Засоби та протоколи віддаленого доступу telnet та SSH

1 Мета: Вивчити особливості реалізації протоколів віддаленого доступу. Навчитись налаштовуати протоколи telnet та SSH за допомогою Cisco Packet Tracer.

2 Ключові положення

2.1 Протокол віддаленого доступу Telnet.

Telnet (англ. *TErminaL NETwork)* - мережний протокол для реалізації текстового інтерфейсу по мережі (в сучасній формі - З допомогою транспорту TCP) .Названіе «telnet» мають також деякі утиліти, що реалізують клієнтську частину протоколу.

Призначення протоколу TELNET в наданні досить загального, двонаправленого, восьмибітових байт-орієнтованого засобу зв'язку. Його основна задача полягає в тому, щоб дозволити термінальним пристроям і термінальним процесам взаємодіяти один з одним. Передбачається, що цей протокол може бути використаний для зв'язку виду термінал-термінал ("зв'язування") або для зв'язку процес-процес («розподілені обчислення»).

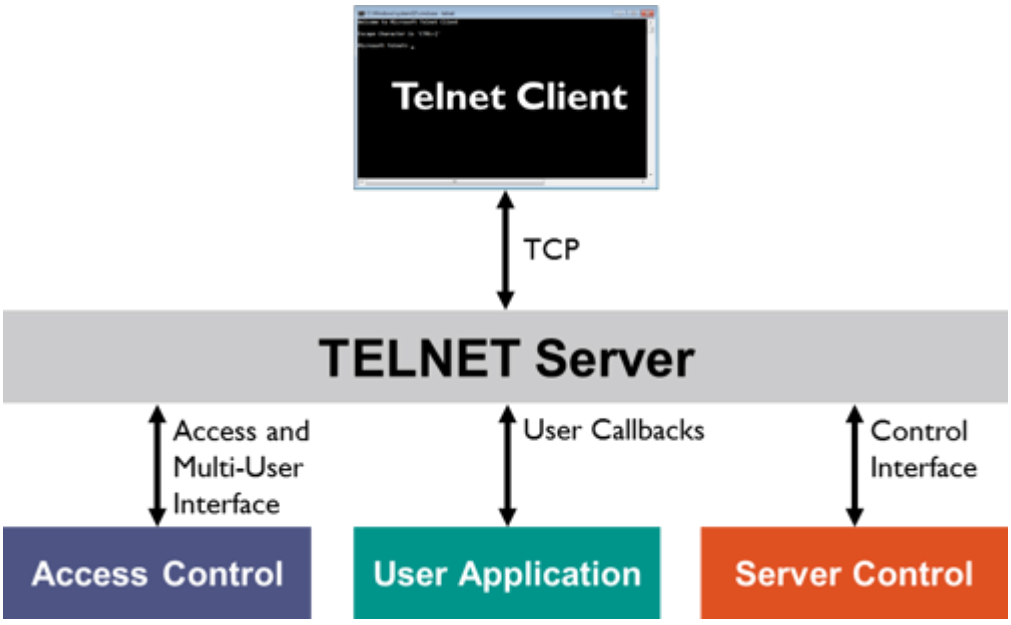


Рисунок 1 - Схема клієнт-сервер для Telnet

Пристрій

Хоча в сесії Telnet виділяють клієнтську і серверну сторону, протокол самому справі повністю симетричний. Після встановлення транспортного з'єднання (як правило, TCP) обидва його кінця грають роль «мережних віртуальних терміналів » (Network Virtual Terminal, NVT), які обмінюються двома типами даних:

- Прикладними даними (то є даними, які йдуть від користувача до текстового додатка на стороні сервера і назад);

- *Командами* протоколу Telnet, приватним випадком яких є *опції,* службовці для з'ясування можливостей і переваг сторін.

Хоча Telnet-сесії, що виконується по TCP, властивий повний дуплекс, NVT повинен розглядатися як напівдуплексний пристрій, що працює по замовчуванням в буферизованому строчном режимі. Прикладні дані проходять через протокол без змін Дані можуть піддаватися змінам (наприклад може бути скинутий старший біт) в разі, якщо на вхід одного терміналу надійшли дані, допустимість прийому яких не була підтверджена іншим. , То є на виході другого віртуального терміналу ми бачимо саме те, що було введено на вхід першого. З точки зору протоколу дані представляють просто послідовність байтів (октетів), по замовчуванням належать набору ASCII, але при включеній опції *Binary* - будь-яких. Хоча були запропоновані розширення для ідентифікації набору символів Telnet Charset Option, але на практиці ними не користуються. Всі значення октетів прикладних даних крім \ 377 (десяткове 255) передаються по транспорту як є. Октет \ 377 передається послідовністю \ 377 \ 377 з двох октетов.Ето пов'язано з тим, що октет \ 377 використовується на транспортному рівні для кодування опцій.

Опції

Протокол надає по замовчуванням міні Рисунокьну функціональність, яка розширюється за рахунок опцій. Принцип обумовлених опцій вимагає проводити переговори при включенні кожної з функцій. Одна сторона ініціює запит, а інша сторона може або прийняти, або відкинути пропозицію. Якщо запит приймається, то опція негайно вступає в силу. Опції описані окремо від протоколу як такого, і їх підтримка програмним забезпеченням довільна. Клієнту протоколу (мережному терміналу) пропонується відкидати запити на включення підтримуються і невідомих опцій.

Структура команд Telnet

Кожна команда TELNET є мультибайтних послідовність, починається з коду \ 377 (десяткове 255) «Interpret as Command" (IAC) і коду команди.Команди, що відповідають за домовленістю по опції, є трьохбайтовіми послідовностями, де третій байт є кодом опції. Наступні коди і кодові послідовності мають відповідний сенс тільки коли йдуть відразу за IAC.

Назва	Код (десятковий / шістнадцятковий)	опис
SE	240 / 0xF0	Завершує узгодження, розпочате командою SB
NOP	241 / 0xF1	Операції відсутні.
Data Mark	242 / 0xF2	Синхронізація (Synch) обміну даними. Ця команда завжди супроводжується TCP Urgent notification.
Break	243 / 0xF3	Натиснута кнопка «Break» або «Attention».
Interrupt Process	244 / 0xF4	Призупиняє, перериває, аварійно припиняє або завершує процес.
Abort output	245 / 0xF5	Відправляє сигнал Synch користувачеві.
Are you There	246 / 0xF6	Відправляє назад відповідь терміналу, що складається з друкованих символів.
Erase character	247 / 0xF7	Одержувач повинен видалити попередній символ, якщо це можливо.
Erase Line	248 / 0xF8	Стерти останній введену рядок, то є всі дані, отримані після останнього переведення рядка.

