Безпека операційних систем і комп’ютерних мереж

Лекція 8

Засоби захисту Windows

Реалізація дискреційного керування доступом у Windows

- Ключову роль у захисті відіграє диспетчер об’єктів.
- Коли якийсь потік намагається здійснити доступ, він запитує визначник об’єкта
  - Модель захисту Windows потребує, щоб потік ще до відкривання об’єкта вказав, які операції він збирається виконувати над цим об’єктом
  - Диспетчер об’єктів та SRM, ґрунтуючись на ідентифікаційних даних процесу, якому належить цей потік, визначають, чи можна надати йому визначник, що дозволяє доступ до потрібного об’єкта
  - Диспетчер об’єктів реєструє права доступу, надані для даного визначника, в таблиці визначників, що належить процесу.
- Контекст захисту потоку може відрізнятися від контексту захисту його процесу. Цей механізм називається *уособленням* (*impersonation*)
  - У випадку уособлення механізми перевірки захисту використовують контекст захисту потоку замість контексту захисту процесу, а без уособлення – контекст захисту процесу, до якого належить потік
  - Всі потоки одного процесу використовують спільну таблицю визначників, тому, коли потік відкриває будь-який об’єкт (навіть при уособленні), всі потоки цього процесу отримують доступ до цього об’єкту.

Відкривання процесом об’єкта, що існує, за іменем

- Це – одна з подій, яка примушує диспетчер об’єктів перевіряти права доступу
- При відкриванні об’єкта за іменем, диспетчер об’єктів шукає його в своєму просторі імен
  - Ім’я об’єкта може належати вторинному простору імен
    - наприклад, простору імен реєстру, що належить диспетчеру конфігурації
    - або простору імен файлової системи, що належить драйверу файлової системи
- Якщо ім’я об’єкта не належить вторинному простору імен, диспетчер об’єктів викликає внутрішню функцію ObpCreateHandle – ця функція створює елемент в таблиці визначників, який зіставляється з об’єктом
  - Для створення елементу в таблиці визначників викликається функція виконуючої системи ExCreateHandle
  - ObpCreateHandle створює визначник лише тоді, коли інша функція диспетчера об’єктів, ObpIncrementHandleCount повідомляє, що потік має право на доступ до даного об’єкту
  - Остання функція для перевірки прав доступу викликає ще одну функцію диспетчера об’єктів, ObCheckObjectAccess, яка повертає результат перевірки

Робота функції ObCheckObjectAccess

- У функцію ObCheckObjectAccess передаються посвідчення (*credentials*) захисту потоку, що відкриває об’єкт, тип доступу, що запитується, а також покажчик на об’єкт
- ObCheckObjectAccess спочатку блокує захист об’єкта та контекст захисту потоку
  - Блокування захисту об’єкта запобігає його зміні іншим потоком в процесі визначення прав доступу
  - Блокування контексту захисту потоку не дає іншому потокові того самого або іншого процесу змінити ідентифікаційні дані захисту першого потоку під час перевірки його прав доступу
- Далі ObCheckObjectAccess викликає метод захисту об’єкта, щоб отримати параметри захисту об’єкта
  - Виклик методу захисту може привести до виклику функції з іншого компоненту виконавчої системи (якщо об’єкт використовує нестандартний захист), але багато об’єктів покладаються на стандартну підтримку керування захистом, що пропонується системою.
- Отримавши інформацію про захист об’єкта, ObCheckObjectAccess викликає SRM-функцію SeAccessCheck, на яку спирається вся модель захисту Windows
  - Вона приймає параметри захисту об’єкта, ідентифікаційні дані захисту потоку (у тому вигляді, в якому вони отримані від ObCheckObjectAccess) і тип доступу, який запитує потік. SeAccessCheck вертає True або False залежно від того, чи дозволяє вона потоку тип доступу що він запросив

Об’єкти із стандартним та нестандартним захистом

- Кожного разу, коли здійснюється виклик методу захисту об’єкта, спочатку перевіряється, чи не використовує об’єкт стандартний захист
- Об’єкт зі стандартним захистом зберігає інформацію про захист у своєму заголовку і пропонує метод захисту з іменем SeDefaultObjectMethod
  - Стандартний захист використовують такі об’єкти як м’ютекси, події та семафори
- Об’єкт, що не використовує стандартний захист, має сам підтримувати інформацію про захист, і пропонувати власний метод захисту
  - Приклад об’єкта з нестандартним захистом – файл
    - У диспетчера введення-виведення, що визначає об’єкти типу “файл”, є драйвер файлової системи, який керує захистом своїх файлів (або не реалізовує захисту)
    - При відкриванні файлу ObCheckObjectAccess не виконується, оскільки об’єкти “файл” знаходяться у вторинних просторах імен
    - Система викликає метод захисту об’єкту файл, тільки якщо потік явно запитує або встановлює параметри захисту файлу (наприклад, через Win32-функції SetFileSecurity або GetFileSecurity)

Посилання процесу на об’єкт по існуючому визначнику

- Це – інша подія, що вимушує диспетчер об’єктів виконати перевірку прав доступу
- Подібні посилання часто робляться опосередковано, наприклад при маніпуляціях з об’єктом через Win32 API з передаванням його визначника
- Наприклад, нехай потік, що відкриває файл, запитує доступ для читання з файлу
  - Якщо потік має відповідні права, що визначаються його контекстом захисту та параметрами захисту файлу, диспетчер об’єктів створює визначник даного файлу в таблиці визначників, що належить процесові – власникові цього потоку
  - Інформація про наданий процесу тип доступу зіставляється з визначником і зберігається диспетчером об’єктів
- В подальшому потік може спробувати щось записати в цей файл через Win32-функцію WriteFile, надавши в якості параметру визначник файлу
  - Системний сервіс NtWriteFile, котрий WriteFile викличе через Ntdll.dll, звернеться до функції диспетчера об’єктів ObReferenceObjectByHandle, щоб отримати посилання на об’єкт «файл» по його визначнику
  - ObReferenceObjectByHandle приймає запитаний тип доступу як параметр
  - Знайшовши у таблиці визначників елемент, що відповідає потрібному визначнику, ObReferenceObjectByHandle порівняє запитаний тип доступу з тим, що був наданий при відкриванні файлу
  - В даному випадку ObReferenceObjectByHandle вкаже, що операція запису має завершитись невдало, оскільки потік, що викликає, відкриваючи файл, не отримав право на записування в нього.

