Bezpečnost v IT

11. přednáška

Radek Janoštík

Univerzita Palackého v Olomouci

25. 4. 2024

Outline

- Aktuální (kyber)bezpečnostní situace
- Filtrace (+ firewall)
- Proxy
- Brány
- SOCKS

Aktuální (kyber)bezpečnostní situace

- Sledujete zprávy z dění v kyberprostoru? Co se stalo?
- Záplaty na chybu Intel BHI (minule) snižují výkon
  - Kompilace kolem 2 %
  - Testy Kraken a Octane kolem 7 %
  - Použití v databázích až 12 %
- 24 let stará chyba v glibc při převodu mezi znakovými sadami
  - Postiženo PHP při převodu na znakovou sadu `ISO-2022-CN-EXT` možný buffer overflow
  - $\Rightarrow$ možnost napadení aplikace $\Rightarrow$ poté serveru
- 37 lidí zatčeno v Kanadě, US a UK za provoz LabHost
  - Poskytovali službu *Phishing-as-a-Service*, nabízeli skoro 200 falešných webů ke krádeži osobních informací

Filtrace

- = kontrola dat, které procházejí aktivním prvkem; polopropustný filtr
  - Možné některou komunikaci nepovolit
  - Nemění data
- Kritéria pro filtraci
  - Režijní informace protokolů – adresy, porty, příznaky (SYN/ACK)
  - Data aplikačního protokolu (musíme znát a „rozumět“)
- Filtrační politiky
  - Co není zakázáno, je povoleno – „blacklisting“
  - Co není povoleno, je zakázáno – „whitelisting“
- Provádí *managovatelný switch*(linková vrstva), *router*(IP a TCP)
  - Případně specializovaný HW přes který „teče veškerý provoz“
  - SW(firewall) na klientských stanicích

Filtrace protokolu IP

- Povolení/zakázání komunikace mezi počítači; údaje ze záhlaví paketu
  - IP adresa odesílatele/příjemce
  - Protokol vyšší vrstvy
  - Příznaky (explicitní směrování)
- Filtrace protokolu ICMP – nežádoucí zprávy (redirect, změna směrování, echo?)
- Možné blokovat i rozsahy adres (jak je na tom geolokace?)
- Bez kombinace s TCP/UDP filtrem poměrně bezzubé
- *Reflexivní filtr* – sleduje spojení vyššího protokolu
  - Umožní zevnitř navázat relaci
  - Dovnitř propustí pouze data s ní související
- *Adress-spoofing attack* – útočník nastaví adresu z vnitřní sítě
  - Nedorazí k němu však odpověď
  - Možné takto doručit falešnou DNS odpověď a tím přesměrovat oběť „jinam“

Filtrace na TCP/UDP

- Povolení/zakázání komunikace určitých aplikací
  - Kombinace s IP filtrem – „tyto počítače můžou komunikovat jen těmito aplikacemi“
  - TCP/UDP záhlaví: porty, příznaky SYN/ACK
- Kvůli fragmentaci nemusí všechny pakety obsahovat TCP záhlaví
  - Filtr zastaví pouze první paket
  - Ostatní jsou doručeny k cíli, pokusí se stavit paket
  - Neuspěje a informuje protějšek ICMP zprávou
  - Čekaní na celý TCP segment a zamítnutí všech(náročnější, pomalejší, jedna cesta)
  - Filtrace ICMP
- Filtrace příchozích spojení (SYN)
  - Po navázání spojení dočasné povolení obousměrného provozu (bez SYN, ale s ACK, RST)
  - Kontrola čísel paketů, stejných IP adres, portů
- Nutné rozumět (a nahlížet do) aplikačním protokolům
  - Např. FTP vytváří datový kanál, aktivní režim (spojení z druhého směru)
  - UDP + DNS – pouštět jen odpovědi po dotazu

Firewall

- = Aplikace či HW, který provádí filtraci provozu + logování
- Dnes často specializovaný HW se podpůrným SW
  - Pro velký provoz potřeba větší výkon
  - Kvalitní síťové karty (větší buffery), redundance
  - Porozumění protokolům vyšších vrstev
  - Někdy podporují i inspekci SSL provozu (MITM)
- Často stovky až tisíce filtračních pravidel

Firewall – ukázky pravidel

- Při práci s větší sítí je dobré zadávat pravidlo pro rozsahy IP adres
  - ⇒ Rozumně rozdělit rozsahy dle logického rozdělení sítě
  - Pojmenovat si rozsahy ⇒ větší přehlednost pravidel

```
ip firewall address-list
add address=158.194.92.201-158.194.92.218 list=Ucebna-5002
add address=158.194.92.180-158.194.92.200 list=Ucebna-5004+vsechn
add address=158.194.92.219-158.194.92.236 list=Ucebna-5003
add address=158.194.92.237-158.194.92.249 list=Ucebna-1029
add address=158.194.92.180-158.194.92.249 list=Vsechny-PC-ucebny
add address=158.194.80.210-158.194.80.212 list=tiskarny
...
add address=158.194.80.0/24 list=vlan-80/92
add address=158.194.92.0/24 list=vlan-80/92
add address=158.194.0.0/16 list=Sit-UP
...
```