Go ahead	249 / 0xF9	Очікується передача даних.
SB	250 / 0xFA	Початок узгодження опції, вимагає передачі параметрів.
WI LL *опція*	251 / 0xFB	Вказує на бажання виконувати підтверджує, що зараз виконується вказане опція.
WILL NOT *опція*	252 / 0xFC	Вказує на відмову почати або продовжити виконувати зазначену опцію.
DO *опція*	253 / 0xFD	Запит на те, щоб інша сторона виконала або підтвердила виконання зазначеної опції.
DO NOT *опція*	254 / 0xFE	Вимога на те, щоб інша сторона зупинила виконання або підтвердила те, що зазначена опція більш не виконується.
IAC	255 / 0XFF	Байт даних 255.

Застосування

Історично Telnet служив для віддаленого доступу до інтерфейсу командного рядка операційних систем. Згодом його стали використовувати для інших текстових інтерфейсів, аж до ігор MUD і анімованого ASCII-art. Теоретично, навіть обидві сторони протоколу можуть бути програмами, а не людиною. Іноді клієнти telnet для доступу до інших протоколів на основі транспорту TCP. Протокол telnet використовується в керуючому з'єднанні FTP, то є заходити на сервер командою telnet ftp.example.net ftp для виконання налагодження і експериментів не тільки можливо, але **і** правильно (в відміну від застосування клієнтів telnet для доступу до HTTP, IRC і більшості інших протоколів).

Безпека

У протоколі не передбачено використання ні шифрування, ні перевірки достовірності даних. Тому він вразливий для будь-якого виду атак, в яких вразливий його транспорт, то є протокол TCP. Для функціональності віддаленого доступу до системи в наше час *[Коли?]* Застосовується мережний протокол SSH (особливо його версія 2), основний причиною створення якого були питання безпеки. Так що слід мати в виду, що сесія Telnet дуже беззахисна, якщо тільки не здійснюється в повністю контрольованої мережі або з застосуванням захисту на мережному рівні (різні реалізації віртуальних частнихсетей). Через ненадійність від Telnet як засобу управління операційними системами давно відмовилися.

Telnet і інші протоколи

У середовищі фахівців по технологіям internet поширене думка, що клієнт Telnet придатний для здійснення ручного доступу (наприклад, з метою налагодження) до таких протоколів прикладного рівня як HTTP, IRC, SMTP, POP3 та іншим текст-орієнтованим протоколами на основі транспорту TCP. Однак, використання клієнта telnet в якості клієнта TCP викликає такі побічні ефекти

- Клієнт може передати дані, які Ви не вводили (опції Telnet)
- Клієнти не буде приймати октет \ 377;
- Клієнт буде спотворювати октет \ 377 при передачі;
- Клієнт взагалі може відмовитися передавати октету зі старшим бітом 1.

Такі програми як netcat дійсно забезпечують чистий доступ до TCP, однак потрібні спеціальні хитрощі для передачі перекладу рядка як CR LF (що потрібно багатьом протоколам). Зазвичай клієнт Telnet по замовчуванням передає будь-який переклад рядка як CR LF, незалежно від його кодування в системі клієнта. Також для доступу до прикладних протоколів (крім FTP і, власне, Telnet) з метою налагодження є використання клієнта PuTTY в режимі «Raw» (чистий доступ до TCP) - PuTTY перетворює перекладу рядка окремо від підтримки протоколу Telnet.

Telnet was developed in 1 968 beginning with RFC 15, extended in RFC 854, and standardized as Internet Engineering Task Force (IETF) Internet Standard STD 8, one of the first Internet standards.

2.2 R утиліти

rlogin - утиліта Юнікс, що дозволяє користувачам авторизуватися на хост по мережі , використовуючи 513-ї порт протоколу TCP.

Утиліта rlogin спочатку поширювалася як частина 4.2 BSD релізу.

rlogin також є назвою протоколу прикладного шару стека протоколів TCP / IP. Минулі перевірку можуть діяти як якщо б вони були фізично бути присутнім за комп'ютером. Стандарт RFC 1258, в якому були визначені rlogin, стверджує, що «rlogin забезпечує віртуальний термінал віддаленого відповіддю і з локально керованим потоком і з належним скиданням буферів виходу.» Rlogin зв'язується з демоном rlogind на віддаленому хості.

rlogin подібний telnet, але по порівнянні з telnet його недоліками є нездатність настройки і то що він здатний з'єднуватися тільки до серверів Юнікс.

В останній час через незахищеності rlogin практично не використовується. Його заміною є SSH.

Протокол RLOGIN (англ. *Remote LOGIN* - Віддалений вхід в систему) - Протокол прикладного рівня (7ой рівень моделі OSI), частина стека TC P / IP. Дозволяє користувачам UNIX підключатися до систем UNIX на друге машинах і працювати так само, як при прямому підключенні терміналу до машіне. Цей протокол забезпечує такий же сервіс, як протокол TELNET.

Принцип роботи протоколу Rlogin

- установка зв'язку

Після установки з'єднання, клієнт посилає серверу 4 безлічі символів, що закінчуються нулями (уявлення рядки в C -подібних мовах):

1. порожня рядок (складається з одного нульового байта),
2. ім'я користувача клієнта,
3. ім'я користувача сервера,
4. тип терміналу та швидкість.

наприклад:

<Null>
bostic <null>
kbostic <null>
vt100 / +9600 <null>

Сервер повертає нульовий байт в підтвердження того, що він отримав Відправлені клієнтом дані і тепер знаходиться в режимі передачі даних.

- завершення з'єднання

Коли TCP-з'єднання закривається в будь-якому з напрямків, інший кінець повинен помітити це і провести закриття зі своєї сторони, Відновивши режими терміналів і повідомивши користувача або процеси обриву з'єднання.

Безпека

rlogin має кілька значних недоліків в системі безпеки:

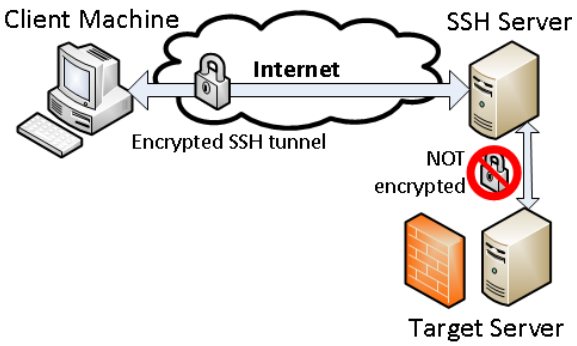
- Вся інформація, включаючи паролі, передаються без шифрування (перехоплення даних дав би повний контроль над жертвою)
- Легко скористатися файлом .rlogin (або .rhosts) в корисливих цілях (відкривається доступ до аккаунту anyone, в якого немає пароля) - По цій причині безліч корпоративних системних адміністраторів забороняють .rlogin файли і активно шукають правопорушників в їх мережах
- Протокол частково довіряє клієнту rlogin віддаленої машини, надаючи йому інформацію чесно (включаючи порт і ім'я хоста). Таким чином, Зловмисник може скористатися цим і отримати доступ, в силу того, що цей протокол ніяк не поділяє віддалений хости на зони довіри
- Загальна практика монтування домашніх директорій користувачів через NFS (Network File System) піддає rlogin атакувати шляхом підроблення файлів .rhosts - Це означає, що єдиною захистом в цьому випадку є безпеку самої NFS.

