Chapter 7
Network Management
NET3011 – 17W

Network Management Topics

- This section covers some tools for Network-Management, specifically:
  - NTP = Network Time Protocol (RFC1305, 5905)
  - AAA = Authentication, Authorization, Accounting
  - RADIUS = Remote Authentication Dial-In User Service
  - TACACS+ = Terminal Access Controller Access-Control System
  - 802.1X: port-based Network Access Control (NAC)
- Other than NTP, all relate to securing the network … except that hackers have invented ways to DDoS machines using NTP servers, so still need security!
- NET3006 coverage of SNMP exceeds what is relevant to this course, so our coverage of SNMP will be very limited

NTP – Why Bother?

- Key question #1: What part of a networking device *doesn't* run according to a timer? (*Brainstorming*!)
- Key question #2: What part of the internet *doesn't* depend on time in one way or another?
- Reading even a few articles helps illustrate how critical *accurate* timing is to networking:

http://www.informationweek.com/it-life/ntps-fate-hinges-on-father-time/d/d-id/1319432
The Network Time Protocol is important enough that the likes of Google and Apple speak up if they find a bug in the protocol that needs fixing, or a modification they think is needed.

http://www.nist.gov/pml/div688/timing-031915.cfm
Our fast-approaching future of driverless cars and "smart" electrical grids will depend on billions of linked devices making decisions and communicating with split-second precision to prevent highway collisions and power outages

http://www.networkworld.com/article/2996260/security/researchers-warn-computer-clocks-can-be-easily-scrambled.html
http://arstechnica.com/security/2015/10/new-attacks-on-network-time-protocol-can-defeat-https-and-create-chaos/
https://threatpost.com/400-gbps-ntp-amplification-attack-alarmingly-simple/104256/

NTP – Basics

- NTP can use a simple client-server relationship (not unlike DNS) where the query-response sequence results in time synchronization to ~1 msec for LANs or ~10 msec for WANs *(despite variable network speeds and latencies!)*
- Accurate time is becoming increasingly critical:
  - "networks are quickly evolving … to a model where performance and reliability need to be quantified and in many cases, guaranteed with service level agreements (SLAs)... The foundation of many metric methodologies is the measurement of time."
  - "The need for accurate time is increasing year by year. Coordinating events, marking logs, and kicking-off scripts all run based on a system clock. The accuracy of the system is becoming more important as networks become faster and faster."

NTP – Organization (1)

- NTP servers are organized into stratum levels, numbered from 1-15 (remind you of RIP?)
  - stratum level 0 is the atomic clock or GPS source
  - stratum level 1 is the directly-attached server(s)
  - stratum 2 servers get their time from level 1 svrs
  - stratum 15 servers are the bottom-most servers
  - stratum 16 = unsynchronized to any external source
- With the stratum concept, NTP servers are also clients!
- For redundancy, devices using NTP are often configured with multiple NTP servers
- NTP recipients automatically choose the source with the lowest stratum level as their time source. This builds a self-organized tree of NTP speakers.
  (No elections and no preempt are necessary!)

NTP – Organization (2)

- Every NTP client-server exchange is processed using advanced statistics to produce accurate time synchronization, even with all the variations due to:
  - network latency (non-deterministic, packet-switched)
  - dispersion (jitter) between the clocks
  - clock offsets (the net difference)
- Some of the sensible rules for NTP are:
  - a NTP client will *not* synchronize to any machine that isn't already synchronized
  - a NTP client compares time from several servers and won't synchronize to a server that differs significantly, even if it's stratum is lower (better)

NTP – Organization (3)

- Small networks can use a flat structure:
  - every device can be configured to peer with every other (devices are both client and server)
  - pro: extremely resilient and stable
  - con: not scalable, high administrative overhead
- Large networks should use hierarchical structure:
  - several highest level servers with an absolute or external reference source (& use peer mode)
  - lower levels synch to level above, and have successively larger numbers of clients
- Large networks may also use the broadcast / multicast mode (IPv4/IPv6) for access LANs

NTP – Characteristics: Modes, Timers

- There are four different NTP modes: client, server, broadcast/multicast, and peer
- Most devices support client mode, and often server Network devices typically support all 4 modes
- Both client and server use UDP, port 123
- The client determines the polling interval and it may or may not be configurable
- The polling interval varies according to performance
- Cisco devices:
  – poll between 64 – 1024 secs, *not* configurable
  – may take up to 5 mins for initial synchronization!

