Přednáška 10

X Window. Secure shell.

X Window systém I

- Systém pro správu oken.
- **Poskytuje nástroje pro tvorbu GUI** (Graphical User Interface) **a grafických aplikací**.
- Nezávislý na hardwaru.
- Transparentní vůči sítím a OS.

- **Historie**
  - 1984 počátek vývoje (Massachusetts Institute of Technology)
  - 1985 verze 9
  - 1986 X10R4
  - 1987 X11R1

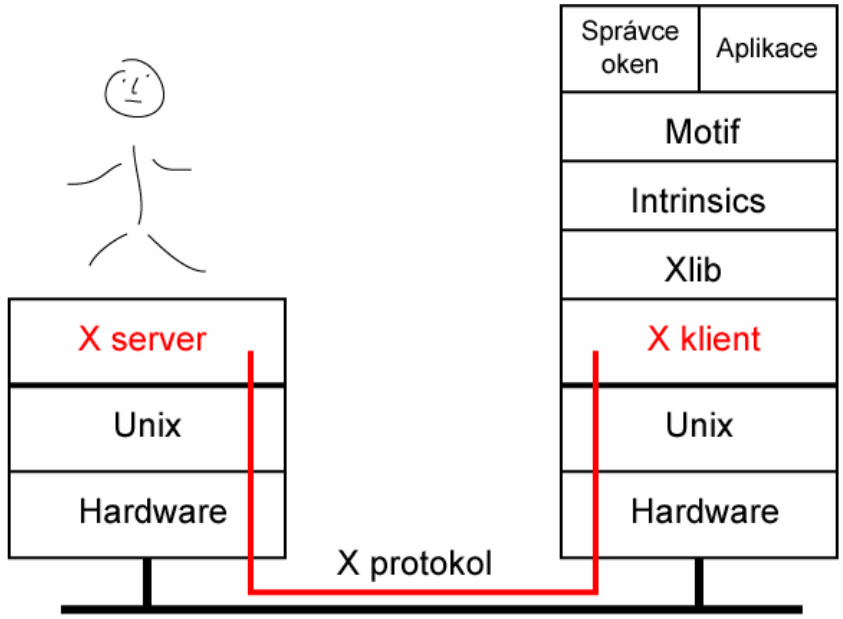
X Window systém II

- **Architektura**
  - model klient/server
  - dva nezávislé procesy, které spolu komunikují pomocí přesně definovaného **X protokolu**
- **X server**
  - obsluhuje grafické zařízení (obrazovka, myš, klávesnice) a předává klientům zprávy o akcích uživatele
  - každý X server může obsluhovat více X klientů
- **X klient**
  - aplikace, která vysílá požadavky na otevírání oken, manipulaci s nimi a vykreslování textu a grafiky
  - hardwarově nezávislý → přenositelnost
  - každý X klient může využívat služeb více X serverů

X Window systém III

- **Komunikace** mezi X klientem a X serverem je **asynchronní**:
  - X klienti vysílají požadavky k serveru, kde jsou řazeny do fronty a postupně vyřizovány (zachovává se pořadí v čase, nemusí být vyřízeny okamžitě)
  - X klient obvykle nečeká, ale může si synchronní zpracování požadavků vyžádat → zpomalení systému
  - X server je obvykle realizován softwarově, někdy je součástí hardware a firmwaru.

X Window systém IV



- **Xlib** – knihovna nejnižší úroveň
- **Intrinsics** – knihovna, která spravuje základní přípravky „widget“ (tlačítka, lišty,...)
- **Motif** – knihovna pro tvorbu GUI
- **Správce oken** – např. FVWM, CDE, KDE, GNOME,...
- **Aplikace** – xterm, xclock, xcalc, ...