2.3 Протокол RSH

RSH розшифровується як Remote SHell - протокол, що дозволяє підключатися віддалено до пристрою і виконувати на ньому команди. Зазвичай RSH-клієнт запускається в консолі, і відповідь на команду виводить прямо в екран консолі. Протокол RSH не є захищеним - дані про авторизації і передаються дані не шифруються.

2.4 Протокол SSH

SSH (Secur e Shell) - мережний протокол , що дозволяє виробляти віддалене управління комп'ютером і передачу файлів. Схожий по функціональності з протоколом Telnet і rlogin , проте використовує алгоритми шифрування переданої інформації.



Криптографічний захист протоколу SSH не фіксована, можливий вибір різних алгоритмів шифрування . Клієнти і сервери, що підтримують цей протокол, доступні для різних платформ. Крім того, протокол дозволяє не тільки використовувати безпечний віддалений she ll на машині, але і туннелировать графічний інтерфейс - X Tunnelling (тільки для Unix -подібних ОС або додатків, що використовують графічний інтерфейс X Window System). SSH також здатний передавати через безпечний канал (Port Forwarding) будь інший мережний протокол, забезпечуючи (при належному конфігурації) можливість безпечної пересилки не тільки X-інтерфейсу, але і, наприклад, звуку.

Підтримка SSH реалізована у всіх UNIX системах, і на більшості з них в числі стандартних утиліт присутні клієнт і сервер ssh. Існує безліч реалізацій SSH-клієнтів і для НЕ-UNIX ОС. Велику популярність протокол отримав після широкого розвитку sniffer'ов , як альтернативне небезпечному ТЕЛНЕТ рішення для управління важливими вузлами.

На даний момент відомо дві гілки версій - 1 і 2. Однак гілка 1 зупинена, так як в кінці 90-х в ній було знайдено багато вразливостей, деякі з яких до цих пір накладають серйозні обмеження на її використання, тому перспективною, що розвивається і найбільш безпечної є версія 2.

Стандарти та програмні реалізації

Перша версія протоколу, SSH-1, була розроблена в 1995 році дослідником Тату Улененом з Технологічного університету Хельсінкі, Фінляндія.SSH-1 був написаний для забезпечення більшої конфіденційності, ніж протоколи rlogin, telnet і rsh. У 1996 році була розроблена безпечніше версія протоколу, SSH-2, несумісна з SSH-1. Протокол набув ще більшу популярність, і до 2000 році у нього було близько двох мільйонів користувачів. В даний час під терміном «SSH» зазвичай мається в виду саме SSH- 2, так як перша версія протоколу огляду істотних недоліків зараз практично не застосовується.

У 2006 році протокол був затверджений робочою групою IETF в якості Інтернет-стандарту.

Однак, в деяких країнах (Франція, Росія, Ірак і Пакистан) до сих пір потрібно спеціальне дозвіл в відповідних структурах для використання певних методів шифрування, включаючи SSH. Див закон Російської Федерації «Про федеральних органах урядової зв'язку та інформації» (закон втратив силу з 1 липня 20 03 року в зв'язку з прийняттям федерального закону від 30.06.2003 № 86-ФЗ).

Поширені дві реалізації SSH: пропріетарна комерційна і безкоштовна вільна. Вільна реалізація називається OpenSSH. До 2006 року 80% комп'ютерів мережі Інтернет використовувало саме OpenSSH. Пропрієтарна реалізація розробляється організацією SSH Inc., вона безкоштовна для некомерційного використання. Ці реалізації містять практично однаковий набір команд.

Протокол SSH-2, в відміну від протоколу telnet, стійкий до атак прослуховування трафіку («сніфінг»), але нестійкий до атак «людина посередині» .Протокол SSH-2 також стійкий до атак шляхом приєднання посередині (англ. Session hijacking) - неможливо включитися в уже встановлену сесію або перехопити її.

Для запобігання атак «людина посередині» при підключенні до хосту, ключ якого ще не відомий клієнту, клієнтське ПЗ показує користувачеві "зліпок ключа" (key fingerprint). Рекомендується ретельно перевіряти показується клієнтським ПО "зліпок ключа" (key fingerp rint) зі зліпком ключа сервера, бажано отриманим по надійним каналах зв'язку або особисто.

Підтримка SSH реалізована у всіх UNIX-подібних системах, і на більшості з них в числі стандартних утиліт присутні клієнт і сервер ssh.Существует безліч реалізацій SSH-клієнтів і для НЕ-UNIX ОС. Велику популярність протокол отримав після широкого розвитку аналізаторів трафіку і способів порушення роботи локальних мереж, як альтернативне небезпечному протоколу Telnet рішення для управління важливими вузлами.

Для роботи по SSH потрібен SSH-сервер і SSH-клієнт. Сервер прослуховує з'єднання від клієнтських машин і при встановленні зв'язку виробляє аутентифікацію, після чого починає обслуговування клієнта. Клієнт використовується для входу на віддалену машину і виконання команд.

Для з'єднання сервер і клієнт повинні створити пари ключів - відкритих і закритих - і обмінятися відкритими ключами. Зазвичай використовується також і пароль.

SSH-сервери

BSD

OpenSSH

Linux

dropbear, lsh-server, openssh-server, ssh

Windows

freeSSHd, copssh, Wi nSSHD, KpyM Telnet / SSH Server, MobaSSH, OpenSSH через Cygwin

SSH-клієнти і оболонки

- GNU / Linux, * BSD: kdessh, lsh-client, openssh-client, putty, ssh, Vinagre
- MS Windows і Windows NT: PuTTY, SecureCRT, ShellGuard, Axessh, ZOC, SSHWindows, ProSSHD, XShell
- MS Windows Mobile : PocketPuTTy, mToken, sshCE, PocketTTY, OpenSSH, PocketConsole
- Mac OS: NiftyTelnet SSH
- Symbian OS: PuTTY
- Java: MindTerm, AppGate Security Server
- J2ME: MidpSSH
- iOS: i-SSH, ssh (в комплекті з Terminal)
- Android: connectBot
- Blackberry: BBSSH
- Maemo 5: OpenSSH

Техніка безпеки використання SSH

- Заборона віддаленого root-доступу.
- Заборона підключення з порожнім паролем або відключення входу по паролю.
- Вибір нестандартного порту для SSH-сервера.

- Використання довгих SSH2 RSA-ключів (2048 біт і більше). Системи шифрування на основі RSA вважаються надійними, якщо довжина ключа не менш 1024 біт.
- Обмеження списку IP-адрес, з яких дозволений доступ (наприклад, настроюванням брандмауера).
- Заборона доступу деяких потенційно небезпечних адрес.
- Відмова від використання поширених або широко відомих системних логінів для доступу по SSH.
- Регулярний перегляд повідомлень про помилки аутентифікації.
- У становленнЯ систем виявлення вторгнень (IDS - Intrusion Detection System).
- Використання пасток, підробляють SSH-сервіс (honeypots).