NTP – Characteristics: Versions

- NTPv3 (RFC1305, 1992) pre-dates the W W W
- NTPv4 (RFC5905) is the most current
- Versions are fully backwards compatible
- NTPv4 added support for IPv6 and security enhancements
- **S**NTP is a **S**imple client-only version
  - accuracy is only assured to within 100 msec
  - performs none of the complex filtering and statistical analysis of full NTP
  - time source is selected ("elected") by:
    1. best (lowest) stratum
    2. in a tie: configured servers rather than broadcast
    3. in a tie: server who's NTP packet received first

NTP – Characteristics: Security

- NTP is a *very* open protocol with servers widely available across the internet; it has been subject to numerous abuses for hacking/DoS purposes
- Within NTP, servers do no authenticity verification; it is solely the client's responsibility to verify the server's MD5 hash against configured password
- ACLs are common and effective for securing NTP
  - should configure a NTP server's source address as a loopback for consistency in ACLs
  - consider limiting NTP requests to intranet only
  - user VLANs should *only* be allowed to send NTP requests and *never* NTP responses

NTP – Actual Servers

- There are many sources for NTP
  - time.nist.gov
  - 0.pool.ntp.org – 3.pool.ntp.org
  - time.nrc.ca
  - 0.ca.pool.ntp.org – 3.ca.pool.ntp.org
  - time.windows.com

  … and also realize that there are hundreds or thousands of servers behind these DNS entries:
  http://www.pool.ntp.org/en/
  http://www.pool.ntp.org/en/use.html
- Since servers & clients can be anywhere in the world, NTP time is *always* UTC (or GMT)

NTP – Configuration and Verification

- Client & server config *may* seem backwards:

```
L3(config)# ntp master {stratum}   ! SERVER
L3(config)# ntp server {ip-addr}   ! CLIENT

L3(config)# access-list {#} permit udp host {ip} eq ntp
L3(config)# ntp authentication-key {#} md5 {pswd}
L3(config)# ntp trusted-key {#}
L3(config)# ntp source {i/f}
```

```
L3# clock set 15:59:00 16 Mar 2016
L3#(config) clock timezone EST -5
L3#(config) clock summer-time EDT recurring
```

```
L3# show ntp status
L3# show ntp associations
L3# show clock detail
```

AAA – Basics

- AAA, or Triple-A, is:
  Authentication: proving who you are
  Authorization: determining what you can do
  Accounting: tracking & logging what you do
- AAA is an alternative to local configuration, which you have been using since day 1. If just a single set of credentials is sufficient:
  – line passwords: **A**uthentication to connect
  – enable secret: **A**uthorization to make configuration changes
  – local logging: **A**ccounting (sort of) where only system responses (if any) to changes are recorded

  … OR for multiple credentials, where Authentication and Authorization may be inter-mixed:

  – local accounts, possibly with an associated privilege level

```
A(config)# user {usr} [privilege {#}] secret {pwd}
A(config-line)# login local
```

AAA – Motivation

- AAA is a solution to many limitations of local config
  - flexibility / customization: AAA methods can be customized and applied on a per interface or per command basis
  - scalability: all parameters collected into a single database, rather than distributed across devices
  - standardization: using RADIUS allows AAA to be a plug-and-play solution from any vendor
  - redundancy: AAA servers can be configured in groups, allowing validation by alternate servers
- AAA does not (should not!) replace a last-chance locally configured admin account on all devices

AAA – Authentication (1)

- Authentication defines *who* is allowed to connect to/via a device and also *how* (to which service) they may connect:
  - login service: direct console or vty login for CLI access
  - enable service: access privileged exec (at level 15)
  - 802.1X service
  - ppp service

A partial list of services

- Authentication lists have up to four methods used to verify credentials for a connection request. Choose from:
  - group: use a (named) RADIUS/TACACS+ group
  - krb5: use Kerberos ver 5 authentication

Remote

  - local: use local account definitions from **`user`** …
  - line: use **`password`** … from **`line xxx`** config
  - enable: use the **`enable secret`** … password
  - ~~none: don't require any authentication~~ BAD!!

