Безпека операційних систем і комп'ютерних мереж

Лекція 16

Безпека прикладних протоколів Інтернету

Питання

- Застарілі протоколи
  - Telnet
  - Мережні служби UNIX
- Актуальні протоколи
  - FTP
  - SSH

Безпека протоколу Telnet

- З погляду безпеки стандартний протокол Telnet має такі суттєві вади:
  - протокол не передбачає ідентифікації й автентифікації, а також контролю послідовності переданих даних, у цьому він цілком покладається на протокол TCP
  - у всіх режимах, крім лінійного, не передбачене шифрування (нагадаємо, що протокол Telnet на практиці дуже часто використовується для віддаленого адміністрування серверів і мережного обладнання, отже, ймовірною є передача конфіденційних даних, зокрема паролів)
- Порушник шляхом прослуховування Telnet-сеансів може отримати ім’я користувача і його пароль, і в подальшому скористатись ними
  - Деякі сучасні реалізації клієнтів і серверів підтримують шифрування даних і знімають зазначену проблему. Однак важливо пам’ятати, що шифрування повинні підтримувати обидві сторони, інакше буде встановлено незахищений сеанс обміну.
- Ряд сучасних RFC, що мають статус пропозицій стандартів, специфікують різні опції автентифікації і шифрування в Telnet
- Іншим рішенням є використання протоколу SSH замість Telnet
  - Так поступають для задач адміністрування UNIX-серверів і мережного обладнання деяких виробників

Атаки на сервер Telnet

- Протокол Telnet має не дуже широко відому можливість, яка суттєво впливає на безпеку сервера
- Клієнт має можливість впливати на змінні оточення сервера ще до автентифікації
  - Ця можливість підтримується багатьма сучасними серверами Telnet
  - Вона може бути використана, наприклад, для атаки на FTP-сервер, який дозволяє анонімне завантаження файлів
  - Порушник може, користуючись протоколом FTP, спочатку завантажити модифіковану бібліотеку (наприклад, **libc**), яка містить бажані для нього функції, або взагалі “троянського коня” під добре відомим ім’ям (наприклад, **ls**), а потім відкрити Telnet-сесію, і змінити змінну оточення, що вказує на бібліотеку, або змінну PATH
  - Такі атаки нелегко виявити, тому що і змінні оточення, і каталоги, в які є доступ для анонімного завантаження файлів за протоколом FTP, ретельно перевіряють далеко не всі адміністратори
  - Оскільки в наш час описана вразливість добре відома, її наявність в системі (тобто, дозвіл на віддалену модифікацію змінних оточення) слід вважати помилкою адміністрування.
- Деякі сервери Telnet мали класичні помилки переповнення буфера
  - У наш час зловмисникам сподіватись на це не варто, але все ж різні нестандартні сервери Telnet (наприклад, вбудовані в мережне обладнання) повинні ретельно перевірятись на наявність таких помилок

Атаки на клієнта Telnet

- Традиційно – помилки переповнення буфера
  - Якщо клієнт має таку помилку, то використати її зловмисник може, наприклад, розмістивши на Web-сервері посилання вигляду **telnet://server.edu/xxxxxxxx** (вважаємо, що за **xxxxxxxx** якраз і схований спеціально підібраний рядок, що дозволяє виконати на комп'ютері клієнта певну команду)
  - Як тільки недбалий користувач активує таке посилання, його браузер запустить Telnet-клієнта і передасть останньому параметри
  - Такі помилки протягом довгих років були актуальними для Telnet-клієнтів ОС Windows 9x
- Ще одна можливість атаки на клієнта була прихована у драйвері ANSI
  - Основні можливості терміналів обмежувались зміною кольорів і шрифтів
    - Використання таких можливостей не несло небезпеки комп'ютерам клієнтів
  - Але існували такі драйвери ANSI, які підтримували макрокоманди
    - При взаємодії Telnet-клієнта з таким драйвером існувала можливість виконання на комп'ютері клієнта макрокоманд, що були передані з сервера. А це вже є потенціальною вразливістю.

