OS Administration lesson #6

Main 3rd party network services/protocols

Agenda

- DNS
- Firewall
- NAT
- SSH
- FTP, SFTP,
- rsync
- VPN

Domain Name System (DNS)

When DNS did not exist, one had to download a host file containing hostnames and their corresponding IP address.

But with increase in number of hosts on the Internet, the size of host file also increased. This resulted in increased traffic on downloading this file.

To solve this problem DNS was introduced.

Domain Name System (DNS)

- is the way that Internet domain names are located and translated into internet protocol (IP) addresses, e.g. maps the name people use to locate a website to the IP address that a computer uses to locate a website.
- is a client/server network communication system
- implements a **distributed** database to store the names and last-known address information for all public hosts on the Internet.

Domain Name System (DNS)

It is a database that maintains the names of resources and links them to particular IP addresses.

Main characteristics:

- locates IP addresses to specific names, and then storing this data, aka **“maintaining records”**.
- has a hierarchical tree structure, aka **“domain namespace”**.
- distributes the data over a vast network of connections
- stores the data on millions of servers around the world
- each server holds only a minor portion of the hostname to IP address mapping details

DNS Architecture

● is defined by a **hierarchical distributed database** and a **set of protocols**
● is a mechanism for updating, replicating information and a schema of the database
● stores the names in a hierarchical tree structure, aka the **domain namespace**.
● lays domain names at the top of the hierarchy; the names are of individual **labels (a-zA-Z0-9-, max 63, can be 0?)**, which are subsequently divided through dots.

As a result, a **fully qualified domain name (FQDN) (max?)** is unique enough to be easily identified by the host position in the DNS structure.

DNS Hierarchy Levels

- **Root Domain –** highest level of the tree it is often stated by a **trailing period** *(.)*
- **Top-level Domain –** describes a country, a region, or a type of an organization. *(com)*
- **Second-level Domains** *(portaone.com)*
- **Sub-domains (how many?)** – *(int.portaone.com)*
- **Hosts** or **Resources –** in the leaf of the DNS tree of names *(task.int.portaone.com)*

DNS Zones

An **authoritative name server** is a name server that only gives answers to DNS queries from data that has been configured by an original source, e.g. by:

- the domain administrator
- dynamic DNS methods

DNS zone - at least **2 authoritative** name servers:

- **master** server – stores the original (master) copies of all zone records.
- **slave** server – automatic updating mechanism in the DNS

DNS Zone Files

- is a text file that describes a DNS zone

- contains mappings between domain names and IP addresses and other resources, organized in the form of text representations of resource records (RR)

- is a sequence of entries for resource records

DNS Resource Records

- more than 30 currently used
- described by a lot of RFCs (1035, 1183, 2230, 2535, 2930, 4255, 5155, 6672, 7553, etc)

- most common types of records stored in the DNS database are:
  - Start of Authority (**SOA**),
  - IP addresses (**A** for IPv4 and **AAAA** and IPv6),
  - SMTP mail exchangers (**MX**),
  - service (**NAPTR/SRV**)
  - name servers (**NS**),
  - pointers for reverse DNS lookups (**PTR**),
  - and domain name aliases (**CNAME**).

DNS Zone File Example

```
$ORIGIN example.com.     ; designates the start of this zone file in the namespace
$TTL 1h                  ; default expiration time of all resource records without their own TTL value
example.com.  IN  SOA   ns.example.com. username.example.com. ( 2007120710 1d 2h 4w 1h )
example.com.  IN  NS    ns                    ; ns.example.com is a nameserver for example.com
example.com.  IN  NS    ns.somewhere.example. ; ns.somewhere.example is a backup nameserver for example.com
example.com.  IN  MX    10 mail.example.com.  ; mail.example.com is the mailserver for example.com
@             IN  MX    20 mail2.example.com. ; equivalent to above line, "@" represents zone origin
@             IN  MX    50 mail3              ; equivalent to above line, but using a relative host name
example.com.  IN  A     192.0.2.1             ; IPv4 address for example.com
              IN  AAAA  2001:db8:10::1        ; IPv6 address for example.com
ns            IN  A     192.0.2.2             ; IPv4 address for ns.example.com
              IN  AAAA  2001:db8:10::2        ; IPv6 address for ns.example.com
www           IN  CNAME example.com.          ; www.example.com is an alias for example.com
wwwtest       IN  CNAME www                   ; wwwtest.example.com is another alias for www.example.com
mail          IN  A     192.0.2.3             ; IPv4 address for mail.example.com
mail2         IN  A     192.0.2.4             ; IPv4 address for mail2.example.com
mail3         IN  A     192.0.2.5             ; IPv4 address for mail3.example.com
```

DNS Root Zone

- is a global list of top-level domains:

- **original top-level domains** (.com, .org, .net, .int, .edu, .gov, .mil)

- **two letter country codes** (.ua, .au, .bb, .ca, .cb, .cc, cd, etc...)

- **special-use domains**:

.example - Not installed as a domain name, but usable in text as an example.

.invalid - Not installed as a domain name, but usable in testing as a domain which wouldn't work.

.local - Local network

.localhost - Points back to own computer

.onion - Connection to the Tor network

.test - Meant for testing DNS software

DNS Root Zone Name Servers

- are name servers for the root zone of DNS of the Internet

- directly answer requests for records in the root zone and answers other requests by returning a list of the authoritative name servers for the appropriate top-level domain (TLD)

- are 13 actual servers due to limits in DNS and certain protocols (e.g. UDP)

- can be much more (up to 600) when using anycast addressing

DNS Root Zone Name Servers

● directly answer requests for records in the root zone and answers other requests by returning a list of the authoritative name servers for the appropriate top-level domain (TLD)

● are 13 logical servers only due to limits in DNS itself and certain protocols (e.g. UDP), that are referred to as **<letter>**.root-servers.net (the letter ranges from **A** to **M**), e.g.:
a.root-servers.net, b.root-servers.net, k.root-servers.net

● can be actually much more (up to 600) using anycast addressing (via BGP routing)

DNS configs

> cat /etc/resolv.conf
Generated by NetworkManager
search portaone.com
nameserver 8.8.8.8
nameserver 10.1.1.100

Up to 3 name servers may be listed, one per keyword.