Local

http://www.cisco.com/c/en/us/td/docs/ios/12_2/security/configuration/guide/fsecur_c/scfathen.html

AAA – Authentication (2)

- Authentication lists are *named* to allow the option of using any one of multiple definitions
  - a reserved name of *default* applies when AAA is enabled and a service has no (other) named list
- Authentication proceeds only far enough through the list of methods to get accepted or denied:
  - failure to connect to specified method (e.g. a group of RADIUS servers) is *not* the same as an authentication denial
  - having *no* defined local accounts is *not* the same as an authentication denial
  - using a non-existent account, incorrect password, or other invalid credentials results in a denial

AAA – Remote Servers (1)

- There are two prominent standards for centralized, and thus remote, AAA servers:
  - RADIUS: open standard, RFC2865 + RFC2866
  - TACACS+: proprietary; an info-only RFC is available at: http://tools.ietf.org/html/draft-grant-tacacs-02
- Other protocols (NTLM, LDAP, Active Directory)
  - don't support all functions (e.g. no accounting)
  - aren't supported by Cisco routers & switches
  - may be supported by Cisco security appliances
- RADIUS is often preferred because it is required by 802.1X (… think WPA2-Enterpise; what campus network doesn't use it?)

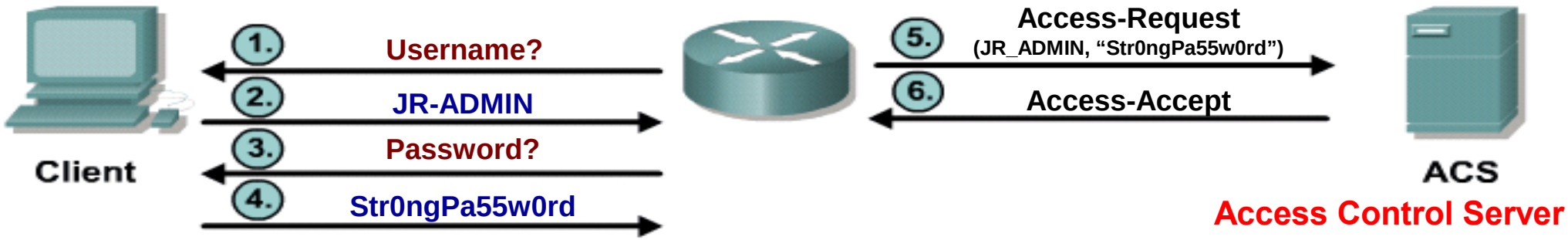
AAA – Remote Servers (2)