Технічна інформація про протоколі

SSH зазвичай використовується для входу на віддалену машину і виконувати команди, але він також підтримує тунельний протокол, портове експедирування TCP і UDP порт | TCP порти. він може передавати файли з допомогою відповідних SSH File Transfer Protocol | SSH File Transfer (SFTP) або Secure Copy (SCP) протоколів SSH використовує клієнт-сервер модель.

SSH програма зазвичай використовується для підключення прийняття віддалених підключень. І то і інше зазвичай присутній на більшості сучасних операційних систем, включаючи Mac OS, більшість розподілів GNU / Linux, OpenBSD, FreeBSD, [[NetBSD]], Solaris і OpenVMS. Відомо, що Windows є одним з небагатьох сучасних / серверних операційних систем, які не включають SSH по замовчуванням.

SSH-сервер зазвичай прослуховує з'єднання на TCP-порту 22. Специфікація протоколу SSH-2 міститься в RFC +4251. Для аутентифікації сервера в SSH використовується протокол аутентифікації сторін на основі алгоритмів електронно-цифровий підпис RSA або DSA.Для аутентифікації клієнта також може використовуватися ЕЦП RSA або DSA, але допускається також аутентифікація з допомогою пароля (режим зворотної сумісності з Telnet) і навіть ip-адреси хоста (режим зворотної сумісності з rlogin). Аутентифікація по паролю найбільш поширена, вона безпечна, так як пароль передається по зашифрованому віртуального каналу. Аутентифікація по ip-адресою небезпечна, цю можливість частіше відключають. Для створення загального секрету (сеансового ключа) використовується алгоритм Діффі - Хеллмана (DH). Для шифрування переданих даних використовується симетричне шифрування, алгоритми AES, Blowfish або 3DES. Цілісність переданих даних перевіряється з допомогою CRC32 в SSH1 або HMAC-SHA1 / HMAC-MD5 в SSH2.

Для стиснення шифруючих даних може використовуватися алгоритм LempelZiv (LZ77), який забезпечує такий же рівень стиснення, і архіватор ZIP. Стиснення SSH включається тільки по запиту клієнта, і на практиці використовується рідко.

Передача файлів з використанням SSH

Є кілька механізмів передачі файлів з допомогою захищених протоколів Shell.

- Secure Copy (SCP), який стався від RCP Протокол по SSH

- Rsync, повинен бути більш ефективним, ніж SCP
- SSH File Transfer Protocol (SFTP), безпечна альтернатива FTP (Не плутати з FTP через SSH)
- Файли передаються по протоколу оболонки (так званий FISH), випущений в 1998 році, який перетворився з Unix Shell команди через SSH

SSH-2 протокол має внутрішню архітектуру (визначено в RFC 4251) з чітко розділеними шарами. До них відносяться:

- *Транспортний шар* (RFC +4253). Цей шар обробляє початковий обмін ключами, перевірку справжності сервер, встановлює шифрування, стиснення і перевірку цілісності даних. Він надає в вищий шар інтерфейс для відправки і отримання текстових пакетів розміром до 32768 байт кожен (цей розмір може бути збільшений в певній реалізації). Транспортний рівень також організовує повторний обмін ключами. Як правило, це відбувається після передачі 1 Гб даних або коли пройшла 1:00, що швидше настане.
- *Шар аутентифікації користувача* (RFC 4252). Цей шар обробляє перевірку справжності клієнта і надає ряд методів аутентіфікації.Аутентіфікація є *клієнто-орієнтованої: коли користувач отримує запит на введення пароля, то це може бути запит SSH клієнта, а не сервера.* Сервер просто відповідає на запити клієнта про автентичності. Широко застосовуються методи аутентифікації користувачів включають наступне:
    - *Пароль: Метод простий пароль аутентифікації, в тому числі засіб, що дозволяє пароль, щоб бути змінені.* Цей метод не реалізований всіма програмами.
    - *з відкритим* ключем: метод відкритого ключа перевірки справжності на основі, як правило, підтримують по крайней мере, DSA або RSA пари ключів, з іншими реалізаціями також підтримує X.509 сертифікати.
    - *Інтерактивний* (RFC +4256): універсальний метод, коли сервер відправляє один або кілька запитів введення інформації і клієнт відображає їх і відправляє назад відповідь введені в користувачем. Використовується для забезпечення одноразовий пароль аутентифікації, такі як S / Key або SecurID. Використовується деяких конфігураціях OpenSSH, коли PAM є основним постачальником хост аутентифікації для забезпечення ефективної аутентифікації по паролю, що іноді призводить до нездатності увійти в систему з клієнтом, який підтримує тільки прості *пароль* метод перевірки автентичності.
    - GSSAPI методи аутентифікації, які забезпечують розширюється схеми для виконання SSH аутентифікації з використанням зовнішніх пристроїв, таких як Kerberos 5 або NTLM, забезпечуючи [[Single Sign On]] Можливість SSH сесій. Ці методи, як правило, здійснюються комерційними реалізаціями SSH для використання в організаціях, хоча OpenSSH дійсно робоча реалізації GSSAPI.
- *З'єднання шарів* (RFC 4254). Цей шар визначає поняття канали, канал запити і глобальний запитів з допомогою якої SSH послуги. Одне з'єднання SSH можна розмістити кілька каналів одночасно, кожна передача даних в обох напрямках. Канал запити використовуються для релейний вихід за смугу каналу конкретні дані, такі, як змінився розмір вікна терміналу або код виходу з серверного процесу. SSH клієнт запитує сервер на стороні порту, який буде спрямовуватися з використанням глобальної запитом. Стандартні типи каналів включають в себе:

- *Оболонка* для терміналу, SFTP і Exec запитів (в тому числі SCP трансфертів)
- *Прямий* TCP / IP для клієнт-сервер передаються з'єднання
- *Спрямований-TCP / IP* для сервер-клієнт направляється з'єднань

- SSHFP DNS-запис (RFC чотири тисячі двісті п'ятдесят-п'ять) надає громадськості відбитки пальців ключа хоста для того, щоб допомогти в перевірці достовірності господаря.

Це відкрита архітектура забезпечує значну гнучкість, дозволяючи SSH, який буде використовуватися для різних цілей, крім безпечної оболонки. Функціональність транспортного рівня тільки порівнянна з Transport Layer Security (TLS) шар аутентифікації користувачів вельми розширюваної з допомогою призначених для користувача методів аутентифікації і з'єднання шарів забезпечує можливість мультиплексування багатьох середніх сесій в одне з'єднання SSH, функція порівнянна з BEEP і не доступні в TLS.