DNS Utiles – nslookup

```
06:34:01 MR60.0 porta-one@configurator-smlab.portaone.com:~
> nslookup
> portaone.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   portaone.com
Address: 217.182.15.195
> set type=mx
> portaone.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
portaone.com    mail exchanger = 5 smtp-in.portaone.com.

Authoritative answers can be found from:
> smtp-in.portaone.com.
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
*** Can't find smtp-in.portaone.com.: No answer
```

```
Authoritative answers can be found from:
portaone.com
        origin = ns.portaone.com
        mail addr = hostmaster.portaone.com
        serial = 2017032710
        refresh = 300
        retry = 300
        expire = 604800
        minimum = 900
> server ns.portaone.com
Default server: ns.portaone.com
Address: 217.182.15.193#53
> smtp-in.portaone.com.
Server:         ns.portaone.com
Address:        217.182.15.193#53

*** Can't find smtp-in.portaone.com.: No answer
> set type=A
> smtp-in.portaone.com.
Server:         ns.portaone.com
Address:        217.182.15.193#53

Name:   smtp-in.portaone.com
Address: 195.138.219.147
>
 configurator-smlab 0.87 0.34 0.22
```

DNS Utiles – nslookup

```
> set type=mx
> google.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
google.com      mail exchanger = 50 alt4.aspmx.l.google.com.
google.com      mail exchanger = 20 alt1.aspmx.l.google.com.
google.com      mail exchanger = 10 aspmx.l.google.com.
google.com      mail exchanger = 30 alt2.aspmx.l.google.com.
google.com      mail exchanger = 40 alt3.aspmx.l.google.com.

Authoritative answers can be found from:
> google.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
google.com      mail exchanger = 50 alt4.aspmx.l.google.com.
google.com      mail exchanger = 20 alt1.aspmx.l.google.com.
google.com      mail exchanger = 10 aspmx.l.google.com.
google.com      mail exchanger = 30 alt2.aspmx.l.google.com.
google.com      mail exchanger = 40 alt3.aspmx.l.google.com.

Authoritative answers can be found from:
```

```
> set type=a
> google.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   google.com
Address: 216.58.209.46
> google.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   google.com
Address: 216.58.209.78
> google.com
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   google.com
Address: 216.58.209.78
> google.com
Server:         8.8.8.8
Address:        8.8.8.8#53
```

DNS Utiles – nslookup

```
06:39:55 MR60.0 porta-one@configurator-smlab.portaone.com:~
> nslookup
> pool.ntp.org
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   pool.ntp.org
Address: 134.249.140.162
Name:   pool.ntp.org
Address: 78.26.180.80
Name:   pool.ntp.org
Address: 31.28.161.68
Name:   pool.ntp.org
Address: 91.198.10.4
> pool.ntp.org
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   pool.ntp.org
Address: 79.142.192.4
Name:   pool.ntp.org
Address: 195.138.69.242
Name:   pool.ntp.org
Address: 193.27.209.20
Name:   pool.ntp.org
Address: 195.138.82.247
>
```

DNS Utiles – nslookup

```
06:39:55 MR60.0 porta-one@configurator-smlab.portaone.com:~
> nslookup
> pool.ntp.org
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   pool.ntp.org
Address: 134.249.140.162
Name:   pool.ntp.org
Address: 78.26.180.80
Name:   pool.ntp.org
Address: 31.28.161.68
Name:   pool.ntp.org
Address: 91.198.10.4
> pool.ntp.org
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   pool.ntp.org
Address: 79.142.192.4
Name:   pool.ntp.org
Address: 195.138.69.242
Name:   pool.ntp.org
Address: 193.27.209.20
Name:   pool.ntp.org
Address: 195.138.82.247
>
```

DNS Utiles – dig

```
07:13:00 MR60.0 porta-one@configurator-smlab.portaone.com:~
> dig @8.8.8.8 portaone.com

; <<>> DiG 9.9.4-RedHat-9.9.4-29.el7 <<>> @8.8.8.8 portaone.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22703
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;portaone.com.                  IN      A

;; ANSWER SECTION:
portaone.com.           3599    IN      A       217.182.15.195

;; Query time: 70 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Tue Apr 04 07:13:11 UTC 2017
;; MSG SIZE  rcvd: 57
```

DNS Utiles – dig

```
07:13:11 MR60.0 porta-one@configurator-smlab.portaone.com:~
> dig @8.8.8.8 portaone.com MX

; <<>> DiG 9.9.4-RedHat-9.9.4-29.el7 <<>> @8.8.8.8 portaone.com MX
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 23458
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;portaone.com.                  IN      MX

;; ANSWER SECTION:
portaone.com.           3403    IN      MX      5 smtp-in.portaone.com.

;; Query time: 35 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Tue Apr 04 07:13:18 UTC 2017
;; MSG SIZE  rcvd: 65
```

DNS Utiles – dig

```
07:13:18 MR60.0 porta-one@configurator-smlab.portaone.com:~
> dig @8.8.8.8 portaone.com ANY

; <<>> DiG 9.9.4-RedHat-9.9.4-29.el7 <<>> @8.8.8.8 portaone.com ANY
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 55673
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;portaone.com.                  IN      ANY

;; ANSWER SECTION:
portaone.com.           3389    IN      SOA     ns.portaone.com. hostmaster
.portaone.com. 2017032710 300 300 604800 900
portaone.com.           3389    IN      TXT     "v=spf1 ip4:195.138.219.1/2
4 ip4:193.28.87.1/24 ip4:91.212.34.1/24 ip4:193.34.92.88/29 ipv4::217.182.1
5.193/27" " a:cn.portaone.com a:mail.portaone.com a:firefly.portaone.com a:
kiev-mail.portaone.com a:avatar.portaone.com a:twix.portaone.com a:snickers
.portaone.com a:www.portaone.com a:itnoc.portaone.com    a:pufik.portaone.c
om  a:sumy-mail.portaone.com -all"
portaone.com.           3389    IN      MX      5 smtp-in.portaone.com.
portaone.com.           3389    IN      NS      ns1.portaone.com.
portaone.com.           3389    IN      NS      ns.portaone.com.
portaone.com.           3389    IN      NS      ns2.portaone.com.
portaone.com.           3389    IN      A       217.182.15.195

;; Query time: 37 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Tue Apr 04 07:13:26 UTC 2017
;; MSG SIZE  rcvd: 551
```

DNS Most Common Issues

- monitoring
- NTP
- SSL domains
- ???