r-служба

- За допомогою програм, що входять в r-службу (rlogin, rsh тощо), користувач з деяким реєстраційним ім'ям з довіреного хосту може виконувати команди на даному хості від імені користувача з таким самим реєстраційним ім'ям
- При використанні r-програм з довіреного хосту користувачу не потрібно проходити стандартну процедуру парольної автентифікації — його авторизація відбувається автоматично
- Довірені, або адміністративно-еквівалентні комп'ютери визначаються у файлах, що містять списки імен та IP-адрес довірених хостів
  - **/etc/hosts.equiv** (спільний для усієї системи)
  - **~/.rhosts** (можуть створюватись для будь-якого користувача)
- Проблема безпеки:
  - У UNIX системах існує значна кількість так званих спеціальних користувачів (bin, daemon тощо), для яких інтерактивний вхід в систему заборонений і від імені яких працюють системні процеси
  - Деякі із спеціальних користувачів можуть мати значні права у системі, або бути включеними до груп з підвищеними правами (наприклад, wheel)
  - Одна із специфічних особливостей rsh полягає в тому, що авторизація користувача відбувається не тими засобами, які використовуються під час інтерактивного входу в систему.
  - За допомогою rsh віддалений користувач міг увійти в систему, скажімо, як спеціальний користувач **bin**, для цього він повинен був мати змогу на своєму комп'ютері працювати саме як **bin**

NFS

- NFS (Network File System) — мережна файлова система призначена для організації спільного використання UNIX-машинами файлових систем
  - Система реалізована за архітектурою клієнт-сервер і працює за принципом “stateless”
  - Протоколи NFS описані в RFC-1094, 1813
  - Sun Microsystems першою впровадила NFS у 1985 р.
- NFS базується на засобах, розроблених Sun Microsystems
  - XDR (*External Data Representation* — зовнішнє подання даних)
  - RPC (*Remote Procedure Call* — віддалений виклик процедур)
- NFS має в своєму складі
  - протокол і сервер віддаленого монтування файлових систем (демон **mountd**)
  - протокол і сервер блокування файлів
  - демони, що реалізують файловий сервіс (демон **nfsd**)
- NFS надає зручний спосіб доступу до файлів через мережу, і, як наслідок, створює потенційні загрози безпеці

Безпека NFS

- Основним заходом захисту при використанні NFS є жорсткий контроль за файлом /etc/exports (або /etc/dfs/dfstab у системі Solaris)
- Користувач, який має привілейований доступ на машині — клієнті NFS, має у своєму розпорядженні кілька елегантних методів доступу до файлів, що належать іншим користувачам, навіть у тому випадку, коли привілейований доступ на експорт файлів не надається
- Якщо користувач, якому з якихось міркувань не довіряють, має привілейований доступ на деякому комп'ютері, то на той комп'ютер не слід експортувати жодних файлових систем

Сценарій атаки на NFS

- Можливі дії порушника, що має права суперкористувача на деякій машині, на яку дозволений експорт деякої файлової системи за NFS:
  - Спочатку порушник монтує віддалену файлову систему
  - При цьому він отримує доступ не як **root**, а як **nobody** (UID = –2)
  - Тим не менше, він може отримати список файлових систем, які експортуються
  - Якщо серед них є домашні каталоги користувачів, то порушник обирає жертву і заводить на своєму комп'ютері користувача з таким самим реєстраційним ім'ям і такими самими UID та GID
  - Тепер, зареєструвавшись на своєму комп'ютері під цим ім'ям, порушник отримує через NFS без введення пароля доступ до домашнього каталогу цього користувача на віддаленому комп'ютері
  - Порушник редагує файл ~/.rhosts, додаючи в нього свій комп'ютер
  - Наступним кроком він отримує доступ до віддаленого комп'ютера вже через rsh, знову ж таки без пароля

NIS

- NIS (*Network Information Service* — мережна інформаційна служба), яку раніше називали Yellow Pages, — це засіб, що був розроблений Sun Microsystems для розповсюдження баз даних, зокрема, файлів **/etc/passwd**, **/etc/group**, **/etc/hosts**, а також поштових псевдонімів
- NIS, як і NIS+, що прийшла їй на заміну, є дуже привабливою для зловмисників, оскільки сама її ідея — легкий доступ до інформації, яку ніяк не можна віднести до загальнодоступної — є неприйнятною для сучасних мереж
- Тому вже багато років в усіх настановах адміністраторам переконливо радять повністю відмовитись від цих служб

