Advanced IOS Device Instrumentation

May 2006
Agenda

• Introduction

• Embedded Management Tools
  Tool Command Language (TCL)
  Embedded Event Manager (EEM)
  Embedded Syslog Manager (ESM)
  Embedded Resource Manager (ERM)
  Command Scheduler (Kron)
  Configuration Replace and Rollback
  Contextual Configuration Diff Utility

• Enhanced Device Interface
Agenda (Cont.)

- Practical Applications
- Summary and Conclusion
Got Tools?
Agenda

• Introduction

• Embedded Management Tools
  Tool Command Language (TCL)
  Embedded Event Manager (EEM)
  Embedded Syslog Manager (ESM)
  Embedded Resource Manager (ERM)
  Command Scheduler (Kron)
  Configuration Replace and Rollback
  Contextual Configuration Diff Utility
Tool Command Language
Tool Command Language (TCL) Overview

- TCL is a multithreaded interpreted scripting language
- Language resources found at: [http://www.tcl.tk/](http://www.tcl.tk/)
- TCL 7.x has been in Cisco IOS Software since 1994
- TCL 8.3.4 first released in Cisco IOS Software Release 12.3(2)T and merged into Release 12.2(25)S
Tool Command Language (TCL) Features

• Support for scripts compiled with the TCLPro bytecode
• Support for TCL namespaces
• Allows execution of exec commands and Cisco IOS Software configuration
Tool Command Language (TCL) Uses Within Cisco IOS Software

- Build custom show commands
- Access SNMP objects
- Integrate with the Embedded Syslog Manager and Embedded Event Manager
- Build Interactive Voice Response (IVR) scripts
- Consolidate complex configuration commands
- Autoconfiguration
Tool Command Language (TCL)
Starting the Interpreter

Router# tclsh
Router(tcl)#
Tool Command Language (TCL) Configuration

Router(config)#scripting tcl ?

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encdir</td>
<td>Specify path for TCL character encoding files</td>
</tr>
<tr>
<td>init</td>
<td>Specify path for TCL initialization script</td>
</tr>
<tr>
<td>low-memory</td>
<td>Configure low water memory mark</td>
</tr>
</tbody>
</table>

- The encdir and init values can be any Cisco IOS URI (ie: disk:, slot:, tftp:, etc.)
- Use the low-memory command to avoid crashes due to memory allocation (do not go less than 10% of total available memory)
Interactive Shell

Router#tclsh
Router(tcl)#puts "Hello Networkers"
Hello Networkers

Router(tcl)#exit
Router#
## Tool Command Language (TCL) Configuration (Cont.)

### Running Cisco IOS Commands

```tcl
Router(tcl)#set output [exec "show interface fa0/0 description"]
Interface                      Status         Protocol Description
Fa0/0                          up             up       FlashNet
Management Connection

Router(tcl)#log_user 0
0

Router(tcl)#set output [exec "show interface fa0/0 description"]

Router(tcl)#puts $output

Interface                      Status         Protocol Description
Fa0/0                          up             up       FlashNet
Management Connection
```