SSH-тунелювання

SSH-тунель - це тунель, який створюється з допомогою SSH-з'єднання і використовується для шифрування тунельованіх даних. Використовується для того, щоб убезпечити передачу даних в Інтернеті (аналогічне призначення має IPsec). Особливість полягає в тому, що зашифровані трафік-небудь протоколу шифрується на одному кінці SSH-з'єднання і розшифровується на іншому. Тунель SSH може забезпечити безпечний шлях через Інтернет, через брандмауер на віртуальній машині. [2]

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 додатків, які вміють працювати через SSH-тунель.
- Створенням VPN-тунелю, підходить практично для будь-яких додатків.

Якщо програма працює з одним певним сервером, можна налаштувати SSH-клієнт таким чином, щоб він пропускав через SSH-тунель TCP-з'єднання, приходять на певний TCP-порт машини, на якій запущений SSH-клієнт. Наприклад, клієнти Jabber підключаються по замовчуванням на порт 443. Тоді, щоб налаштувати підключення до сервера Jabber через SSH-тунель, SSH-клієнт налаштовується на перенаправлення підключень будь-якого порту локальної машини (наприклад, з порту 4430) на віддалений сервер (наприклад , jabber. example.com і порт 443):

$ Ssh -L 4430: jabber.example.com 443 somehost

В даному випадку Jabber-клієнт налаштовується на підключення до порту 4430 сервера localhost (якщо ssh-клієнт запущений на тій же машині що і Jabber-клієнт).

Для створення ssh-тунелю необхідно машина з запущеним ssh-сервером і доступом до jabber.example.com. Така конфігурація може використовуватися в разі, якщо з локальної машини доступ до jabber.example.com закритий файерволом, але є доступ до деякого ssh-сервера, у якого обмеження доступу в Інтернет відсутні.

3 Контрольні питання:

1. Що таке telnet?
2. На якому рівні в OSI / ISO працює telnet?
3. Можна чи використовувати Telnet на будь-якому комп'ютері?

4. Є чи Telnet єдиним способом підключення до іншого комп'ютера?
5. Чим відрізняється протокол RSH від SH?
6. Що таке SSH?
7. На якому рівні в OSI / ISO працює SSH?
8. Які аналоги SSH існують?
9. Яке програмне забезпечення необхідно для роботи SSH?
10. Для чого застосовують шифрування даних, які передають в мережі?

4 Лабораторне завдання

Налаштування протоколів telnet та SSH в Cisco Packet Tracer

1. **Побудуйте мережу** зазначену на Рисунку 4.1, самостійно задайте адресацію на PC та порти маршрутизаторів.

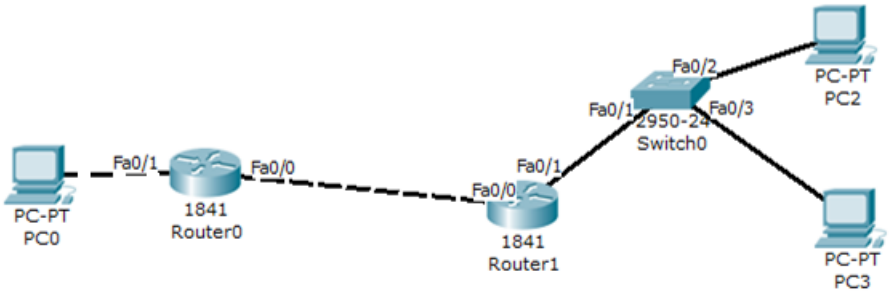
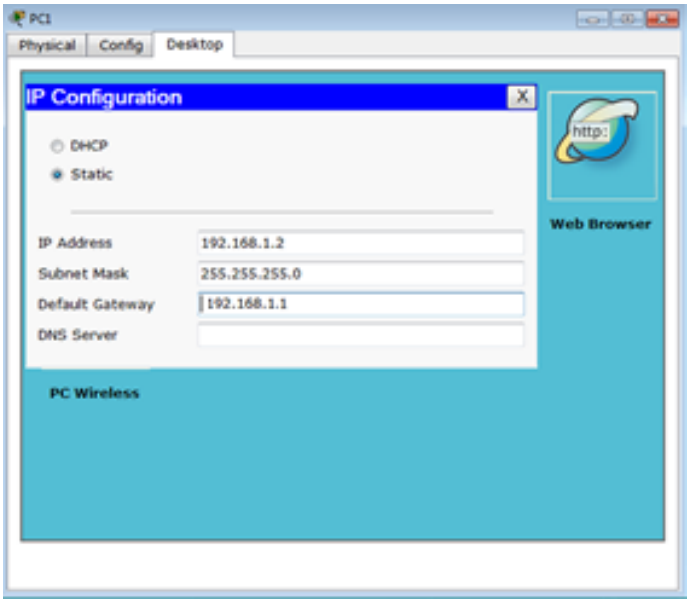


Рисунок 4.1 - Інформаційна мережа

Для прикладу налаштуємо адресацію лівого сегменту мережі (PC 0 і Router0)

Налаштовуємо IP-адреса комп'ютера через Desktop.



Налаштовуємо IP-адреса маршрутізатора.
Команда для переходу в **режим конфігурації інтерфейсу** FastEthernet 0/0:

Router (config) # interface fa0 / 0

За замовчуванням всі інтерфейси відключені (стан administratively down). Включаємо інтерфейс:

Router (config-if) #no shutdown

Налаштуємо IP-адреса:

Router (config-if) #ip address 192.168.1.1 255.255.255.0

shutdown - означає "вимкнути інтерфейс". Відповідно, якщо ви хочете скасувати дію команди, то використовуйте слово **no** перед нею. Це правило загальне для CLI і може бути застосовано до більшості команд.

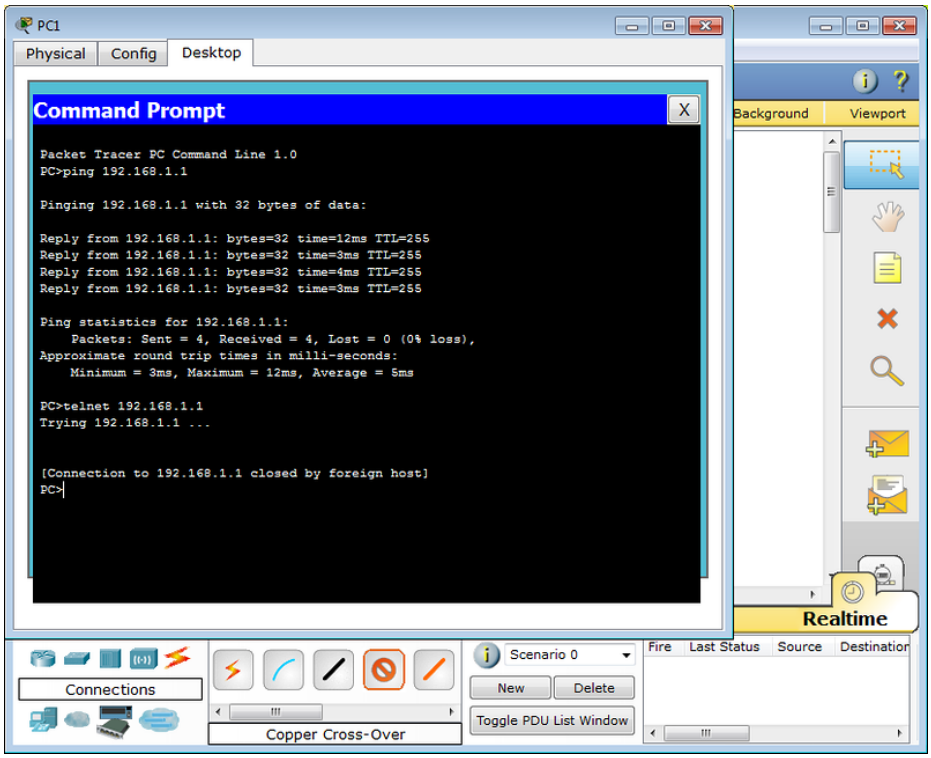
2. Налаштуйте адресацію всієї мережі та перевірте працездатність мережі (надішліть повідомлення з кожного пристрою кожному) .