Модель захисту Windows для Win32

- Функції захисту Windows дозволяють застосуванням Win32 визначати власні закриті об’єкти і викликати SRM-сервіси для застосування до цих об’єктів засобів захисту Windows
- Сутність моделі захисту SRM відображає математичний вираз з трьома вхідними параметрами:
  - ідентифікаційними даними захисту потоку,
  - типом доступу, що запитується,
  - інформацією про захист об’єкту.
- Його результат – значення «так» або «ні», які визначають, чи надасть модель захисту доступ, що запитується
- Багато з функцій режиму ядра, які використовує диспетчер об’єктів та інші компоненти виконавчої системи для захисту своїх об’єктів, експортуються у вигляді Win32-функцій режиму користувача
  - Наприклад, еквівалентом SeAccessCheck для режиму користувача є AccessCheck
- Таким чином, застосування Win32 можуть застосовувати модель захисту Windows й інтегруватися з інтерфейсами автентифікації й адміністрування цієї операційної системи

Ідентифікатори захисту SID (1/2)

- Для ідентифікації активних об'єктів (суб'єктів) в системі, Windows використовує *ідентифікатори захисту* (*Security Identifiers*, *SID*)
  - SID є у користувачів, локальних груп та груп домену, локальних комп'ютерів, доменів та членів доменів
- SID являє собою числове значення змінної довжини, що формується з
  - номера версії структури SID
  - 48-бітного коду агента ідентифікатора (*Identifier Authority Value*)
    - Код агента ідентифікатора визначає агента, що видав SID
    - Таким агентом зазвичай є локальна система або домен під керуванням Windows
  - змінної кількості 32-бітних кодів субагентів та/або відносних ідентифікаторів (*Relative Identifiers*, *RID*)
    - Коди субагентів ідентифікують попечителів, яких уповноважив агент, що видав SID
    - RID – засіб створення унікальних SID на основі спільного базового SID (*common-based SID*)
    - Оскільки довжина SID достатньо велика і Windows намагається генерувати дійсно випадкові значення для кожного SID, ймовірність появи двох однакових SID практично дорівнює нулю.
- В текстовій формі кожен SID починається з префікса S, за яким слідують групи чисел, розділених дефісами, наприклад

  S-1-5-21-746385570-2913517877-2667279727-1023
  - В цьому SID номер версії дорівнює 1, код агента ідентифікатора – 5 (Windows, SECURITY_NT_AUTHORITY), далі йдуть коди чотирьох субагентів та один RID на кінці (1023).

Ідентифікатори захисту SID (2/2)

- SID призначається комп'ютеру під час інсталяції Windows програмою Windows Setup
- Далі Windows призначає SID локальним обліковим записам на цьому комп'ютері
  - SID кожного локального облікового запису формується на основі SID комп'ютеру з додаванням RID
  - RID облікових записів користувача починається з 1000 і збільшується на 1 для кожного нового користувача або групи
- Аналогічно Windows видає SID кожному новоствореному доменові Windows
  - Нові облікові записи домену отримують SID, що формуються на основі SID домену з додаванням RID (який також починається з 1000 й збільшується на 1 для кожного нового користувача або групи)
  - RID з номером 1023 вказує на те, що його SID є 24-м, що був виданий доменом
- Багатьом наперед заданим обліковим записам й групам Windows видає SID, що складається з SID комп'ютера або домену та наперед заданого RID
  - RID облікового запису адміністратора дорівнює 500
  - RID облікового запису гостя – 501
- В основі SID облікового запису локального адміністратора лежить SID комп'ютера, до якого додано RID, що дорівнює 500:

  S-1-5-21-746385570-2913517877-2667279727-500
- Для груп Windows також визначає ряд вбудованих локальних та доменних SID
  - SID, що репрезентує будь-який обліковий запис (має назву Everyone або World), має вигляд S-1-1-0
  - SID мережної групи має вигляд S-1-5-2 (SECURITY_NETWORK_RID)

Маркери (1/3)

- Інформація, що описує привілеї, облікові записи та групи, зіставлені з процесом або потоком, складає *контекст захисту*
- Для ідентифікації контексту захисту SRM використовує об’єкт, що зветься *маркером* (*token*), або *маркером доступу* (*access token*)
- Довжина маркеру варіюється через те, що облікові записи різних користувачів мають неоднакові набори привілеїв й зіставленні з різними обліковими записами груп
- Усі маркери зберігають одну й ту ж саму інформацію:
  - джерело маркера;
  - тип уособлення;
  - ідентифікатор маркера;
  - ідентифікатор автентифікації;
  - ідентифікатор модифікації;
  - час закінчення дії;
  - основна група по умовчанню;
  - DACL по умовчанню;
  - SID облікового запису користувача;
  - SID групи 1 … SID групи n;
  - обмежений SID 1 … обмежений SID m;
  - привілей 1 … привілей k.