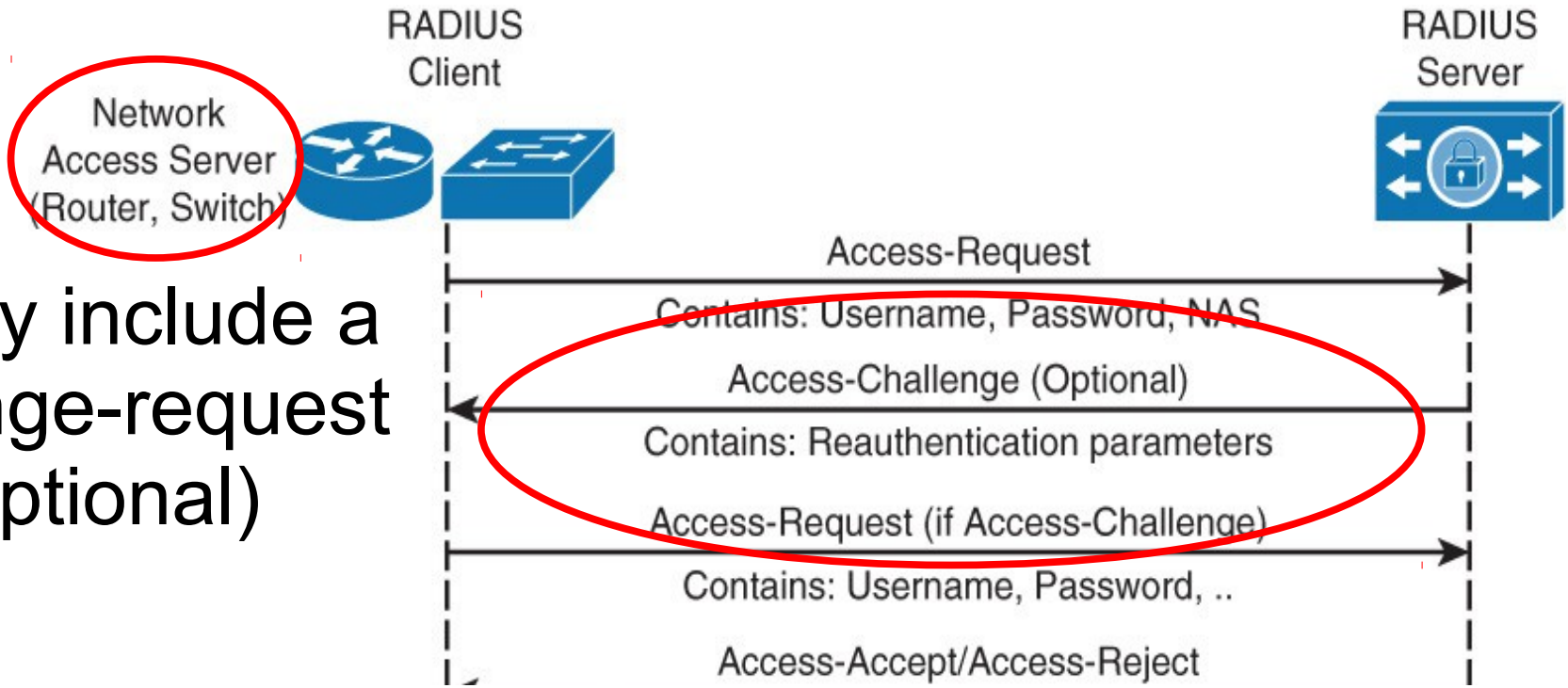
- You need to know the main operating characteristics and differences of RADIUS and TACACS+

Feature	RADIUS	TACACS+
Support	RFC-based	Proprietary
Transport	UDP port 1812/1813	TCP port 49
Latency	Faster (connectionless)	Slower
AAA	Authentication + Authorization combined	all three are separate
Challenge flow	1-way: server->client authentication only	2-way: permits mutual authentication
Encryption	password only	entire packet
Bindings	single: character **or** PPP	multi: character/PPP

AAA – Remote Servers (3)