Firewall

- **network security system**
- typically establishes a barrier between a **trusted**, secure internal network and another outside network, such as the Internet
- can be:
  - **network firewall**
  - **host-based firewall**

- often have **network address translation (NAT)** functionality (RFC 1918)

*NIX Firewalls

- IPFilter: included with (Open)Solaris, FreeBSD and NetBSD, available for many other Unix-like operating systems;
- ipfirewall (ipfw): FreeBSD-native packet filter
- NPF: NetBSD-native Packet Filter
- PF: OpenBSD-native Packet Filter

- Netfilter with **iptables**/nftables: the Linux packet filter

Iptables Tables :)

● set of predefined **chains**, which contain **rules** which are traversed in order

● are actually 5 five these tables:

- **raw** – configuring packets
- **filter** – the **default table**
- **nat**
- **mangle** – used for specialized packet alterations.
- **security** – used for Mandatory Access Control networking rules (e.g. SELinux).

filter and **nat** – most commonly used tables

The other tables are aimed at complex configurations involving multiple routers and routing decisions and are beyond the scope of this lesson

Iptables Chains

filter: **INPUT**, **OUTPUT** and **FORWARD**

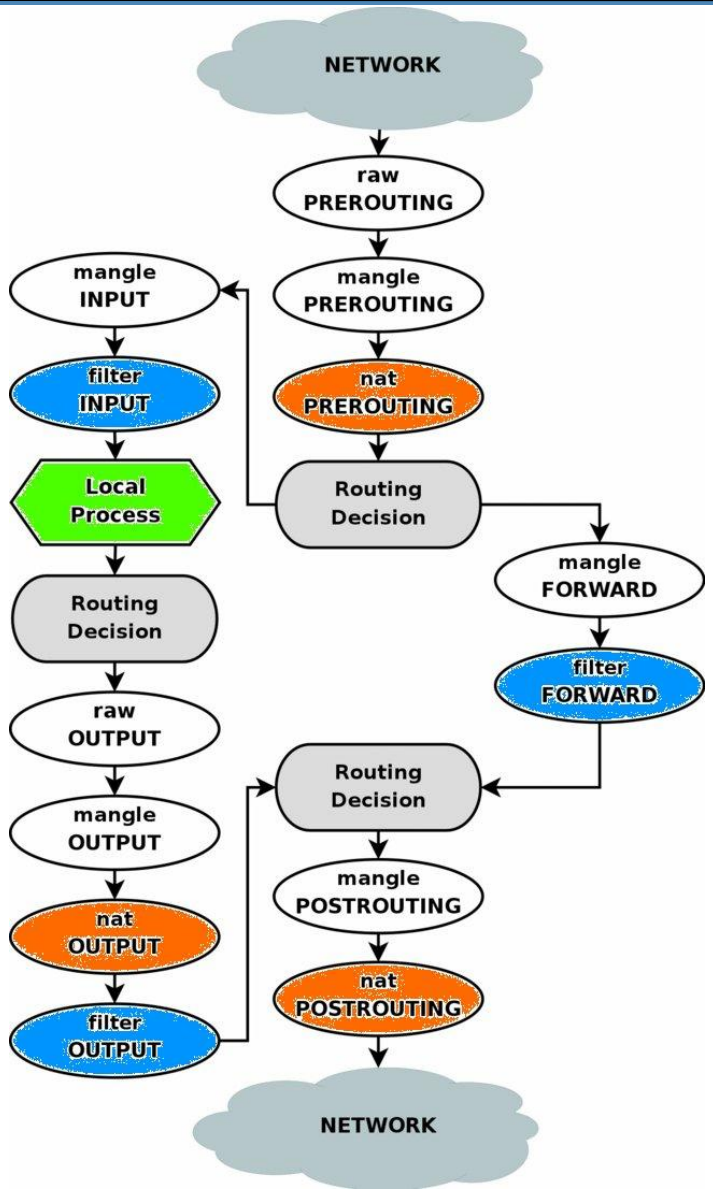
nat table includes **PREROUTING**, **POSTROUTING**, and **OUTPUT** chains.

By default, none of the chains contain any rules.

Chains **do have a default policy**, which is generally set to **ACCEPT**, but can be reset to **DROP** and is **always** applied at the end of each chain

User-defined chains can be added to make rulesets more efficient or more easily modifiable

Iptables Flowchart



Iptables Rules

● consist of a predicate of potential matches and the corresponding action (I.e. **target**) which is executed if the predicate is true;

● are specified by **one or multiple matches** (conditions the packet must satisfy so that the rule can be applied), and **one target** (action taken when the packet matches all conditions)

● typically match on things like:
  - interface the packet came in/out on (e.g., eth0 or eth1)
  - type of packet it is (ICMP, TCP, or UDP)
  - the source/destination port of the packet

Iptables Targets

- can be either:
  - one of the special **built-in** targets (ACCEPT, DROP, QUEUE and RETURN)
  - **target extensions** (e.g., REJECT and LOG)
  - **user-defined** chains (i.e. if these conditions are matched, jump to the following user-defined chain and continue processing there)
- are specified using the **-j** or **--jump** option

If the target is:
– a **built-in** target, the fate of the packet is decided immediately and processing of the packet in current table is stopped.
– a **user-defined** chain and the fate of the packet is not decided by this second chain, it will be filtered against the remaining rules of the original chain.

Target extensions:

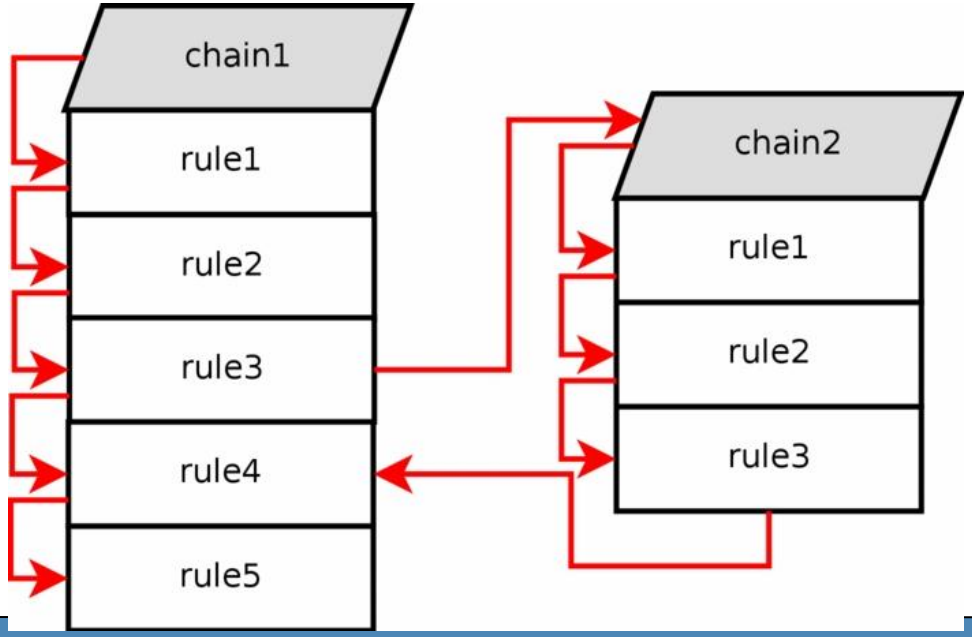
– either terminating (as built-in targets)
– or non-terminating (as user-defined chains)

Iptables Targets

The 3 **most commonly used targets** are ACCEPT, DROP, and jump to a user-defined chain.