Маркери (2/3)

- Під час визначення набору методів доступу, що дозволені потоку чи процесу, механізми захисту в Windows використовують два елемента маркера
  - Перший елемент складається з SID облікового запису користувача та полів SID груп. SID груп в маркері вказує, до яких груп належить обліковий запис користувача.
    - При обробці клієнтських запитів, серверні застосування можуть блокувати визначені групи для обмеження посвідчень захисту, зіставлених з маркером
    - Блокування групи дає майже той самий ефект, що її виключення з маркера
    - Однак, блоковані SID використовуються при перевірці прав доступу.
  - Другим елементом маркера, що визначає, що може робити потік або процес, є список привілеїв – прав, зіставлених з маркером
    - Прикладом привілею може буди право процесу або потоку, зіставленого з маркером, виключати комп'ютер

Маркери (3/3)

- Поля основної групи маркера по умовчанню і списку дискреційного керування доступом (DACL), являють собою атрибути захисту, що застосовуються Windows до об’єктів, які створюються процесом чи потоком з використанням маркера
- Маркер може бути
  - основним (*Primary Token*)
    - ідентифікує контекст захисту процесу
  - уособленим (*Impersonation Token*)
    - застосовується для тимчасового запозичення потоком іншого контексту захисту – зазвичай іншого користувача
- Поле джерела маркера зберігає відомості (в текстовій формі) про творця маркера
  - Дозволяє розрізняти такі джерела, як диспетчер сеансів Windows, мережевий файл-сервер або RPC-сервер

Ідентифікатори маркера

- Ідентифікатор маркера являє собою локально унікальний ідентифікатор (*Locally Unique Identifier*, *LUID*), який SRM присвоює маркеру при його створенні
  - Виконавча система підтримує свій LUID-лічильник, за допомогою якого вона призначає кожному маркеру унікальний числовий ідентифікатор.
- Ідентифікатор автентифікації (*Authentication ID*) – це також LUID. Він призначається маркерові творцем
  - Як правило, Lsass є єдиним творцем маркерів у системі й формує LUID з LUID виконавчої системи
  - Lsass копіює ідентифікатор автентифікації для всіх маркерів – нащадків початкового маркеру
  - Використовуючи цей ідентифікатор, програма може визначити, чи належить якийсь маркер тому самому сеансові, що й інші маркери, які аналізуються цією програмою
- Ідентифікатор модифікації оновлюється виконавчою системою при кожній зміні характеристик маркера.
  - Перевіряючи ідентифікатор модифікацій, програма може виявляти зміни в контексті захисту з моменту останнього використання
- Маркери мають поле терміну закінчення дії, яке існує в них, починаючи з Windows NT 3.1, але досі не використовується і не описане в MSDN
  - Якщо термін дії облікового запису закінчується в той момент, коли користувач все ще знаходиться в системі, він все ще може звертатися до системних ресурсів, доступ до яких було дозволено по простроченому обліковому запису
  - Якби Windows підтримувала маркери з обмеженим часом дії, система могла б заборонити користувачу доступ до ресурсів одразу після завершення терміну дії маркера.

Дескриптори захисту

- Дескриптор захисту (*Security Descriptor*) – це структура даних, що зберігає інформацію про захист, що зіставлена з об’єктом і вказує кому й які дії дозволено виконувати над об’єктом
- Дескриптор захисту включає такі атрибути:
  - *Номер версії*. Версія моделі захисту SRM, що використана для створення дескриптора
  - *Прапори*. Необов’язкові модифікатори, що визначають поведінку чи характеристики дескриптора
    - Приклад – прапор SE_DACL_PROTECTED, який забороняє успадкування дескриптором параметрів захисту від іншого об’єкта
  - *SID власника*. Ідентифікатор захисту власника
  - *SID групи*. Ідентифікатор захисту основної групи даного об’єкта
    - використовується тільки POSIX
  - *Список дискреційного керування доступом* (DACL)
    - Вказує, хто і за якими методами доступу може отримати доступ до об’єкта
  - *Системний список керування доступом* (SACL)
    - Вказує, які операції доступу до цього об’єкта й яких користувачів мають реєструватися в журналі аудиту безпеки

Списки керування доступом

- Список керування доступом складається із заголовку й може містити елементи (ACE)
- Існує два типи ACL: DACL та SACL
- У DACL кожен ACE має SID та маску доступу (а також набір прапорців)
  - ACE можуть бути чотирьох типів:
    - “доступ дозволено” (*access allowed*)
    - “доступ заборонено” (*access denied*)
    - “дозволений об’єкт” (*allowed-object*)
    - “заборонений об’єкт” (*denied-object*)
  - ACE типів “дозволений об’єкт” та “заборонений об’єкт” використовуються лише в Active Directory
  - Якщо в дескрипторі захисту немає DACL (DACL = null), то будь-який користувач отримає повний доступ до об’єкту. Якщо DACL пустий (тобто в ньому немає ACE), доступ до об’єкта не отримає ніхто.
  - ACE, що використовуються в DACL, також мають набір прапорів, які контролюють і визначають характеристики ACE, що пов’язані з успадкуванням
- SACL складається з ACE двох типів:
    - “системного аудиту” (*System Audit ACE*)
    - “об’єкта системного аудиту” (*System Audit-Object ACE*).
  - Ці ACE визначають, аудит яких операцій, що виконуються над об’єктами конкретними користувачами або групами, має бути виконаний

Алгоритми з’ясування прав доступу

- Для визначення прав доступу до об’єкта використовуються два алгоритми:
  - алгоритм, що порівнює запитані права з максимально можливими для даного об’єкта та експортується в режим користувача у вигляді Win32 функції *GetEffectiveRightsFromAcl*
  - алгоритм, що перевіряє наявність конкретних прав доступу та активується через Win32-функцію *AccessCheck* або *AccessCheckByType*
- Права доступу кодуються за допомогою бітової маски, кожний біт якої відповідає певному праву доступу
- На початку перевірки створюються маски **g**=0 і **d**=0, а в масках доступу **m**, **a**(1), …, **a**(n) відображаються всі відображувані права доступу
  - Таким чином, в подальшому всі маски доступу **a**(i) містять лише специфічні і стандартні права доступу, а маски **m**, **g** і **d** можуть містити ще й віртуальні права доступу (причому **g** і **d** – лише віртуальне право Access System Security)

Позначення в описі алгоритмів

m – маска доступу, що описує права доступу, які суб'єкт намагається отримати, але ще не отримав; на початку алгоритму маска m містить всі права, які суб'єкт запитав;

g – маска доступу, що описує права доступу, вже надані суб'єкту ("granted"); на початку роботи алгоритму g=0;

d – маска доступу, що описує права доступу, заборонені суб'єкту ("denied"); на початку роботи алгоритму d=0;

a(i) – маска доступу i-го ACE;

n – кількість ACE у дескрипторі захисту об'єкта;

xi – i-й біт маски доступу x;

~ – операція побітової інверсії;

& – операція побітової кон'юнкції (побітове логічне І);

| – операція побітової диз'юнкції (побітове логічне АБО).