Проблеми з безпекою протоколу FTP

- Одною з найголовніших проблем протоколу FTP є те, що обмін по каналу керування здійснюється у відкритому вигляді, без криптографічного закриття інформації
  - Пересилання даних також не передбачає додаткового захисту, але це не є великою проблемою:
    - якщо дані потребують захисту конфіденційності, то вони можуть зберігатись у зашифрованому вигляді на сервері,
    - якщо потребують захисту цілісності, то вони повинні супроводжуватись електронним підписом
  - Відкритість каналу керування, натомість, відкриває для порушників можливості перехоплення ідентифікаторів і паролів користувачів FTP-сервера
    - у FTP атрибути передаються відповідними командами протоколу, кожна з яких передається окремим пакетом. Тому в одному з пакетів є послідовність символів USER: <login_name>, а в наступному PASS: <password>, де login_name і password — ідентифікатор і пароль, що введені користувачем
- Для боротьби з цією вадою в межах протоколів прикладного рівня існують лише два шляхи:
  - відмовитись від протоколу FTP на користь захищених протоколів (які, на жаль, не набули достатнього поширення),
  - або взагалі відмовитись від автентифікації користувачів, зробивши FTP–сервер повністю відкритим та доступним
- Інша проблема FTP — відсутність контролю за походженням пакетів: ці завдання покладаються на засоби транспортного і мережного рівнів

Протокол FTP і брандмауери

- Типовою вимогою політики безпеки, яку виконують міжмережні екрани, є заборона ініціювання з зовнішньої мережі з'єднань з комп'ютерами захищеної мережі
  - Тобто ініціаторами з'єднань можуть виступати лише "свої" комп'ютери
  - Це нормально працює для більшості протоколів, заснованих на технології "клієнт-сервер": клієнт з захищеної мережі робить запит до сервера, що знаходиться у зовнішній мережі, сервер йому відповідає
  - У протоколі FTP таким чином встановлюється з'єднання лише для каналу керування
  - Як тільки сервер отримує команду, що вимагає пересилання файлів, він зі своєї ініціативи намагається встановити з'єднання з комп'ютером-клієнтом
  - Крім пересилання файлів, так само окремим з'єднанням передається, наприклад, список файлів у поточному каталозі
    - Більшість клієнтів, що мають розвинутий інтерфейс користувача, передають запит списку файлів невідкладно після встановлення з'єднання. Все, що побачить користувач, який знаходиться за міжмережним екраном, — це вікно, яке інформує про підключення до сервера і про очікування з'єднання. Список файлів ніколи не буде отриманим, також неможлива буде будь-яка передача файлів.
- Вихід з проблеми передбачено у самому протоколі FTP — це так званий *пасивний режим*
  - У пасивному режимі сервер через канал керування передає клієнту параметри каналу передачі даних (зокрема, номер порту), який він дозволяє створити
  - Ініціатором з'єднання виступає клієнт
  - Це дозволяє нормально працювати з FTP-серверами, навіть знаходячись за брандмауером
- Проблеми взаємодії FTP з міжмережними екранами описані у RFC 1579
- Удосконалення і додаткові можливості протоколу FTP, значна частина яких безпосередньо впливає на безпеку, описані також у RFC 2228, 2428, 2773 та інших