TCL Cisco IOS
Extended Commands
TCL Built In Command
Cisco IOS Command
**TCL and CLI Configuration Commands**

```tcl
Router(tcl)#ios_config "interface fa0/0" "description Networkers Uplink"

Router(tcl)#set output [exec "show interface fa0/0 description"]

Router(tcl)#puts $output

<table>
<thead>
<tr>
<th>Interface</th>
<th>Status</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fa0/0</td>
<td>up</td>
<td>up</td>
</tr>
<tr>
<td>Uplink</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**Tool Command Language (TCL)**

Configuration (Cont.)

TCL Cisco IOS Extended Commands
TCL Built In Command Cisco IOS Command
Writing to the Input Buffer

Router(tcl)#typeahead "show run\n"

Router(tcl)#show run
Building configuration...
Current configuration : 8245 bytes
!
! Last configuration change at 22:05:49 CET Sat Mar 10 2005
!
version 12.0
no service pad
...

TCL Cisco IOS
Extended Commands
TCL Built In Command
Cisco IOS Command
Capturing Cisco IOS Errors

Router(tcl)#set line "snmp server community RO"
Router(tcl)#if {[catch {ios_config $line} result]} {
    +>puts "Bad config command: \"$line\"
    +>}
Bad config command: "snmp server community RO"
Tool Command Language (TCL) Configuration (Cont.)

Loading External Scripts

Router(tcl)#source slot0:myscript.tcl

Router(tcl)#source tftp://10.10.10.10/myscript.tcl

Router#tclsh tftp://10.10.10.10/myscript.tcl

Tip: Keep common scripts in a central TFTP archive
Tool Command Language (TCL)
SNMP Support

• Requires an SNMP community to be configured on the router
• Provides easy access to SNMP objects and commands
  - snmp_getbulk—retrieves a large section of the MIB tree
  - snmp_getid—retrieves the system table
  - snmp_getnext—retrieves the next object in the MIB tree
  - snmp_getone—retrieves one object in the MIB tree
  - snmp_setany—sets an object in the MIB tree
• Data is returned in an XML format
• First introduced in Release 12.3(7)T
Tool Command Language (TCL)
SNMP Example

Router(tcl)#snmp_getid public
Copyright (c) 1986-2005 by Cisco Systems, Inc. Compiled Fri 25-Mar-05 14:01 by yiyan'/>}
{<obj oid='system.2.0' val='products.108'/>}
{<obj oid='sysUpTime.0' val='71184284'/>}
{<obj oid='system.4.0' val='Dan Jerome'/>}
{<obj oid='system.5.0' val='dj.cisco.com'/>}
{<obj oid='system.6.0' val='Networkers 2005'/>}

Router(tcl)#snmp_setany private system.6.0 -d "Networkers 2006"
{<obj oid='system.6.0' val='Networkers 2006'/>}
The following is a list of differences between the TCL 8.3.4 standards and Cisco IOS Software

<table>
<thead>
<tr>
<th>Command</th>
<th>Keyword</th>
<th>Argument</th>
<th>Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>after</td>
<td>ms</td>
<td>script</td>
<td>Yes</td>
</tr>
<tr>
<td>file</td>
<td>atime</td>
<td>atime</td>
<td>No</td>
</tr>
<tr>
<td>file</td>
<td>mtime</td>
<td>mtime</td>
<td>No</td>
</tr>
<tr>
<td>fileevent</td>
<td></td>
<td></td>
<td>Yes*</td>
</tr>
<tr>
<td>history</td>
<td>!n</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>load</td>
<td></td>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>
• **Script Debugging**

  Use a UNIX or Windows TCL 8.3 interpreter to “sanity check” code

  Make sure log_user is set to 1 to get all possible errors

  Use Control+Shift+6 to interrupt a runaway script
Tool Command Language (TCL) Caveats

- Use Release 12.3(14)T or later for best results
- Use low-memory to prevent malloc failures
- **TCL process runs at medium priority**, so be careful with loops
Tool Command Language (TCL) Security Concerns

- No implied trust with TCL scripts
- Load scripts from network servers with care
- Use privilege levels to control access to the tclsh

```bash
Router(config)#username admin privilege 7 password cisco
Router(config)#username dan privilege 3 password cisco
Router(config)#privilege tcl all level 7 tclsh
Router(config)#line vty 0 903
Router(config-line)#login local

