In addition, there are two special wildcard masks for matching all IP addresses and matching only a single IP address.

Matching all addresses can be simplified using the keyword 'any':

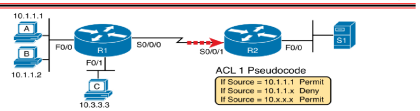
```
Router(config)#access-list 1 permit 0.0.0.0 255.255.255.255

Can be written as:
Router(config)#access-list 1 permit any
```

Matching one single address can be simplified using 'host' keyword:

```
Router(config)#access-list 1 permit 172.30.16.29 0.0.0.0

Can be written as:
Router(config)#access-list 1 permit host 172.30.16.29
```

Note: 'host' keyword is not required for newer versions of router IOS.

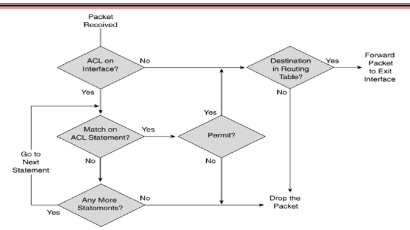
An example of standard numbered ACL for implementing the required security policy in the previous slide with explanations of the configuration commands in the following slides.




```
R2(config)# access-list 1 permit 10.1.1.1
R2(config)# access-list 1 deny 10.1.1.0 0.0.0.255
R2(config)# access-list 1 permit 10.0.0.0 0.255.255.255
R2(config)# interface S0/0/1
R2(config-if)# ip access-group 1 in
```

Note: Standard ACLs can only filter based on source IP. To avoid unintentionally filtering more packets than necessary, it is recommended to be implemented near the destination; e.g. at router R2 instead of R1.

For inbound ingress-filtering ACL, a router will ① filter all IP packets received on that interface, and ② only the permitted packets are then processed and forwarded.



Note: Packets destined to the router itself can be filtered by appropriate inbound ACLs.

In this lab you will experiment with access control list

4.1 Standard numbered ACL is used to filter network traffic based on source IP addresses.

4.2 Extended numbered ACL is more powerful as it can filter network traffic based on protocol type, source/destination IP addresses, and source/destination port numbers.

4.3 Named ACL enables ACL to be configured with meaningful name to make it easier to remember.

4.4 Time-based ACL enhances ACL with time range to automatically turn on/off ACL which can be useful in practice.

4.5 Object group-based ACL is a new way of writing ACL to make it more readable and manageable.

To verify that the ACL is applied correctly at the interface to filter packets, use the following command:

```
R2# show ip interface s0/0/1
Serial0/0/1 is up, line protocol is up
  Internet address is 10.1.2.2/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Multicast reserved groups joined: 224.0.0.9
  Outgoing access list is not set
  Inbound  access list is 1
! Lines omitted for brevity
```


In general, numbered ACL is written by (1) specifying the related ACEs using the same access-list-number, and (2) applying the complete ACL on the required interface.

1) Each ACE is configured using an access-list command:
   - 'remark' is optional but helps you remember what the ACL statements are for
   - 'log' is optional which will generate a log message if matches

Step 1

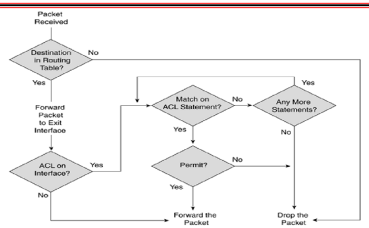
```
Router(config)#access-list access-list-number
   remark test
Router(config)#access-list access-list-number
   {permit | deny} {test-conditions} log
```

2) Apply the ACL onto the required interface and specify whether to filter inbound or outbound packets

Step 2

```
Router(config-if)#ip access-group access-list-number
   {in | out}
```


In contrast, for outbound egress-filtering ACL, ① only after forwarding decision has been made, ② IP packets are then filtered before permitted to be sent out from that interface.

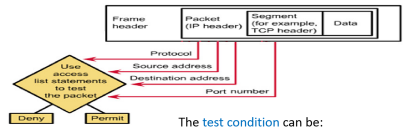


Note: Packets originated from the router itself are not filtered by outbound ACLs.

An access control list (ACL) is a sequential list of one or more ACL statement, also called access control entry (ACE) to filter network traffic.

Each ACL statement or ACE contains two components:

- Action to be taken
- When test condition is met



The action is either permit or deny the network traffic when condition is met.

The test condition can be:

- Standard – matches source IP address
- Extended – matches source IP address, destination IP address, protocol type, port number.

You may also view the statistics of the number of matches of each ACL statement as follows, which can be useful for network security monitoring.

```
R2# show ip access-lists
Standard IP access list 1
    10 permit 10.1.1.1 (107 matches)
    20 deny   10.1.1.0, wildcard bits 0.0.0.255 (4 matches)
    30 permit 10.0.0.0, wildcard bits 0.255.255.255 (10 matches)
```

The number of matches in () at the end of each ACL statement is the count of IP packets that matches it so far.

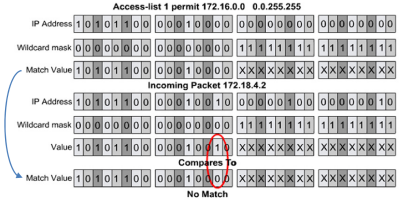
Note: To have a count of 'deny any' at end of ACL, you need to explicitly add 'deny any' to the end of ACL instead of using the default.

To clear the count, use the command:

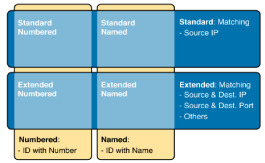
```
R2# clear access-list counters
```


The test-conditions for standard ACL statements are source IP addresses specified with wildcard mask similar as configuring OSPF and NAT in ICT1010.

E.g.: The following ACL with wildcard mask 0.0.255.255 represents IP addresses from 172.16.0.0 – 172.16.255.255. Thus an IP packet with IP 172.18.4.2 will not match the test condition.



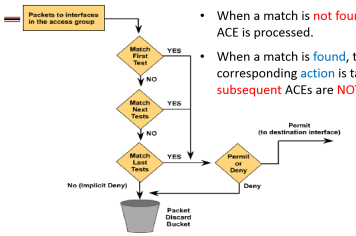
In Cisco network devices, ACLs are divided into standard and extended ACLs, and may be identified by a number or name, called numbered or named ACLs respectively.



- Standard ACLs filter at layer 3 based on source IP only.
- Extended ACLs filter at layer 3 and 4 based on more attributes.

- Standard numbered ACLs is configured using any number from 1 – 99 or 1300 – 1999.
- Extended numbered ACLs is configured using any number from 100 – 199 or 2000 – 2699.
- Named ACLs is configured using any alphanumeric characters.

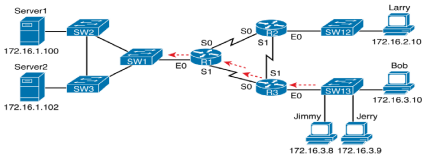
The ACEs are processed from top to bottom in order, and hence the sequence of ACEs in an ACL is important.



- When a match is not found, the next ACE is processed.
- When a match is found, the corresponding action is taken, and subsequent ACEs are NOT processed.
- If no match is found after processing all ACEs, there is a default deny statement at the end to drop the traffic.

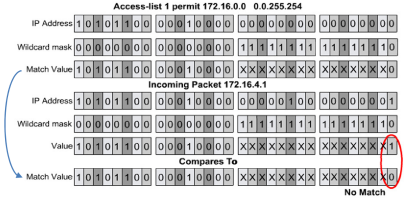
(4.2) More powerful extended numbered ACL for controlling access to networks or end hosts/servers.

E.g.: Server 1 and 2 host both web servers and tftp servers. The company's security policy allows Bob to access web servers but not tftp servers. In addition, Larry is not allowed to access web server at Server 1. How can you implement this?



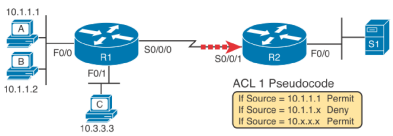
In fact, wildcard mask can be used to specify any bit patterns of the IP address to be matched. An example as follows:

E.g.: The following ACL with wildcard mask 0.0.255.254 denotes even-numbered IP addresses from 172.16.0.0 – 172.16.255.255. Thus an IP packet with IP 172.16.4.1 will not match.



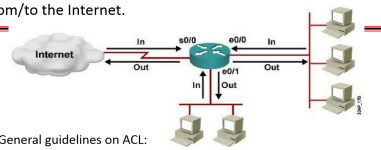
(4.1) Standard numbered ACL for controlling access to a network or end host/server.

E.g.: A company's security policy allows any users at subnet C but only one user at host A to access server S1. How can you implement this?



Typically, ACL is written with more specific ACEs (in terms of IP addresses) at the top and more general ACEs at the bottom.

In practice, ACL are commonly applied on the switch SVIs or router interfaces to filter ingress or egress IP packets, especially from/to the Internet.



General guidelines on ACL:

- One ACL per direction: ACL can only filter packets in one direction as configured. To filter both ingress and egress traffic, two separate ACLs must be written.
- One ACL per interface: Only one in ACL and one out ACL can be applied on each interface.
- One ACL per protocol: Separate ACLs must be written to filter IPv4 and IPv6 packets (not covered in this module)

(4.5) To be more readable, a new way of writing ACL called object group-based ACL has been introduced in Cisco routers and ASA firewalls.

First, configure related objects into object groups:

- Network
  - host IP
  - IP ranges
  - subnet
  - etc
- Service
  - protocol
  - port number
  - etc
  - e.g. tcp www

Note: User, ICMP-type and Security object groups are less commonly used and are only available in ASA firewalls.

With sequence numbers being inserted automatically, both numbered and named ACLs can be edited easily as follows:

To delete ACL statement, use no sequence-number command:

```
! Step 3: Still in ACL 24 configuration mode, the line with
 sequence number 20 is deleted.
R1(config-std-nacl)# no 20
```

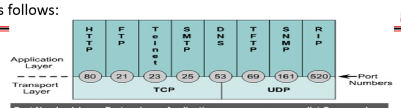
To insert new ACL statement, configure it starting with a sequence-number to indicate the position:

```
! Step 4: Inserting a new first line in the ACL.
R1(config-std-nacl)# 5 deny 10.1.1.1
```

Verify the resultant ACL:

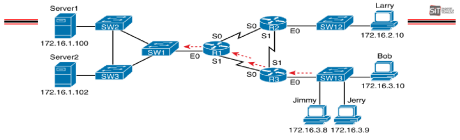
```
! Step 5: Displaying the ACL's contents with the new statement
R1(config-std-nacl)# do show ip access-list 24
Standard IP access list 24
    5 deny   10.1.1.1
   10 permit 10.1.1.0, wildcard bits 0.0.0.255
   30 permit 10.1.3.0, wildcard bits 0.0.0.255
```


The port numbers may be specified directly using numbers, or names representing well-known port numbers. Some examples as follows:



Port Number(s)	Protocol	Application	access-list Command Keyword
20	TCP	FTP data	ftp-data
21	TCP	FTP control	ftp
22	TCP	SSH	—
23	TCP	Telnet	telnet
25	TCP	SMTP	smtp
53	UDP, TCP	DNS	domain
67, 68	UDP	DHCP	nameserver
69	UDP	TFTP	tftp
80	TCP	HTTP (WWW)	www
110	TCP	POP3	pop3
161	UDP	SNMP	snmp
443	TCP	SSL	—

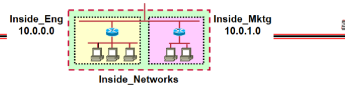
An example of extended numbered ACL for implementing the required security policy in the previous slide.




```
R1(config)# access-list 101 remark Stop Bob to TFTP servers, and Larry to Server1 web
R1(config)# access-list 101 deny udp host 172.16.3.10 172.16.1.0 0.0.0.255 eq tftp
R1(config)# access-list 101 deny tcp host 172.16.2.10 host 172.16.1.100 eq www
R1(config)# access-list 101 permit ip any any
R1(config)# interface Ethernet0
R1(config-if)# ip access-group 101 out
```

- The permit statement is to allow other traffic to pass through. Otherwise, the default deny any any will block all traffic.

An example of configuring network object group.




```
router(config)# object-group network Inside_Eng
router(config-network-group)# 10.0.0.0 /24
router(config-network-group)# exit
router(config)# object-group network Inside_Mktg
router(config-network-group)# host 10.0.1.1
router(config-network-group)# host 10.0.1.2
router(config-network-group)# exit
router(config)# object-group network Inside_Networks
router(config-network-group)# group-object Inside_Eng
router(config-network-group)# group-object Inside_Mktg
```

Notes:

- Object group may be nested using the group-object command
- Configuration commands on ASA firewalls are slightly different

In addition, the automatic numbering of the ACL statements can be resequenced as follows, e.g. if you need to insert more ACL statements in between:

Resequence numbering of ACL statements in ACL list:

```
! Step 6: resequence the ACL's contents
R1(config)# ip access-list resequence 24 10 20
```

starting sequence number

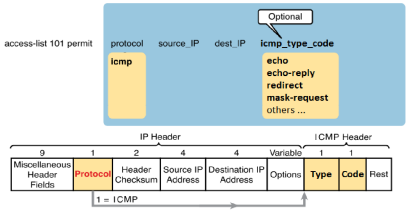
increment step

Verify the resultant ACL again:

```
! Step 7: Displaying the ACL's contents one last time
R1(config)# do show ip access-list 24
Standard IP access list 24
    10 deny   10.1.1.1
    30 permit 10.1.1.0, wildcard bits 0.0.0.255
    50 permit 10.1.3.0, wildcard bits 0.0.0.255
```


For icmp protocol type, the test-conditions may include optional special keywords representing different types of ICMP messages.

Note that there is no keyword like 'eq' for icmp protocol_type_code as compared to that for tcp and udp protocol types.



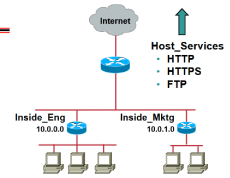
Alternatively, two extended numbered ACLs may be written for implementing the required security policy in the previous slide as follows:

```
R2(config)# access-list 102 remark Stop Larry to Server1 web
R2(config)# access-list 102 deny tcp host 172.16.2.10 host 172.16.1.100 eq www
R2(config)# access-list 102 permit ip any any
R2(config)# interface Ethernet0
R2(config-if)# ip access-group 102 in
```

```
R3(config)# access-list 103 remark Stop Bob to TFTP servers
R3(config)# access-list 103 deny udp host 172.16.3.10 172.16.1.0 0.0.0.255 eq tftp
R3(config)# access-list 103 permit ip any any
R3(config)# interface Ethernet0
R3(config-if)# ip access-group 103 in
```

Note: Since extended ACLs can filter based on destination addresses, it is recommend to be configured near the source of packets to avoid wasting bandwidth to forward packets and then drop at the end.