While built-in chains can have default policies, user-defined chains can not.

If every rule in a chain that you jumped fails to provide a complete match, the packet is dropped back into the calling chain:



Iptables Usage

```
07:28:34 MR60.0 porta-one@porta-billing-slave:~
> sudo iptables -nL
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
dbfw-10.10.2.45  tcp  --  0.0.0.0/0            0.0.0.0/0           multiport dports 3307

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain dbfw-10.10.2.45 (1 references)
target     prot opt source               destination
ACCEPT     tcp  --  10.10.15.2           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.15.3           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.15.5           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.45           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.54           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.55           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.56           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.59           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.66           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.70           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  10.10.2.73           0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  182.50.64.103        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  182.50.64.107        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  182.50.64.108        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  182.50.64.110        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  182.50.64.112        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  182.50.64.113        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  182.50.64.126        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  198.18.1.0/24        0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  198.18.1.100         0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  198.18.1.111         0.0.0.0/0           multiport dports 3307
ACCEPT     tcp  --  43.245.94.2          0.0.0.0/0           multiport dports 3307
DROP       all  --  0.0.0.0/0            0.0.0.0/0
```

Iptables Usage

```
> sudo iptables -nL -v
Chain INPUT (policy ACCEPT 13M packets, 11G bytes)
 pkts bytes target     prot opt in     out     source               destination
 266K   16M dbfw-10.10.2.45  tcp  --  !lo    *       0.0.0.0/0            0.0.0.0/0           multiport dports 3307

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 12M packets, 921M bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain dbfw-10.10.2.45 (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  *      *       10.10.15.2           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.15.3           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.15.5           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.2.45           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.2.54           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.2.55           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.2.56           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.2.59           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.2.66           0.0.0.0/0           multiport dports 3307
 266K   16M ACCEPT     tcp  --  *      *       10.10.2.70           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       10.10.2.73           0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       182.50.64.103        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       182.50.64.107        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       182.50.64.108        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       182.50.64.110        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       182.50.64.112        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       182.50.64.113        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       182.50.64.126        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       198.18.1.0/24        0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       198.18.1.100         0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       198.18.1.111         0.0.0.0/0           multiport dports 3307
    0     0 ACCEPT     tcp  --  *      *       43.245.94.2          0.0.0.0/0           multiport dports 3307
   11   660 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0
```

Iptables Usage

```
> sudo iptables -nL -v --line-numbers
Chain INPUT (policy ACCEPT 13M packets, 11G bytes)
num   pkts bytes target     prot opt in     out     source               destination
1     266K   16M dbfw-10.10.2.45  tcp  --  !lo    *       0.0.0.0/0            0.0.0.0/0           multiport dports 3307

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
num   pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 12M packets, 922M bytes)
num   pkts bytes target     prot opt in     out     source               destination

Chain dbfw-10.10.2.45 (1 references)
num   pkts bytes target     prot opt in     out     source               destination
1        0     0 ACCEPT     tcp  --  *      *       10.10.15.2           0.0.0.0/0           multiport dports 3307
2        0     0 ACCEPT     tcp  --  *      *       10.10.15.3           0.0.0.0/0           multiport dports 3307
3        0     0 ACCEPT     tcp  --  *      *       10.10.15.5           0.0.0.0/0           multiport dports 3307
4        0     0 ACCEPT     tcp  --  *      *       10.10.2.45           0.0.0.0/0           multiport dports 3307
5        0     0 ACCEPT     tcp  --  *      *       10.10.2.54           0.0.0.0/0           multiport dports 3307
6        0     0 ACCEPT     tcp  --  *      *       10.10.2.55           0.0.0.0/0           multiport dports 3307
7        0     0 ACCEPT     tcp  --  *      *       10.10.2.56           0.0.0.0/0           multiport dports 3307
8        0     0 ACCEPT     tcp  --  *      *       10.10.2.59           0.0.0.0/0           multiport dports 3307
9        0     0 ACCEPT     tcp  --  *      *       10.10.2.66           0.0.0.0/0           multiport dports 3307
10    266K   16M ACCEPT     tcp  --  *      *       10.10.2.70           0.0.0.0/0           multiport dports 3307
11       0     0 ACCEPT     tcp  --  *      *       10.10.2.73           0.0.0.0/0           multiport dports 3307
12       0     0 ACCEPT     tcp  --  *      *       182.50.64.103        0.0.0.0/0           multiport dports 3307
13       0     0 ACCEPT     tcp  --  *      *       182.50.64.107        0.0.0.0/0           multiport dports 3307
14       0     0 ACCEPT     tcp  --  *      *       182.50.64.108        0.0.0.0/0           multiport dports 3307
15       0     0 ACCEPT     tcp  --  *      *       182.50.64.110        0.0.0.0/0           multiport dports 3307
16       0     0 ACCEPT     tcp  --  *      *       182.50.64.112        0.0.0.0/0           multiport dports 3307
17       0     0 ACCEPT     tcp  --  *      *       182.50.64.113        0.0.0.0/0           multiport dports 3307
18       0     0 ACCEPT     tcp  --  *      *       182.50.64.126        0.0.0.0/0           multiport dports 3307
19       0     0 ACCEPT     tcp  --  *      *       198.18.1.0/24        0.0.0.0/0           multiport dports 3307
20       0     0 ACCEPT     tcp  --  *      *       198.18.1.100         0.0.0.0/0           multiport dports 3307
21       0     0 ACCEPT     tcp  --  *      *       198.18.1.111         0.0.0.0/0           multiport dports 3307
22       0     0 ACCEPT     tcp  --  *      *       43.245.94.2          0.0.0.0/0           multiport dports 3307
23      11   660 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0
```

