Post 10

Thursday, January 24, 2019 11:26 PM

SYSLOG/SNMP/NETFLOW

Intro: Common method of accessing sys msgs: Protocol named syslog

- Term used to describe standard: Syslog dev for UNIX: 1980s: 1st doc RFC 3164: IETF: 2001
- Syslog uses UDP: 514 to send event notification msgs across networks to event msg collectors
- Possible to build special OOB: Out-of-band network for this purpose
- Diff syslog server SW packages for Win/UNIX

Service provides 3 primary functions:

1. Gather logging info for monitoring/troubleshooting
2. Select type of logging info captured
3. Specify dest of captured syslog msgs

Syslog Op: Cisco: Syslog starts by sending sys msgs/debug output to local logging process internal to device

- Msgs may be sent to internal buffer: Only viewable through CLI of device

Popular dest for syslog msgs: Logging buffer: RAM inside rtr/switch/Console/Terminal/Syslog server
Syslog Msg Fmt: Every msg contains severity lvl/facility: Smaller # lvls more critical alarms
Each syslog lvl has meaning:

Warning/Emergency	Error msgs about SW/HW malfuncs: Func of device affected: Severity determines lvl
Debugging	Output generated from issuing debug cmds
Notification	Only for info: Func not affected: Int up/downs/sys restart msgs displayed

In addition to severity: Msgs also contain info on facility:
Syslog facilities: Service identifiers: ID/categorize sys state data for error/event msg reporting
Common syslog msg facilities on IOS rtrs:

IP	**OSPF**	**SYS OS**	**IPsec**	**Int IP: IF**

Default: Fmt of syslog msgs on IOS:
seq no: timestamp: %facility-severity-MNEMONIC: description
00:00:46: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up
Syslog Severity Lvl

Severity Name	Severity Level	Explanation
Emergency	0	Sys unusable
Alert	1	Immediate action needed
Critical	2	Critical condition
Error	3	Error condition
Warning	4	Warning condition
Notification	5	Normal: Sig condition
Informational	6	
Debugging	7	

Syslog Msg Fmt

Field	Explanation
seq no	Stamps log msgs w/seq # only if **service sequence-numbers** [global] config
timestamp	Date/time of msg/event: Only if **service timestamps** [global] config
facility	Facility to which msg refers
severity	From 0-7: Severity of msg
MNEMONIC	Txt str that uniquely describes msg

description	Txt str containing detailed info about event reported

Service Timestamp: Msgs can be time-stamped/source address of syslog msgs can be set
- Enhances real-time debugging/mgmt

service timestamps log uptime [global] Amt of time since switch last booted displayed on logged events
- More useful ver applies **datetime** keyword in place of **uptime**
- Forces each log event to display date/time associated w/event
- When using datetime: Clock on device must be set

Accomplished 1 of 2 ways:
Manually: clock set
Auto: Using NTP
Config NTP:
R1(config)# ntp master 1
R1(config)# ntp server IP
Syslog Server: To view syslog msgs: syslog server must be installed on a workstation
- Advantage: Ability to perform granular searches through data: Can quickly del unimpt msgs from db

Default Logging: Default: Cisco: Rtrs/switches send log msgs for all severity lvls to con: Some IOS ver: Default: Device buffers msgs
To enable:
R1(config)# logging console
R1(config)# logging buffered
R1# sh logging Displays default logging service settings on rtr
- 1st line of output lists info about logging process: End of output log msgs
- Also lists msgs in buffer: Can view sys msgs logged at end of output

Rtr/Switch Cmds for Syslog Clients
R1(config)# logging 192.168.1.3
R1(config)# logging trap 4
R1(config)# logging source-interface g0/0
R1(config)# int loopback 0
3 Steps to config rtr to send sys msgs to syslog server:
1. Config dest hostname/IP of syslog server **logging 192.168.1.3**
2. Control msgs to be sent to syslog server w/**logging trap level** [global] cmd