An example of configuring service object group.

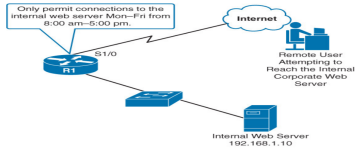



```
router(config)# object-group service Host_Services
router(config-service-group)# tcp www
router(config-service-group)# tcp 443
router(config-service-group)# tcp ftp
```

Note: Configuration commands on ASA firewalls are slightly different, use ? for help.

(4.4) Time-based ACLs are basically ACLs enhanced with time-range which will only become operational during certain time of the day or week.

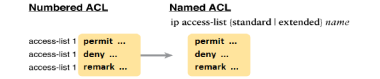
For example, a company may wish to only allow access to its internal web server during office hours. If without time-based ACL, it will be tedious and error-prone to manually remove the ACL during office hours, and add in after office hours.



Time-range is referencing the router's internal clock, which for accuracy can be synchronized using NTP (to be covered in later labs).

(4.3) Named ACLs provide the same functionalities as numbered ACLs except that names are used to identify ACLs which are easier to remember than numbers.

To configure standard or extended named ACLs, just convert the equivalent standard or extended numbered ACLs as follows:

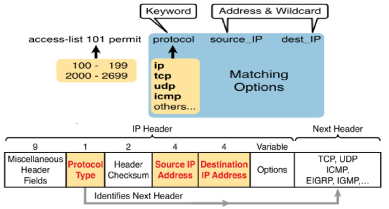


An example of configuring extended named ACL:

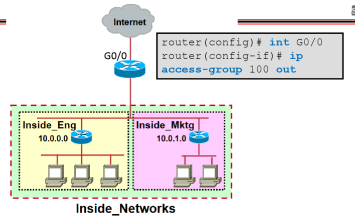
```
Router(config)# ip access-list extended barney
Router(config-ext-nacl)# permit tcp host 10.1.1.2 eq www any
Router(config-ext-nacl)# deny udp host 10.1.1.1 10.1.2.0 0.0.0.255
Router(config-ext-nacl)# deny ip 10.1.3.0 0.0.0.255 10.1.2.0 0.0.0.255
Router(config-ext-nacl)# deny ip 10.1.2.0 0.0.0.255 10.2.3.0 0.0.0.255
Router(config-ext-nacl)# permit ip any any
Router(config-ext-nacl)# interface serial1
Router(config-if)# ip access-group barney out
```


In general, extended ACLs are similar as standard ACLs except that the test-conditions are more complex but useful because they can filter based on more data fields.

At a minimum, the test-conditions must specify protocol type, source IP address and destination IP address:



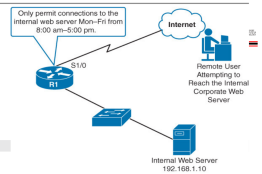
Finally, configure object group-based ACL based on the configured network and service object groups as follows:




```
router(config)# access-list 100 permit object-group
Host_Services object-group Inside_Networks any
```


An example of configuring time-based ACL.

```
R1# conf term
R1(config)# time-range WEEKDAYS
R1(config-time-range)# periodic ?
  Friday      Friday
  Monday      Monday
  Saturday    Saturday
  Sunday      Sunday
  Thursday    Thursday
  Tuesday     Tuesday
  Wednesday   Wednesday
  daily       Every day of the week
  weekdays    Monday thru Friday
  weekend     Saturday and Sunday

R1(config-time-range)# periodic weekdays 8:00 to 17:00
R1(config-time-range)# exit
R1(config)# access-list 100 permit tcp any host 192.168.1.10 eq 80 time-range
  WEEKDAYS

... OUTPUT OMITTED FOR OTHER PERMIT ACL STATEMENTS NOT RELEVANT TO THIS EXAMPLE ...

R1(config)# interface serial 1/0
R1(config-if)# ip access-group 100 in
R1(config-if)# end
```



In newer versions of Cisco IOS, numbered ACLs are also able to be configured in the same way as named ACLs.

Just use number instead of name to configure numbered ACL using the same style as named ACL:

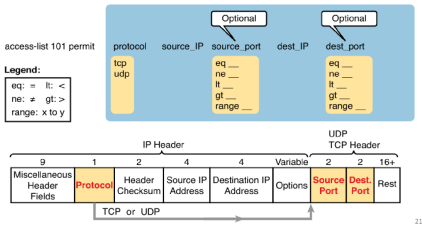
```
! Step 1: The Standard Numbered IP ACL is configured.
R1# configure terminal
Enter configuration commands, one per line.  End with Ctrl-Z.
R1(config)# ip access-list standard 24
R1(config-std-nacl)# permit 10.1.1.0 0.0.0.255
R1(config-std-nacl)# permit 10.1.2.0 0.0.0.255
R1(config-std-nacl)# permit 10.1.3.0 0.0.0.255
```

Note that sequence numbers are automatically inserted by IOS:

```
! Step 2: Displaying the ACL's contents, without leaving configuration mode.
R1(config-std-nacl)# do show ip access-list 24
Standard IP access list 24
    10 permit 10.1.1.0, wildcard bits 0.0.0.255
    20 permit 10.1.2.0, wildcard bits 0.0.0.255
    30 permit 10.1.3.0, wildcard bits 0.0.0.255
```

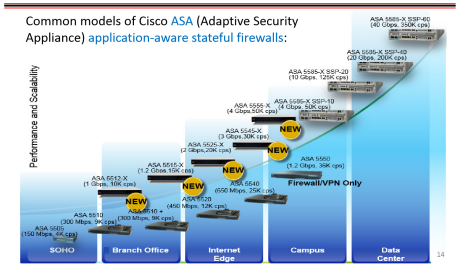

For tcp and udp protocol type, the test-conditions can further specify to filter based on layer 4 source port numbers and/or destination port numbers if need to.

Note the use of keywords like 'eq', etc before port numbers:



To have a better understanding of firewall, we'll now study the Cisco ASA firewall which is an application-aware stateful firewall.

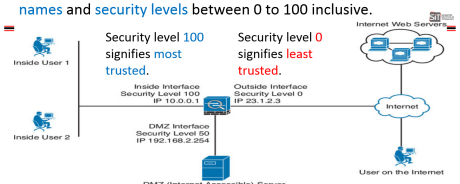
Common models of Cisco ASA (Adaptive Security Appliance) application-aware stateful firewalls:



Lab Exercise 1.1

1.1 To get Cisco ASA up and operational, configure interface security level to segment networks into different security zones.

ASAs are typically deployed in routed mode like routers and its interfaces must be configured with IP addresses, and also names and security levels between 0 to 100 inclusive.




```
ciscoasa(config)# interface g0/1
ciscoasa(config-if)# nameif dmz
ciscoasa(config-if)# security-level 50
ciscoasa(config-if)# ip address 192.168.2.254 255.255.255.0
ciscoasa(config-if)# no shutdown
```


To verify that the interfaces of ASA are up and operational after configuration, use the similar CLI command as routers but note the slight difference as shown:

```
ciscoasa# show interface ip brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0     23.1.2.3        YES manual up                    up
GigabitEthernet0/1     unassigned      YES unset  administratively down down
GigabitEthernet0/2     unassigned      YES unset  administratively down down
  .
  .
Management0/0          192.168.1.1     YES manual up                    up
ciscoasa#
```

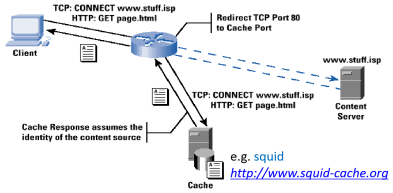
To show the details of each interface:

```
ciscoasa# show interface if_number
```

```
ciscoasa# show interface name
```

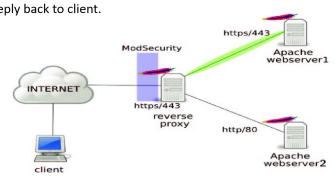

Nowadays, it is more common to have transparent proxy which will automatically intercept and restrict what users are allowed to access.

An example of transparent web proxy is the free open source Squid with WCCPv2 router interception which you may try in your team project. https://wiki.squid-cache.org/Features/Wccp2



Application proxy firewall may also be deployed in reverse proxy mode; e.g. WAFs to protect web servers.

In reverse proxy mode, client will connect to WAF which acts as the web server. If request is non-malicious, WAF will relay request to web server and reply back to client.



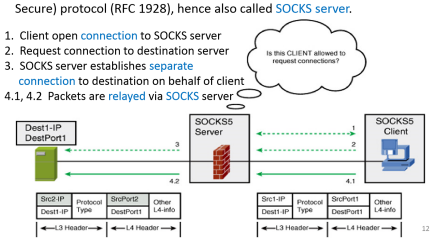
An example of WAF is the free open source ModSecurity which you may also try in your team project. https://github.com/SpiderLabs/ModSecurity

(D) Circuit-level proxy firewall, also referred as generic proxy works similarly by connecting to server on behalf of user, but does not understand application protocols.

Circuit-level proxy is commonly implemented using SOCKS5 (Socket Secure) protocol (RFC 1928), hence also called SOCKS server.

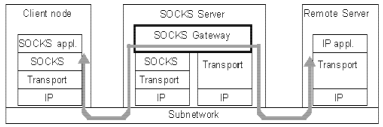
1. Client open connection to SOCKS server
2. Request connection to destination server
3. SOCKS server establishes separate connection to destination on behalf of client

4.1, 4.2 Packets are relayed via SOCKS server



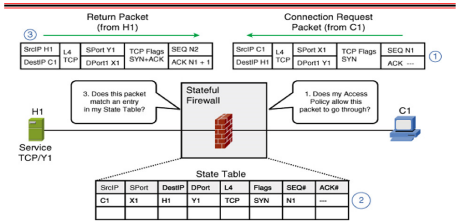
Technically, SOCKS works as a 'shim-layer' between transport and application layers to carry any application-layer traffic.

With circuit-level proxy, instead of opening socket to transport layer directly, applications are modified to call SOCKS5 API to carry its traffic to the firewall, hence called SOCKS applications.



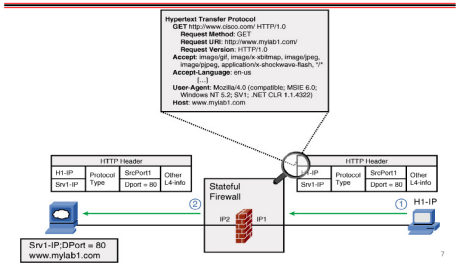
The SOCKS server/firewall will then connect to destination server on behalf of users, and then relay the traffic between them.

(B) To provide better security, stateful inspection firewalls are developed which maintain connection information in state table for packet filtering, and are commonly used today.

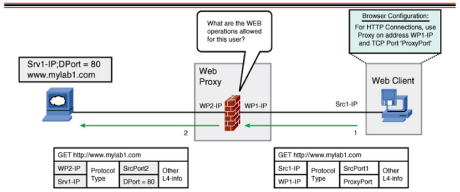


(2) For permitted packet, an entry is created in firewall state table which will be used to verify return packets.

To achieve even better access control, modern stateful firewalls are often application-aware, i.e. able to inspect application-level protocol such as HTTP, e.g. Cisco ASA firewall.



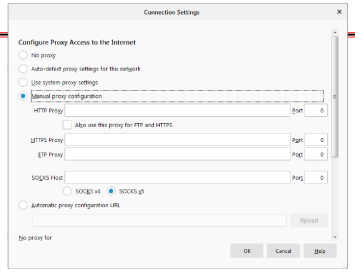
(C) Application proxy firewall, aka as application layer gateway (ALG) is specially designed to filter certain application, e.g. web proxy or WAF (Web Application Firewall).



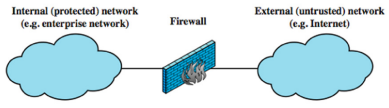
Proxy firewall and application-aware stateful firewall are different:

- Proxy firewall: two spliced connections between user and proxy, and proxy and destination server
- Stateful firewall: one direct connection between user and server

An example of configuring proxy in Firefox browser:



Theoretically, a network firewall is a device or software that segments networks into different security zones, and enforces access control policy on traffic crossing between them.

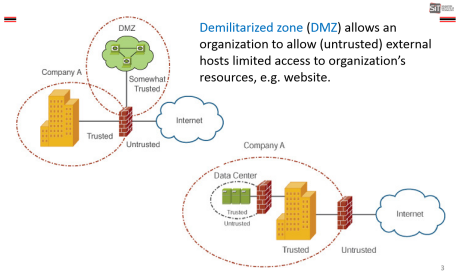


To be effective in enforcing access control policy, a firewall must be:

- Only transit point between different security zones
- Resistant to attacks itself

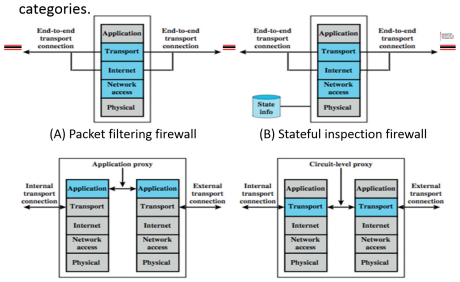
Limitation: Firewall cannot enforce access control policy on network traffic within the same security zone that do not pass through the firewall.

To segment networks into more than two security zones, an enterprise may implement one or more firewalls, e.g.:



Demilitarized zone (DMZ) allows an organization to allow (untrusted) external hosts limited access to organization's resources, e.g. website.

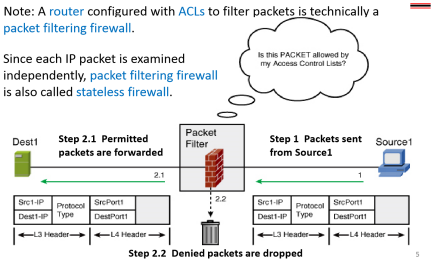
Broadly, firewall technologies may be divided into 4 different categories.



(A) In packet filtering firewall, access control policy rules are implemented as ACLs to filter each IP packet independently, but have limitations as discussed in previous lectures.