Алгоритм з'ясування максимальних прав доступу

- За відсутності DACL (DACL = null) об'єкт є незахищеним і система захисту надає до нього повний доступ
- Якщо потік, що викликає, має привілей заволодіння об'єктом у власність (Take-Ownership Privilege), система захисту надає право доступу для заміни власника (Write-Owner Access) до аналізу DACL: $g_{WRITE\_OWNER}=1$
- Якщо потік, що викликає, є власником об'єкта, йому надаються права керування читанням (Read-Control) й доступу до DACL для записування (Write-DACL Access): $g_{READ\_CONTROL}=1$ і $g_{WRITE\_DAC}=1$.
- Виконується цикл по i від 1 до n. В циклі виконуються такі дії:
  - Якщо i-й ACE має SID, що співпадає з SID маркера доступу потоку, що викликає, і має тип "доступ заборонено", то права доступу з **a**(i), які не містяться в **g**, додаються до **d**, тобто ті права, які ще не були у явному вигляді надані, помічаються як заборонені: **d**=**d**|(**a**(i) & ~**g**)
  - Якщо i-й ACE має SID, що співпадає з SID маркера доступу потоку, що викликає, і має тип «доступ дозволено», то права доступу з **a**(i), які не містяться в **d**, додаються до **g**, тобто надаються ті права, які ще не були у явному вигляді заборонені: **g**=**g**|(**a**(i) & ~**d**)
- Після аналізу всіх елементів DACL розрахована маска наданих прав доступу **g** повертається потоку як максимальні права доступу
  - Ця маска відображає повний набір методів доступу, які потік може вдало запитувати при відкриванні даного об'єкта
- Все вищезгадане відноситься лише до того різновиду алгоритму, що працює в режимі ядра. Його Win32-версія, реалізована функцією *GetEffectiveRightsFromAcl*, відрізняється відсутністю кроку 2, а також тим, що замість маркеру доступу вона розглядає SID єдиного користувача або групи

Алгоритм з’ясування можливості доступу з заданими правами (1/3)

- Цей алгоритм перевіряє чи можливо задовольнити конкретний запит на доступ, виходячи з маркера доступу потоку, що викликає
  - У кожної Win32-функції відкривання захищених об’єктів є параметр, що вказує бажану маску доступу **m** – останній елемент виразу, що описує захист об’єкта
- За відсутності DACL (DACL = null) об’єкт є незахищеним і система захисту надає до нього тип доступу, що запитується
- Якщо у потоку, що викликає, є привілей захоплення у володіння, і $m_{WRITE\_OWNER}=1$ то система захисту надає право на запис власника: $g_{WRITE\_OWNER}=1$, $m_{WRITE\_OWNER}=0$
  - Якщо після цього виявляється, що **m**=0, тобто потік запитав лише доступ на запис власника, система надає йому цей тип доступу і на цьому завершує перевірку
- Якщо потік, що зробив виклик, є власником об’єкта, то:
  - Якщо $m_{READ\_CONTROL}=1$, то йому надається право керування читанням: $g_{READ\_CONTROL}=1$, $m_{READ\_CONTROL}=0$
  - Якщо $m_{WRITE\_DAC}=1$, то йому надається право на запис до DACL: $g_{WRITE\_DAC}=1$, $m_{WRITE\_DAC}=0$
  - Якщо потік, що викликає, запитав лише ці права (тобто, на цьому кроці **m**=0), то система захисту надає їх без перегляду DACL
- Якщо потік має привілей аудитора і цей привілей не заблокований і $m_{ACCESS\_SYSTEM\_SECURITY}=1$, то йому надається право на зчитування і записування параметрів аудиту об’єкта: $g_{ACCESS\_SYSTEM\_SECURITY}=1$, $m_{ACCESS\_SYSTEM\_SECURITY}=0$
  - Якщо **m**=0, то подальша перевірка не виконується, доступ надається

Алгоритм з’ясування можливості доступу з заданими правами (2/3)

- У циклі по i від 1 до n переглядаються всі ACE у DACL – від першого до останнього. Обробка ACE виконується у випадку однієї з наступних умов:
  - SID в ACE збігається з неблокованим SID (SID можуть бути блоковані та неблоковані) у маркері доступу потоку, що викликає, незалежно від того, який це SID – основний чи групи;
  - SID в ACE типу “доступ дозволено” збігається з SID у маркері доступу потоку, що викликає, і цей SID не має атрибуту перевірки тільки на заборону;
  - Йде вже другий прохід пошуку в дескрипторі обмежених SID, та SID в ACE збігається з обмеженим SID у маркері доступу потоку, що викликає.
- Якщо i-й ACE має тип “доступ заборонено”, то права доступу з **a**(i), які містяться в **m**, але не містяться в **g**, додаються до **d**, тобто ті права, рішення за якими ще не було прийнято, помічаються як заборонені: **d=d**|(**a**(i) & **m** & ~**g**)
- Якщо i-й ACE має тип “доступ дозволено”, то права доступу з **a**(i), які містяться в **m**, але не містяться в **d**, додаються до **g**, тобто надаються ті права, які ще не були у явному вигляді заборонені, і при цьому надані права вилучаються із списку прав, які були запитані:
  g=g|(**a**(i) & **m** & ~**d**), **m=m** & ~**g**