- RADIUS authentication sequence can be simple

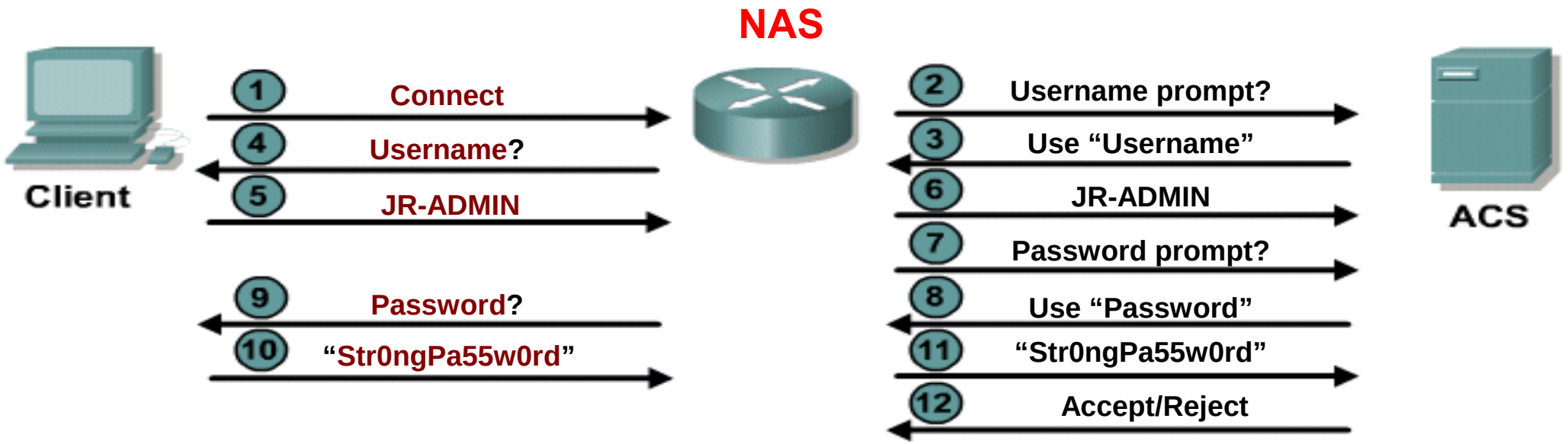




- but may include a challenge-request step (optional)

AAA – Remote Servers (4)

- TACACS+ authentication sequence involves multiple challenge-response sequences, relayed by the NAS device



AAA – Cisco Quirks (1)

- Until AAA is enabled globally (**`aaa new-model`**), all other AAA commands are hidden/unavailable

```
A(config-line)# login ?
  local  Local password checking
  <cr>
A(config-line)# exit
A(config)# aaa new-model
A(config)# line con 0
A(config-line)# login ?
  authentication  Authentication parameters.
  ctrlc-disable   Disable CONTROL-C during login.
```

- Enabling AAA immediately applies local authentication to all lines and interfaces (except the console). To avoid being locked out, you should define a local account before starting AAA
- RADIUS has incorrect default ports (1645-1646)

AAA – Cisco Quirks (2)

- Authentication is:
  - always required as soon as AAA is enabled
  - checked against the *local* user accounts if no default authentication list exists (except console)
  - checked according to the "*default*" list if it exists and no other list is configured on a line (except console which would have no checking)
  - checked according to the named list configured on a line
- We'll see in the next section that Cisco handles the requirement for Authorization a little differently from Authentication

AAA – Authentication: Configuration (1)

```
A(config)# aaa new-model
A(config)# aaa authentication login default none
A(config)# aaa authentication login StdConsoleAuth line
```

- Reverting back to default config conditions, with AAA enabled (no password req'd for console):

```
A(config)# aaa new-model
A(config)# line con 0
A(config-line)# login authentication default
```

- The equivalent of what you've always configured:

```
A(config)# aaa new-model
A(config)# line con 0
A(config-line)# password class
A(config-line)# login authentication StdConsoleAuth
```

AAA – Authentication: Configuration (2)

- There are multiple ways of configuring RADIUS and TACACS+ servers singly and/or in groups
- Cisco CLI syntax depends strongly on IOS version; compare FLG textbook 15.0(x) with format below

```
A(config)# ! Syntax below is valid for IOS 12.2-12.4
A(config)#aaa group server radius SomeNameYouPick
A(config-sg-radius)#server 10.1.1.1 ?
  acct-port UDP port for accounting server (default: 1646)
  auth-port UDP port for authentication server (default: 1645)
  <cr>
A(config-sg-radius)#server 10.1.1.1
A(config-sg-radius)#server 10.2.2.2
A(config-sg-radius)#server 10.3.3.3
```

AAA – Authentication: Configuration (3)

Here's a suggested configuration sequence:

1. Define a fall-back local user account
2. Define fall-back enable and line passwords
3. Enable AAA: **`aaa new-model`**
4. Define one or more RADIUS, TACACS+ groups
5. Define default login authentication (include local!):
   `aaa authentication login default group xxx local`
6. Define additional login lists, as required:
   `aaa authentication login {altName} group xxx local`
7. Define the enable list if required (include 'enable'):
   `aaa authentication enable default group xxx enable`
8. Apply authentication lists to desired lines
   `line con 0: login authentication [default|{altName}]`
   `line vty 1 {n}: login authentication [default|{altName}]`

AAA – Authorization (1)

- Authorization defines *what* a user is allowed to do and may be quite granular, for example:
  - commands: access to given level 0-15 of commands
  - exec: access to privileged exec (level 15)
  - network: PPP, SLIP

A partial list of privileges

- Authorization lists again have up to four methods used to confirm permission for a user action. Choose from:
  - group: use a (named) RADIUS/TACACS+ group
  - krb5-instance: Kerberos instance privilege maps

Remote

  - local: use local account definitions from **`user`** …
  - if-authenticated: succeed if *authenticated*
  - ~~none: don't require any authentication~~ AVOID!!