Iptables Usage

```
07:36:29 MR55.5 porta-one@portasip:~
> sudo iptables -n -L sipdos-162.247.220.17 -v
Chain sipdos-162.247.220.17 (1 references)
 pkts bytes target     prot opt in     out     source               destination
16721   18M DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0
1358K 1208M DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0
```

```
 udp dpt:5060 STRING match  "INVITE sip:" ALGO name bm TO 65 limit: above 30/min burst 16 mode srcip
 udp dpt:5060 STRING match  "REGISTER sip:" ALGO name bm TO 65 limit: above 30/min burst 16 mode srcip
 udp dpt:5060 STRING match  "OPTIONS sip:" ALGO name bm TO 65 limit: above 30/min burst 16 mode srcip
```

Iptables Usage

```
07:41:34 MR55.5 porta-one@portasip:~
> sudo iptables -n -L INPUT -v
Chain INPUT (policy ACCEPT 1906M packets, 615G bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       82.205.4.227         0.0.0.0/0
    0     0 ACCEPT     all  --  *      *       162.247.220.18       0.0.0.0/0
    0     0 DROP       all  --  *      *       69.30.245.58         0.0.0.0/0
    0     0 DROP       all  --  *      *       5.189.190.171        0.0.0.0/0
    0     0 DROP       all  --  *      *       192.96.201.135       0.0.0.0/0
    0     0 DROP       all  --  *      *       199.168.102.90       0.0.0.0/0
    0     0 DROP       all  --  *      *       176.227.209.42       0.0.0.0/0
    0     0 DROP       all  --  *      *       107.150.50.162       0.0.0.0/0
    0     0 DROP       all  --  *      *       85.25.197.23         0.0.0.0/0
    3   984 ACCEPT     all  --  *      *       162.247.220.1        0.0.0.0/0
    0     0 ACCEPT     all  --  *      *       162.211.139.52       0.0.0.0/0
    0     0 ACCEPT     udp  --  *      *       192.168.10.0/24      0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       162.211.139.10       0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       66.33.146.52         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       169.132.196.11       0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       169.132.136.9        0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       66.33.167.74         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       216.115.69.144       0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       70.167.153.130       0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       216.115.69.144       0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       88.208.208.34        0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       70.84.58.18          0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       67.15.180.14         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       67.15.128.18         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       67.15.128.14         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       64.246.22.119        0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       209.62.66.242        0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       198.101.50.4         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       198.101.50.2         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       162.211.139.10       0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       66.33.146.52         0.0.0.0/0           udp dpt:5060
    0     0 ACCEPT     udp  --  *      *       198.101.50.2         0.0.0.0/0           udp dpt:5060
    0     0 DROP       udp  --  *      *       198.50.129.134       0.0.0.0/0           udp dpt:5060
    0     0 DROP       all  --  *      *       199.217.116.25       0.0.0.0/0
    0     0 DROP       all  --  *      *       213.136.75.214       0.0.0.0/0
    0     0 DROP       udp  --  *      *       69.64.50.92          0.0.0.0/0           udp dpt:5060
    0     0 DROP       all  --  *      *       23.95.84.202         0.0.0.0/0
    0     0 DROP       udp  --  *      *       23.95.84.202         0.0.0.0/0           udp dpt:5060
    0     0 DROP       udp  --  *      *       23.95.11.250         0.0.0.0/0           udp dpt:5060
    0     0 DROP       udp  --  *      *       23.94.144.154        0.0.0.0/0           udp dpt:5060
    0     0 DROP       tcp  --  *      *       23.95.11.250         0.0.0.0/0           tcp dpt:80
    0     0 DROP       tcp  --  *      *       23.95.11.250         0.0.0.0/0           tcp dpt:80
 225K   10M dbfw-10.10.104.101  tcp  --  !lo    *       0.0.0.0/0            0.0.0.0/0           multiport dports 3307
1664M  304G sipdos-162.247.220.17  all  --  *      *       0.0.0.0/0            162.247.220.17
```

Iptables Usage

– to add the rule:

> sudo iptables -A INPUT -s 1.2.3.4 -j DROP

– to insert the rule:

> sudo iptables -I INPUT -s 1.2.3.4 -j DROP

– to save changes to the default file (/etc/sysconfig/iptables.save):

> sudo service iptables save

– to make a backup:

> sudo iptables-save > /home/porta-one/my.active.firewall.rules

– to restore the previous state:

> sudo iptables-restore < /home/porta-one/my.active.firewall.rules

Iptables Use Cases

- DB protector
- SIP protector
- custom rules
- NAT/forwarding

???

Network Address Translation (NAT)

- is the process where a network device, usually a firewall, assigns a public address to a computer (or group of computers) inside a private network
- is mainly aimed at limiting the number of public IP addresses an organization or company must use, for both economy and security purposes
- conserves the number of globally valid IP addresses a company needs, and in combination with Classless Inter-Domain Routing (CIDR) has done a lot to extend the useful life of IPv4 as a result
- is described in general terms in IETF RFCs 1631 and 2663

Network Address Translation (NAT)

- is a router feature and it is often a part of a corporate firewall.
- can map IP addresses in several ways:
  - from a local IP to one global IP statically;
  - from a local IP to any of a rotating pool of global IPs;
  - from a local IP plus a TCP/UDP port to a global IP and a port from a pool of ports;
  - from a global IP to any of a pool of local IP on a round-robin basis.

As network address translation modifies the IP address information in packets, it:

- has serious consequences on the quality of Internet connectivity
- requires careful attention to the details of its implementation

Basic NAT

- provides a one-to-one translation of IP addresses
- is often also called a one-to-one NAT
- changes only the IP addresses, IP header checksum and any higher level checksums that include the IP address
- can be used to interconnect two IP networks that have incompatible addressing

One-to-many NAT

- maps multiple private hosts to one publicly exposed IP address
- translated on the fly the source address in each packet from a private address to the public address
- tracks basic data about each active connection (particularly the destination address and port)
- when a reply returns to the router, uses the connection tracking data it stored during the outbound phase to determine the private address on the internal network to which to forward the reply
- enables communication through the router only when the conversation originates in the masqueraded network since this establishes the translation tables.

