Telnet, SSH and DNS

Unit 06 - Hands-On Networking - 2018

Prof. Dr.-Ing. Thorsten Herfet, Andreas Schmidt, Pablo Gil Pereira

Telecommunications Lab, Saarland Informatics Campus, 20th Feb. 2018

Recap

- Application layer is where **services** are provided to **users**.
- Resolving names to machine-readable data is a service provided by **DNS**.
- **Accessing systems remotely** is a service that is particularly important in todays widespread network infrastructures.

Telnet

TELetype NETwork (Telnet) (RFC854)

Defintion: Remote terminal protocol of the Internet architecture.

Services:

- Allows to interact with a remote system (*virtual terminal*).
- Behaves as if terminal was directly connected to the system.

Requirements: non-real-time protocol (even though it benefits from low delay).

Transport: TCP on port 23

Details:

- Uses a channel of 8 bits (1 byte) to send 7-bit ASCII data (RFC20)
- High bit set to `1` indicates control message (e.g. the `AYT` command).
  AYT = Are you there?
- Telnet intersperses data and control in the same channel.
  An approach commonly called in-band.

Telnet | History

- Again this protocol stems from the **old times** of the Internet (1969).
- **Nobody cared about security**, because only few people had access and everyone trusted each other.
- Telnet does not *encrypt messages*.
  Sensitive information (e.g. credentials) is transmitted in plain text.
- Telnet does not *authenticate messages*.
  Sender can be forged, messages can be changed on their way.
- Several common Telnet implementations have known vulnerabilities.

⚠ **Telnet is fundamentally broken regarding security!**

→ You should **not use** it for remote login.

Telnet | Current Uses

Why is it still there?

- Debugging: Test simple TCP Protocols, e.g. SMTP, POP3, IMAP, HTTP, ...
  See tutorial sheet.
- Port Scanning: Testing if port is open or not.
  If nothing such as `nmap` is available.
- Switch / Router Configuration and Debugging
- Emergency-Modes of certain Devices
- Legacy Systems

Telnet | Usage

Logging into `host` on `port` (last part can be omitted if default port 23 is used):

```
$ telnet host [port]
```

Secure SHell (SSH)

SSH (RFCs 4250 - 4256 and more)

Definition: Cryptographic network protocol for operating network services securely over an unsecured network.

Services:

- secure any kind of communication
- remote login to another server
- tunneling, forwarding TCP ports, forwarding X11 connections, ...

Requirements: see Telnet

Transport: TCP on port 22

Details:

- Client-server architecture (remember pattern!)
- Uses public-key cryptography (RSA, Elliptic Curves, etc.)
- Unknown public keys must be approved and should not change (key pinning).
  Validity checked using out-of-band methods.

SSH | Authentication Modes

𝙸 Passwords

- Same process as **normal system login** through attached console.
- Get prompted for **username**, type in in clear.
- Get prompted for **password**, type without feedback on `stdout`.

🔑 Keys

- Use public-key (asymmetric) cryptography.
- Store the private key on the local machine (secret) and the public key on the remote server to be accessed.
- The server also has private and public keys to prove its identity.
- Upon login, a cryptographic protocol runs, granting access if user is valid and also the remote host can be authenticated.

SSH | Configuration and Operation Files

Server Configuration (`/etc/ssh/sshd_config`)
Ports, Protocols, Keys, Authentication Methods, ...

Client Configuration

- **Options**:
  - Host Defaults (`User`, `IdentityFile`, `Port`)
  - Forwardings (ports, X11, ...)

- **System-wide Configuration** (`/etc/ssh/ssh_config`)
  Global SSH configuration for all users.

- **User-Specific Configuration** (`/home/<user>/.ssh/`) contains the following **files**:
  - `config`: Contains the local SSH config of the user (overwriting global settings).
  - `authorized_keys`: Stores public keys that can be used to login as `<user>`.
  - `*.pub`: Public keys of local identities.
  - `known_hosts`: Fingerprints of remote hosts that have been approved.
    Upon first access, key is presented and has to be validated by user.

SSH | Commands

Execute Command / Login to Shell (if no command specified)

```
ssh [user@]hostname [-p port] [command]
```

Copy a File (much like `cp` but with login information)

```
scp [[user@]host1:]file1 ... [[user@]host2:]file2
```

SSH | Configure Passwordless Login

Generating an SSH key pair:

```
ssh-keygen -t rsa -C "YourFancyKeyName" -b 4096
```

When asked for name, you might keep the standard name `id_rsa`.
This default identity can then be used without further configuration.

The `id_rsa.pub` contains the public key, which can be pasted in on the remote system (e.g. SSH-Server, GitHub, etc.).

The remote account can be configured using `ssh-copy-id`:

```
ssh-copy-id [-i [identity_file]] [user@]machine # by default, the `id_rsa.pub` is used.
```

DNS

dig

Installation

```
apt-get install bind9utils
```

Usage:

- Query DNS server `9.9.9.9` (free, recursive, DNS platform) for `uni-saarland.de`:

```
$ dig @9.9.9.9 uni-saarland.de
```

- Query for A record (works for all types, e.g. MX, ...):

```
$ dig uni-saarland.de A
```

- Reverse lookup:

```
$ dig -x 192.168.2.17
```

- Get a full trace of a DNS resolution:

```
$ dig +trace cms.nt.uni-saarland.de
```

Wrap-Up

HON

Questions?

⚠ Action Points

- Solve the task sheet.
- Create yourself an SSH identity if you want to become
  - a serious software-developer (`git push`, etc.) or
  - system administrator (don't administer by foot).
- `dig` into DNS a bit further, e.g. by querying the University's DNS server
  For instance for MX records and CNAMEs.
- If you are bored, then `telnet towel.blinkenlights.nl`

📖 Further Reading

- SSH Essentials by DigitalOcean
- SSH Server Config by ubuntu.com
- `telnet`, `ssh`, `scp`, and `dig` man pages
- DiG Tutorial