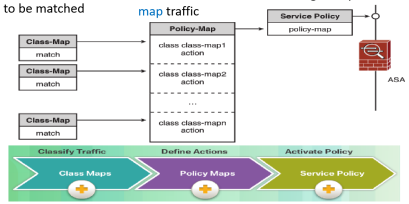
Note: A router configured with ACLs to filter packets is technically a packet filtering firewall.

Since each IP packet is examined independently, packet filtering firewall is also called stateless firewall.



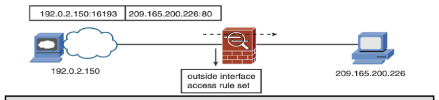
Essentially, configuring ASA using Modular Policy Framework (MPF) involves 3 steps:

1. Configure class-map: specify traffic to be matched
2. Configure policy-map: specify actions to be taken on class-map traffic
3. Configure service-policy: apply policy-map to an interface or globally to all



If needed, ACLs can be configured at ASA to override security level; e.g. to allow lower security-level to access higher security-level, or limit what higher-security level can send.

e.g.: To allow outside users to access web server 209.165.200.226 at DMZ




```
ciscoasa(config)# access-list OUTSIDE_TO_DMZ permit tcp any
host 209.165.200.226 eq www
ciscoasa(config)# access-group OUTSIDE_TO_DMZ in interface
outside
```

Note some differences between configuring ACLs in ASAs and routers:

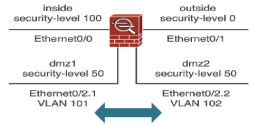
1. ASAs assume ACLs are extended and 'extended' keyword is optional
2. ASAs use standard subnet mask but not wildcard mask to specify IP address block

Another state table in ASA is the local host table which maintains per-host statistics as follows:

```
FIREWALL# show local-host 10.0.0.108
Interface management: 0 active, 0 maximum active, 0 denied
Interface DMZ: 2 active, 69 maximum active, 0 denied
Interface inside: 130 active, 320 maximum active, 0 denied
local host: <10.0.0.108>,
    TCP flow count/limit = 15/unlimited
    TCP embryonic count to host = 0
    TCP intercept watermark = unlimited
    UDP flow count/limit = 10/unlimited

  Xlate:
    NAT from any:10.0.0.108 to outside:209.165.200.238 flags I idle 0:00:00
      timeout 3:00:00

  Conn:
    TCP outside 192.0.2.65:22 inside 10.0.0.108:50124, idle 0:00:58, bytes 443,
      flags UIO
    ... output omitted ...
Interface outside: 120 active, 1940 maximum active, 0 denied
```


Note that if you choose to configure two or more interfaces with the same security level, traffic are by default not allowed to pass between them.

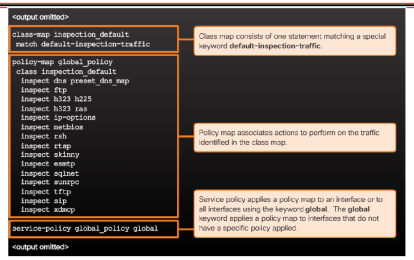
e.g.: you can use any security level between 0 and 100 to configure DMZ, but say you choose to use the same security level 50 for dmz1 and dmz2 as follows:



To override default security policy and allow traffic to be initiated from same security level:

```
ciscoasa(config)# same-security-traffic permit inter-interface
```


By default, ASA is pre-configured with a class-map named inspection-default and a policy-map named global_policy which is applied globally using service-policy on all ASA interfaces.



Similar as Cisco IOS routers, line numbers are automatically added into ACLs in ASAs to support editing as follows:

To show ACL, use the command:

```
ciscoasa# show access-list OUTSIDE_TO_DMZ
access-list OUTSIDE_TO_DMZ; 1 elements; name hash: 0x5113f0a2
access-list OUTSIDE_TO_DMZ line 1 extended permit tcp any host
209.165.200.226 eq www (hitcnt=9) 0x4a02e414
```

To insert an ACL statement, use 'line' keyword to indicate the position:

```
ciscoasa(config)# access-list OUTSIDE_TO_DMZ line 1 remark
allow internet access to web
```

To delete an ACL statement, use 'no' keyword in front:

```
ciscoasa(config)# no access-list OUTSIDE_TO_DMZ line 1 remark
allow internet access to web
```

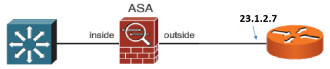
To delete the whole ACL:

```
ciscoasa(config)# clear configure access-list OUTSIDE_TO_DMZ
```


Note that for connection with other network devices, ASAs will also need to be configured with static and/or default routes, or routing protocols to learn routes dynamically.

To configure static route in ASA:

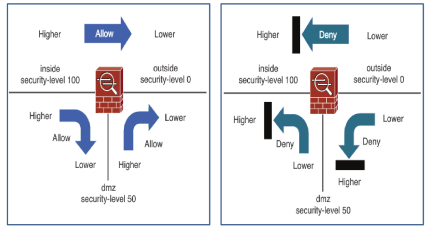
```
ciscoasa(config)# route interface ip_address netmask gateway_ip
```



e.g.: Suppose your ISP can be reached via 23.1.2.7, you can configure a static default route in ASA as follows:

```
ciscoasa(config)# route outside 0.0.0.0 0.0.0.0 23.1.2.7
```


By default, traffic initiated from higher security level to lower security level are allowed but not the reverse.



Specifically, the default-inspection-traffic in class-map inspection_default is matching commonly encountered network traffic as follows:

```
default-inspection-traffic  Match default inspection traffic:

        ctiqbe----tcp--2748       dns-------udp--53
        ftp-------tcp--21         gtp-------udp--2123,3386
        h323-h225-tcp--1720       h323-ras--udp--1718-1719
        http------tcp--80         icmp------icmp
        ils-------tcp--389        ip-options-----rsvp
        mgcp------udp--2427,2727  netbios---udp--137-138
        radius-acct---udp--1646   rpc-------udp--111
        rsh-------tcp--514        rtsp------tcp--554
        sip-------tcp--5060       sip-------udp--5060
        skinny----tcp--2000       smtp------tcp--25
        sqlnet----tcp--1521       tftp------udp--69
        waas------tcp--1-65535    xdmcp-----udp--177
```

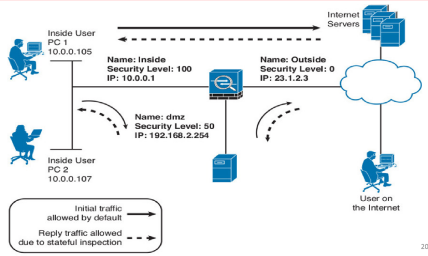

Lab Exercise 1.3

1.3 Configure ASA service policy using Modular Policy Framework (MPF) for advanced features, e.g. enabling inspection engine, etc.

To verify the routing table in the ASA, use the similar command as in routers but note the slight difference as shown:

```
ciscoasa# show route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is 23.1.2.7 to network 0.0.0.0

C    23.1.2.0 255.255.255.0 is directly connected, outside
C    192.168.1.0 255.255.255.0 is directly connected, management
S*   0.0.0.0 0.0.0.0 [1/0] via 23.1.2.7, outside
ciscoasa#
```


Nevertheless, due to stateful firewall feature which maintains state tables of initiated connections, return traffic from lower to higher security level are automatically allowed.



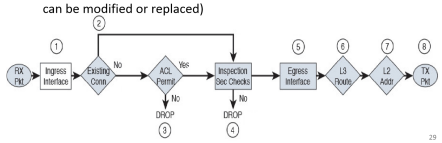
To view statistics of the matching traffic inspected by the ASA, use the show command as follows:

```
ciscoasa# show service-policy
Global policy:
  Service-policy: global_policy
    Class-map: inspection_default
      Inspect: dns preset_dns_map, packet 0, drop 0, reset-drop 0
      Inspect: ftp, packet 0, drop 0, reset-drop 0
      Inspect: h323 h225 _default_h323_map, packet 0, drop 0, reset-drop 0
               tcp-proxy: bytes in buffer 0, bytes dropped 0
      Inspect: h323 ras _default_h323_map, packet 0, drop 0, reset-drop 0
      Inspect: rsh, packet 0, drop 0, reset-drop 0
      Inspect: rtsp, packet 0, drop 0, reset-drop 0
               tcp-proxy: bytes in buffer 0, bytes dropped 0
      Inspect: esmtp _default_esmtp_map, packet 0, drop 0, reset-drop 0
      Inspect: sqlnet, packet 0, drop 0, reset-drop 0
      Inspect: skinny, packet 0, drop 0, reset-drop 0
               tcp-proxy: bytes in buffer 0, bytes dropped 0
      Inspect: sunrpc, packet 0, drop 0, reset-drop 0
               tcp-proxy: bytes in buffer 0, bytes dropped 0
      Inspect: xdmcp, packet 0, drop 0, reset-drop 0
      Inspect: sip, packet 0, drop 0, reset-drop 0
               tcp-proxy: bytes in buffer 0, bytes dropped 0
      Inspect: netbios, packet 0, drop 0, reset-drop 0
      Inspect: tftp, packet 0, drop 0, reset-drop 0
      Inspect: ip-options _default_ip_options_map, packet 0, drop 0, reset-drop 0
```


Before discussing the configuration of Modular Policy Framework (MPF), it's useful to understand packet processing order of operations in ASA.

When a packet arrives at ASA and before it is allowed to pass through, it will go through the following filtering as shown:

② States table (maintain automatically after packets are permitted by interface security level)
③ Access control list (to override interface security level if needed)
④ MPF inspection security policy (shipped with a factory default, but can be modified or replaced)



Lab Exercise 1.2

1.2 Configure access control list in Cisco ASA to override security level if needed.

Specifically, one of the state tables maintained by ASA is the connection table which automatically tracks the connection parameters of TCP and UDP:

Protocol	Extent of Stateful Tracking
TCP	Source and destination addresses and ports, TCP flags, sequence numbers (in both directions), and idle time
UDP	Source and destination addresses and ports, idle time, and for some applications, request identifiers

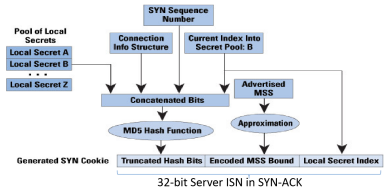
To view the content of connection table:

```
ciscoasa# show conn
352 in use, 5985 most used
UDP DMZ 172.16.0.25:161 inside 10.0.0.30:1879, idle 0:00:27, bytes 706509, flags -
UDP outside 192.0.2.18:123 inside 10.0.0.108:123, idle 0:00:34, bytes 79, flags -
TCP outside 192.0.2.150:80 DMZ 172.16.0.5:59512, idle 0:00:00, bytes 0, flags Uf
TCP outside 192.0.2.150:80 inside 10.0.0.102:59559, idle 0:00:03, bytes 1488,
    flags UfFRIO
...
```

Or to view more details: `ciscoasa# show conn detail`

The procedure for generating SYN cookie varies with different implementations, an example as follows:

- Connection Info Structure: IP addresses and TCP ports
- SYN Sequence Number: 32-bit Client ISN



32-bit Server ISN in SYN-ACK

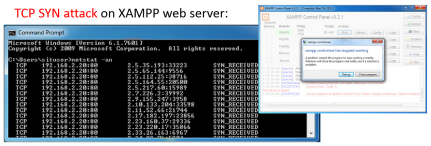
- To validate reply, decrease ACK number – 1 to recover SYN cookie, and local secret index, then apply the same procedure. If valid, allow TCP connection to internal server.

hping3 in Kali Linux may be used to conduct TCP SYN attack which you will try in the lab.

e.g.: TCP SYN attack on port 80 at victim

```
root@kali:~# hping3 -S -P --flood -V --rand-source -p 80
target_domain_name_or_ip
```

TCP SYN attack on XAMPP web server:



- Notice all TCP connections for the web server are stuck at SYN_RCVD state, consuming all system resources.
- Latest version of XAMPP may not crash, but you may notice system becomes sluggish during attack.

An example of configuring MPF layer 3/4 policy map based on previously configured class maps:

```
ciscoasa(config)# policy-map p1
ciscoasa(config-pmap)# class anything
ciscoasa(config-pmap-c)# set connection ...
ciscoasa(config-pmap-c)# inspect ...
ciscoasa(config-pmap-c)# exit
ciscoasa(config-pmap)# class voice
ciscoasa(config-pmap-c)# priority
ciscoasa(config-pmap-c)# exit
ciscoasa(config-pmap)# class data-center
ciscoasa(config-pmap-c)# set connection timeout ...
ciscoasa(config-pmap)# exit
```

Note:

- When a packet matches a class map for a feature type, ASA does not match it to any subsequent class maps for same feature type.
- If the packet matches a subsequent class map for a different feature type, then ASA also applies the actions for the subsequent class map.

For more information, refer
https://www.cisco.com/c/en/us/td/docs/security/asa/asa96/configuration/firewall/asa-96-firewall-config/inspect-service-policy.html

To implement your required access control policy in ASA, you may modify the default policy-map as follows or configure your own MPF as discussed in the next few slides.

E.g.: For security reason, ICMP traffic is denied in ASA by default. But if your company's security policy allows ICMP, you can configure ASA to maintain stateful tracking of ICMP to permit return ICMP traffic by adding ICMP inspection engine in the default policy-map as shown below:

```
ciscoasa(config)# policy-map global_policy
ciscoasa(config-pmap)# class inspection_default
ciscoasa(config-pmap-c)# inspect icmp
ciscoasa(config-pmap-c)# exit
ciscoasa(config-pmap)# exit
ciscoasa(config)#
```

Note: You may also write your own policy-map and apply it on appropriate interface of the ASA.

To verify and monitor attacks intercepted by TCP intercept, you may configure threat detection statistics as follows:

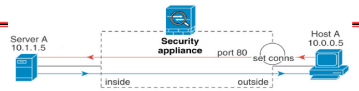
- Enabling threat detection statistics for TCP intercept:

```
ciscoasa(config)# threat-detection statistics tcp-intercept
rate-interval 3 burst-rate 60 average-rate 30
```

- rate-interval sets the size of monitoring window in mins during which ASA samples the attacks 60 times
- burst-rate and average-rate are in attacks/events per sec (eps), when exceeded will generate syslog message
- Monitoring TCP intercept statistics:

```
ciscoasa(config)# show threat-detection statistics tcp-intercept
Top 10 protected servers under attack (sorted by average rate)
Monitoring window size: 3 mins    Sampling interval: 3 secs
<Rank> <Server IP:Port> <Interface> <Ave Rate> <Cur Rate> <Total> <Source IP (Last Attack Time)>
------------------------------------------------------------------------------
1    10.1.1.5:80 inside 1249  9503  2249245  10.0.0.3 (0 secs ago)
```


To defend TCP SYN attack, ASA can be configured using MPF to limit the maximum number of embryonic connections.