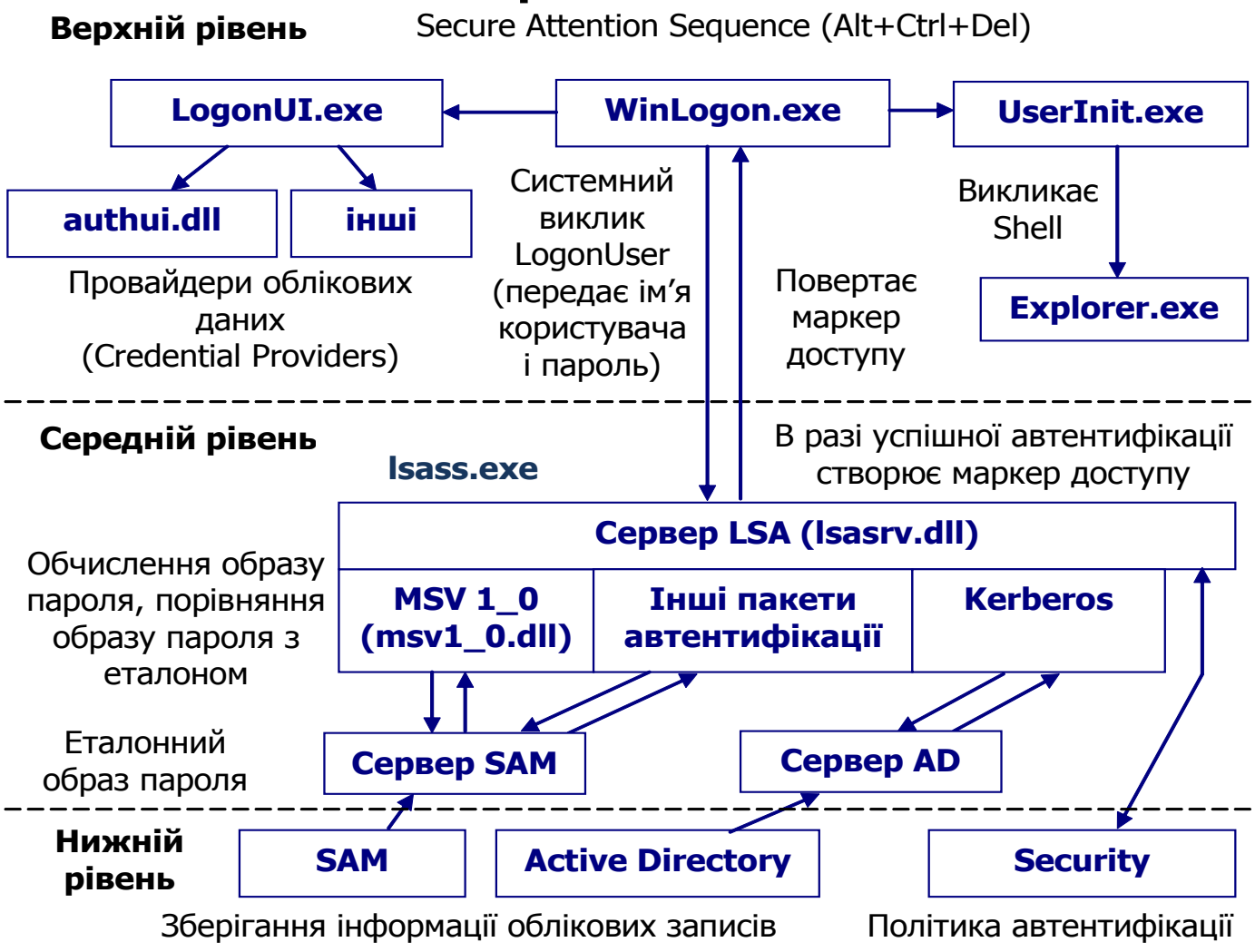
Алгоритм з’ясування можливості доступу з заданими правами (3/3)

- Після кожної ітерації циклу здійснюється перевірка **m**
  - Якщо **m**=0, це означає, що всі запитані права вже надані, при цьому цикл завершується і потоку надається доступ до об’єкта за списком методів, що заданий маскою **g**
  - Якщо **m=d**, це означає, що всі запитані і досі не надані права вже заборонені, при цьому цикл завершується і потоку доступ до об’єкта не надається
  - Якщо досягнуто кінець DACL, і деякі із запитаних прав доступу ще не надані (**m**≠0), доступ до об’єкта забороняється
- Якщо всі права доступу надані, але в маркері доступу потоку, що викликає, є хоча б один обмежений SID, тоді система повторно сканує DACL у пошуку ACE, маски доступу яких відповідають наборові запитаних прав доступу
  - При цьому, також йде пошук ACE, SID яких збігається з будь-яким з обмежених SID потоку, що викликає. Потік отримує доступ до об’єкту, якщо запитані права доступу надавались після кожного проходу по DACL

Особливості алгоритмів з'ясування прав доступу

- Поведінка обох алгоритмів перевірки прав доступу залежить від відносного розташування ACE, що дозволяють та що забороняють
  - Вбудовані засоби Windows завжди організують DACL таким чином, що передують ACE, що забороняють
- Обробка DACL системою захисту при кожному використанні визначнику процесом була б неефективною, тому SRM перевіряє права доступу тільки при відкриванні визначнику, а не при кожному його використанні
  - Таким чином, якщо процес один раз вдало відкрив визначник, система захисту не може анулювати надані при цьому права доступу – навіть коли DACL об'єкту змінився
  - Оскільки код режиму ядра звертається до об'єктів за покажчиками, а не за визначниками, то при використанні об'єктів операційною системою права доступу не перевіряються. Інакше кажучи, виконавча система Windows повністю довіряє собі в сенсі захисту.
- Власник об'єкта завжди отримує право на запис DACL при доступі до об'єкту
  - Це право надається через привілей власника ще до початку перевірки DACL
  - Його буде надано навіть якщо з якихось причин DACL об'єкту пустий
- Власник привілею захоплення у власність (наприклад, адміністратор) має доступ до будь-якого об'єкту в системі
  - Маючи привілей заволодіння у власність, користувач може відкрити об'єкт з правом "запис власника" (Write-Owner Permission) і змінити власника на "Администратор"
  - Після цього він може закрити об'єкт і повторно відкрити його з правом доступу до DACL для запису, а потім змінити DACL так, щоб обліковому записові адміністратора був наданий повний доступ до об'єкту.