3 Налаштування доступу по Telnet

З цього-то режиму ми і налаштуємо інтерфейс для підключення комп'ютера через telnet:

І пробуємо підключитися, вибравши Command Prompt в панелі Desktop:

PC> telnet 192.168.1.1



Як і очікувалося, циска не пускає без пароля. У реальному житті зазвичай видає фразу "Password required, but none set"

Паролі

Підключення по telnet або ssh називається віртуальним терміналом (vt) і налаштовується таким чином:

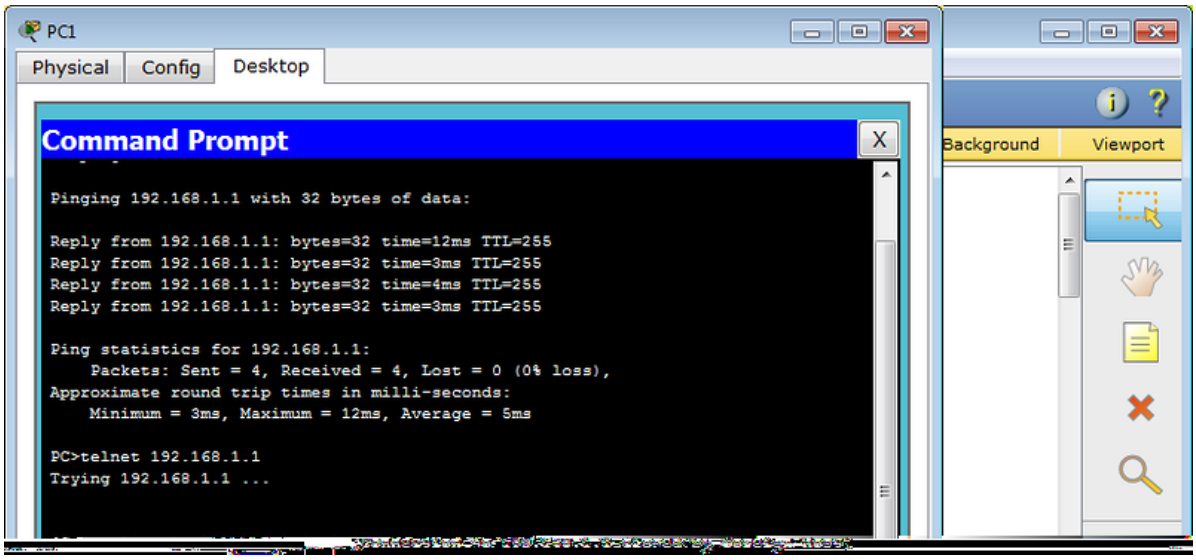
Router (config) #line vty 0 4

Router (config-line) #password <i> cisco </ i>

Router (config-line) #login

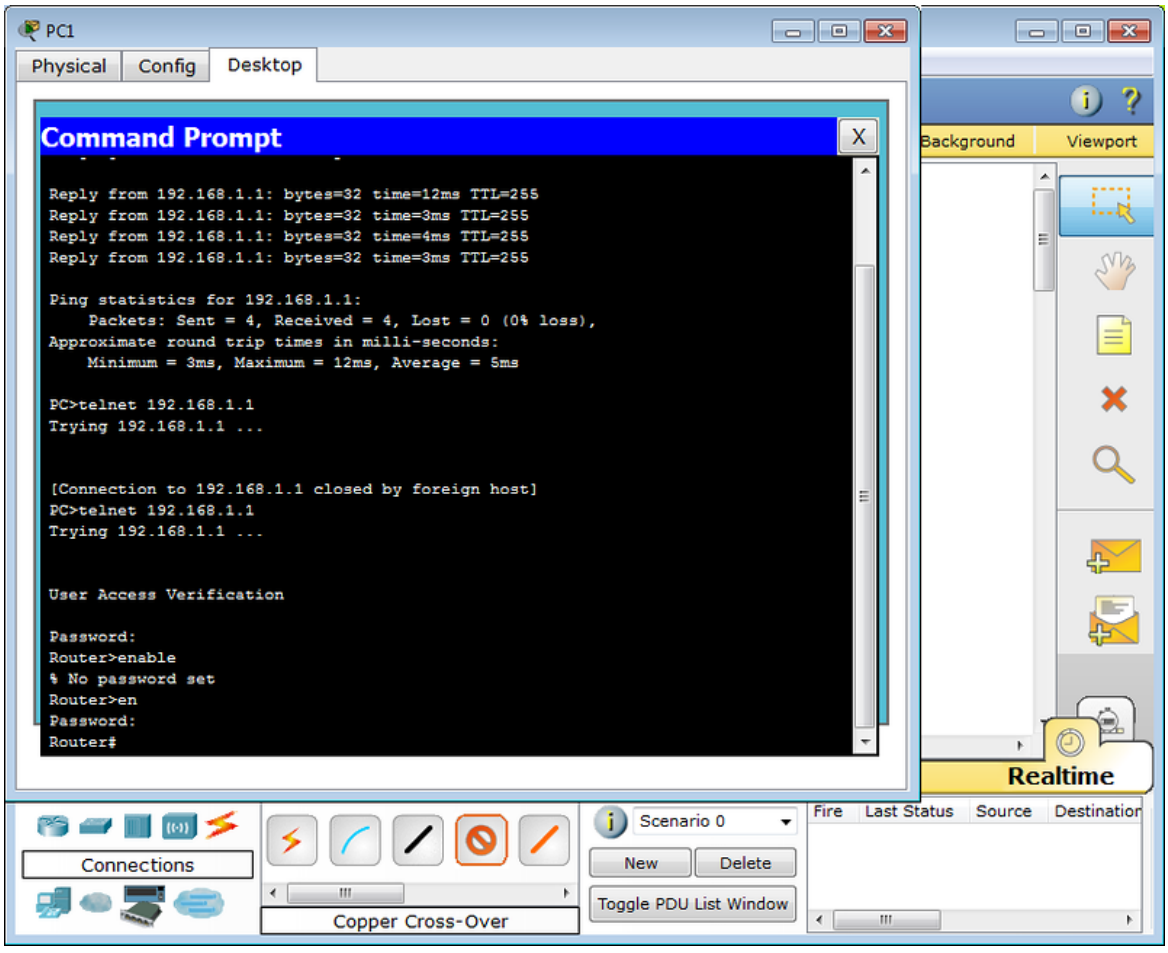
0 4 - це 5 призначених для користувача віртуальних терміналів = telnet сесій.