Step 1: Create class map to specify server to protect

```
ciscoasa(config)# access-list serverA extended permit tcp any host 10.1.1.5 eq 80
ciscoasa(config)# class-map http_serverA
ciscoasa(config-cmap)# match access-list serverA
```

Step 2: Create policy map to specify actions to take

10 and 100 are just examples

```
ciscoasa(config)# policy-map policy_serverA
ciscoasa(config-pmap)# class http_serverA
ciscoasa(config-pmap-c)# set connection embryonic-conn-max 100
ciscoasa(config-pmap-c)# set connection per-client-embryonic-max 10
```

Step 3: Apply policy map on appropriate interface

```
ciscoasa(config)# service-policy policy_serverA interface outside
```


Step 3: Configure MPF service policy to apply policy map to the ASA interface or globally as follows:

```
ciscoasa(config)# service-policy policy_map_name {global | interface if_name}
```

Based on where service policy is configured, the actions for different feature types will be applied to traffic bi-directionally or uni-directionally:

Feature	Single Interface Direction	Global Direction
Application inspection (multiple types)	Bidirectional	Ingress
NetFlow Secure Event Logging filtering	N/A	Ingress
QoS input policing	Ingress	Ingress
QoS output policing	Egress	Egress
QoS standard priority queue	Egress	Egress
TCP and UDP connection limits and timeouts, and TCP sequence number randomization	Bidirectional	Ingress
TCP normalization	Bidirectional	Ingress
TCP state bypass	Bidirectional	Ingress

- Each interface can have only one service-policy applied to it. In addition, only one global policy can be applied to all interfaces.
- Interface service-policy takes precedence over global service-policy.

Alternatively, to configure new MPF – Step 1: Configure MPF class map to specify what type of network traffic to be matched according to some conditions, e.g.:

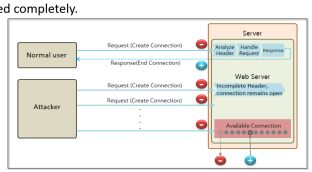
```
ciscoasa(config)# class-map class_map_name
ciscoasa(config-cmap)# description text
ciscoasa(config-cmap)# match ...
ciscoasa(config-cmap)# exit
```

Common match commands for specifying traffic:

Matching Condition	Command Syntax
Any traffic	ciscoasa(config-cmap)# **match any**
Default traffic types	ciscoasa(config-cmap)# **match default-inspection-traffic**
Destination port number	ciscoasa(config-cmap)# **match port** {**tcp** | **udp**} {**eq** *port* | **range** *start end*}
Access list	ciscoasa(config-cmap)# **match access-list** *acl_name*
RTP port number range	ciscoasa(config-cmap)# **match rtp** *starting_port range*

Slowloris attack is an example of DoS attack on web servers using low-rate but maliciously crafted packets to exploit design weakness of HTTP protocol.

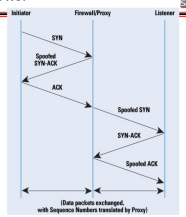
By design, web server will only process HTTP request when header is received completely.



By creating many concurrent connections to the web server, and for each connection, intentionally sending HTTP header slowly in many small fragments, Slowloris is thus able to consume all connections and make the server unavailable.

When the embryonic connection threshold is crossed, the ASA has a component called TCP Intercept which will be activated to act as a proxy for the server as follows:

- Intercept external SYN request to server and reply spoofed SYN-ACK on behalf of server.
- If subsequent ACK is received which indicates non-malicious request, then initiate spoofed SYN request to server to setup connection.



The maximum concurrent and embryonic connections can exceed the configured numbers, up to *n*-1 extra connections and embryonic connections, where *n* is the number of cores on your ASA.

Lab Exercise 2

2.1 TCP SYN flooding attack

2.2 Defense:

- Configure MPF service policy in ASA

An example of configuring MPF class map:

```
ciscoasa(config)# class-map anything
ciscoasa(config-cmap)# match any
ciscoasa(config-cmap)# exit
!
ciscoasa(config)# class-map voice
ciscoasa(config-cmap)# match rtp 2000 100
ciscoasa(config-cmap)# exit
!
ciscoasa(config)# access-list dc extended permit ip any 10.100.0.0 255.255.0.0
ciscoasa(config)# class-map data-center
ciscoasa(config-cmap)# match access-list dc
ciscoasa(config-cmap)# exit
```


To try Slowloris attack in the lab, you may use the slowhttptest in Kali Linux.

```
root@kali:~# slowhttptest -c number_connections -u target_url
```

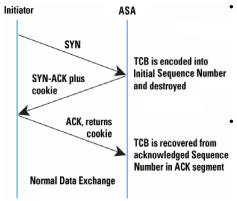
Refer *https://tools.kali.org/stress-testing/slowhttptest* on the use of slowhttptest.

An example of Slowloris attack on XAMPP Apache web server with default setting.

Notice that a single client making 150 connections is sufficient to make Apache web server unavailable!

Similar as defending TCP SYN attack, MPF inspection security policy may be configured to defend Slowloris. Try it yourself!

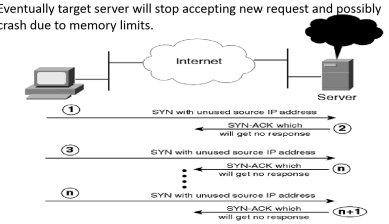
In addition, ASA implements TCP SYN cookie mechanism to protect itself as a proxy from TCP SYN flooding attack.



- The TCP TCB (Transmission Control Block), i.e. data structure/memory used to store TCP half-open state in 3-way handshake is encoded into ISN and destroyed.
- The reply, if arrives, will contain ACK sequence number ISN+1 which will enable proxy to reconstruct TCB, and if valid, to continue with the connection request.

TCP SYN flooding attack is an effective DoS attack which sends TCP SYN segments to a target server but deliberately never completes the TCP handshake.

(Recall ICT1010 on TCP 3-way handshake.) Sending SYN request without subsequent ACK will result in half-open (embryonic) connection. Eventually target server will stop accepting new request and possibly crash due to memory limits.



Step 2: Configure MPF layer 3/4 policy map to specify one or more actions to take on the matched traffic, e.g.:

```
ciscoasa(config)# policy-map policy_map_name
ciscoasa(config-pmap)# description text
ciscoasa(config-pmap)# class class_map_name
ciscoasa(config-pmap-c)# ...
```

Some commands for configuring actions to take on matched traffic:

Action	Command Syntax
Set connection limits	ciscoasa(config-pmap-c)# **set connection** ...
Inspect applications	ciscoasa(config-pmap-c)# **inspect** *engine_name*
Police the traffic	ciscoasa(config-pmap-c)# **police** {**output** | **input**} *conform_rate* [*burst_bytes*] [**conform-action** {**drop** | **transmit**}] [**exceed-action** {**drop** | **transmit**}]
Apply priority handling	ciscoasa(config-pmap-c)# **priority**

HTTP Inspection – STEP 1

The list of regular expression metacharacters and their usages

Metacharacter	Name	Function
.	Dot	Matches any single character. Example: b.d matches bad, bbd, bcd, bdd, bed, and so on
()	Subexpression	Groups the characters inside the parentheses as a single expression for matching with other metacharacters.
|	Or	Matches either expression that | separates. Example: com|net matches whatever.com or whatever.net Example: Ma(r|y) matches Mar or May
?	Question mark	Matches 0 or 1 of the expression just before the ?. Example: e?smtp matches smtp (zero e's) or esmtp (1 e) Example: (12)? matches 4444, 12444, 1212444, and so on
*	Asterisk	Matches 0, 1, or any number of the expression just before the *. Example: w* matches cisco.com and www.cisco.com
+	Plus	Matches at least one of the expressions just before the +. Example: w+ matches www.cisco.com, but not cisco.com

ASA: Application Layer Inspection

Step 1: Application layer protocol inspection class map

```
host(config)# class-map type inspect application [match-all | match-any]
              application_class_map
```

Step 2: Application layer protocol inspection policy map

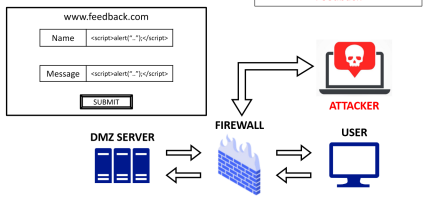
```
host(config)# policy-map type inspect application application_policy_map
```

Step 3: Inspection policy map over MPF layer 3/4 policy map

```
host(config)# policy-map policy_map_name
host(config-pmap)# class class_map_name
host(config-pmap-c)# inspect application application_policy_map
```


Cross-Site Scripting (XSS) Attack

Step 2
Insert HTML/Java Script in the Feedback



ICT2203 Network Security

Attacks and Defense of IP Networks with Firewalls II

Lab 6.2 Notes

Application Layer Protocol Inspection

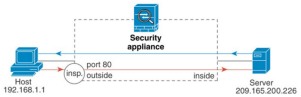
HTTP Inspection – STEP 1

The list of regular expression metacharacters and their usages

Metacharacter	Name	Function
{n}	Repeat	Matches if the expression just before {n} is repeated exactly n times. Example: (test){2} matches testtest but not testtesttest
{n,}	Minimum repeat	Matches if the expression just before {n,} is repeated at least n times. Example: (test){2} matches testtest and also testtesttest
[abc]	Character class	Matches any of the characters listed between the square brackets. Example: [dfhl]og matches dog, fog, hog, and log, but not frog
[^abc]	Not character class	Matches any character that is not listed between the brackets. Example: [^dfhl]og matches cog, but not dog, fog, hog, or log
[a-c]	Character range class	Matches any character in the range from a to c. Example: [a-z] matches any lowercase letter, [A-Z] matches any uppercase letter, [0-9] matches any digit

How to Defend XSS?

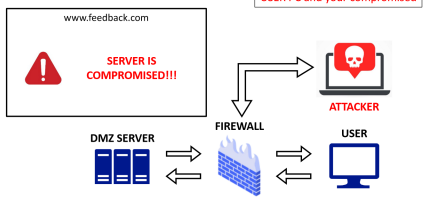
We will use the HTTP Application Layer Protocol Inspection




```
ciscoasa(config)# class-map type inspect http [match-all | match-any] cmap_name
ciscoasa(config-cmap)# match [not] {request | response | req-resp} ...
ciscoasa(config-cmap)# ...
ciscoasa(config-cmap)# exit
ciscoasa(config)# policy-map type inspect http pmap_name
ciscoasa(config-pmap)# class cmap_name
ciscoasa(config-pmap-c)# {drop-connection [log] | log | reset [log]}
ciscoasa(config-pmap-c)# exit
```


Cross-Site Scripting (XSS) Attack

Step 3
Access the website from the USER PC and your compromised



What To Do in This LAB?

- Different Application Layer Attacks: SQL Injection, Cross-Site scripting, Session Hijacking, Remote Code Execution, File Upload Abuse, Distributed Denial of Service and so on….
- In this LAB, we will implement Cross-Site Scripting (XSS) Attack using DVWA
- We will then use ASA to defend the Cross-Site Scripting Attack

HTTP Inspection – STEP 1

The list of regular expression metacharacters and their usages

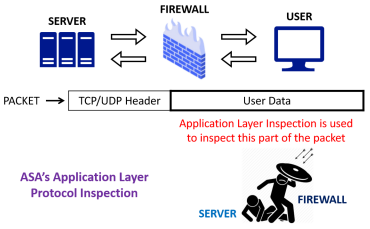
Metacharacter	Name	Function
^	Caret	Matches the beginning of a line; any expression following the caret will be matched only if it appears at the beginning of a line. Example: ^Dear matches "Dear John" but not "John Dear"
\	Escape	The metacharacter following \ will be treated as a literal character; this is useful when you need to match against something that is normally interpreted as a metacharacter. Example: \^Test matches "^Test"
\r	Carriage return	Matches a carriage return character (ASCII 13 or 0x0d).
\n	Newline	Matches a newline character (ASCII 10 or 0x0a).
\t	Tab	Matches a tab character (ASCII 9 or 0x09).
\f	Form feed	Matches a form feed character (ASCII 12 or 0x0c).
\xNN	Escaped hex number	Matches an ASCII character that has the two-digit hex code NN. Example: \x20 matches a space (ASCII 32)
\NNN	Escaped octal number	Matches an ASCII character that has the three-digit octal code NNN. Example: \040 matches a space (ASCII 32)

HTTP Inspection – STEP 1

HTTP inspection policy map can be configured using the match commands as below

```
ciscoasa(config-cmap)# match [not] request header {[field]
[regex [regex_name | class regex_class_name]] |
[length gt max_length_bytes | count gt max_count_bytes]}

ciscoasa(config-cmap)# match [not] request method {[method] |
[regex [regex_name | class regex_class_name]]

ciscoasa(config-cmap)# match [not] request uri {regex [regex_name | class
regex_class_name] | length gt max_bytes}

ciscoasa(config-cmap)# match [not] request args regex [regex_name | class
regex_class_name]

ciscoasa(config-cmap)# match [not] request body {regex [regex_name | class
regex_class_name] | length gt max_bytes}

ciscoasa(config-cmap)# match [not] req-resp content-type mismatch

ciscoasa(config-cmap)# match [not] response header {[field]
[regex [regex_name | class regex_class_name]] |
[length gt max_length_bytes | count gt max_count]}

ciscoasa(config-cmap)# match [not] response status-line [regex [regex_name | class
regex_class_name]]

ciscoasa(config-cmap)# match [not] response body {[active-x] | [java-applet] |
[regex [regex_name | class regex_class_name]] | length gt max_bytes}
```

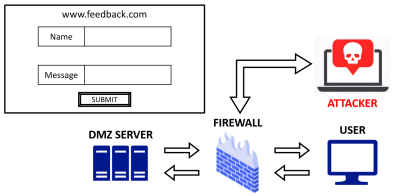

XSS – Defence:



https://www.cisco.com/c/en/us/td/docs/security/asa/asa72/configuration/guide/conf_gd/inspect.html#wp1514315

Cross-Site Scripting (XSS) Attack

- What is XSS?
  - Occurs when an attacker injects code.
  - Types: Reflected, **Stored** and DOM-Based.