Firewall – ukázky pravidel

- U každého pravidla nezapomínejte psát komentáře
  - Za chvíli zapomenete, proč tam to pravidlo máte
  - Zvýšení zastupitelnosti, snížení následků nízkého *bus factoru*
- Proč jsou v našem FW následující pravidla:

```
add action=accept chain=forward  dst-port=445 \
    protocol=tcp src-address=158.194.0.0/16
add action=accept chain=forward  dst-port=445 \
    protocol=udp src-address=158.194.0.0/16
...
add action=drop chain=forward dst-port=445 protocol=tcp
add action=drop chain=forward dst-port=445 protocol=udp
```

- Původní komentáře:

```
comment="TCP-ACCESS SMB z UP"
comment="TCP-DROP SMB"
```

Firewall – ukázky pravidel

- Co dělají následující pravidla:

```
add action=accept chain=forward  dst-address-list=tiskarny \
    src-address=158.194.80.0/24
add action=accept chain=forward dst-address-list=tiskarny \
    src-address=158.194.92.0/24
add action=accept chain=forward dst-address-list=tiskarny \
    src-address-list=SMTP
...
add action=drop chain=forward dst-address-list=tiskarny
add action=drop chain=forward dst-address-list=diskova-pole \
    src-address-list=tiskarny
add action=drop chain=forward  src-address-list=tiskarny \
   dst-address-list=nodes-studentsky-cluster
add action=drop chain=forward src-address-list=tiskarny \
   dst-address-list=infrastrukturni-hypervizory
```

Firewall – komentáře

- Komentáře:

```
comment="ACCESS komunikace k tiskarnam z vlan80"
comment="ACCESS komunikace k tiskarnam z vlan92"
comment="ACCESS SMTP UP -> VLAN 80"
comment="DROP komunikace k tiskarnam"
comment="DROP z tiskaren -> diskova pole (kvuli ramsomware)"
comment="DROP z tiskaren -> studentsky cluster (kvuli ramsomware)
comment="DROP z tiskaren -> infrastrukturni hypervizory (kvuli ra
```

Skenování portů

- = Posílání specifických paketů protokolem TCP či UDP pro zjištění běžících služeb
- Snaha o (většinou) navázání spojení $\rightarrow$ rozumná odpověď $\Rightarrow$ na portu *poslouchá* nějaká služba
- Regulérní taktika při diagnostice sítě
  - Běží daná služba? Má otevřený port?
  - Neblokuje ji nějaký firewall?
  - Co nám odpovídá?
- Avšak také taktika útočníků k mapování sítě
  - Objevení potenciálně zranitelných aplikací
  - Odhad běžících operačních systém
  - Odhad počtu stanic
- Hromadné, masivní skenování může být považováno za útok
  - Zvlášť když po sobě neuzavíráme spojení (čerpání zdrojů)
- Velké množství sw scannerů, pro základy postačuje `nmap`
  - `shodan.io`

Skenování portů – typy

- **TCP connect** – vykoná kompletní třífázový handshake
  - Trvá poměrně dlouho
  - Pravděpodobně bude na cílovém systému zalogován
  - Možné provádět (non-root) běžným uživatelem
- **TCP SYN** – odeslán pouze SYN paket
  - Po obdržení `SYN,ACK` můžeme odvodit, že na portu služba běží
  - Pokud přijde `RST,ACK` je nejspíš port uzavřen
  - ACK už neposíláme $\Rightarrow$ možný DOS cíle (neukončeno spojení)
- **TCP FIN** – na port odešleme pouze `FIN`
  - Dle RFC 793 by měl systém pro všechny uzavřené porty odpovědět `RST`
  - Nemusí však RFC striktně dodržovat
- **TCP Null** – všechny příznaky nulové, měl by odpovědět stejně jako výše
- **UDP** – odešle paket na cílový port (není spojení)
  - Je-li odpověď `ICMP port unreachable` je port uzavřen
  - Jinak můžeme usuzovat, že je otevřen