Simple traversal of UDP over NATs
(STUN, RFC 3489)

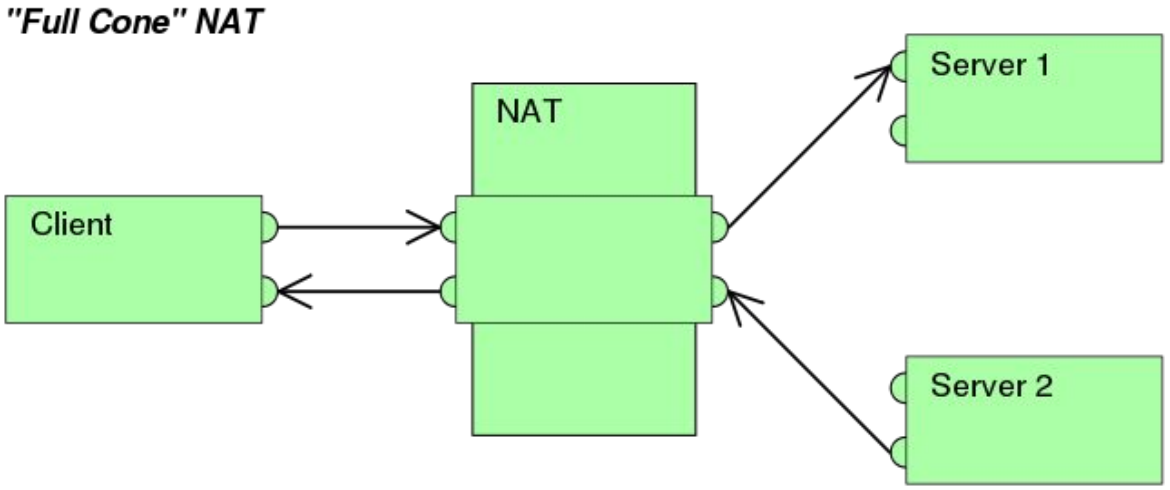
- is used when an application behind NAT needs to:
  - determine the external address of the NAT
  - examine and categorize the type of mapping in use

- requires assistance from a third-party network server (STUN server) located on the Internet

- classified NAT implementation as:
  - full-cone
  - (address) restricted-cone
  - port-restricted cone
  - symmetric

Full-cone NAT

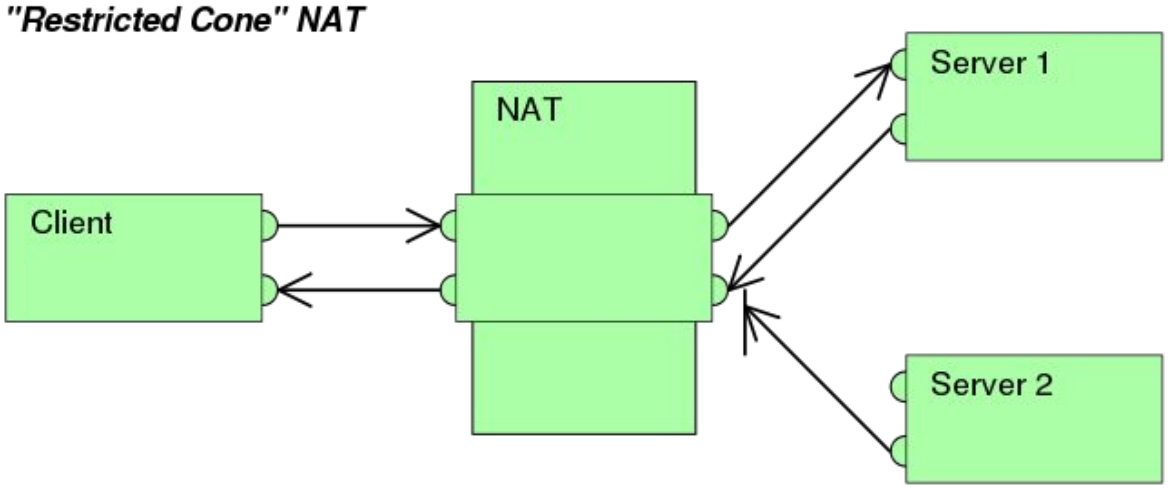
- also known as one-to-one NAT
- once an internal address (iAddr:iPort) is mapped to an external address (eAddr:ePort), any packets from iAddr:iPort are sent through eAddr:ePort.
- any external host can send packets to iAddr:iPort by sending packets to eAddr:ePort.



(Address)-restricted-cone NAT

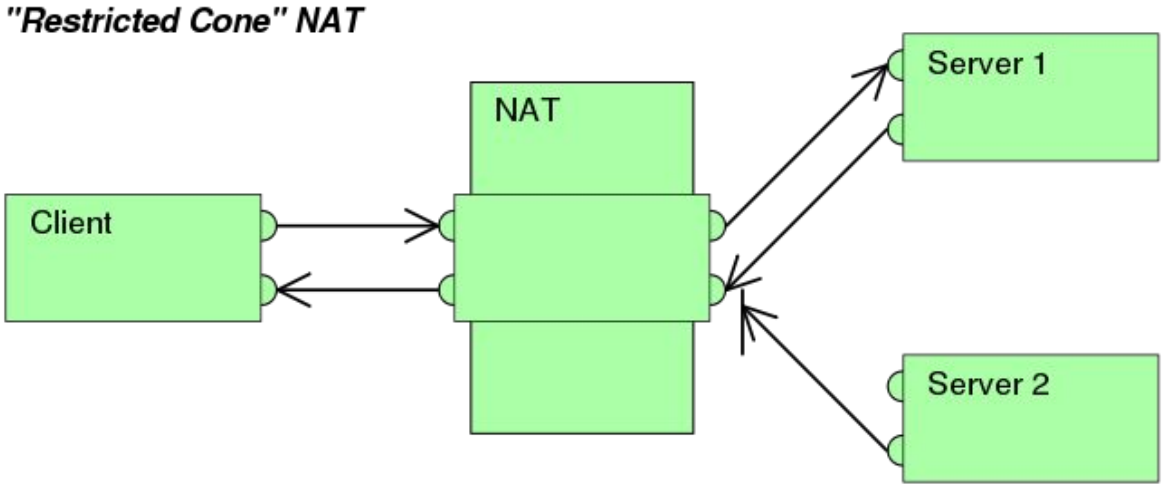
● once an internal address (iAddr:iPort) is mapped to an external address (eAddr:ePort), any packets from iAddr:iPort are sent through eAddr:ePort.

● an external host (hAddr:any) can send packets to iAddr:iPort by sending packets to eAddr:ePort only if iAddr:iPort has previously sent a packet to hAddr:any. "Any" means the port number doesn't matter.