HTTP Inspection – STEP 2

Configure HTTP inspection policy map to specify actions to be performed on matched traffic

```
ciscoasa(config-pmap-c)# {[drop [send-protocol-error] |
drop-connection [send-protocol-error]] | mask | reset} [log] |
rate-limit message_rate}
```

- **drop** keyword drops the packets
- **send-protocol-error** keyword sends a protocol error message
- **drop-connection** keyword drops the packet and closes the connection
- **mask** keyword masks out the matching portion of the packet
- **reset** keyword drops the packet, closes the connection, and sends a TCP reset
- **log** keyword sends a system log message
- **rate-limit** *message_rate* argument limits the rate of messages

Note that not all actions are allowed for different match command.

Use CLI ? help for exact options available.

HTTP Inspection – STEP 1

Regular expression (regex) is a powerful way to specify search patterns for matching and can be configured in two ways as follows:

Single **Regular Expression**

```
ciscoasa(config)# regex regex_name regular_expression
ciscoasa(config)# regex ...
```

Group of **Regular Expressions** using class-map

```
ciscoasa(config)# class-map type regex match-any regex_cmap_name
ciscoasa(config-cmap)# match regex regex_name
ciscoasa(config-cmap)# match regex ...
ciscoasa(config-cmap)# exit
```

While formulating your regular expression, you may test it as follows:

```
ciscoasa# test regex input_text regular_expression

ciscoasa# test regex https://shadycontent.com/toolz https?://
INFO: Regular expression match succeeded.
```

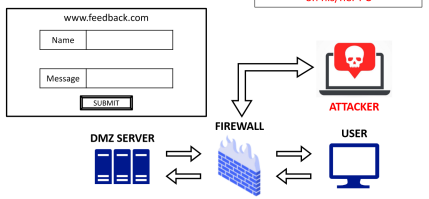

ASA: Application Layer Inspection

Configuration of Application Inspection in ASA

Step 1	Step 2	Step 3
Configure application layer protocol inspection class map	Configure application layer protocol inspection policy map	Apply inspection policy map over MPF layer 3/4 policy map discussed in the last lecture

Cross-Site Scripting (XSS) Attack

Step 1
The attacker opens the website on his/her PC



HTTP Inspection – STEP 2

In addition, HTTP inspection policy map has special parameters option for specifying action to take if HTTP protocol violation occurs.

To enter into **parameters configuration** mode in inspection policy map:

```
ciscoasa(config-pmap)# parameters
ciscoasa(config-pmap-p)#
```

To **configure** action for **HTTP protocol violation**, enter the command:

```
ciscoasa(config-pmap-p)# protocol-violation [action
                         [drop-connection | reset | log]]
```


HTTP Inspection – STEP 1 & 2

To summarise, here is an example of HTTP inspection class map and inspection policy map.

```
ciscoasa(config)# regex url_example example\.com
ciscoasa(config)# regex url_example2 example2\.com
ciscoasa(config)# class-map type regex match-any URLs
ciscoasa(config-cmap)# match regex url_example
ciscoasa(config-cmap)# match regex url_example2

ciscoasa(config)# class-map type inspect http match-all http-traffic
ciscoasa(config-cmap)# match req-resp content-type mismatch
ciscoasa(config-cmap)# match request body length gt 1000
ciscoasa(config-cmap)# match not request uri regex class URLs

ciscoasa(config)# policy-map type inspect http http-map1
ciscoasa(config-pmap)# class http-traffic
ciscoasa(config-pmap-c)# drop-connection log
ciscoasa(config-pmap-c)# match req-resp content-type mismatch
ciscoasa(config-pmap-c)# reset log
ciscoasa(config-pmap)# parameters
ciscoasa(config-pmap-p)# protocol-violation action log
```

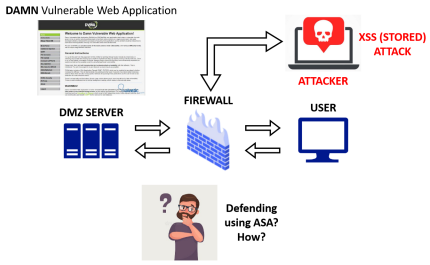

HTTP Inspection – STEP 3

```
ciscoasa(config)# policy-map test
ciscoasa(config-pmap)# class test
ciscoasa(config-pmap-c)# inspect http http-map1

ciscoasa(config)# service-policy test interface outside
```

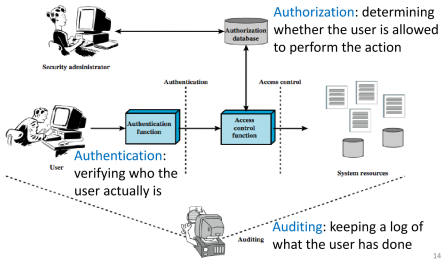
Note: If you modify an in-use HTTP inspection policy map, you must remove and reapply it for the changes to take effect.

```
ciscoasa(config)# policy-map test
ciscoasa(config-pmap)# class http
ciscoasa(config-pmap-c)# no inspect http http-map1
ciscoasa(config-pmap-c)# inspect http http-map1
```


LAB 6.2: What To Do in the LAB?



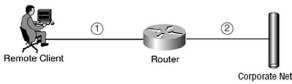
After connecting to network devices, the next problem is how to manage access control, which is achieved by authentication, authorization and accounting, known as AAA (or triple-A).



Broadly, there are two main uses of AAA for access control – device administration and network access. (Our focus will be on device administration.)

Device administration: controlling access to who can log in to a network device to configure and manage it.

Network access: essentially about verifying the identity of the user/ device before permitting to use the network.

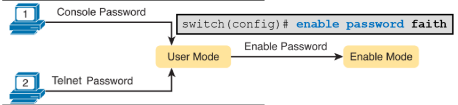


Lab Exercise 1A

1.1 Configure device access authentication using locally stored passwords and/or usernames.

Cisco supports simple password security without username for console and telnet login. But, everyone will use the same password!

```
switch(config)# line con 0
switch(config-line)# password hope
switch(config-line)# login
```

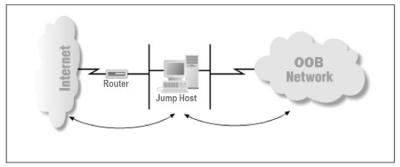



```
switch(config)# line vty 0 15
switch(config-line)# password love
switch(config-line)# login
```

- password *password-string* command defines the password
- login command enables simple password security
- enable password *password-string* command for enable mode password

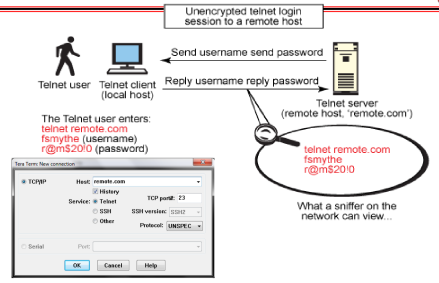
To extend remote access via the Internet, it can be risky to simply expose the network devices. Instead, a jump server/host/box can be implemented.

A jump server/host/box is a device which connects two networks in different security zones to allow access of more sensitive security zone from insecure zone.



Note that the jump server/host/box itself must be hardened to prevent attacks!

Once the network devices are remotely accessible, Telnet was once commonly used but was insecure because all messages are transmitted in clear including login password!



Today, SSH (Secure Shell) is recommended so that all messages are encrypted within an established secure channel (you'll learn the theory in ICT2205 Applied Cryptography).

Configuring network device to support SSH:

```
switch(config)# hostname SW1
SW1(config)# ip domain-name remote.com

SW1(config)# crypto key generate rsa modulus 2048
The name for the keys will be: SW1.remote.com

% The key modulus size is 2048 bits
% Generating 2048 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 74 seconds)

SW1(config)# ip ssh version 2

SW1(config)# ip ssh timeout 120
SW1(config)# ip ssh authentication-retries 3
```

1. Configure hostname and domain name for key generation
2. Generate a key-pair
3. SSHv2 recommended for security reason
4. Optional

For troubleshooting, you may verify the status of SSH using the following commands:

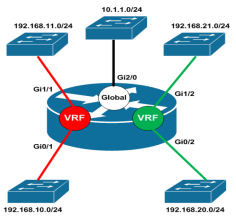
```
SW1# show ip ssh
SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3

SW1# show ssh
Connection Version Mode  Encryption    Hmac        State           Username
0          2.0     IN    aes128-cbc    hmac-sha1   Session started wendell
0          2.0     OUT   aes128-cbc    hmac-sha1   Session started wendell
%No SSHv1 server connections running.
```

Above command shows that user Wendell is currently connected to switch SW1 via SSHv2.

For routers without a management port, you may also configure Virtual Routing and Forwarding (VRF) on normal port to isolate it and connect it to the OOB management LAN.

A router configure with VRF will create multiple instances of routing tables, essentially dividing a physical router into multiple logical routers:

- A global routing table which exists by default
- Separate routing tables for different VRF instances, e.g. Red and Green VRF as shown
- Each interface will either belong to global or a VRF instance
- Traffic on different VRFs are separated because of different routing tables



For Cisco ISR 2900 routers, some of the commands for VRF are as follows:

Create a VRF instance, e.g. named Mgmt-vrf:

```
Router(config)# vrf definition Mgmt-vrf
```

Associate an interface to VRF, e.g. g0/0:

```
Router(config)# int g0/0
Router(config-if)# vrf forwarding Mgmt-vrf
```

Configure static route into specific VRF routing table:

```
Router(config)# ip route vrf Mgmt-vrf ...
```

Ping through a specific VRF:

```
Router# ping vrf Mgmt-vrf ...
```

Note: VRF commands vary between different devices. Use ? to find out.

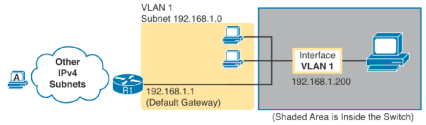
To verify the routing table of VRF, use the following command:

```
Router# show ip route vrf Red
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

C    192.168.10.0/24 is directly connected, GigabitEthernet0/1
C    192.168.11.0/24 is directly connected, GigabitEthernet1/1
```

For comparison:

```
Router# show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

C    10.1.1.0/24 is directly connected, GigabitEthernet2/0
```


For switches without a management port, you may also configure an isolated switch virtual interface (SVI) to connect it to the OOB management LAN.

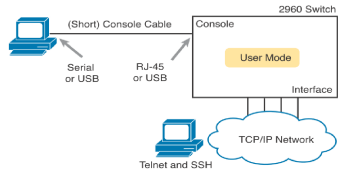


Configuring SVI using the command interface vlan *n*, the switch can now send and receive frames using the configured IP address.

```
switch(config)# interface vlan 1
switch(config-if)# ip address 192.168.1.200 255.255.255.0
switch(config-if)# no shutdown
switch(config-if)# exit
switch(config)# ip default-gateway 192.168.1.1
```

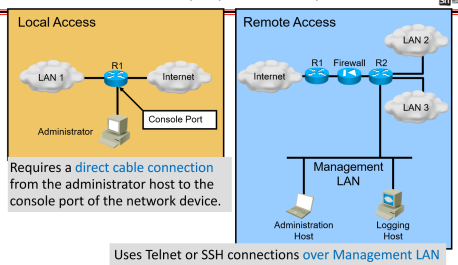

In this lab, we are going to discuss how to secure the management of network devices to prevent unauthorised access and configuration.

You have been using the console cable to access and configure network devices in the lab exercises.



In fact, accessing and configuring of network devices may also be done via TCP/IP network using Telnet or SSH.

Console access is the most secure but requires local access which may not be convenient. Thus, Telnet or SSH which allows remote access is commonly implemented in practice.

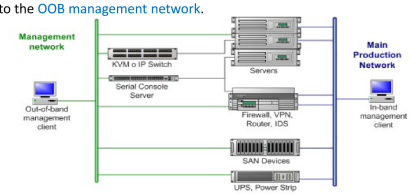


Requires a direct cable connection from the administrator host to the console port of the network device.

Uses Telnet or SSH connections over Management LAN from the administrator host to the network device.

To be secure in remote management, out-of-band (OOB) network is recommended which is a separate network isolated from the main production network carrying user data traffic.

Note that now every network device has an additional link to connect to the OOB management network.

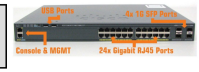


Although the devices seem to link the management network and main production data network together, traffic are unable to flow between them due to the use of management ports on the network devices.

Management port is a layer 3 port specially constructed for device management using Virtual Routing and Forwarding (VRF) to isolate it from the normal ports on the network devices.

Configuring management port on Catalyst 2960-X layer-2 switch:

```
S1(config)# int fastethernet0
S1(config-if)# ip address ...
S1(config-if)# no shut
S1(config)# ip default-gateway ...
```



Similarly for Catalyst 3650 layer-3 switch:


```
S1(config)# int gigabitethernet0/0
S1(config-if)# ip address ...
S1(config-if)# no shut
S1(config)# ip default-gateway ...
```

An example of RADIUS server is the free open source and most widely deployed freeRADIUS server which you will also try out in the lab exercise.

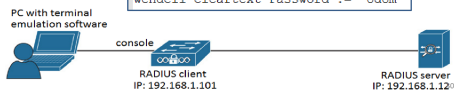
It's easier to install and run freeRADIUS in Kali Linux, follow the guide at https://wiki.freeradius.org/guide/Getting Started

In clients.conf specify IP addresses of supported network devices and shared password:

```
client networkdevices {
    ipaddr      = 192.168.1.0/24
    secret      = RADIUS-pa55w0rd
}
```

In file "users", configure username and password:

```
wendell Cleartext-Password := "odom"
```



Here is an explanation of the common methods for use with aaa authentication command to configure the method list:


```
Router(config)#aaa authentication type {default | list-name}
 method1[...[method4]]
```

Methods	Description
local	Use local username for authentication.
local-case	Use case-sensitive local username for authentication.
enable	Use enable password for authentication.
none	No authentication.
group radius	Use the configured RADIUS servers for authentication.
group tacacs+	Use the configured TACACS+ server for authentication.
group *group-name*	Use the configured RADIUS or TACACS+ servers for authentication as defined by the command **aaa group server radius** *group-name* **or aaa group server tacacs+** *group-name*.