Příklady

- **Správa přístupu k X serveru**
  - pomocí konfig. souboru `/etc/X*.hosts`
  - pomocí příkazu `xhost +počítač -počítač`
    např. `xhost +sunray1 -dray1`
- **Přesměrování grafických výstupů aplikací**
  - pomocí proměnné shellu `DISPLAY`
    `DISPLAY=[počítač]:číslo_serveru.[obrazovka]`
    např. `export DISPLAY=počítač:0.0`
  - pomocí přepínače `-d (-display)`
    `program -d [počítač]:číslo_serveru.[obrazovka]`
    `např. xterm -d sunray1:0.0`

Příklady

- **Nastavení parametrů**
- **`xset`**
  - nastavení chování
  - např. **`xset b on/off`** (nastaveni zvonku)

    `xset fp adresař` (nastavení cesty k fontům)

- **`xterm -g šířkaxvýška±xoff±yoff`**

Secure shell – SSH

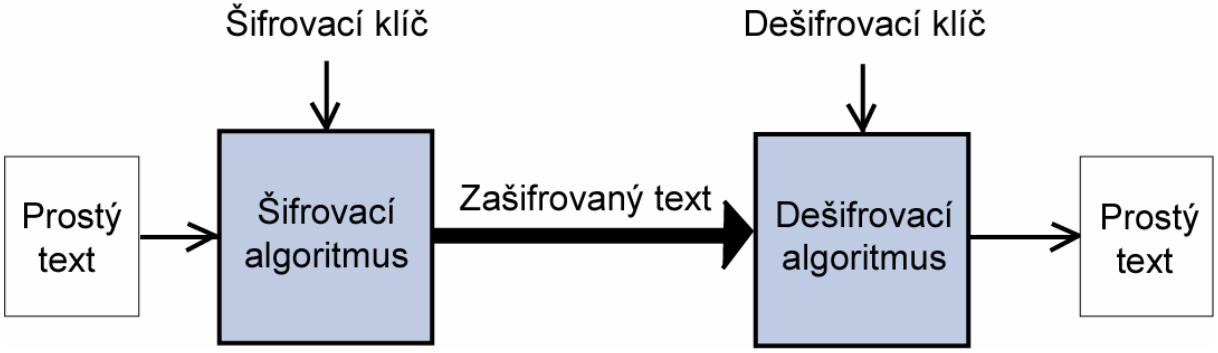
- **Softwarové řešení síťového zabezpečení** (na aplikační úrovni).

- **Založené na architektuře klient/server**
  - **klienti**: `ssh`, `slogin`, `scp` (v Unixu)
    např. `putty`, `winscp` (MS Windows)
  - **server**: `sshd`
  - transportní protokol je TCP/IP a server obvykle naslouchá na portu 22
  - různé implementace SSH1, SSH2, OpenSSH, ...

- **Přehled vlastností:**
  - **Soukromí (šifrování)** = ochrana dat před rozkrytím (odposloucháváním)
  - **Integrita dat** = garance, že se data nezmění během přenosu
  - **Autentizace** = ověření identity (jak serveru tak uživatele)
  - **Autorizace** = definice co smí příchozí uživatel dělat
  - **Směrování** (tunelování) = zapouzdření jiného protokolu využívající služeb TCP/IP do šifrované relace SSH

Základy kryptografie I

- **Šifrování (šifra)**
  - proces kódování dat takovým způsobem, aby je nebylo možné přečíst neoprávněnými osobami