Архітектура підсистеми автентифікації Windows



Послідовність входу користувача в систему (1/4)

1. Надходить SAS – Secure Attention Sequence
   - Або програмний виклик функції, або натискання Alt+Ctrl+Del
2. Активізується Winlogon. Winlogon викликає LogonUI для надання користувачеві графічного інтерфейсу для введення даних автентифікації
3. LogonUI викликає необхідного провайдера облікових даних (Credential Provider). Провайдер – це COM об’єкт.
   - Стандартні провайдери:
     - authui.dll
     - SmartcardCredentialProvider.dll
   - Можуть реєструватись численні інші провайдери (у тому числі від сторонніх розробників)

Послідовність входу користувача в систему (2/4)

4. Провайдер отримує від користувача інформацію, що ідентифікує та автентифікує його
   - Для парольної автентифікації:
     - інформація, що ідентифікує, – це умовне ім’я користувача
     - інформація, що автентифікує – це пароль
5. Winlogon передає інформацію, що ідентифікує та автентифікує користувача, на середній рівень (до Lsasrv)
   - Для цього використовується системний виклик LogonUser
6. На середньому рівні Lsasrv викликає потрібний пакет автентифікації (відповідно до політики) і передає йому отриману від верхнього рівня інформацію, що ідентифікує та автентифікує користувача.
   - Для входу в домен ActiveDirectory – Kerberos
   - Для локального входу в систему – MSV 1_0

Послідовність входу користувача в систему (3/4)

7. Пакет автентифікації обробляє отримані дані належним чином (наприклад, генерує образ пароля)
8. Звертаючись до нижнього рівня підсистеми, пакет автентифікації отримує еталонний образ пароля (або інших даних) і здійснює його порівняння з образом даних, що він отримав з верхнього рівня
9. У разі збігу паролів, LSA отримує з нижнього рівня інформацію про те, чи може цей користувач в цей момент розпочинати роботу з системою
   - чи не застарілий пароль?
   - чи не заблоковано обліковий запис користувача?
   - тощо

Послідовність входу користувача в систему (4/4)

10. При позитивному результаті останньої перевірки LSA створює маркер доступу користувача
    - Для цього LSA додатково отримує всю необхідну інформацію з нижнього рівня підсистеми автентифікації
    - Сформований маркер доступу повертається на верхній рівень підсистеми
11. Якщо маркер доступу було створено успішно, Winlogon здійснює авторизацію користувача
    - Для цього він запускає процес UserInit.exe і назначає йому щойно створений маркер доступу (тобто, цей процес діє від імені користувача, якого щойно було автентифіковано)
12. Процес UserInit:
    - завантажує індивідуальні настроювання цього користувача
    - монтує його ключ реєстру
    - завантажує програмне середовище (Shell), яке задано у відповідному ключі реєстру, після чого завершує роботу
      - За умовчанням Shell – це Explorer.exe

Особливості підсистеми автентифікації Windows

- Архітектура підсистеми авторизації досить гнучка і дозволяє використовувати будь-які способи перевірки істинності
- WinLogon використовує привілеї псевдокористувача SYSTEM створювати маркери доступу і призначати маркери доступу процесам.
  - Якщо не надати ці привілеї псевдокористувачу SYSTEM, то вхід користувачів в систему стане неможливим.
- Образи паролів зберігаються в спеціальному розділі реєстру
- Авторизація користувача може відбуватися як локально, так і делегуватися контролерові домену
- Windows за допомогою оснастки керування обліковими записами локальних користувачів і групами забезпечує широкі можливості по керуванню обліковими записами користувачів
  - Для кожного користувача може бути задано ряд атрибутів, таких як належність до груп, місцезнаходження профілю користувача (user profile), робочі години, повноваження на доступ по комутованим лініям і т.д.
  - Крім того, може бути задана політика керування обліковими записами, що регламентує:
    - мінімальний та максимальний терміни життя паролю;
    - мінімальну довжину паролю;
    - унікальність паролю як вимога не належати до заданої кількості востаннє використаних;
    - кількість невдалих спроб автентифікації, після яких обліковий запис блокується, та відрізок часу, протягом якого вони мають відбутися;
    - тривалість блокування облікового запису.
  - Також ця оснастка дозволяє
    - призначати користувачам привілеї (права на всю систему, а не на конкретний об’єкт)
    - задавати політику аудиту.

Аудит (1/4)

- Реєстрація подій у Windows здійснюється шляхом виклику функцій ядра ОС, що додають записи у файли ***.evt** директорії **\WINNT\system32\config**.
- У клієнтських версіях Windows усього є три журнали
  - системний
  - прикладного ПЗ
  - безпеки
- Події аудиту можуть генерувати:
  - диспетчер об’єктів за результатами перевірки прав доступу
    - Для цього використовується список аудиту SACL, який визначає для кожного об’єкту, що саме буде реєструватися при спробах отримати доступ тим чи іншим суб’єктом
  - безпосередньо Win32-функції, що є доступними прикладним програмам
  - код режиму ядра

Аудит (2/4)