Limit logging to level 4/below:
R1(config)# logging trap 4 | R1(config)# logging trap warning
1. Config src int w/**logging source-int int-type int#** [global]
    - Specifies that syslog packets contain IPv4/6 addr of specific int
    - Regardless of which int packet uses to exit rtr

logging trap Limits syslog msgs sent to server based on severity
Verify Syslog
View any msgs logged:
R1# show logging
- When logging buffer large: Helpful to use **|** w/**show**

R1# show logging | include changed state to up
- Ensures only int notifications stating int up displayed

SNMP: Simple Network Mgmt
- Dev to allow admins to manage nodes [servers/workstations/rtrs/switches/sec apps] on IP network
- Enables admins to manage performance/find/solve problems/plan for growth

SNMP: App layer protocol: Provides msg fmt for comm bet managers/agents
Consists of 3 elements:
1. Manager
2. Agents: Managed node
3. MIB: Mgmt Info Base

To config: 1st necessary to define relationship bet manager/agent
- Part of NMS: Network Mgmt Sys: Runs SNMP mgmt SW
- Can collect info from SNMP agent using “get” action
- Can change configs on agent using “set” action
- SNMP agents can fwd info directly to NMS using “traps”

SNMP agent/MIB reside on networking device clients
- MIBs store data about device op/meant to be avail to auth remote users
- Agent responsible for providing access to local MIB of objects that reflects resources/activity
- Defines how mgmt info exchanged bet apps/agents

SNMP: UDP: 162: Retrieve/send mgmt info

SNMP Op

Operation	Description
get-request	Retrieves value from specific var
get-next-request	Retrieves value from var w/in table: SNMP manager doesn't need to know exact var name • Sequential search performed to find needed var
get-bulk-request	Retrieves large blocks of data [like multiple rows in table] • That would otherwise req transmission of many small blocks of data (only SNMPv2/later)
get-response	Replies to get-request, get-next-request, set-request sent by NMS
set-request	Stores value in specific var

2 primary SNMP manager requests: get | set
get request: Used by NMS to query device for data
set request: Used by NMS to change config vars in agent device

- Can also initiate actions w/in device

SNMP agent responds to SNMP manager reqs as follows:

Get MIB var	Agent performs func in response to GetRequest-PDU from NMS • Agent retrieves value of req MIB var/responds to NMS w/that value
Set MIB var	Agent performs func in response to SetRequest-PDU from NMS • Agent changes value of MIB var to value specified by NMS • SNMP agent reply to set request includes new settings in device

Agent Traps: NMS periodically polls SNMP agents residing on managed devices, by querying device for data using get req

- Can collect info to monitor traffic loads/verify device configs of managed devices
- SNMP manager samples value periodically/presents info in graph for network admin to use in creating baseline

Periodic SNMP polling disadvantages:

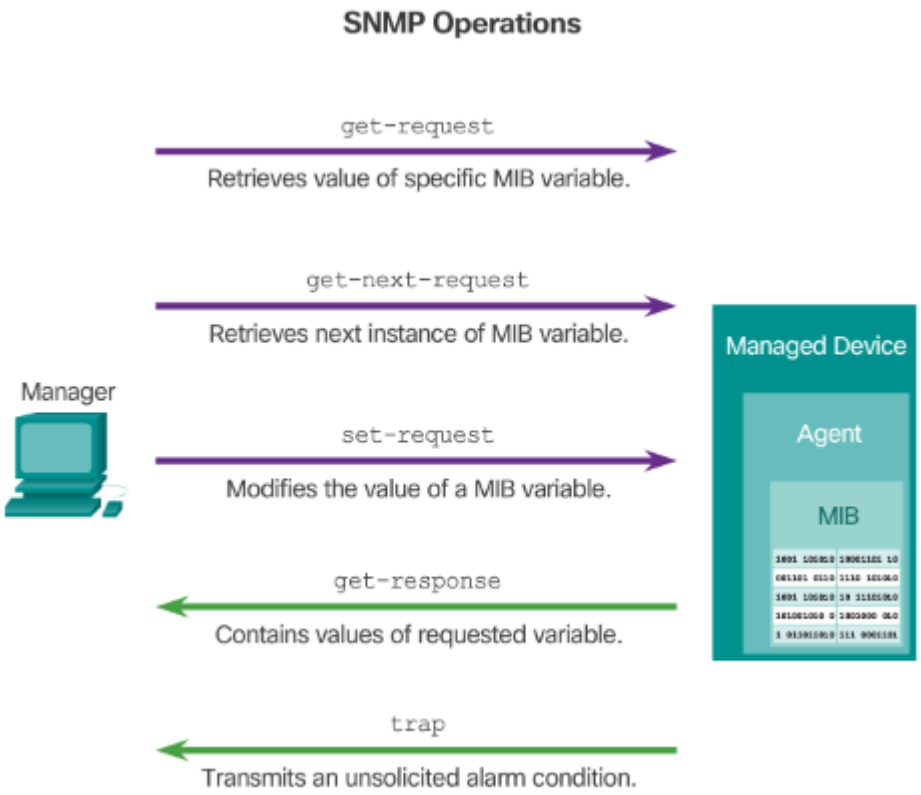
- Delay bet time event occurs/time noticed by NMS
- Trade-off bet polling frequency/BW usage