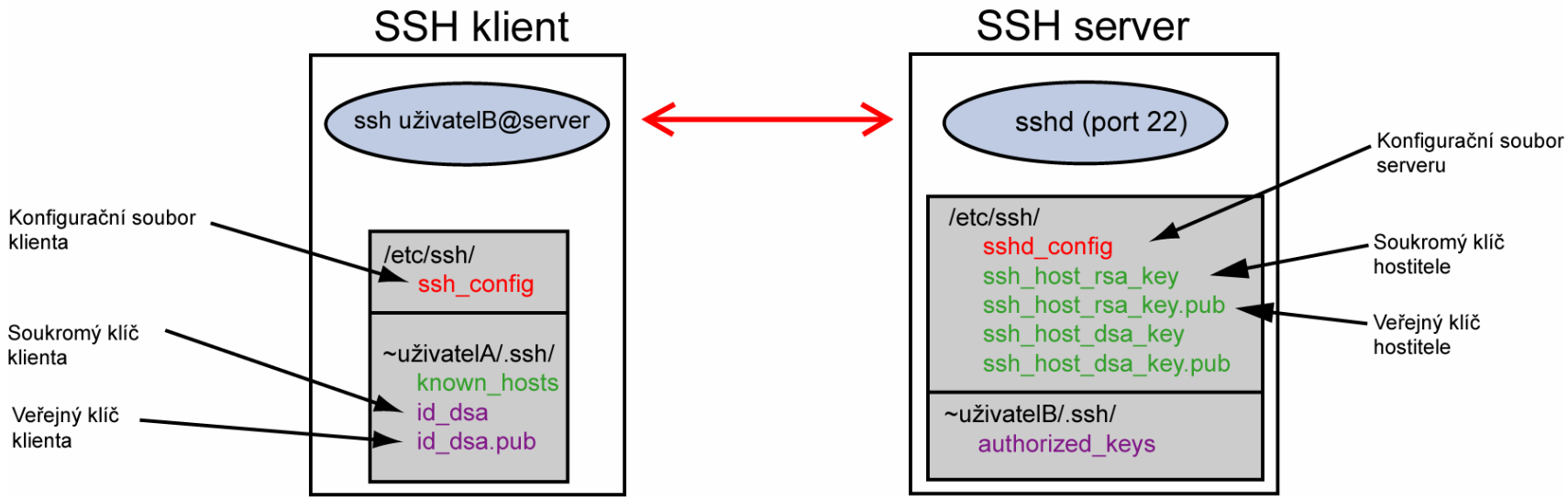
- **Šifrovací algoritmus**
  - konkrétní metoda, kterou se kódování provádí (např. DES, RSA, DSA, ...)
  - považuje se za **bezpečný** tehdy, když je pro ostatní osoby **"neproveditelné" přečíst data bez znalosti klíče**
  - v současnosti **nelze dokázat, že nějaký algoritmus je 100% bezpečný**
- **Kryptoanalýza**
  - pokus dešifrování dat bez znalosti klíče

Základy kryptografie II

- **Symetrické šifry (šifry s tajným klíčem)**
  - k zašifrování a rozšifrování se používá tentýž klíč
  - **výhoda:** rychlost šifrování/dešifrování
  - **nevýhoda:** problém s distribucí klíče
  - např. Blowfish, DES, IDEA, RC4

- **Asymetrické šifry (šifry s veřejnými klíči)**
  - používá se dvojce klíčů: **veřejný a soukromý klíč**
  - data zašifrovaná veřejným klíčem lze rozšifrovat pouze jeho soukromým klíčem
  - **nelze odvodit soukromý klíč z veřejného**
  - **výhoda:** odpadá problém s distribucí klíče
  - **nevýhoda:** pomalé šifrování/dešifrování
  - např. RSA, DSA, ElGamal, Elliptic Curve, ...

Ustanovení zabezpečeného spojení



1. **Klient kontaktuje server** (port TCP na serveru obvykle 22)
2. Klient a server si vzájemně sdělí **jaké verze SSH podporují**.
3. **Server se identifikuje klientovi** a dodá mu parametry relace
   - veřejný klíč hostitele
   - seznam šifrovacích, kompresních a autentizačních metod, které server podporuje
4. **Klient odešle serveru tajný klíč relace** (zašifrovaný pomocí veřejného klíče hostitele).
5. **Obě strany zapnou šifrování a dokončí autentikaci serveru** (klient čeká na potvrzovací zprávu od serveru).