- Рішення про аудит конкретного типа подій безпеки приймається у відповідності до політики аудиту локальної системи
  - Політика аудиту є частиною політики безпеки (*local security policy*), що підтримується Lsass у локальній системі
  - Під час ініціалізації системи та зміні політики LSass надсилає SRM повідомлення, що інформує його про поточну політику аудиту
- Lsass відповідає за
  - отримання записів аудиту, що генеруються на основі подій аудиту, від SRM
  - їх редагування
  - їх передачу реєстратору подій (Event Logger)
    - Ці записи надсилає саме Lsass (а не SRM), оскільки він додає в них відповідні подробиці, наприклад інформацію, потрібну для більш повної ідентифікації процесу, по відношенню до якого здійснюється аудит
- З аудитом пов’язані два привілеї:
  - SeSecurityPrivilege
    - Для керування файлом журналу безпеки, а також для перегляду та зміни SACL об’єктів, процес повинен мати привілей SeSecurityPrivilege
  - SeAuditPrivilege
    - Процес, що викликає системні сервіси аудиту, повинен мати привілей SeAuditPrivilege, щоб вдало генерувати запис аудиту в цьому журналі

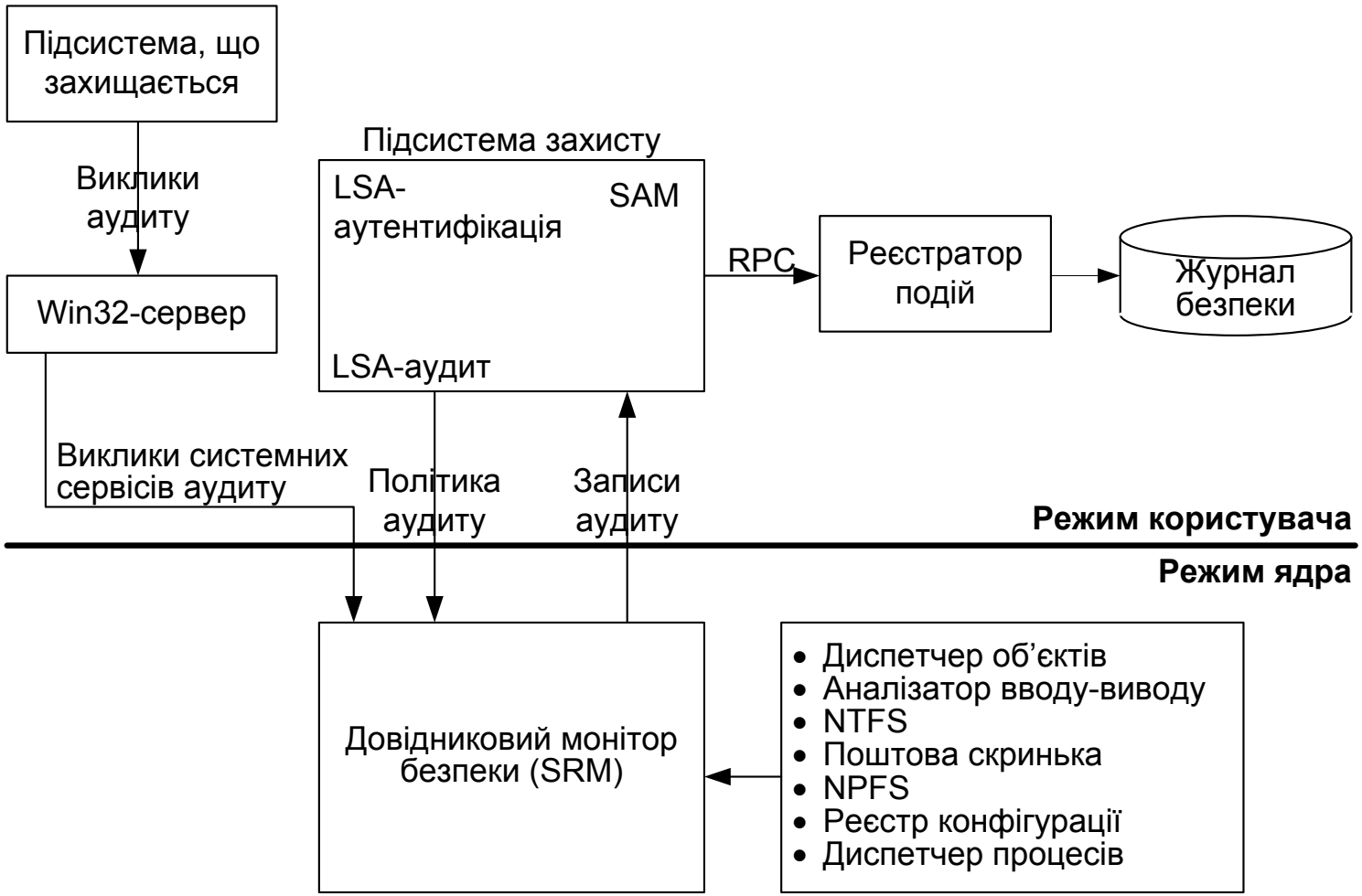
Аудит (3/4)