Port-restricted cone NAT

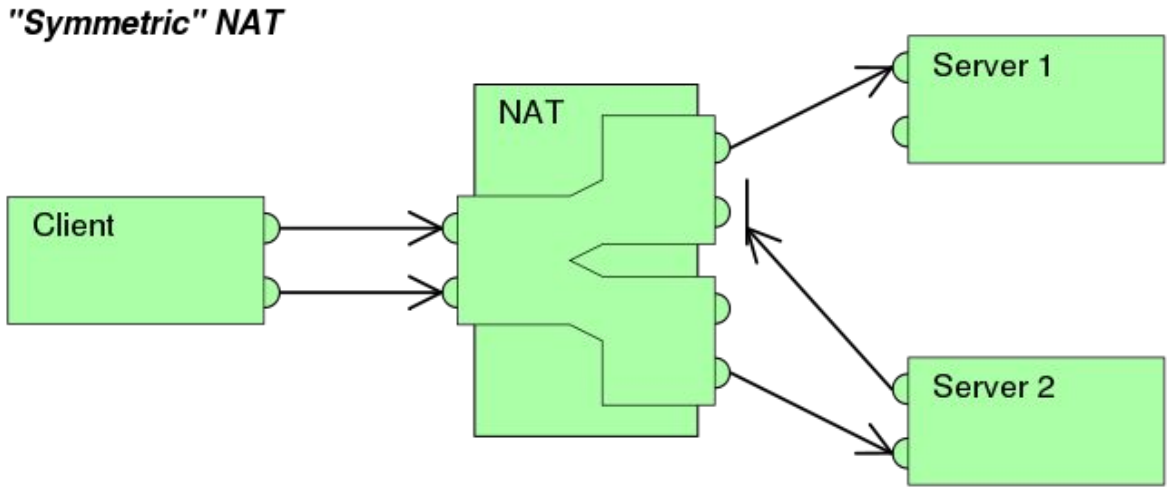
- is like an address restricted cone NAT, but the restriction includes port numbers.
- once an internal address (iAddr:iPort) is mapped to an external address (eAddr:ePort), any packets from iAddr:iPort are sent through eAddr:ePort.
- an external host (hAddr:hPort) can send packets to iAddr:iPort by sending packets to eAddr:ePort only if iAddr:iPort has previously sent a packet to hAddr:hPort.



Symmetric NAT

● each request from the same internal IP address and port to a specific destination IP address and port is mapped to a unique external source IP address and port; if the same internal host sends a packet even with the same source address and port but to a different destination, a different mapping is used.

● only an external host that receives a packet from an internal host can send a packet back.



~~Simple traversal of UDP over NATs~~

(STUN, RFC 5389)

However,

- those procedures have since been deprecated from standards status
- those methods are inadequate to correctly assess many devices and cases
- new methods have been standardized in RFC 5389 (Oct 2008)
- the STUN acronym now represents **Session Traversal Utilities for NAT**

Session Traversal Utilities for NAT

(STUN, RFC 5389)

- does work with:
  - full cone NAT
  - restricted cone NAT
  - port restricted cone NAT

- doesn't with (due to different IPs):
  - symmetric NAT

NAT & PortaSwitch

- Audio issues
- STUN servers
- IP Forwarding
- custom configurations
- ???

File-Transfer Protocol (FTP)

- is a standard network protocol used for the transfer of files between computers
- is built on a client-server model architecture
- uses separate control and data connections between the client and the server
- typically uses a clear-text sign-in protocol
- is often secured with SSL/TLS (FTPS)
- can also be replaced by SSH File Transfer Protocol (SFTP) for security reasons

FTP Workflow

- requires multiple network ports to work properly.

When an FTP client application initiates a connection to an FTP server, it connects to port 21 on the server — known as the command port.

This port is used to issue all commands to the server.

Any data requested from the server is returned to the client via a data port.

The port number for data connections and the way in which data connections are initialized vary depending upon whether the client requests the data in active or passive mode.

FTP Modes

- **active mode** – is the original method used by the FTP protocol for transferring data to the client application.

The server opens a connection from port 20 on the server to the IP address and a random, **unprivileged port (greater than 1024)** specified by the client.
This arrangement means that the client machine must be allowed to accept connections over any port above 1024. Because these client-side firewalls often deny incoming connections from active mode FTP servers, passive mode was devised.

- **passive mode** – like active mode, is initiated by the FTP client application.

Then, the FTP client indicates it wants to access the data in passive mode and the server provides the IP address and a random, unprivileged port (greater than 1024) on the server. **The client then connects to that port on the server** to download the requested information.

Secure Shell (SSH)

- is a **cryptographic network protocol** for operating network services securely over an unsecured network.
- provides a secure channel over an unsecured network in a client-server architecture, connecting an **SSH client** application with an **SSH server**.
- usually used for remote command-line login and remote command execution
- can be used for securing **any** network service.
- has two major versions (considered as obsolete v.1 and v.2)
- **is not** natively supported by Windows OSs

Secure Shell (SSH)

● uses **public-key cryptography** to authenticate the remote computer and allow it to authenticate the user, if necessary.

Ways to use SSH:

● to use automatically generated public-private key pairs to simply encrypt a network connection, and then use password authentication to log on.

● to use a manually generated public-private key pair to perform the authentication, allowing users or programs to log in without a password.

Secure Shell (SSH) Keys

- the public key is placed on all servers that must allow access to the owner of the matching private key (the owner keeps the private key secret).
- the private key itself is never transferred through the network during authentication, as SSH only verifies whether the same person offering the public key also owns the matching private key.

The list of authorized public keys is typically stored on the SSH server in the home directory of the user that is allowed to log in remotely, in the file **~/.ssh/authorized_keys**.
This file is respected by SSH server only if it is not writable by anything apart from the owner and root.

The **ssh-keygen** utility produces the public and private keys, always in pairs.

SSH password-based authentication

- relies on using a login password pair
- is still encrypted by **automatically** generated keys

However, in this case:

- the legitimate server side can be imitated by an attacker
- the illegitimate SSH server (man-in-the-middle) then can ask for and obtain the password
- as an additional condition, such an attack is possible only if the two sides have never authenticated before, as SSH client remembers the key that the server side previously used (stored in **~/.ssh/known_hosts**)
- the SSH client raises a warning before accepting the key of a new, previously unknown server

SSH is typically used for:

- logging to a shell on a remote host (replacing **Telnet** and **rlogin**)
- executing a single command on a remote host (replacing rsh)
- setting up automatic (passwordless) login to a remote server (e.g, using OpenSSH)
- secure file transferring (SFTP)
- (In combination with rsync) backing up, copying and mirroring files efficiently and securely
- forwarding or tunneling a port
- using as a full-fledged encrypted VPN. (OpenSSH only)
- forwarding X from a remote host (possible through multiple intermediate hosts)
- browsing the web through an encrypted proxy connection with SSH clients via SOCKS
- mounting a directory on a remote server as a filesystem on a local computer via SSHFS.
- monitoring and management of servers
- ...