Local

AAA – Authorization (2)

- As noted, Cisco handles the requirement for Authorization a little differently from Authentication
- Authorization is:
  - *Not* checked if no authorization lists exist
  - checked according to the "*default*" list if it exists and no other named list is configured on a line (except console which has no check in this case)
  - checked according to the named list configured on a line

AAA – Authorization: Configuration

Assuming the standard Authentication steps are complete:

– Define a fall-back local user account
– Enable AAA: **`aaa new-model`**
– Define one or more RADIUS, TACACS+ groups

Configuring Authorization is a shorter sequence:

1. Define default exec authorization (include local!):
   ```
   aaa authorization exec default group xxx local
   ```
2. Define additional exec lists, as required:
   ```
   aaa authorization exec {altName} group xxx local
   ```
3. Define additional lists as required (include local!):
   ```
   aaa authorization commands {lvl-#} default group xxx local
   ```
4. Apply authorization lists to desired lines
   ```
   line con 0: authorization exec [default|{altName}]
   line vty 1 {n}: authorization exec [default|{altName}]
   ```

AAA – Accounting

- Accounting tracks & logs what's *done* on a system
  – there are ~18+ categories of actions / events
- Accounting has an extra parameter of *how* to track and report actions/events (ie. style). Choose from:
  – none: no accounting
  – start-stop: record start and stop without waiting
  – stop-only: record stop when service terminates
- Accounting lists may specify *where* as up to *two* destinations for reporting. Choose from:
  – broadcast: send accouting via broadcasts
  – group: send to a server group

AAA – Accounting: Configuration

- No Accounting is generated & reported, unless specifically configured
- Define default and/or named Accounting lists:

```
aaa accounting {type}   default {style}  {dest-list}
aaa accounting resource default start-stop group Svrs
```

- Activating accounting takes only a single line: on the desired line or interface, identify the type of accounting and the list to be used

```
line con 0: accounting exec [default|{altName}]
line vty 1 {n}: accounting resource [default|{altName}]
```

- As with the others, syntax varies according to IOS

AAA – Did You Notice?

- Each of the three parts of AAA have separate list definitions
- Each list definition may specify a different server
- So the three AAA functions can be split across different servers to obtain load distribution and/or balancing
- The load distribution allows AAA to be a highly scalable solution, even in situations where a high volume of traffic occurs from a large number of nodes