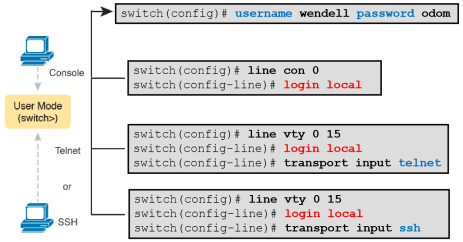
For accounting purpose (to be discussed later), it is also useful to log login attempts.

```
R1(config)#
login on-success log [every login]
R1(config)#
login on-failure log [every login]
```

```
Router#
*Nov  1 08:53:54.571: %SEC_LOGIN-4-LOGIN_FAILED: Login failed [user: wendell]
[Source: 192.168.1.10] [Localport: 23] [Reason: Login Authentication Failed -
BadPassword] at 08:53:54 UTC Thu Nov 1 2018
Router#
```

```
Router#
Router#show login failures
Total failed logins: 2
Detailed information about last 50 failures

Username      IPort  Count     TimeStamp              SourceIPAddr
wendell         23     2     08:53:54 UTC Thu Nov 1 2018 192.168.1.10
```

A better authentication method will be to configure for each user a locally stored (login local) unique username and password which is a requirement for SSH.



In summary, here is a table comparing TACACS+ vs RADIUS for implementing AAA.

	TACACS+	RADIUS
Functionality	Separates authentication, authorization and accounting, allowing modularity of the security server implementation	Combines authentication and authorization but separates accounting, allowing less flexibility in implementation than TACACS+.
Standard	Mostly Cisco supported	Open/RFC standard
Transport Protocol	TCP port 49	UDP port 1812 and 1813
Security	Entire packet encrypted	Password encrypted
Authorization	Provides authorization of commands on a per-user or per-group basis.	Has no option to authorize router commands on a per-user or per-group basis
Primary Use	Device administration	Network access

You'll study theory of authentication protocols in ICT2205 Applied Crypto.

Next, setup AAA server, e.g. TACACS+ server using the free tacacs.net which you'll try out in the lab exercise.

After downloading and installing TACACS+, go to config folder and modify the following configuration files:

In tacplus.xml, modify IP address to match your TACACS+ server:

```
<Server xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Port>49</Port>
  <LocalIP>127.0.0.1</LocalIP>
```

In clients.xml, specify IP addresses of network devices (TACACS+ clients), and the shared password with TACACS+ server:

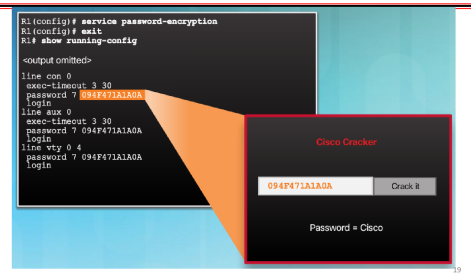
```
<ClientGroup Name="INTERNAL">
  <Secret ClearText="TACACS-pa55w0rd" DES="" </Secret>
  <Clients>
    <Client>10.0.0.0/8</Client>
    <Client>172.16.0.0/12</Client>
    <Client>192.168.0.0/16</Client>
  </Clients>
</ClientGroup>
```

Read the documentation at https://www.tacacs.net/documentation/

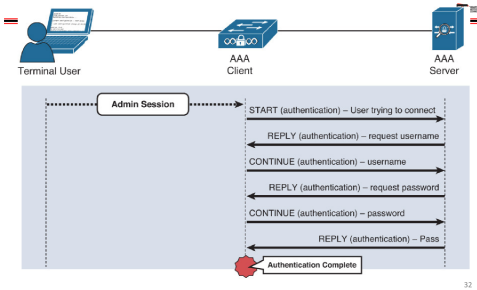
Lab Exercise 1B/1C

1.2 Configure device access authentication using centralized AAA servers to store usernames and passwords.

By default, Cisco stores passwords in the configuration file in clear, which can be encrypted using the service password-encryption command but the protection is very weak.



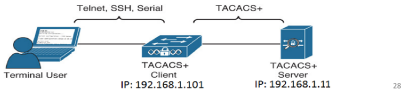
Specifically, authentication in TACACS+ is done separately from authorization and accounting as shown:



In addition, configure usernames and passwords centrally in AAA server, e.g. in tacacs.net as follows:

Configuring username and password in authentication.xml

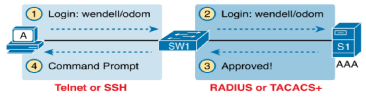
```
<UserGroup>
  <Name>Network Engineering</Name>
  <AuthenticationType>File</AuthenticationType>
  <Users>
    <User>
      <Name>wendell</Name>
      <LoginPassword ClearText="odom" DES=""> </LoginPassword>
      <EnablePassword ClearText="" DES=""></EnablePassword>
      <CHAPPassword ClearText="" DES=""> </CHAPPassword>
      <OutboundPassword ClearText="" DES=""> </OutboundPassword>
    </User>
  </Users>
</UserGroup>
```



A better and more practical way is to store passwords in an external centralised AAA server (server-based AAA) for authentication, especially when managing many devices.

Benefits of server-based AAA:

- Scalability and manageability: the configuration of usernames and passwords is centralized and not repeated in every device
- Fault tolerance: multiple AAA servers can be configured for backup purpose



Communication between network devices and AAA servers typically uses RADIUS or TACACS+ (Terminal Access Controller Access-Control System, pronounced as 'tack-axe' from Cisco textbooks) protocols.

To provide better protection, Cisco has now implemented cryptographic algorithms like md5, sha256 and scrypt which you will be studying the details in ICT2205 Applied Crypto.

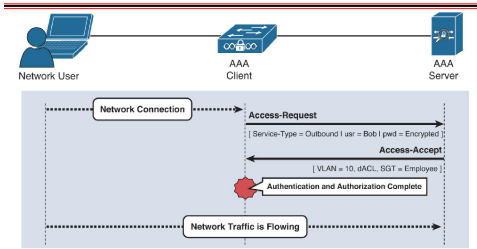
By replacing 'password' command with 'secret', MD5 hash will be used to protect user passwords in the configuration file by default.

```
Router(config)#
username name secret unencrpyted-password
Router(config)#
enable secret unencrpyted-password
```

For even better protection, sha256 and scrypt are recommended:

```
Router(config)#
username name algorithm-type {md5 | scrypt | sha256 } secret unencrpyted-password
Router(config)#
enable algorithm-type {md5 | scrypt | sha256 } secret unencrpyted-password
```

In contrast, authentication in RADIUS is done together with authorization in the same Access-Accept reply message as shown:



For redundancy, AAA can be configured with multiple methods, e.g. TACACS+ server, RADIUS server, and local database (to avoid being locked out if AAA servers are down).

```
R1(config)# aaa new-model
R1(config)# tacacs server TACACSERVER
R1(config-server-tacacs)# address ipv4 192.168.1.11
R1(config-server-tacacs)# key TACACS-pa55w0rd
R1(config)#
R1(config)# radius server RADIUSERVER
R1(config-radius-server)# address ipv4 192.168.1.12 auth-port
1812 acct-port 1813
R1(config-radius-server)# key RADIUS-pa55w0rd
R1(config)#
R1(config)# username ADMIN secret Str0ngPa55w0rd
R1(config)#
R1(config)# aaa authentication login default group tacacs+
group radius local
```

To enable authentication using centralised AAA server, configure using AAA commands, e.g. with TACACS+ as follows:

```
SW1(config)# aaa new-model                          1. Enable AAA

SW1(config)# tacacs server TACACSVR                 2. Specify AAA server
SW1(config-server-tacacs)# address ipv4 192.168.1.11
SW1(config-server-tacacs)# key TACACS-pa55w0rd
SW1(config-server-tacacs)# exit

! Default method list is automatically applied to console and
! vty(Telnet and SSH) if no custom method list is configured
SW1(config)# aaa authentication login default group tacacs+
```

3. Configure authentication method list (either default above or custom below)

```
! Or, configure custom method list for authentication
SW1(config)# aaa authentication login MY-LIST group tacacs+

SW1(config)# line vty 0 15
SW1(config-line)# login authentication MY-LIST
```

4. Unlike default method list, custom method list must be applied at interface to be effective

In addition, Cisco has added features to slow down brute-force attack on login as follows:

```
R1(config)#
login delay seconds

login block-for seconds attempts tries within seconds
```

To prevent DoS attack denying authorized administrators from login during this block-for period called quiet-mode, you may configure ACL to allow login before block-for timer expires as follows:

```
R1(config)#
login quiet-mode access-class {acl-name|acl-number}
```

```
R1(config)# ip access-list standard PERMIT-ADMIN
R1(config-std-nacl)# remark Permit only Administrative hosts
R1(config-std-nacl)# permit 192.168.10.10
R1(config-std-nacl)# permit 192.168.11.10
R1(config-std-nacl)# exit
R1(config)# login quiet-mode access-class PERMIT-ADMIN
```

Some examples of configuring CLI views:

```
R1(config)# parser view SHOWVIEW
R1(config-view)# secret cisco
R1(config-view)# commands exec include show version
R1(config-view)# exit
R1#
R1(config)# parser view VERIFYVIEW
R1(config-view)# secret cisco5
R1(config-view)# commands exec include ping
R1(config-view)# exit
R1#
R1(config)# parser view CONFIGVIEW
R1(config-view)# secret cisco10
R1(config-view)# commands exec include configure terminal
R1(config-view)# commands configure include interface
R1(config-view)# exit
```

13

After configuration, you may verify available commands in the CLI view as follows:

```
R1> enable view SHOWVIEW
Password:
R1#
R1# ?
Exec commands:
  enable      Turn on privileged commands
  exit        Exit from the EXEC
  show        Show running system information
R1#
R1# show ?
  parser      Display parser information
  version     System hardware and software status
```

By default, enable, exit and show parser commands are always available.

14

Optionally, superview can be configured, examples as follows:

Creating superviews USER, SUPPORT and ADMIN:

```
R1(config)# parser view USER superview
R1(config-view)# secret cisco
R1(config-view)# view SHOWVIEW
R1(config-view)# exit
R1#
R1(config)# parser view SUPPORT superview
R1(config-view)# secret cisco1
R1(config-view)# view SHOWVIEW
R1(config-view)# view VERIFYVIEW
R1(config-view)# exit
R1#
R1(config)# parser view ADMIN superview
R1(config-view)# secret cisco2
R1(config-view)# view SHOWVIEW
R1(config-view)# view VERIFYVIEW
R1(config-view)# view CONFIGVIEW
R1(config-view)# exit
```

15

You may also verify available commands in superviews as follows:

```
R1# enable view ADMIN
Password:
R1# ?
Exec commands:
  configure   Enter configuration mode
  enable      Turn on privileged commands
  exit        Exit from the EXEC
  ping        Send echo messages
  show        Show running system information
R1# config t
R1(config)# ?
Configure commands:
  do-exec     To run exec commands in config mode
  end         Exit from configure mode
  exit        Exit from configure mode
  interface   Select an interface to configure
```

16

Lab Exercise 1B

1.2 Configure device access authorization using role-based CLI views.

9

Privilege level has the problem that commands at lower level are always available for higher privilege users. Thus a more flexible RBAC using views to define roles is developed.

Views allow you to pick and choose which commands are available according to roles, and are divided into 3 types:

- Root view: for creating views and superviews, similar as privilege level 15

- CLI view: for defining a set of available commands

- Superview (optional): for grouping CLI views, but not commands

10

To begin defining roles using views, enable root view in aaa new-model and then configure new CLI view as follows:

```
R1(config)# aaa new-model
R1(config)# exit
R1# enable view
Password:
R1#
R1# configure terminal
R1(config)# parser view VIEWNAME

R1(config-view)# secret viewpassword
R1(config-view)# commands exec include ...
R1(config-view)# commands exec exclude ...
R1(config-view)# commands exec include all ...
R1(config-view)# ...
R1(config-view)# exit
R1(config)#
```

1. Enable AAA to create views
2. Go to root view
3. Enter the same enable secret password
4. Create new CLI view
5. Password for new view
6. Pick and choose commands for view

Note: Only a maximum of 15 views can be created.

11

The parameters for you to pick and choose commands for CLI views are as follows:

```
commands parser-mode {include | include-exclusive | exclude}
[all] command
```

Parameter	Description
parser-mode	Specifies the mode in which the specified command exists (e.g. exec, configure).
include	Adds a command or an interface to the view and allows the same command or interface to be added to an additional view.
include-exclusive	Adds a command or an interface to the view and excludes the same command or interface from being added to all other views.
exclude	Excludes a command or an interface from the view; that is, users cannot access a command or an interface.
all	(Optional) Specifies a "wildcard" that allows every command in a specified configuration mode that begins with the same keyword or every subinterface for a specified interface to be part of the view.
command	Specifies a command that is added to the view.

12

Except for the five level 0 commands, all CLI commands are initially assigned to level 1 or level 15, but can be reconfigured to different custom privilege levels between 2 – 14 as follows:

To reconfigure command(s) to new privilege level:

```
R1(config)# privilege exec level level command-string
```

e.g.

```
R1(config)# privilege exec level 5 ping
```

For security, a password can be configured for the new privilege level; e.g.

```
R1(config)# enable secret level 5 cisco5
```

To reset command(s) to default privilege level:

```
R1(config)# privilege exec reset command-string
```

5

To run a command, a user must be at the same or higher privilege level as the command as shown below:

```
R1> show privilege
Current privilege level is 1
R1# ping 10.10.10.1
          ^
% Invalid input detected at '^' marker.

R1> enable 5
Password: <enter cisco5>
R1#
R1# show privilege
Current privilege level is 5
R1#
R1# ping 10.10.10.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
R1#
```

6

To simplify administration, privilege levels can be assigned to users directly so that they will enter at the assigned privilege level upon successful login.

An example of assigning 4 different users – USER, SUPPORT, JR-ADMIN and ADMIN – with different privilege levels to access the network device:

```
R1(config)# username USER privilege 1 secret cisco
R1(config)#
R1(config)# privilege exec level 5 ping
R1(config)# enable secret level 5 cisco5
R1(config)# username SUPPORT privilege 5 secret cisco5
R1(config)#
R1(config)# privilege exec level 10 reload
R1(config)# enable secret level 10 cisco10
R1(config)# username JR-ADMIN privilege 10 secret cisco10
R1(config)#
R1(config)# username ADMIN privilege 15 secret cisco123
```

7

If using AAA, privilege levels can also be assigned directly to users using the AAA authorization commands as follows:

```
R1(config)# aaa new-model
R1(config)# ...
R1(config)# aaa authentication login default group tacacs+
group radius local
R1(config)# aaa authorization exec default group tacacs+
group radius local
```

Correspondingly, users in AAA servers must be configured with appropriate privilege levels, e.g.