Příklady

- **Připojení na neznámý server (první připojení)**

```
$ ssh trdlicka@dray1.feld.cvut.cz
The authenticity of host 'dray1.feld.cvut.cz (147.32.192.154)' can't be established
RSA key fingerprint is d8:d4:05:fe:a7:b5:a1:42:6b:79:d4:58:3e:fe:44:1f.
Are you sure you want to continue connecting (yes/no)? yes
```

- Pro kontrolu, zda dva klíče jsou stejné, se používá **otisk klíče (fingerprint).**

- **Jak získat otisk klíče?**

```
$ ssh-keygen -l -f ssh_host_rsa_key.pub
1024 d8:d4:05:fe:a7:b5:a1:42:6b:79:d4:58:3e:fe:44:1f ssh_host_rsa_key.pub
```

Autentizace klienta

- Po ustanovení zabezpečeného spojení klient pokouší prokázat svou identitu (existuje několik metod):
- **Heslem**
- **Veřejným klíčem klienta**
  1. klient pošle svůj veřejný klíč serveru

     (např. ~uživatelA/.ssh/id_dsa.pub)
  2. server se pokusí najít záznam o klíči

     (např ~uživatelB/.ssh/authorized_keys)
  3. server vygeneruje náhodný řetězec, zašifruje ho veřejným klíčem klienta a pošle ho klientovi
  4. klient rozšifruje zašifrovaný řetězec svým soukromým klíčem a pošle serveru

Příklady

- **Jak najít problém při připojování**

```
sh -v trdlicka@dray1.feld.cvut.cz
```

Příklady

- **Vygenerování dvojce klíčů**

  `$ ssh-keygen -t dsa`

- Klíče se explicitně uloží do souborů

  `~uživatelA/.ssh/id_dsa` (soukromý klíč)

  `~uživatelA/.ssh/id_dsa.pub` (veřejný klíč)

- Pokud zadáte přístupovou frázi, pak bude klíč zašifrován a při autorizaci pomocí klíče budete muset zadávat přístupovou frázi.