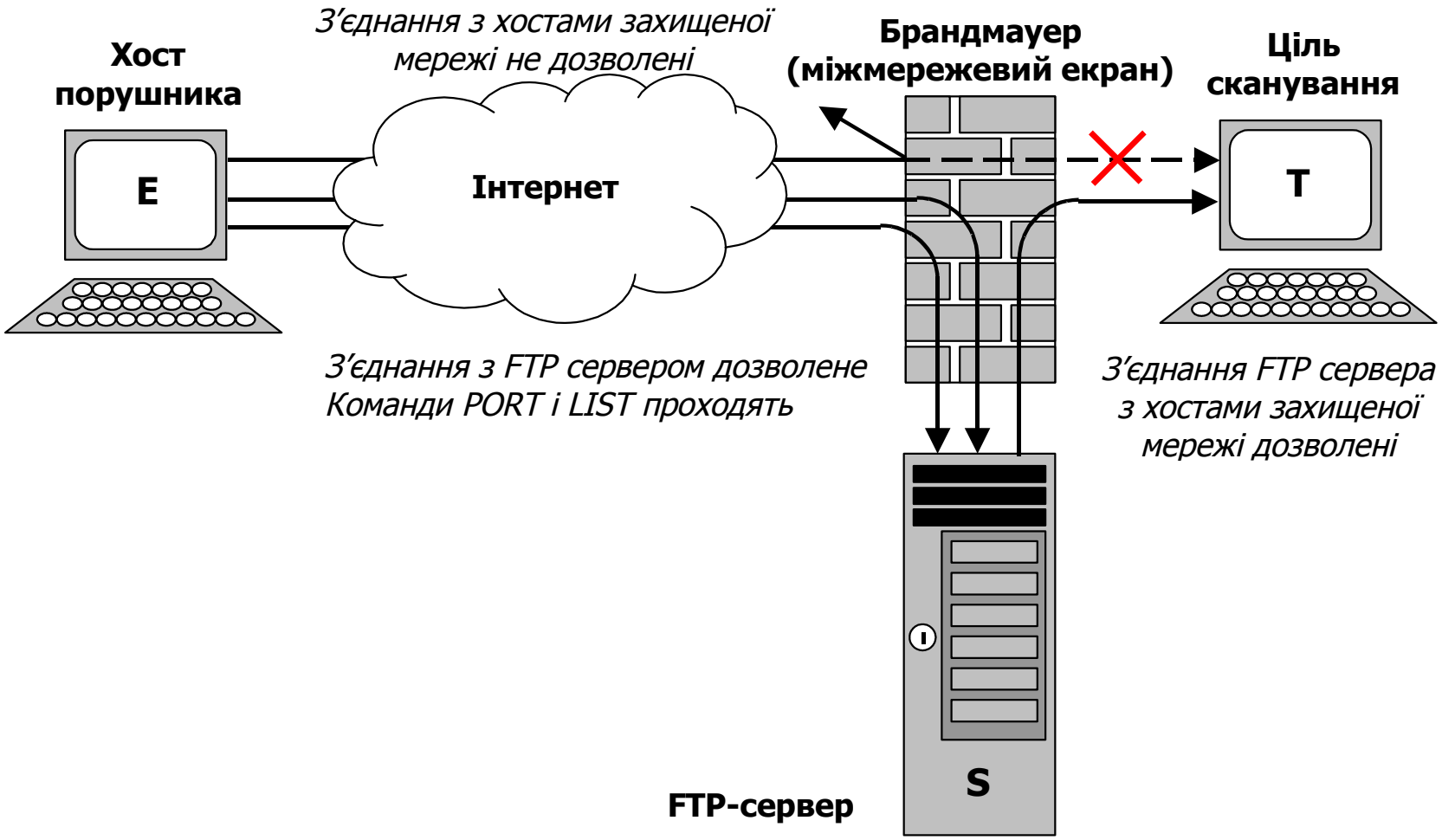
Проксі-режим у протоколі FTP

- Обмеження проксі-режиму практично не описані в документації, але його можливість визначена дуже чітко
  - Суть проксі-режиму полягає в тому, що користувач, зв'язавшись із сервером, може запросити пересилання файлів не лише собі, а й на інший сервер
  - Користувач повинен встановити з'єднання з обома серверами
  - Очевидно, що одному з серверів при цьому необхідно видати команду **PASV** для переведення його у режим прослуховування порту, а специфікацію порту, яку він надішле у відповідь, необхідно передати другому серверу командою **PORT**
  - Після цього команда, що викликає передачу файлів, надіслана другому серверу, спричинить встановлення ним з'єднання з першим сервером і спробу відповідної передачі файлів між ними
- Розробники RFC не дуже детально описали такий режим, можливо, маючи на увазі його подальше детальне опрацювання
  - Очевидно, що сервер може легко розпізнати, що канал передачі даних встановлюється не з тим хостом, з яким встановлено сесію керування
  - Сервери поділяються на ті, що підтримують проксі-режим, і ті, що його не підтримують
- Можна було б вважати, що великої шкоди у проксі-режимі бути не повинно, оскільки користувач все одно повинен мати права встановлювати з'єднання з кожним із серверів — учасників обміну
- Існує одна унікальна можливість, яку надає проксі-режим. Мова йде про техніку прихованого сканування портів — FTP Bounce Attack (прихована атака по FTP)

