Обьектно-ориентированное проектирование с примерами применения. //М.: Конкорд, 1992 г.