Nmap – ukázka

- Doporučuji pročíst manuál

Proxy

- = aplikace poskytující službu počítačům (většinou) v interní síti
  - Serverová část – přijímá požadavky od klientů
  - Klientská část – chová se jako klient a „posílá požadavky dál“
  - Prostředník mezi komunikujícími uzly
- Většinou na rozhraní dvou sítí
- Zná aplikační protokol, který zpracovává
- Hlavní účely
  - Filtrace – před „přeložením paketů“ aplikace sady pravidel
    - Kdo se kam může připojit
    - Vidí aplikační data – HTTP metody, JavaScript, ...
    - Blacklist DNS, URL
  - Cache – ukládání provedený požadavků a znovupoužití (s dnešními SPA nepoužitelné)
- Některé protokoly mají *chování jako proxy* (DNS, SMTP), Obrázek

Klasická proxy

- Klient (=aplikace) se přímo připojuje na proxy (např. `proxy.inf.upol.cz`)
- Po připojení řekne proxy: „Chci se připojit na `server.nekde.v.internetu`“
- Proxy toto připojení zprostředkuje
- Obrázek
- Jakým způsobem bude proxy a klient komunikovat?
  - Např. rozšiřující příkazy v protokolech (FTP, telnet, SMTP)
  - Nestandardní domluva – úprava aplikací klienta i proxy $\Rightarrow$ nákladné
- Klient potřebuje umět adresovat pouze proxy, překlad `server.nekde.v.internetu` již provádí proxy

Generická proxy

- Upravování klientských aplikací někdy není možné (proprietární software)
- Idea: Do proxy „zadrátujeme“ adresu cílového serveru a port
- Pro každý server, kam je potřeba se připojit vytvoříme instanci na jiném portu
- Obrázek
- Není potřeba dělat úpravy klientských aplikací
- Omezené množství koncových serverů
- Jednoduchá implementace, stejný program pouze s jiným nastavením cílové IP a portů
- Vhodné pro aplikační protokoly bez rozšířených příkazů (SSH, IMAP, POP, . . .)

Transparentní proxy

- RFC-1919 – pojednává o transparentní proxy, pěkné čtení
- Klient musí umět adresovat server v internetu, vytvoří pakety pro něj a normálně odešle
- Paket přesměrován na transparentní proxy
- Proxy vykoná dotaz(+ filtrace, cache) za klienta, příjme odpověď a vrátí ji klientovi
- Není potřeba upravovat (ani nastavovat) klientské aplikace
- Obrázek

Brána (gateway)

- = uzel podobný proxy, avšak mění aplikační protokol mezi serverovou a klientskou části
- Nejčastěji – brána HTTP $\Leftrightarrow$ FTP
  - SW brány generuje webovou stránku zobrazující obsah FTP
  - Poté i překlad přenášených dat
- Rozlišovat druhy bran
  - Aplikační – překládají aplikační protokoly (výše)
  - Síťové/protokolové – např IoT gateway (LoRaWAN, Zigbee gateway, . . .)
- Obrázek

SOCKS

- Zatím jsme museli pro každý aplikační protokol vytvářet proxy
- SOCKS-server – proxy společná pro všechny aplikační protokoly (obrázek)
- SOCKS – komunikační protokol, cílem je otevřít proxy pro aplikační protokol
- SOCKS verze 5 (RFC-1928)
- Dohoda na autentizační metodě
  - Klient nabízí podporované autentizační metody
  - Server jednu z nich vybírá
  - Možná dohoda i na šifrování a podpis dat (zejména v „opačném módu“)
- Autentizační dialog – dle dohody
  - Jménem a heslem – RFC-1929
  - *Generic Security Service API(GSS-API)* – RFC-1508, RFC-1961
  - Bez autentizace (možná kontrola IP klienta)

SOCKS

- Vytvoření potřebné proxy, příkazy:
  - `CONNECT` – nejčastější, specifikace cílového serveru, portu, protokolu, (obrázek)
    - SOCKS-server alokuje port, naváže spojení
    - Informuje klienta o vytvoření proxy (ip + port)
    - Klient se připojí na vytvořenou proxy
  - `BIND` – zohledňuje potřeby aktivního FTP
  - `ASSOCIATE` – pro zřízení UDP proxy
- Poté již je možná datová komunikace klienta s cílovým serverem přes proxy
- WIN SOCKS = MS řešení téhož problému, méně síťové, více aplikační („podvržení DLL“), uzavřené

Doporučená četba / zdroje

- Dostálek L. a kolektiv. Velký průvodce protokoly TCP/IP: Bezpečnost (2. aktualizované vydání). Computer Press, 2003. ISBN 807226849X
  - Kapitola 5 – Filtrace, Proxy, Brány, SOCKS

- McClure S., Scambray J., Kurtz G.: Hacking Exposed 7: Network Security Secrets and Solutions (7th. edition). CompuMcGraw Hill, 2012. ISBN 978-0071780285
  - Port scanning

- Fridrich I.: Pravidla firewallu KI