- **Přidání veřejného klíče klienta do souboru na serveru**

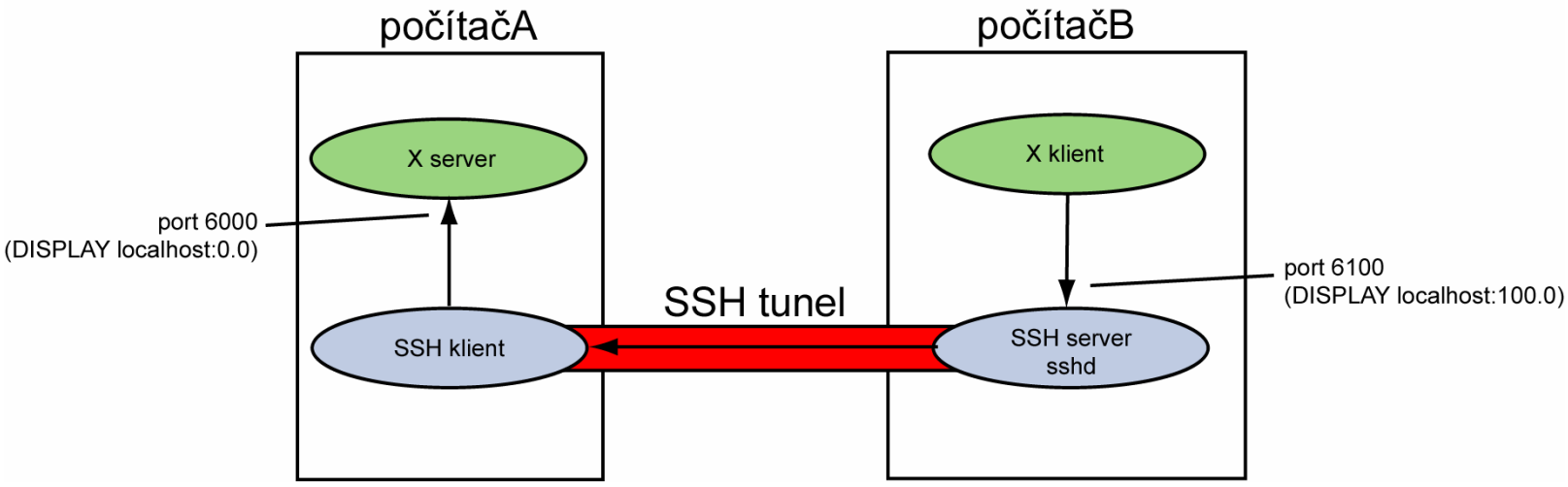
  `~uživatelB/.ssh/authorized_keys`

- **Autorizace pomocí veřejného klíče**

  uživatelA@klient$ `ssh uživatelB@server`

  uživatelB@server$

Tunelování protokolu X11



- Spojení přes **protokol X lze přesměrovat přes SSH**, které mu poskytne bezpečnost.
- Klient SSH si při připojení k serveru SSH vyžádá směrování protokolu X (musí být zapnuto na klientovi SSH a povoleno na serveru SSH):
  - **SSH server nastaví proměnnou**

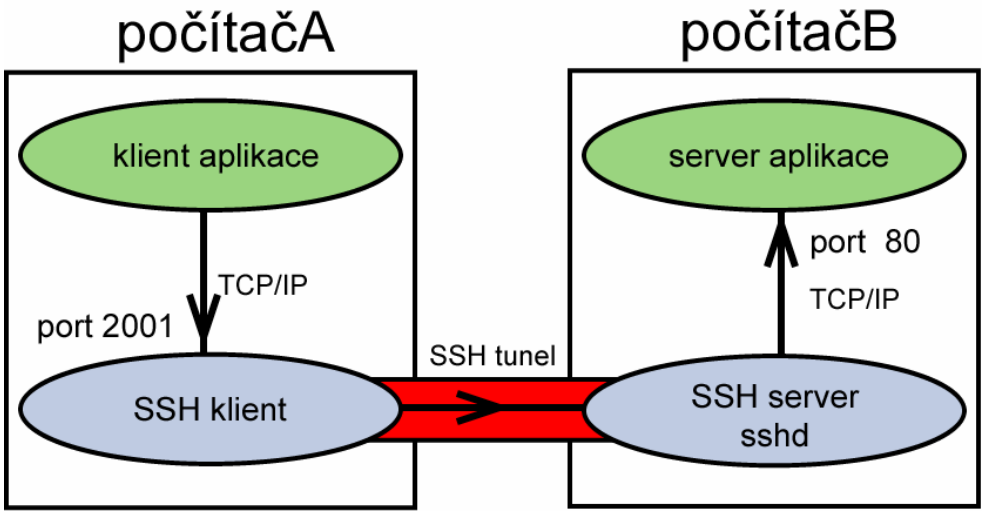
    `DISPLAY=localhost:pořadové_číslo_serveru.0`
  - SSH server začne **poslouchat na lokálním portu 6000+pořadové_číslo_serveru** a vše přeposílá na SSH klienta
  - klient SSH se chová jako X klient a obdržené výstupy posílá X serveru

Příklad

- **Připojení z počítačA na počítačB a zapnutí tunelování X11**

```
$ ssh -X uživatel@počítačB
```