To mitigate disadvantages:

- Possible for SNMP agents to generate/send traps to inform NMS immediately of certain events

Traps: Unsolicited msgs alerting SNMP manager to condition/event on network

- Reduce network/agent resources, by eliminating need for some polling req



SNMP Versions

SNMPv1	SNMP: Full Internet Standard: RFC 1157
SNMPv2c	RFC1901-1908: Utilizes comm-str-based Admin Framework
SNMPv3	Interop standards-based protocol: Originally RFCs2273-2275 • Sec access to devices by auth/encrypting packets over network **Sec features:** • Msg integrity to ensure packet not tampered with in transit; auth to determine msg from valid src • Encryption to prevent contents of msg from being read by unauth source

SNMPv1/SNMPv2c: Comm-based form of sec: Managers able to access agent's MIB defined by ACL/passwd

SNMPv2c: Includes bulk retrieval mech/more detailed error msg reporting to mgmt stations

- It retrieves tables/large quantities of info, min # of round-trips req
- Improved error-handling: Expanded error codes: Distinguish diff kinds of error conditions
- Conditions reported through single error code in SNMPv1: Error returns in SNMPv2c include error type

SNMP Sec Models/Levels

Model	Level	Auth	Encryption	Result
SNMPv1	noAuthNoPriv	Comm Str	No	Uses comm str match for auth
SNMPv2	NoAuthNoPirv	Comm Str	No	Uses comm str match for auth
SNMPv3	NoAuthNoPriv	Usrname	No	Usrname match for auth (improvement over SNMPv2c)
SNMPv3	authNoPriv	MD5/SHA	No	Provides auth based on HMAC-MD5/HMAC-SHA algs
SNMPv3	authPriv (crypto SW img)	MD5/SHA	DES/AES	Auth based on HMAC-MD5/HMAC-SHA algs • DES-56bit encryption in add to auth based on CBC-DES standard • 3DES 168-bit encryption

				• AES 128/192/256-bit encryption

Comm Strs

- For SNMP to op: NMS must have access to MIB: Some form of auth must be in place
- SNMPv1/SNMPv2c use comm strs that control access to MIB
- **Comm strs** = plaintext passwds: Auth access to MIB objects

2 types of comm strs:

Read-only: RO	Access to MIB vars: Doesn't allow vars to be changed (only read) • Sec minimal in ver 2c: Many orgs use SNMPv2c in read-only mode
Read-write: RW	Read/write access to all objects in MIB