SSH usually uses standard port **22/TCP**, however It can be re-configured.

User-specific SSH configuration:

- is stored in the user's home directory within the **~/.ssh/** directory:

– **authorized_keys** – This file holds a list of authorized public keys for servers. When the client connects to a server, the server authenticates the client by checking its signed public key stored within this file.
– **id_dsa** – Contains the DSA private key of the user.
– **id_dsa.pub** – The DSA public key of the user.
– **id_rsa** – The RSA private key used by ssh for version 2 of the SSH protocol.
– **id_rsa.pub** – The RSA public key used by ssh for version 2 of the SSH protocol
– **identity** – The RSA private key used by ssh for version 1 of the SSH protocol.
– **identity.pub** – The RSA public key used by ssh for version 1 of the SSH protocol.
– **known_hosts** — This file contains DSA host keys of SSH servers accessed by the user. This file is very important for ensuring that the SSH client is connecting the correct SSH server.

System-wide SSH configuration information:

- is stored in the **/etc/ssh/** directory:

– **moduli** — contains Diffie-Hellman groups used for the Diffie-Hellman key exchange which is critical for constructing a secure transport layer. When keys are exchanged at the beginning of an SSH session, a shared, secret value is created which cannot be determined by either party alone. This value is then used to provide host authentication.
– **ssh_config** — the system-wide default SSH client configuration file. It is overridden if one is also present in the user's home directory (~/.ssh/config).
– **sshd_config** — the configuration file for the sshd daemon.
– **ssh_host_dsa_key** — the DSA private key used by the sshd daemon.
– **ssh_host_dsa_key.pub** — the DSA public key used by the sshd daemon.
– **ssh_host_key** — the RSA private key used by the sshd daemon for SSH v.1
– **ssh_host_key.pub** — the RSA public key used by the sshd daemon for SSH v.1
– **ssh_host_rsa_key** — the RSA private key used by the sshd daemon for SSH v.2
– **ssh_host_rsa_key.pub** — the RSA public key used by the sshd daemon for for SSH v.2

Useful SSH commands:

- to copy files between servers using **csp** utility and a key:

**> sudo scp -i .ssh/id_dsa.portaone -rp **
/porta_var/10.0.0.1/mysql porta-one@10.1.0.4:/porta_var/tmp/test/

- to create an SSH tunnel for servers without access to the Internet:

On the Configuration server (IP 192.168.1.115):
> sudo ssh -L 192.168.1.115:80:packages.portaone.com:80 -AfN
porta-one@192.168.1.115

On a server:
In /etc/hosts:
192.168.1.115 packages.portaone.com

Useful SSH commands:

- typical issue when copying files via scp:

```
12:27:56 MR60.0 porta-one@master-smlab.portaone.com:~
> scp -i .ssh/id_dsa.portaone -rp /porta_var/porta-db-master-1/mysql/mysql-bin.000064 \
> porta-one@configurator:/porta_var/tmp/
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0644 for '.ssh/id_dsa.portaone' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
bad permissions: ignore key: .ssh/id_dsa.portaone
porta-one@configurator's password:
```

- **DON'T!!! change** file permissions for **id_dsa.portaone** as this affects other services
- just use **sudo** instead:

```
12:28:15 MR60.0 porta-one@master-smlab.portaone.com:~
> sudo scp -i .ssh/id_dsa.portaone -rp /porta_var/porta-db-master-1/mysql/mysql-bin.000064 \
> porta-one@configurator:/porta_var/tmp/
mysql-bin.000064
12:28:29 MR60.0 porta-one@master-smlab.portaone.com:~
> 
```

SSH tunnels:

- access to Internet resources for servers connected to private networks only
- for Oracle Enterprise Manager

SSH File Transfer Protocol (SFTP)

- is an extension of the Secure Shell protocol (SSH) **version 2.0** to provide secure file transfer capabilities
- itself does not provide authentication and security
- assumes that it is run over a established secure channel, such as SSH
- allows an extended range of operations on remote files:
  - resuming interrupted transfers
  - directory listings
  - remote file removal, etc…
- can be thought of just as encrypted version of FTP
- uses a syntax that is very similar to FTP

Rsync (Remote Synchronization)

- is a utility for efficiently transferring and synchronizing files across computer systems
- is commonly found on Unix-like systems
- can use Zlib for additional compression
- can use SSH or stunnel data security
- requires the specification of a source and of a destination; either of them may be remote, but not both.
- has plenty of CLI options and configuration files to specify:
  - alternative shells
  - options
  - commands, possibly with full path
  - port numbers.

Rsync Features

- support for copying links, devices, owners, groups, and permissions
- exclude and exclude-from options similar to GNU tar
- can use any transparent remote shell, including ssh or rsh
- **does not** require root privileges
- pipelining of file transfers to minimize latency costs
- support for anonymous or authenticated rsync servers (ideal for mirroring)

Rsync Workflow

An rsync process does its job by communicating with another rsync process, a sender and a receiver.

At startup, an rsync client has to connect to a peer process.

If the transfer is **local** – the peer can be created with fork

If a **remote** host is involved, rsync:

- starts a process to handle the connection, typically SSH
- starts an rsync process on the remote host via the established connection

If the remote host runs **an rsync daemon** – rsync clients can connect by opening a socket on port **873/TCP**, possibly using a proxy.

Virtual Private Network (VPN)

- is a technology that creates safe and encrypted connections over less secure networks, e.g. the Internet

- was developed as a way to allow remote users and branch offices to securely access corporate applications and other resources

- ensures the appropriate level of security to the connected systems when the underlying network infrastructure alone cannot provide it

Virtual Private Network (VPN)

- is characterized by:

  – protocols used to tunnel the traffic
  – termination point location, e.g., on the customer edge or network-provider edge
  – type of topology of connections, such as site-to-site or network-to-network
  – levels of security provided
  – OSI layer they present to the connecting network, e.g. Layer 2 or Layer 3
  – number of simultaneous connections

VPN Protocols

- IP security (IPsec)
- Secure Sockets Layer (SSL) & Transport Layer Security (TLS)
- Point-To-Point Tunneling Protocol (PPTP)
- Layer 2 Tunneling Protocol (L2TP)
- Cisco VPN
- OpenVPN

VPN Types

- **Remote-access VPN** – uses a public telecommunication infrastructure like the Internet to provide remote users secure access to their organization's network.

- **Site-to-site VPN** – uses a gateway device to connect the entire network in one location to the network in another -- usually a small branch connecting to a data center.

Thank you!