Цього вже достатньо, щоб потрапити в призначений для користувача режим, але недостатньо для привілейованого режиму :



Налаштуємо пароль для enable-режиму:

Router (config) #enable secret <i> test </ i>



Secret відрізняється від *password* приблизно тим же, чим ssh від telnet. При налаштуванні *secret* пароль зберігається в зашифрованому вигляді в конфігураційному файлі, а *password* - в відкритому. Тому рекомендується використання *secret* .

Якщо ви все-таки задаєте пароль командою **password** , то слід застосувати так само **service password-encryption** , тоді ваш пароль в конфігураційному файлі буде зашифрований:

line vty 0 4

password 7 08255F4A0F0A0111

Зверніть увагу:

Зараз принятно налаштовувати доступи не через віртуальні термінали, а командами **#username** і **#aaa new-model** . У версіях PT вище 5.3.2 релізована саме ця функція.

Для цього потрібно виконати:

Router (config) #aaa new-model

Router (config) #username admin password тисячі двісті тридцять чотири

Перша команда служить для активації нової моделі AAA (Authentication, Authorization, Accounting). Це потрібно для того, щоб була можливість використовувати для аунтетіфікації на пристрої RADIUS або TACACS сервер. Якщо окремо це не налаштоване, то буде використовуватися локальна база користувачів, що задається командою **username** .

***Будьте уважні** : пріоритет команди aaa new-model вище, ніж команд віртуальних терміналів і тому навіть не дивлячись на те, що у вас налаштований password в режимі line vty, якщо у вас не буде користувачів в локальній базі, зайти на пристрій віддалено вже не вийде.*

Тепер при підключенні маршрутизатор запросить ім'я користувача та відповідний йому пароль.

При більш глибокої налаштування *line vty* існує одна небезпека.

Є такий параметр: **access-class** . Його настройка дозволяє обмежити IP-адреси, з яких можливо підключення.

При роботі з access-list'амі і іншими небезпечними речами, неправильна настройка яких може позбавити вас доступу до пристрою, можна використовувати чудову команду **reload in *min*** , де *min* час в хвилинах. Ця команда перезавантажить пристрій по закінченні зазначеного часу, якщо її не перервати командою **reload cancel** .

Задати паролем доступ через консольний порт, необхідно команди

Router (config) #line console 0

Router (config-line) #login

Router (config-line) #password <i> cisco </ i>

Privilege Level

Ще один важливий момент, якому в статтях приділяють Рисуноко уваги: privelege level.

Як зрозуміло з латинського звучання - це рівень прав користувача. Всього існує 16 рівнів: 0-15.

privilege level 0 - це команди disable, enable, exit, help і logout, які працюють у всіх режимах

privilege level 1 - Це команди призначеного для користувача режиму, то є як тільки ви потрапляєте на цискго і побачите запрошення **Router>** ви маєте рівень 1.

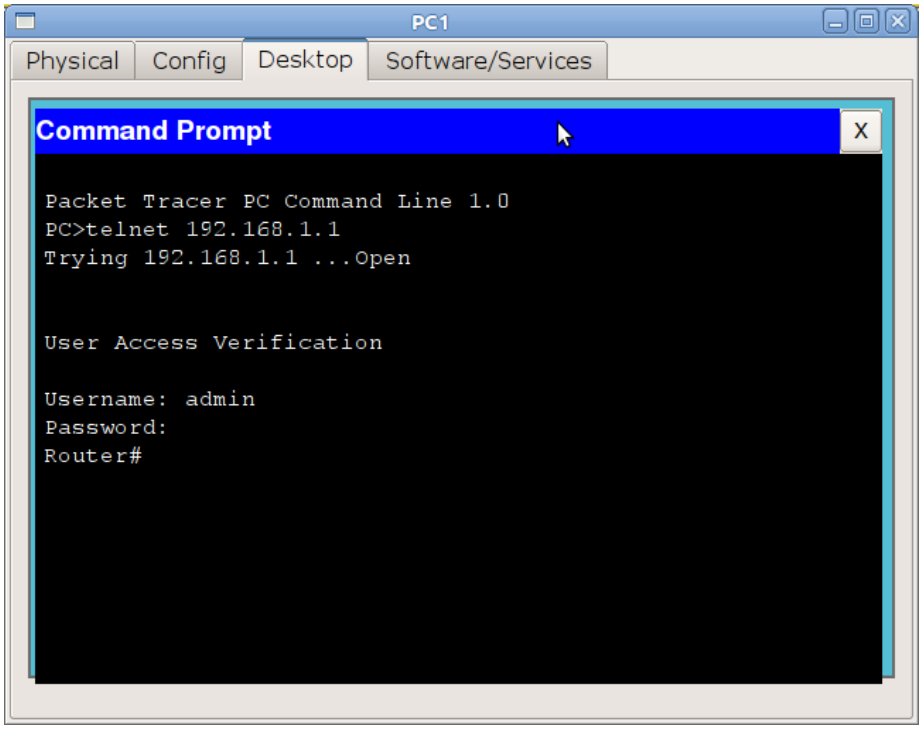
privilege level 15 - Це команди привілейованого режиму, на зразок, як root в UNIX'ах

Встановіть privelege level 15 і просматріте запишіть в протокол наслідки.

Router (config) #line vty 0 4

Router (config-line) privilege level 15

Після входу на маршрутизатор при такій настройці ви відразу побачите Router # зі всіма витікаючими правами.