FTP Bounce Attack

- Якщо FTP-сервер проінструктований встановити з'єднання з деяким хостом Інтернету, відмінним від хосту клієнта, що підтримує в цей момент із сервером FTP-сесію, то сервер не може знати, чи має насправді клієнт з тим іншим хостом у цей момент встановлену FTP-сесію, як того вимагає RFC.
  - Отже, сервер (якщо він підтримує такий режим) просто буде намагатись встановити TCP-з'єднання з указаним хостом
  - Це і надає можливість прихованого сканування.
- Послідовність атаки
  - FTP-серверу надсилається команда PORT з IP-адресою і номером порту, який планується перевірити
  - Після цього надсилається команда LIST
  - За цією командою сервер повинен надіслати на об'єкт, визначений командою PORT, список файлів у поточному каталозі
    - без попередньої команди PORT сервер надсилав би цей список хосту, від якого надійшла команда LIST
  - Внаслідок цього сервер надсилає на заданий порт TCP SYN пакет
  - Якщо порт відкритий, то з'єднання встановлюється і здійснюється передача, а FTP-клієнту надсилаються відповіді 150 і 226
    - якщо віддалений хост замість підтвердження про отримання пакета даних, що містить список файлів, надішле TCP RST-пакет, то замість відповіді 226 буде отримана 426
  - Якщо ж порт закритий, то FTP-клієнту надсилається відповідь 425
- Будь-які спроби з'єднань з хостом, який сканують, здійснюються з FTP сервера
  - отже, якщо атаку і буде виявлено, то саме останній і буде зафіксований як її джерело, а порушник залишиться не викритим
- Типовою вимогою захисту є встановлення правил фільтрації пакетів, коли з FTP-сервером можна встановлювати з'єднання із зовнішньої мережі, а з будь-яким іншим хостом захищеної мережі — ні. Але якщо при цьому хости внутрішньої мережі можуть працювати з FTP сервером, і сервер підтримує проксі-режим, то у зовнішнього порушника з'являється можливість обійти брандмауер через FTP-сервер і просканувати внутрішню мережу

Сканування захищеної мережі з використанням FTP bounce attack



SSH (Secure Shell)

- SSH — мережний протокол прикладного рівня (за деякими джерелами — сеансового рівня)
  - Основна функція - віддалене керування комп'ютером
    - Схожий за функціональністю з протоколами Telnet і rlogin, але шифрує весь трафік
  - Також забезпечує тунелювання TCP-з'єднань (port forwarding)
- Криптографічний захист протоколу SSH не фіксований, можливий вибір різних алгоритмів шифрування
- Клієнти і сервери SSH доступні для різних платформ.
  - Підтримка SSH реалізована у всіх UNIX системах, і на більшості з них в числі стандартних утиліт присутні клієнт і сервер ssh
  - Існує багато реалізацій SSH-клієнтів для не-UNIX ОС
- Версії SSH 1 і 2 є несумісними. Версія 1 має суттєві вразливості, тому перспективною і найбезпечнішою є версія 2
  - Саме її усі й використовують

Ще про SSH

- Історія
  - Перша версія протоколу, SSH-1, була розроблена в 1995 році дослідником Тату Улененом з Технологічного університету Хельсінкі, Фінляндія.
  - У 1996 році була розроблена безпечніша версія протоколу, SSH-2, несумісна з SSH-1.
  - У 2006 році протокол був затверджений робочою групою IETF в якості Інтернет-стандарту.
- Поширені дві реалізації SSH (що містять практично однаковий набір команд): пропріетарна комерційна та безкоштовна вільна
  - Вільна реалізація називається OpenSSH
    - До 2006 року 80% комп'ютерів мережі Інтернет використовувало саме OpenSSH
  - Пропріетарна реалізація розробляється організацією SSH Inc.,
    - Вона безкоштовна для некомерційного використання.