- For tacacs.net, in authorization.xml

```
<AutoExec>
    <Set>priv-lvl=5</Set>
</AutoExec>
```

- For freeRADIUS, in file 'users':

```
wendell   Cleartext-Password := "odom"
          cisco-avpair = "shell:priv-lvl=5"
```

8

ICT2203
Network Security

Lab 7.2 and 7.3 Notes:

Device Access Control with AAA – Authentication, Authorization and Accounting

2021-2022 Trimester 3

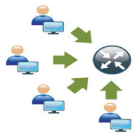
Lab Exercise 1A

1.1 Configure device access authorization using privilege level.

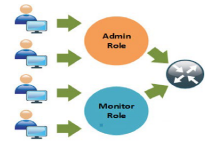
2

After successful authentication, users should not be given more access rights than necessary which brings us to the topic of authorization.

Theoretically, there are different access control models for implementing authorization.



Assigning rights directly to each user; e.g. privilege level



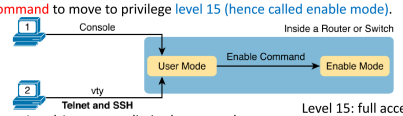
Assigning rights to roles which is then assigned to each user; e.g. CLI views

3

By default, authorization in Cisco devices is implemented using privilege levels in 3 pre-defined levels:

Privilege Level	Result
1	User level only (prompt is router>), the default level for login
15	Privileged level (prompt is router#), the level after going into enable mode
0	Seldom used, but includes five commands: disable, enable, exit, help, and logout

Upon login, users are placed at level 1 (user mode) , and can use enable command to move to privilege level 15 (hence called enable mode).



Level 1: access to limited commands; e.g. read-only commands like show

Level 15: full access to all commands

4

And here is a sample of exec log in tacacs.net server which can be found in c:\Users\All Users\TACACS.net\Logs\Accounting_...

A 'start' message is logged when the exec shell started:

```
<102> 2017-11-09 03:57:41 [192.168.1.200:43972]
11/09/2017 03:57:41 NAS_IP=192.168.1.200 Port=tty1
rem_addr=192.168.1.3 User=wendell Flags=Start task_id=25
timezone=UTC service=shell
```

A 'stop' message is logged when the exec shell ended:

```
<102> 2017-11-09 03:58:04 [192.168.1.200:32751]
11/09/2017 03:58:04 NAS_IP=192.168.1.200 Port=tty1
rem_addr=192.168.1.3 User=wendell Flags=Stop task_id=25
timezone=UTC service=shell disc-cause=1 disc-cause-ext=9
pre-session-time=5 elapsed_time=22 stop_time=731022330
```

25

Finally, AAA accounting is to have a record of what a user has done which is also crucial to security.

One useful type of accounting is 'exec' which provides information about user EXEC terminal session after login.

```
R1(config)# aaa new-model
R1(config)# ...
R1(config)# aaa authentication login default group tacacs+
group radius local
R1(config)# aaa authorization exec default group tacacs+
group radius local
R1(config)# aaa accounting exec default start-stop group
tacacs+ group radius
```

- start-stop: a start message is sent when session begins and a stop message is sent when session ends

21

In addition, you may use show parser view command to verify created views as follows:

```
R1# enable view
Password:
R1#
R1# show parser view
Current view is 'root'
R1#
R1# show parser view all
Views/SuperViews Present in System:
 SHOWVIEW
 VERIFYVIEW
 REBOOTVIEW
 USER *
 SUPPORT *
 ADMIN *
-------(*) represent superview-------
R1#
```

17

In addition, TACACS+ (but not RADIUS) supports another useful type of accounting on commands which provide information about specific privilege level commands that a user issues.

```
R1(config)# aaa accounting exec default start-stop group
tacacs+ group radius
R1(config)# aaa accounting commands 15 default stop-only
group tacacs+
```

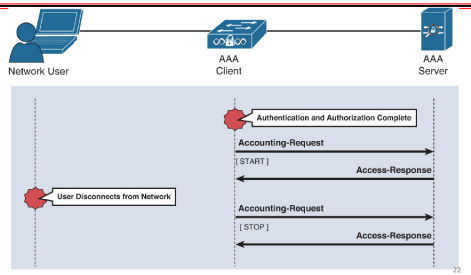
- stop-only: a stop message is sent when command is executed

A sample of commands log in tacacs.net:

```
<102> 2017-11-09 03:57:47 [192.168.1.200:44251]
11/09/2017 03:57:47 NAS_IP=192.168.1.200 Port=tty1
rem_addr=192.168.1.3 User=wendell Flags=Stop task_id=25
timezone=UTC service=shell priv-lvl=15 cmd=show
running-config <cr>
```

26

For RADIUS, accounting is done separately after authentication and authorization are completed as shown:



22

Similar as privilege levels, views can be assigned to users directly so that they will enter at the assigned views upon successful login.

Locally in network device:

```
R1(config)# username Wendell view ADMIN secret odom
```

or centrally using AAA servers:

```
R1(config)# aaa authentication login default group tacacs+
group radius local
R1(config)# aaa authorization exec default group tacacs+
group radius local
```

in tacacs.net:

```
<AutoExec>
    <Set>cli-view-name=ADMIN</Set>
</AutoExec>
```

in freeRADIUS 'users':

```
Wendell   Cleartext-Password := "odom"
          Service-Type = NAS-Prompt-User,
          cisco-avpair = "shell:cli-view-name=ADMIN"
```

18

Here's a sample of exec log in freeRADIUS server which can be found at /var/log/freeradius/...

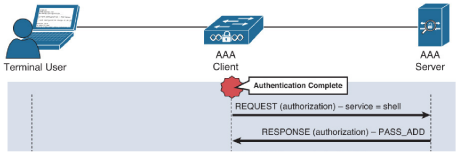
```
Thu Nov  9 13:00:56 2017
    Acct-Session-Id = "0000002F"
    User-Name = "wendell"
    Acct-Authentic = RADIUS
    Acct-Status-Type = Start
    NAS-Port = 1
    NAS-Port-Id = "tty1"
    NAS-Port-Type = Virt
    Service-Type = NAS-P
    NAS-IP-Address = 192
    Acct-Delay-Time = 0
    Event-Timestamp = "N
    Tmp-String-9 = "ai:"
    Acct-Unique-Session-
    Timestamp = 15102324
```

```
Thu Nov  9 13:07:28 2017
    Acct-Session-Id = "0000002F"
    User-Name = "wendell"
    Acct-Authentic = RADIUS
    Acct-Terminate-Cause = User-Request
    Acct-Session-Time = 392
    Acct-Status-Type = Stop
    NAS-Port = 1
    NAS-Port-Id = "tty1"
    NAS-Port-Type = Virtual
    Service-Type = NAS-Prompt-User
    NAS-IP-Address = 192.168.1.200
    Acct-Delay-Time = 0
    Event-Timestamp = "Nov  9 2017 13:07:28 UTC"
    Tmp-String-9 = "ai:"
    Acct-Unique-Session-Id = "f1e1339928a412881fcac9c442105e01"
    Timestamp = 1510232848
```

23

For TACACS+, authorization is done separately from authentication, whereas for RADIUS, authorization is done together with authentication.

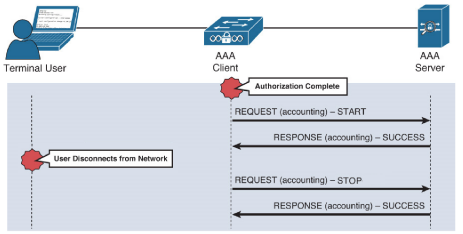
Specifically, TACACS+ authorization consists of a request and a response message to assign privilege level or view as shown:



In contrast, RADIUS authorization is done together with authentication in the same access-request and access-accept message in pg 33.

19

Similarly for TACACS+, accounting is done separately after authorization as shown:



24

Lab Exercise 2

2.1 Configure device access accounting using centralized AAA servers for storing logs.

20

Based on the design of your network, configure network devices as NTP server, client, peer or broadcast:

- Server: provides time information to clients, only used if you are not able to reach reliable external NTP servers

```
R1(config)# ntp master [stratum]
```

- Client: synchronizes time with an NTP server

```
R1(config)# ntp server {ip | hostname} [version] [prefer]
```

- Peer: also called symmetric mode, which work both as server and client to synchronize, or be synchronized by each other

```
R1(config)# ntp peer {ip | hostname} [version] [prefer]
```

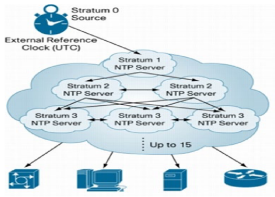
- Broadcast: pushes time announcements from NTP server to clients, used only when time accuracy is not a concern

Server to broadcast:
```
R1(config-if)# ntp broadcast [version]
```

Client to receive:
```
R1(config-if)# ntp broadcast client
```


To be scalable, NTP servers are typically setup in an hierarchical structure to support time synchronisation in practice.

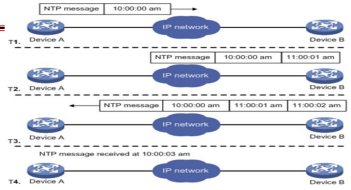
NTP uses the concept of stratum to describe how many NTP hops away a machine is from the most reliable authoritative time source at stratum 0.



As the stratum number increases, its accuracy degrades.

UTC (Coordinated Universal Time), aka GMT (Greenwich Mean Time), is the current worldwide standard for time and date.

In client/server mode, NTP client synchronises its clock by performing the following computation based on NTP reply:



Offset of client clock from server clock: $\theta = \frac{(T2-T1)+(T3-T4)}{2}$ e.g. $\theta = \frac{1h1s+59m59s}{2} = 1h$

Round-trip delay: $\delta = (T4-T1)-(T3-T2)$ e.g. $\delta = 3s-1s = 2s$

(Clock filter and mitigation algorithms will then be used to determine the best and final offset to synchronize the system clock.)

ICT2203
Network Security

SIT SINGAPORE INSTITUTE OF TECHNOLOGY

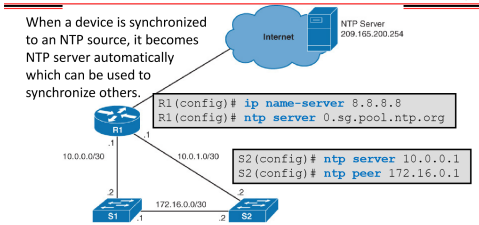
Lab 8 Notes:

Network Security Monitoring with NTP, Syslog and NetFlow

2021-2022 Trimester 3

An example of configuring NTP, with edge router R1 synchronizing with an external NTP time source, which in turn acts as NTP server for internal switches.

When a device is synchronized to an NTP source, it becomes NTP server automatically which can be used to synchronize others.

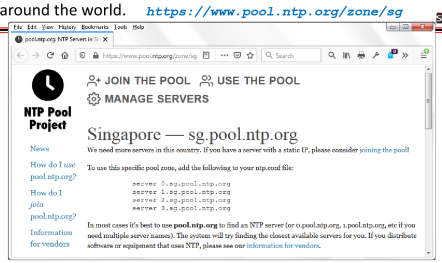



```
R1(config)# ip name-server 8.8.8.8
R1(config)# ntp server 0.sg.pool.ntp.org
```

```
S2(config)# ntp server 10.0.0.1
S2(config)# ntp peer 172.16.0.1
```

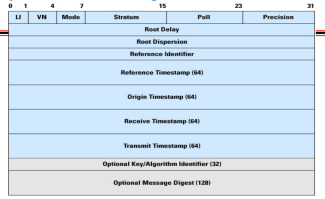
```
S1(config)# ntp server 10.0.0.1
S1(config)# ntp peer 172.16.0.2
```


An example is the NTP pool project which is maintaining a network of NTP servers that are being used by many systems around the world. `https://www.pool.ntp.org/zone/sg`



To use the pool, simply configure your device to point to any of the provided domain name, each representing a random set of NTP servers.

To support NTP computation, NTP client and server register its timestamps in the NTP message format as follows:



	NTP request	NTP reply
Origin timestamp:	ignore	T1
Receive timestamp:	ignore	T2
Transmit timestamp:	T1	T3

Reference timestamp: time when system clock was last updated

Finally, with the various network security implemented, we are now ready to perform network security monitoring by logging all alerts to centralised servers for analysis and reporting.

Logging is commonly implemented using Syslog, SNMP and/or NetFlow protocols to centralised servers or SIEM (Security Information and Event Management) systems.

To facilitate analysis of log data, it is essential to synchronize time between all network devices, which can be achieved by using NTP (Network Time Protocol).

To verify that the device's clock has indeed been synchronized with NTP, use the following show commands:

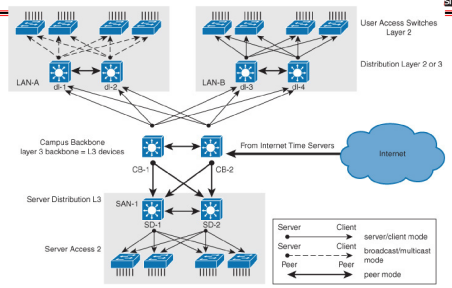
To verify which NTP server your device is synchronizing with, e.g.:

```
R1# show ntp associations

  address         ref clock       st  when   poll reach  delay  offset  disp
*~209.165.200.254 .LOCL.          1     24     64    17  1.000  -0.500  2.820
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

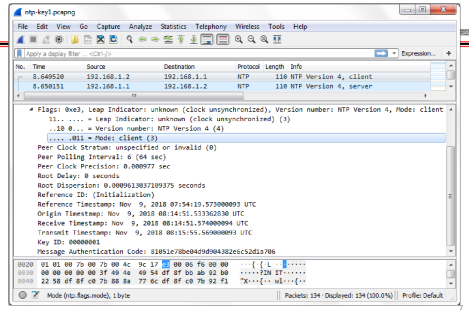
By synchronizing with stratum 1 time source, your device automatically becomes stratum 2 time source as shown, e.g.:

```
R1# show ntp status
Clock is synchronized, stratum 2, reference is 209.165.200.254
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**10
ntp uptime is 1500 (1/100 of seconds), resolution is 4000
reference time is D67E670B.0B020C68 (05:22:19.043 PST Mon Jan 13 2014)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 630.22 msec, peer dispersion is 189.47 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 64, last update was 5 sec ago.
```


After obtaining time from an NTP pool, an enterprise may implement time synchronization within its network in an hierarchical manner:



An example of NTP request sent by NTP client:



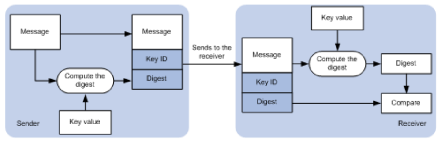
Lab Exercise 1A

SIT SINGAPORE INSTITUTE OF TECHNOLOGY

1.1 Configure NTP for synchronizing time between network devices to facilitate analysis of log data.

Unfortunately, an attacker can inject false NTP replies to confuse time synchronization and log timings. Thus, it is advisable to enable NTP authentication to protect NTP clients.

Both NTP client and server can be configured with the same pre-shared key/password which is used to compute MD5 hash for authentication:



Using the pre-shared key, NTP server generates an MD5 hash and send it together with NTP message.

Using the same pre-shared key, NTP client re-computes the MD5 hash and verifies it with the received MD5 hash before accepting the NTP message.

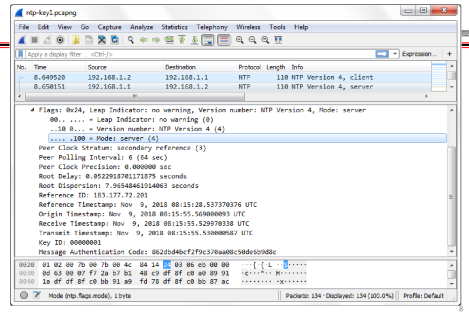
To be specific, we'll now discuss how to configure the network devices to synchronize time using NTP.

Before configuring NTP, you may wish to configure your device clock to current date/time:

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# clock timezone SGT 8
R1(config)# ^Z
R1#
R1# clock set 20:52:00 25 October 2017
*Oct 25 12:52:00.000: %SYS-6-CLOCKUPDATE: System clock has been updated from 20:36:38
  SGT Wed Oct 25 2015 to 20:52:00 SGT Wed Oct 25 2015, configured from console by
  console.
R1# show clock
20:52:55.051 SGT Wed Oct 25 2017
```

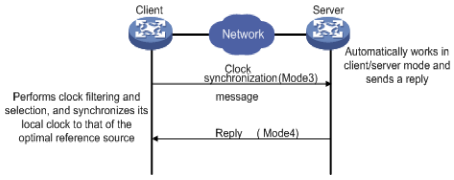
Any letters representing your timezone, e.g. SGT for Singapore Time, which is UTC + 8 hrs.