Tunelování portů (místní) I

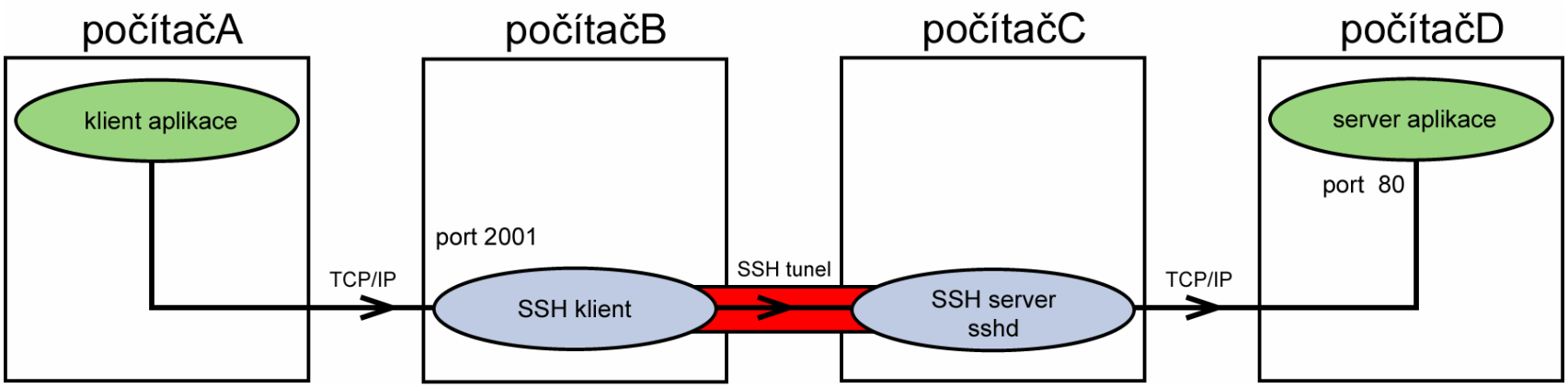


- SSH klient na jedné straně přijímá požadavky na služby, zašifrované je posílá na SSh server, který je na druhé straně doručí cílovému příjemci.

- SSH tunel je pro aplikace **transparentní**.

```
ssh -L 2001:localhost:80 uživatel@počítačB
```

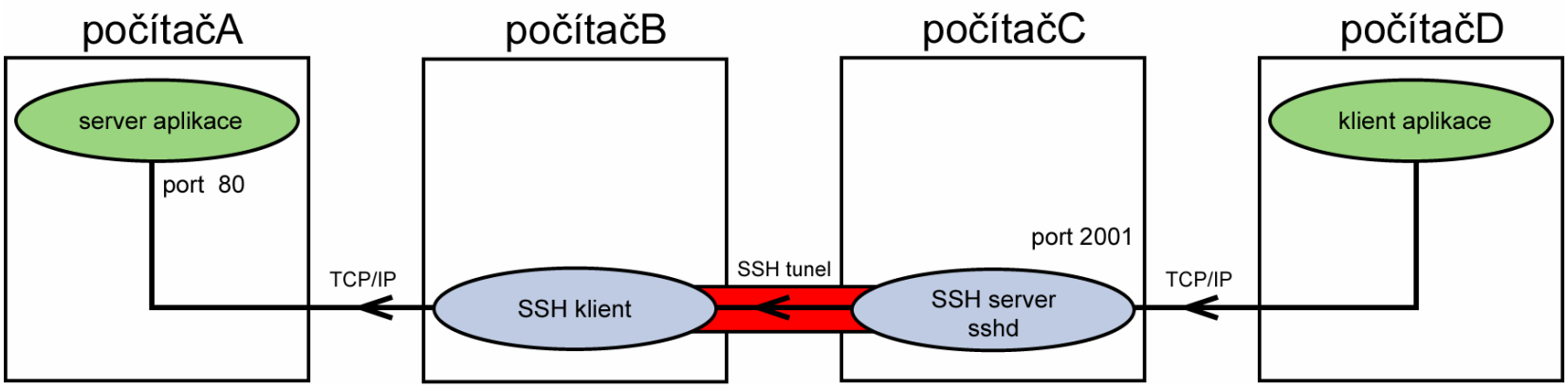
Tunelování portů (místní) II



- Jednotliví klienti a servery můžou běžet na různých počítačích.

```
ssh -g -L 2001:počítačD:80 uživatel@počítačC
```

Tunelování portů (vzdálené) I




```
ssh -g -R 2001:počítačD:80 uživatel@počítačC
```