802.1X – Basics

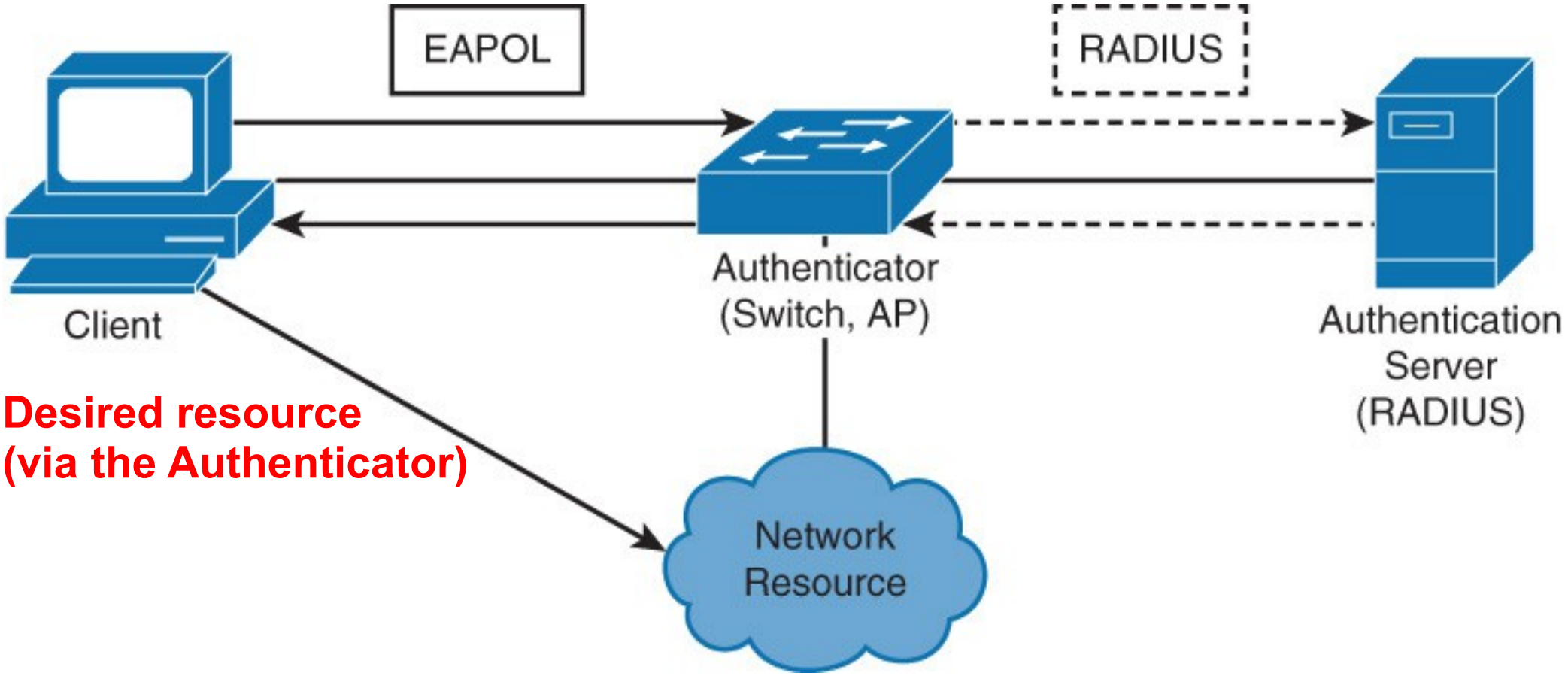
- With the high degree of client mobility (think laptops and wireless), traditional security applied to a specific port or interface is no longer adequate
- Security now needs to be applied to the user, wherever s/he may be and whatever port is used for connectivity
- 802.1X completely blocks (almost) all traffic from a port until a client has successfully authenticated
- The blocking and passing of traffic after validation is referred to as Network Admission Control (NAC)
- Wireless WPA2-Enterprise authentication appears to be exclusively implemented via 802.1X
- 802.1X is tightly tied to RADIUS authentication mechanism

802.1X – Characteristics

- 802.1X uses an expanded client-server model:
  - a client or *supplicant* with 802.1X software
  - the *authenticator* which is typically a switch or AP
  - the *authentication server*
- The authenticator has several responsibilities:
  - does the actual blocking/passing of user traffic
  - acts as a RADIUS client, as a proxy for the user
  - encapsulates Extensible Authentication Protocol (EAP) frames to travel Over the Lan (EAPOL)
  - allows only EAPOL, CDP, STP prior to successful authentication