An example of NTP reply sent by NTP server:



As defined in RFC 5905, there are 3 NTP operation modes – client/server, peer and broadcast. We'll discuss the common client/server mode to understand how NTP works.

NTP operates over UDP port 123. In client/server mode, the NTP client initiates a request to the NTP server, and receives a reply as shown:



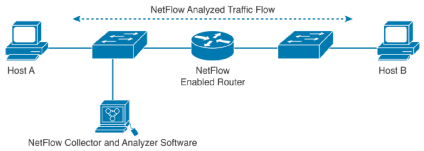
Lab Exercise 2

2.1 Configure NetFlow to provide statistical records of different IP packets that flow through a router.

29

To support network traffic monitoring, Cisco has developed NetFlow (RFC 3954) to collect statistics on IP packets flowing through network devices.

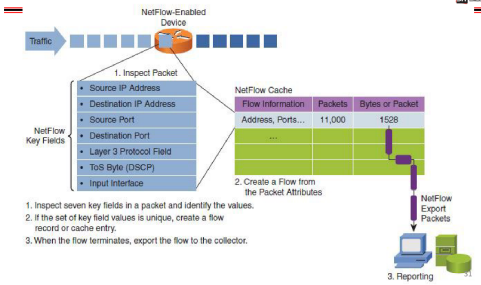
When enabled, NetFlow keeps statistics of different IP packets flowing through a router in a NetFlow cache, which can then be exported to a NetFlow Collector for centralized logging and analysis.



NetFlow has proven valuable and today different variants have been implemented, e.g. IPFIX (RFC 7011, derived from NetFlow), sFlow (RFC 3176) and JFlow (Juniper).

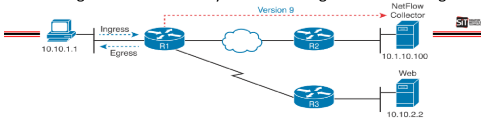
30

A flow is defined as a unidirectional sequence of packets that pass through a network device having the same values for certain key fields in the packet headers, examples:



31

The simplest way to enable NetFlow is to rely on default matching rules and thus only need to configure the followings:




```
R1(config)# interface fastethernet0/0
R1(config-if)# ip flow ingress      <-- Monitor incoming packets
R1(config-if)# ip flow egress       <-- Monitor outgoing packets
R1(config-if)# exit
R1(config)# ip flow-export destination 10.1.10.100 99   <-- Port number
R1(config)# ip flow-export version 9   <-- current version
R1(config)# ip flow-export source loopback 0
R1(config)# end
```

For easier identification of router, use loopback 0 IP address to send NetFlow packets to Collector.

NetFlow is transported over UDP (port not standardized in RFC).

32

To prevent cluttering the log with too much data, Syslog has a severity level rating which you can use it to control which messages to log.

Keyword	Numeral	Description	
Emergency	0	System unusable	Severe
Alert	1	Immediate action required	
Critical	2	Critical Event (Highest of 3)	Impactful
Error	3	Error Event (Middle of 3)	
Warning	4	Warning Event (Lowest of 3)	
Notification	5	Normal, More Important	Normal
Informational	6	Normal, Less Important	
Debug	7	Requested by User Debug	Debug

Service	To Set Message Levels
Console	logging console *level-name | level-number*
Monitor	logging monitor *level-name | level-number*
Buffered	logging buffered *level-name | level-number*
Syslog	logging trap *level-name | level-number*

25

An example of Syslog message which shows the date/time of occurrence, the severity level and the description.

```
*Dec 18 17:10:15.079: %LINEPROTO-5-UPDOWN: Line protocol on Interface
  FastEthernet0/0, changed state to down
```

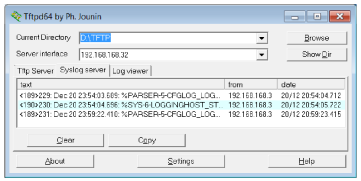
severity level

- **A timestamp:** *Dec 18 17:10:15.079
- **The facility on the router that generated the message:** %LINEPROTO
- **The severity level:** 5
- **A mnemonic for the message:** UPDOWN
- **The description of the message:** Line protocol on Interface FastEthernet0/0, changed state to down

26

There are many syslog servers available. An example is the free open source tftpd64 server which can also function as syslog server that you can try in the lab.

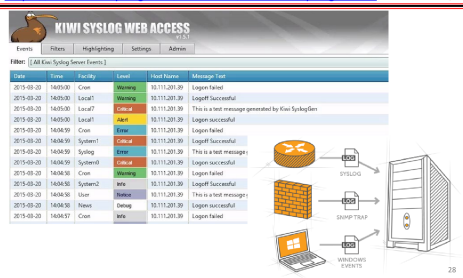
https://pjo2.github.io/tftpd64/



27

Alternatively, you may also try other Syslog servers like Kiwi Syslog which offers a free edition supporting up to 5 clients.

https://www.kiwisyslog.com/free-tools/kiwi-free-syslog-server



28

Lab Exercise 1B

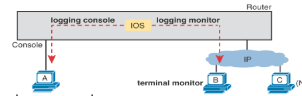
1.2 Configure Syslog which is one of the most commonly implemented method for logging.

22

Syslog (System message logging) (RFC 5424) is a standard protocol for event notification messages which is widely adopted by the industry.

While configuring Cisco devices in the labs, you must have seen or even be annoyed by the syslog messages that appeared, e.g.

```
*Dec 18 17:10:15.079: %LINEPROTO-5-UPDOWN: Line protocol on Interface
  FastEthernet0/0, changed state to down
```



Disable syslog on console:

```
R1(config)# no logging console
```

Enable syslog over Telnet/SSH:

```
R1(config)# logging monitor
```

Prevent syslog from interrupting command entry:

```
R1(config)# logging synchronous
```

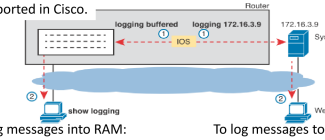
In addition, at B:

```
R1# terminal monitor
```

23

For monitoring of network security, it is more useful to store log messages in the device RAM, or even better to a centralised Syslog server.

By default, Syslog messages are transported over UDP port 514 (RFC 5426) or more securely over TLS over TCP port 6514 (RFC 5425) – may not be supported in Cisco.



To log messages into RAM:

```
R1(config)# logging buffered
```

In addition:

```
R1# show logging
```

To log messages to Syslog server:

```
R1(config)# logging
172.16.3.9
R1(config)# logging trap 4
```

All errors 4 or below (more serious) are logged.

24

For configuring of NTP authentication, the additional commands required are as follows:

- Define one or more NTP key(s), each number specifies a unique NTP key (password):

```
R1(config)# ntp authentication-key key-number md5 string
```

- Specify which key to enable for authentication:

```
R1(config)# ntp trusted-key key-number
```

- Enable NTP authentication:

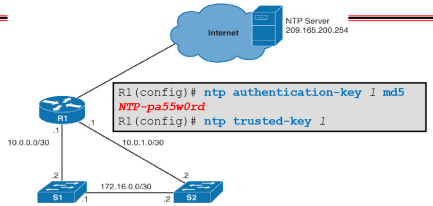
```
R1(config)# ntp authenticate
```

- For NTP client to specify which key to use for authentication:

```
R1(config)# ntp server {ip | hostname} key key-number
```

17

An example of configuring NTP authentication to prevent NTP clients from synchronising with attacker's time source.

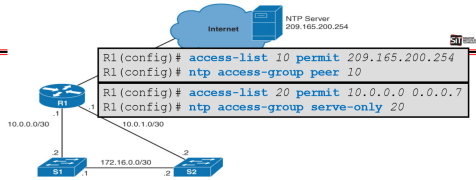



```
R1(config)# ntp authentication-key 1 md5 NTP-pa55w0rd
R1(config)# ntp trusted-key 1
```

```
S1(config)# ntp authentication-key 1 md5 NTP-pa55w0rd
S1(config)# ntp authenticate
S1(config)# ntp trusted-key 1
S1(config)# ntp server 10.0.0.1 key 1
```

18

In addition, NTP can be protected by ACLs.




```
R1(config)# access-list 10 permit 209.165.200.254
R1(config)# ntp access-group peer 10
R1(config)# access-list 20 permit 10.0.0.0 0.0.0.7
R1(config)# ntp access-group serve-only 20
```

Command for configuring ACL:

```
ntp access-group {query-only | serve-only | serve | peer}
access-list-number
```

- peer: accept NTP reply and also respond to NTP request
- serve-only: for server to respond to NTP request only
- serve: similar as 'serve-only', but also accept NTP control queries
- query-only: only accept NTP control queries such as SNMP

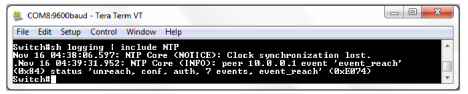
19

Since NTP is important, it is also recommended to enable NTP logging to log significant NTP events.

Enabling NTP logging (logging centrally will be discussed shortly):

```
Switch(config)# ntp logging
```

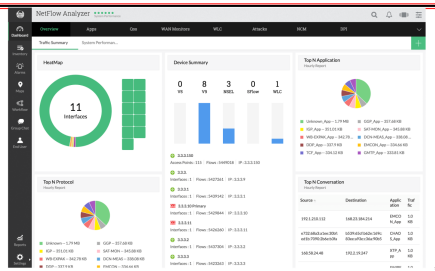
Examples of NTP logs:



20

An example of NetFlow collector is the NetFlow Analyzer by ManageEngine which you can try in your team project.

https://www.manageengine.com/products/netflow/download-free.html



37

To verify NetFlow configuration, use the following commands:

```
R1# show ip flow interface
FastEthernet0/0
  ip flow ingress
  ip flow egress

R1# show ip flow export
Flow export v9 is enabled for main cache
  Export source and destination details :
  VRF ID : Default
   Source(1)       1.1.1.1 (Loopback0)
   Destination(1)  10.1.10.100 (99)
  Version 9 flow records
  0 flows exported in 0 udp datagrams
  0 flows failed due to lack of export packet
  0 export packets were sent up to process level
  0 export packets were dropped due to no fib
  0 export packets were dropped due to adjacency issues
  0 export packets were dropped due to fragmentation failures
  0 export packets were dropped due to encapsulation fixup failures
R1#
```

33

Today, instead of separate logs, SIEM (Security Information and Event Management) solution is getting common to provide unified real-time correlation of all logs and threat alerting.

Figure 1: Magic Quadrant for Security Information and Event Management

38

To view statistics stored locally in the NetFlow cache, use the command:

```
R1# show ip cache flow
IP packet size distribution (5727 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .147 .018 .700 .000 .001 .001 .001 .001 .011 .009 .001 .002 .000 .001

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .001 .001 .097 .000 .000 .000 .000 .000 .000 .000 .000

  :
  :
Protocol         Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
--------         Flows     /Sec     /Flow  /Pkt     /Sec     /Flow     /Flow
TCP-Telnet           4     0.0        27    43      0.2       5.0      15.7
TCP-WWW            104     0.2        14   275      3.4       2.1       1.5
ICMP                 4     0.0      1000   100      8.8      27.9      15.4

SrcIf         SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP  Pkts
  :
  :
```

34

Instead of relying on default matching rules, NetFlow version 9 allows the flexibility to configure matching rules and collection statistics, hence also called flexible NetFlow.

(1) Example of configuring matching rules and collection statistics using flow record:

```
R1(config)# flow record my-record
R1(config-flow-record)# match ipv4 source address
R1(config-flow-record)# match ipv4 destination address
R1(config-flow-record)# match transport source-port
R1(config-flow-record)# match transport destination-port
R1(config-flow-record)# match ipv4 protocol
R1(config-flow-record)# match ipv4 tos
R1(config-flow-record)# match interface input
R1(config-flow-record)# collect counter bytes
R1(config-flow-record)# collect counter packets
```

35

Next, apply the specific matching rules and collection statistics configured onto the interface of the router as follows:

(2) Configuring flow exporter:

```
R1(config)# flow exporter my-exporter
R1(config-flow-exporter)# destination 10.1.10.100
R1(config-flow-exporter)# transport udp 99
R1(config-flow-exporter)# source lo0
R1(config-flow-exporter)# export-protocol netflow-v9
```

(3) Configuring flow monitor:

```
R1(config)# flow monitor my-monitor
R1(config-flow-monitor)# exporter my-exporter
R1(config-flow-monitor)# record my-record
```

(4) Applying flow monitor onto router interface:

```
R1(config)# interface g0/0
R1(config-if)# ip flow monitor my-monitor {input|output}
```

36