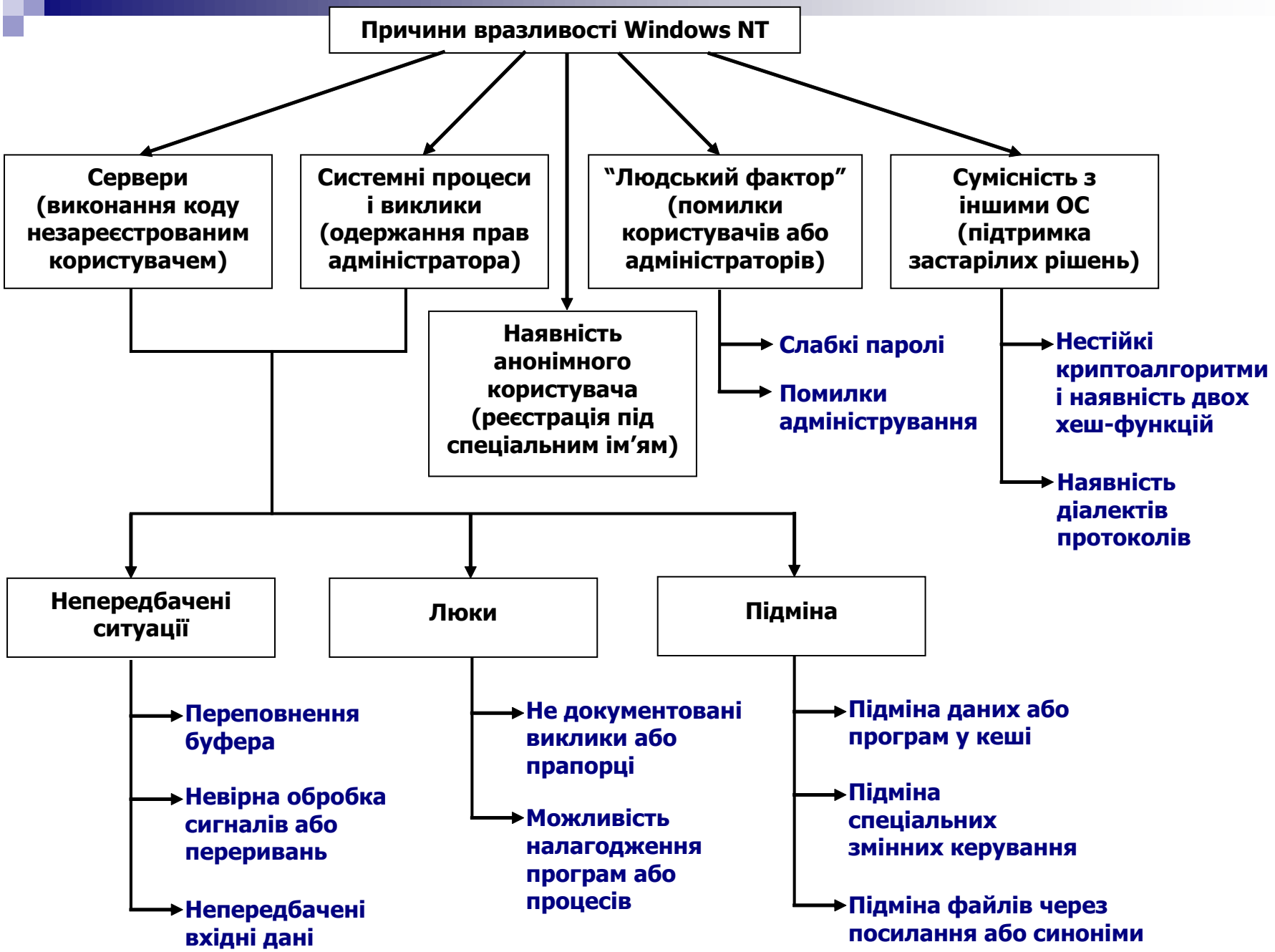
- SRM надсилає записи аудиту до Lsass через своє LPC-з’єднання
- Після того Event Logger вносить записи в журнал безпеки
- Lsass та SAM також генерують записи аудиту, які Lsass пересилає безпосередньо Event Logger
- Записи аудиту, що підлягають пересиланню LSA, поступають в чергу по мірі отримання – вони не передаються пакетами
- Пересилання цих записів здійснюється одним з двох способів
  - Якщо запис аудиту невеликий (менше максимального розміру LPC-повідомлення), він надсилається як LPC-повідомлення, і запис аудиту копіюється з простору адрес SRM у простір адрес процесу Lsass
  - Якщо запис аудиту великий, SRM робить його доступним для Lsass через спільну пам’ять і передає Lsass покажчик на нього, використовуючи для цього LPC-повідомлення

Аудит (4/4)

- Перегляд журналів реєстрації здійснюється оснасткою Event Viewing
- Оснастка надає досить широкі можливості призначення фільтрів відображення записів, зокрема:
  - за часом реєстрації
  - за типом
  - за категорією
  - за джерелом події
- Адміністратори мають доступ до перегляду журналів реєстрації через
  - Control Panel (“Панель управления”), що доступна через стартове меню
  - Пункт меню “Administrative tasks” (“Администрирование”)
- Списки аудиту об’єктів можна задати за допомогою Windows Explorer: “Properties” → “Security” → “Advanced…” → “Auditing” (“Свойства” → “Безопасность” → “Дополнительно...” → “Параметры аудита”)
- Політику аудиту задають окремо – за допомогою “Administrative tasks” → “Local security policy” (“Администрирование” → “Локальная политика безопасности”)

Взаємодія елементів системи аудиту




Причини вразливості Windows NT
Сервери (виконання коду незареєстрованим користувачем)
Системні процеси і виклики (одержання прав адміністратора)
Наявність анонімного користувача (реєстрація під спеціальним ім'ям)
"Людський фактор" (помилки користувачів або адміністраторів)
Сумісність з іншими ОС (підтримка застарілих рішень)
Слабкі паролі
Помилки адміністрування
Нестійкі криптоалгоритми і наявність двох хеш-функцій
Наявність діалектів протоколів
Непередбачені ситуації
Люки
Підміна
Переповнення буфера
Невірна обробка сигналів або переривань
Непередбачені вхідні дані
Не документовані виклики або прапорці
Можливість налагодження програм або процесів
Підміна даних або програм у кеші
Підміна спеціальних змінних керування
Підміна файлів через посилання або синоніми

Загальна оцінка системи Windows

- ОС Windows з міркувань безпеки є досить досконалою системою
  - У Windows на всіх рівнях, починаючи з архітектури системи, впроваджені заходи, спрямовані на забезпечення захисту самої системи та інформації, яка під її керуванням обробляється
- Простота використання ОС Windows є оманливою
  - Windows має інтуїтивно зрозумілий інтерфейс налаштовування політики безпеки, але насправді адміністрування цієї системи досить складне, бо вимагає від системного адміністратора неабияких знань
  - Чутки про “легкість адміністрування ОС Windows” дещо перебільшені
- ОС Windows забезпечує лише довірчу політику керування доступом
- Через велику складність системи і значну кількість функцій, що вона підтримує, неможливо гарантувати відсутність помилок, які призводять до наявності вразливостей