Стійкість протоколу SSH

- Стійкий до атак прослуховування трафіку («сніфінг»)
- Нестійкий до атак «людина посередині» (англ. man in the middle, MITM)
  - Для запобігання атак «людина посередині» при підключенні до хосту, ключ якого ще не відомий клієнту, клієнтське ПО показує користувачеві «зліпок ключа» (key fingerprint)
  - Рекомендується ретельно перевіряти показуваний клієнтським ПО «зліпок ключа» (key fingerprint) зі зліпком ключа сервера, бажано отриманим по надійних каналах зв'язку або особисто
- Стійкий до атак шляхом приєднання посередині (англ. session hijacking) — неможливо включитися у вже встановлену сесію або перехопити її

Стандарти протоколу SSH

- RFC 4250 - The Secure Shell (SSH) Protocol Assigned Numbers
- RFC 4251 - The Secure Shell (SSH) Protocol Architecture
- RFC 4252 - The Secure Shell (SSH) Authentication Protocol
- RFC 4253 - The Secure Shell (SSH) Transport Layer Protocol
- RFC 4254 - The Secure Shell (SSH) Connection Protocol
- RFC 4255 - Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints
- RFC 4256 - Generic Message Exchange Authentication for the Secure Shell Protocol (SSH)
- RFC 4335 - The Secure Shell (SSH) Session Channel Break Extension
- RFC 4344 - The Secure Shell (SSH) Transport Layer Encryption Modes
- RFC 4345 - Improved Arcfour Modes for the Secure Shell (SSH) Transport Layer Protocol
- RFC 4419 - Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol
- RFC 4432 - RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol
- RFC 4716 - The Secure Shell (SSH) Public Key File Format

Використання SSH

- Зазвичай SSH сервер прослуховує 22-й порт TCP
  - Для убезпечення використання рекомендують застосовувати нестандартні порти
- Команда підключення до SSH-сервера **miyserver.example.com** з командного рядка GNU/Linux або FreeBSD для користувача **petrovich** (сервер прослуховує нестандартний порт 3010):

  $ ssh -p 3010 petrovich@miyserver.example.com
- Генерація пари ключів (в UNIX-подібних ОС) здійснюється командою

  $ ssh-keygen
- Генерація пари SSH-2 RSA-ключів довжиною 4096 біта програмою puttygen під UNIX-подібними ОС:

  $ puttygen -t rsa -b 4096 -o sample
- Для використання SSH в Python існують модулі:

  python-paramiko

  python-twisted-conch

Тунелювання SSH

- SSH-тунель — це тунель, який створюється за допомогою SSH-з'єднання і використовується для шифрування тунельованих даних будь-якого протоколу
- Практична реалізація може виконуватися декількома способами:
  - Створенням Socks-проксі для програм, які не вміють працювати через SSH-тунель, але можуть працювати через Socks-проксі
  - Використанням програм, які вміють працювати через SSH-тунель
  - Створенням VPN-тунелю, що підходить практично для будь-яких програм
- Для роботи програми з конкретним сервером можна налаштувати SSH-клієнт таким чином, щоб він пропускав через SSH-тунель TCP-з'єднання, що приходять на певний TCP-порт машини, на якій запущений SSH-клієнт
- Для створення SSH-тунелю необхідна машина з запущеним SSH-сервером і доступом до цільового сервера
  - В результаті трафік між SSH-клієнтом і SSH-сервером буде зашифрований
  - Така конфігурація може також забезпечити прохід через брандмауер — якщо з локальної машини доступ до потрібного нам сервера закритий, але є доступ до деякого SSH-сервера.

Передача файлів з використанням SSH

- Є кілька механізмів передачі файлів за допомогою SSH
  - Secure Copy (SCP)
  - Rsync, повинен бути більш ефективним, ніж SCP
  - SSH File Transfer Protocol (SFTP), безпечна альтернатива FTP
    - не плутати з FTP через SSH, що також можливо!

- Загальний висновок:

  Рекомендовано застосовувати SSH замість Telnet, rlogin, rsh, а SFTP замість FTP!