Всі рівні з 2 по 14 налаштовуються вручну. Те є, наприклад, ви можете дати добро користувачеві з privelege level 2 на виконання команди **show running-config**

Налаштування прав для конкретного користувача

Налаштувати права для конкретного користувача допоможе вже згадана раніше команда **username**

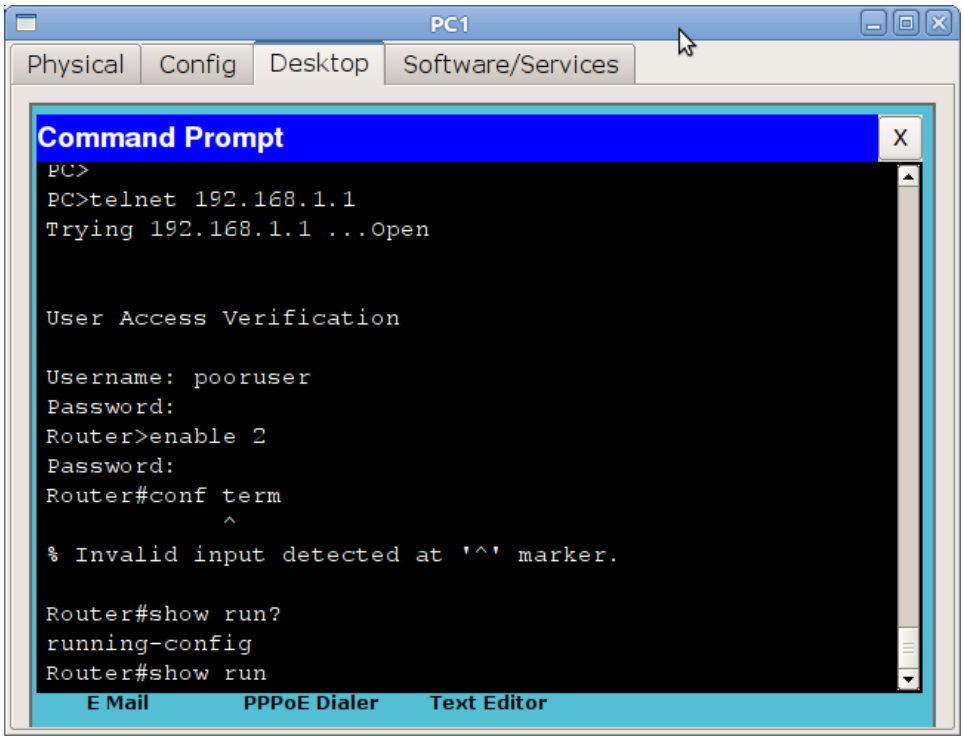
Router (config) #username pooruser privilege 2 secret poorpass

Router (config) #privilege exec level 2 show running-config

Router (config) #enable secret level 2 l2poorpass

У першій рядку призначаємо рівень прав користувачеві, у другій команду, дозволену для цього рівня, в третій задаємо пароль для входу в привілейований режим з цим рівнем.

Після цього з призначеного для користувача режиму ви можете виконати команду **enable 2** і ввівши пароль *l2poorpass* потрапити в привілейований режим, в якому будуть доступні всі команди рівня 1 + команди рівня 2.



4 Налаштування по протоколу SSH

Не можна **не** згадати про те, що telnet - протокол незахищений і передає пароль і дані в відкритому вигляді. З допомогою будь-якого аналізатора пакетів можна обчислити пароль.

Тому вкрай рекомендуємо використовувати ssh - будь-які пристрої cisco з не самої урізаною прошивкою здатні виступати ssh-сервером.

Наступний набір команд дозволить вам включити ssh і відключити доступ по telnet:

Router (config) #hostname R0

Router (config) #ip domain-name cisco-dmn

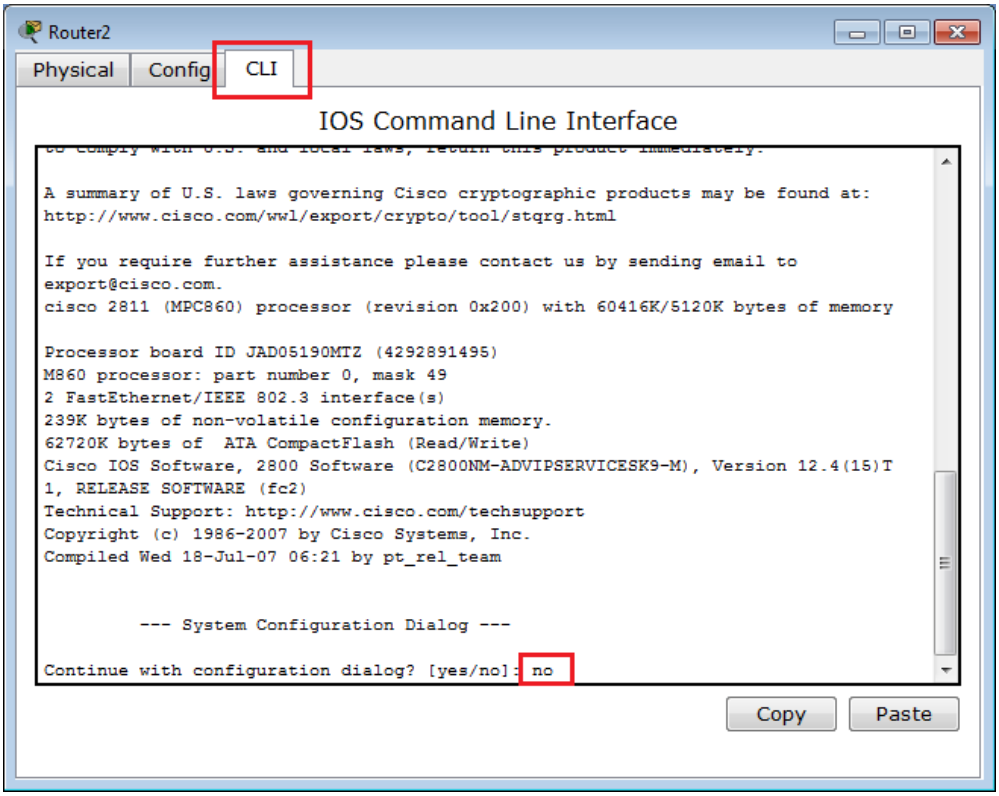
Router (config) #crypto key generate rsa

Router (config) #line vty 0 4

Router (config-line) #transport input ssh

Ім'я хоста повинно відрізнятися від Router, обов'язково повинно бути задано ім'я домену. Третьою рядком генерується ключ і далі дозволяється тільки ssh. Довжина ключа повинна бути більш 768 біт, якщо ви бажаєте використовувати ssh версії 2 .

Використовуючи PT налаштовувати обладнання **не** через термінал або телнет, а безпосередньо через CLI пристрою, яке викликається кліком по іконці роутера - так зручніше:



Примітка. Якщо під час виконання роботи Ви забули пароль тоді потрібно його скинути:

Практично на будь-якому мережному пристрої є можливість скинути пароль, маючи фізичний доступ, але Ви працюєте в PT, тому це зробити неможливо, тоді:

1) Потрібно підключитися до пристрою консольним кабелем,

2) Відправити його в ребут (кнопкою чи командою **#reload**)

3) Коли на екрані побіжить така строчка ######## ... ###, що означає завантаження образу (40-60 секунд після включення), необхідно відправити сигнал *Break* (кнопка на клавіатурі). Ви потрапляєте в режим ROMMON.

4) У цьому режимі введіть команду: **confreg 0x2142** , вона змусить пристрій ігнорувати startup-config при завантаженні.

5) Введіть **reset** для перезавантаження

6) Після завантаження running-config буде невинно чистим, а startup-config містить як і раніше останню збережену конфігурацію. Зараз саме час поміняти пароль або злити конфиг.

7) Саме важливе: **поверніть назад регістри** :

Router (config) # config-register 0x2102