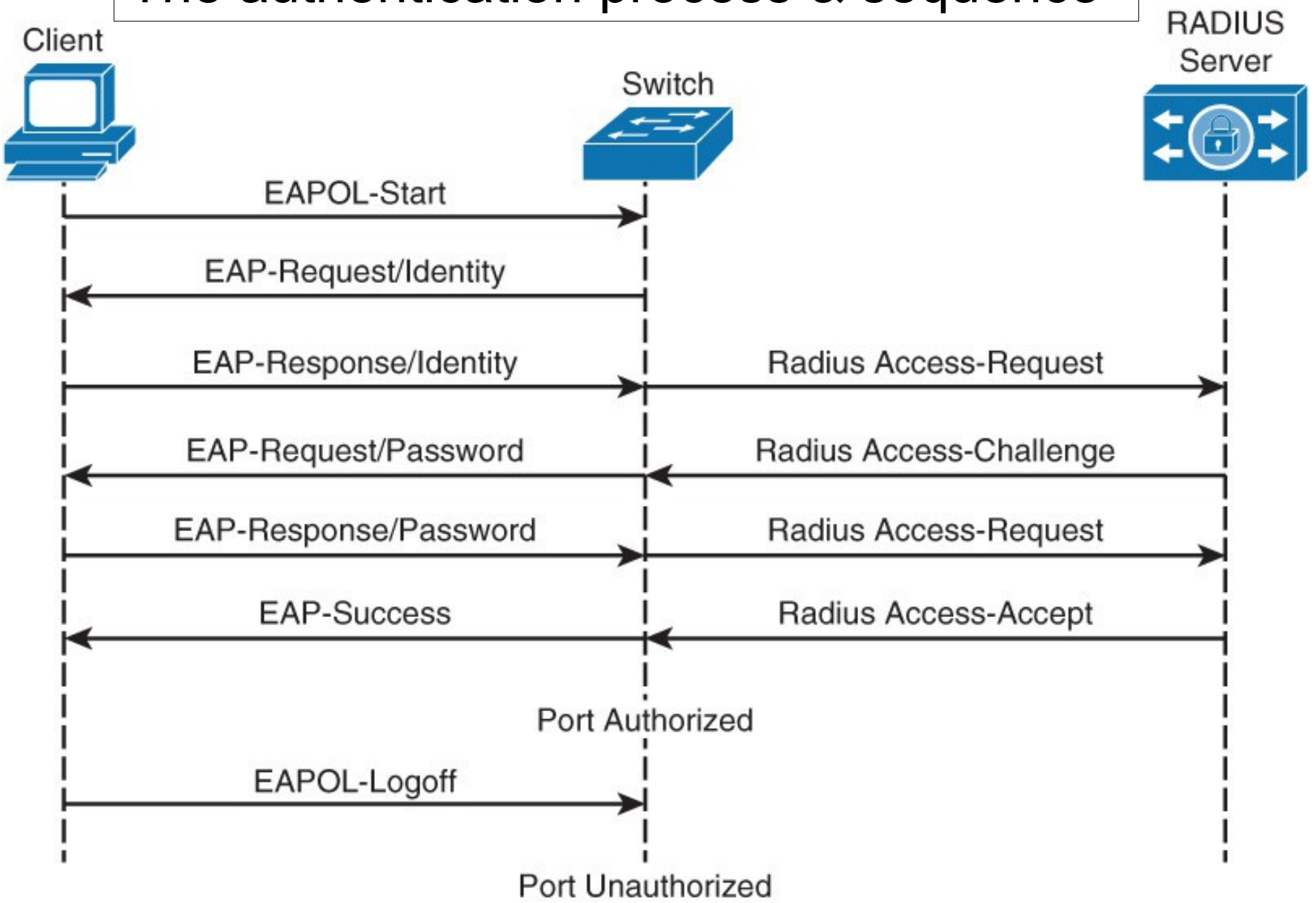
802.1X – Characteristics (2)

An illustration of the three roles:



802.1X – Characteristics (3)

The authentication process & sequence



802.1X – Configuration

- 802.1X relies on RADIUS, which means AAA and server groups must be correctly configured first

```
Sw(config)# aaa new-model
Sw(config)# aaa group server radius {grp-name}
Sw(config-sg-radius)# server 10.1.1.1 auth-p 1812 acct-p 1813
```

- Three steps complete the setup:
  - enable 802.1X globally
  - configure an authentication list for 802.1X
  - apply 802.1X to a port in *access mode*

```
Sw(config)# dot1x system-auth-control
Sw(config)# aaa authentication dot1x default group {grp-name}
Sw(config-if)# dot1x port-control auto|force-auth|force-unauth
```

Reminder

- *Not all* details appear in these slides!
- You are responsible for *reading* the textbook to gain the knowledge (memorization) and understanding (apply the knowledge)