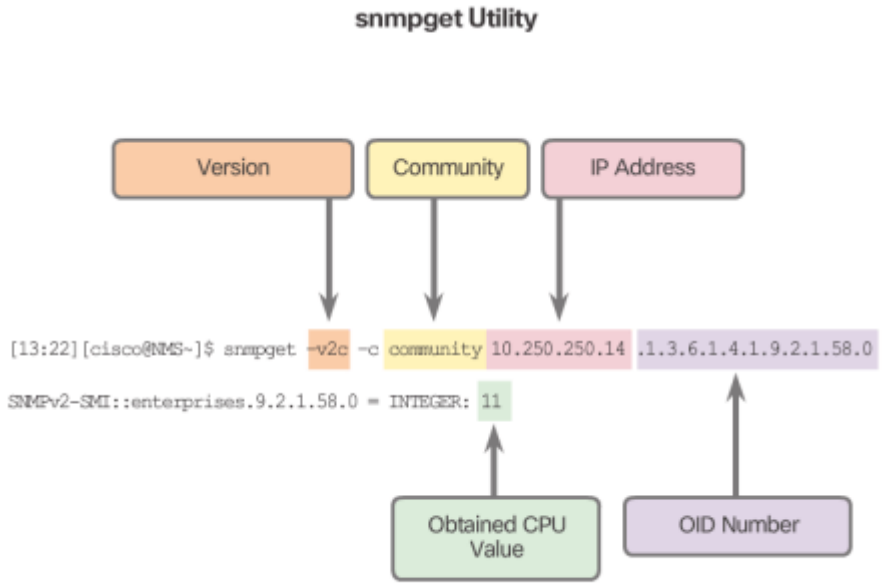
MIB: Mgmt Info Base Object ID

- MIB organizes vars hierarchically
- MIB vars: Enable mgmt SW to monitor/control network device
  - Formerly: Defined each var as an OID: Object ID
  - OIDs uniquely ID managed objects in MIB hierarchy
  - MIB organizes OIDs based on RFC into hierarchy of OIDs: Usually shown as tree

CPU: Key resource: Should be measured continuously: Statistics should be compiled on NMS/graphed

- Observing CPU utilization over an extended time period allows admins to establish baselines
- Threshold values can then be set relative to the baseline
- When CPU utilization exceeds threshold: Notifications sent
- SNMP graphing tool can periodically poll agents

Data retrieved via snmpget utility: Issued on NMS: Can manually obtain values for avg CPU busy %
snmpget req SNMP ver: Correct comm: IP of device to query: OID # set
Config SNMP



R1(config)# snmp-server community batonaug ro SNMP_ACL
R1(config)# snmp-server location NOC_SNMP_MANAGER
R1(config)# snmp-server contact Wayne World
R1(config)# snmp-server host 192.168.1.3 version 2c batonaug
R1(config)# snmp-server enable traps
R1(config)# ip access-list standard SNMP_ACL
R1(config-std-nacl)# permit 192.168.1.3
Steps:

1. Config comm str/access level (ro/rw) **snmp-server community string ro | rw**
2. Doc loc **snmp-server location**n [optional]
3. Doc sys contact **snmp-server contact** [optional]
4. Restrict SNMP access to NMS hosts perm by ACL: [optional]
   - Define ACL/reference w/ snmp-server community string access-list-number-or-name
   - Can be used both to specify comm str/restrict SNMP access via ACLs
5. Specify recipient of SNMP trap ops **snmp-server host host-id [version{1| 2c | 3 [auth | noauth |**

priv]}] comm-str
   - Default: No trap manager defined
6. Enable traps on SNMP agent snmp-server enable traps notification-types
   - If no trap notification types specified: All trap types sent: Repeat if needed

Verify SNMP Config
R1# show snmp
R1# show snmp community

- **show snmp:** Doesn't display info related to SNMP comm str/associated ACL

Best Practices

- Ensure SNMP msgs don't spread beyond mgmt cons
- ACLs should be used to prevent SNMP msgs from going beyond required devices
- ACL should also be used on monitored devices to limit access for mgmt sys only

SNMPv3:
Create new SNMP group on device:
snmp-server group groupname {v1 | v2c | v3 {auth | noauth | priv}}
Add new usr to SNMP group
snmp-server user username groupname v3 [encrypted] [auth {md5 | sha} auth-password] [priv {des | 3des | aes {128 | 192 | 256}} priv-password]
NetFlow: Cisco: Provides statistics on packets flowing through rtr/multilayer switch:

- Standard for collecting IP operational data from networks
- Historically: Dev b/c ppl needed simple/efficient method for tracking TCP/IP flows: SNMP wasn't good for this
- SNMP attempts to provide wide range of mgmt features/options
- NetFlow focused on providing statistics on packets flowing through devices

Flexible NetFlow: Improves original by ability to customize traffic analysis params for specific reqs

- More complex configs for traffic analysis/data export through use of reusable config components
- Uses Version 9 export fmt: Template-based
   - Templates provide extensible design to record fmt

Network Flows: Breaks down TCP/IP comm for statistical record keeping using the concept of a flow
Flow: Unidirectional stream of packets bet specific source sys/dest

- NetFlow: Built around TCP/IP: Src/dest defined by network layer IP: Transport layer src/dest port #'s

Packets safely determined from diff flows:

- Source/Dest IP
- Source/Dest port #
- L3 protocol type
- ToS: Type of Service marking
- Input logical int

Config NetFlow
R1(config)# int g0/1
R1(config-if)# ip flow ingress
R1(config-if)# ip flow egress
R1(config-if)# exit
R1(config)# ip flow-export destination 192.168.1.3 2055
R1(config)# ip flow-export version 5
Implement NetFlow on rtr:

1. Config data capture: Captures data from ingress (inc)/egress (outgoing) packets
2. Config data export: IP/hostname of NF collector must be specified/UDP port NF collector listens on
3. Verify NF/op/stats: – After config: Exported data can be analyzed

NF flow unidirectional: One usr connection to app exists as 2 NF flows: One for each direction
Define data captured for NF in int config mode:

- NF data for monitoring inc packets on int using **ip flow ingress**
- NF data for monitoring outgoing packets on int **ip flow egress**

Enable NF data to be sent to collector:
NF collector's IP/UDP port #
ip flow-export destination ip-address udp-port

- Collector has 1/more ports: Default NF data capture
- SW allows admin to specify which port to accept for capture
- Common UDP ports: 99/2055/9996

NF ver when fmting NF records to collector:
ip flow-export version version

- NF exports data in UDP in 1-5 fmts: 1/5/7/8/9

- Ver 9 most versatile export data fmt: Not backward compatible: Ver 1 default

Src int to use as src of packets sent to collector:
ip flow-export source typenumber
Verify NF
R1# show ip cache flow
Input Descriptors

Field	Description
bytes	# of bytes of mem used by NF cache
active	# of active flows in NF cache at time cmd entered
inactive	# of flow buffers allocated in NF cache, but not currently assigned to specific flow at time cmd entered

Output Descriptors

Operation	Description
Protocol	IP/well-known port #
Total Flows	# of flows in cache for protocol since last time stats cleared
Flows/Sec	Avg # of flows for protocol per sec = total flows divided by # of sec for period
Packets/Flow	Avg # of packets for flows for protocol = total packets for protocol / by # of flows for protocol for period
Bytes/Pkt	Avg # of bytes for packets for protocol = total bytes for protocol / by # of packets for protocol for period
Packets/Sec	Avg # of packets for protocol per second = total packets for protocol / by total # of sec for summary period
Active(Sec)/Flow	# of sec from 1st packet to last packet of expired flow / by # of total flow for protocol for period
Idle(Sec)/Flow	# of sec observed from last packet in each non-expired flow for protocol • until time **sh ip cache verbose flow** entered / by total # of flows for protocol for period

NF Record Descriptors

Operation	Description
SrcIf	Int on which packet was received
SrcIPaddress	IP of device that transmitted packet
DstIf	Int from which packet transmitted; If an * immediately follows DstIf field: Flow shown as egress
Pr	Ip protocol “well-known” port #/displayed in hex fmt
SrcP	Src protocol port # in hex
DstP	Dest protocol port # in hex
Pkts	# of packets switched through this flow

R1# show ip flow interface
R1# show ip flow export