NMS_server% telnet Router
Trying 10.10.10.10...
Connected to Router.cisco.com.
Escape character is '^]'.
User Access Verification
Username: dan
Password:
Router#tclsh
Translating "tclsh"...domain server (10.10.10.10)
```
Tool Command Language (TCL)

• Additional References

  General Language Resources
  http://www.tcl.tk/

  Cisco IOS Scripting with TCL

  Cisco Open Source Initiative (COSI) with scripts found in this session
  http://sourceforge.net/project/showfiles.php?group_id=25401&package_id=154317&release_id=332786
Embedded Event Manager
Embedded Event Manager (EEM) Overview

• Service running in IOS (In-box monitoring)

• Offers the ability to monitor events and take informational, corrective or any desired action, when the monitor event occurs or when a threshold is reached via sw agents

• Advantages
  
  Ability to take proactive actions based on configurable events
  
  Reduce network bandwidth by doing local event monitoring
• Version 1.0 introduced in Releases 12.0(26)S, 12.3(4)T
• Version 2.0 introduced in Release 12.2(25)S
• Version 2.1 introduced in Release 12.3(14)T
• Version 2.1.5 introduced in Release 12.2(18)SXF1
• Version 2.2 introduced in Release 12.4(2)T
EEM Architecture

All of this is internal to Cisco IOS Software

Think of a policy as an action registered to an event

- Applet-based policies
- Defined via CLI
- Simpler

ED notifies EEM Server; which triggers interested policies

- Tcl-based policies
- Programmed in Tcl
- As complex as you want

POLICY ENGINES - TWO TYPES

EEM Applet Policy
Subscribes to receive events, implements policy actions

EEM Tcl Policy
Subscribes to receive events, implements policy actions

Event Subscribers
**EEM Policies**

- Entity which defines an event and actions to be taken
- Policies should be short scripts that require no less than 20 seconds to interpret and run
- Two Engines:
  - CLI Based (Applet Interface)
  - Script Based (Tcl) – supported since EEM 2.1
- Two Policy Types:
  - Synchronous – policy can affect the outcome of the event
  - Asynchronous – policy runs asynchronously with the event
EEM Policy Simple Example

- Write a special syslog message (even with different severity) when we see a particular syslog message

  When someone leaves config mode, this message is seen:

  Router#conf t
  Enter configuration commands, one per line.  End with CNTL/Z.
  Router(config)#exit
  Router#
  *Oct 15 06:29:44.113: %SYS-5-CONFIG_I: Configured from console
  by vty0 (144.254.8.54)
Router(config)# event manager applet CFGMSG
Router(config-applet)# event syslog pattern "%SYS-5-CONFIG_I:"
Router(config-applet)# action 1.0 syslog priority warnings msg
   "Configuration event occurred"

Router(config-applet)# exit
Router(config)# exit
Router#
*Oct 15 06:42:34.773: %SYS-5-CONFIG_I: Configured from console by
vty0 (144.254.8.54)
*Oct 15 06:42:34.789: %HA_EM-4-LOG: CFGMSG: Configuration event
occurred

Router#sh event manager policy registered
No.  Class   Type    Event Type  Trap  Time Registered          Name
1    applet  system  syslog      Off   Sat Oct 15 06:42:31 2005  CFGMSG
    pattern {%SYS-5-CONFIG_I:}
    action 1.0 syslog priority warnings msg "Configuration event occurred"
Cisco IOS Embedded Event Manager 1.0
Basic Architecture

Event Detectors Feed EEM

Syslog Event
SNMP Data
Other Event

Syslog Event Detector
SNMP Event Detector
Other Event Detector

Embedded Event Manager

Network Knowledge

Notify
Switch-Over
Reload

Actions

Note - EEM 1.0 originally developed to support Cisco IOS High Availability, but applicable to more general situation
Embedded Event Manager 1.0

• Introduced the following event detectors

  **SNMP**—The SNMP event detector allows a standard SNMP
  MIB object to be monitored and an event to be generated
  when the object matches specified values or crosses
  specified thresholds

  **Syslog**—The syslog event detector allows for screening
  syslog messages for a regular expression pattern match
Embedded Event Manager 1.0 (Cont.)

• Introduced the following actions

  Generate custom, prioritized syslog messages

  Generate a CNS event for upstream processing by Cisco CNS devices

  Reload the Cisco IOS Software

  Switch to a secondary processor in a fully redundant hardware configuration
Embedded Event Manager (EEM) 1.0
Variables

- Cisco defines read-only environment variables called built-in variables that are pre-set with a specific value when the event is triggered.

- Can be used in “msg” text.

- Environment variable available for all events
  - \$_event_type: The event type that triggered the event
  - \$_event_pub_time: The time at which the event type was published

- Environment variable available for SNMP events
  - \$_snmp_oid: The SNMP object OID that caused the event to be published
  - \$_snmp_oid_val: The SNMP object ID value when the event was published

- Environment variable available for Syslog events
  - \$_syslog_msg: The syslog message that caused the event to be published
Example:

When the primary RP runs low on memory (*ciscoMemoryPoolFree*), an event is triggered at which a certain the threshold is reached.

Then applet named memory-demo runs (2 actions):

1. Syslog message to be written to the console (variables)
2. Switch-over to the Secondary RP

Crash!
EEM 1.0 – SNMP ED Example (Cont.)

event manager applet memory-demo

  event snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type exact entry-op lt entry-val 5120000 poll-interval 10

  action 1.0 syslog priority critical msg "Memory exhausted; current available memory is $\_snmp\_oid\_val$ bytes"

  action 2.0 force-switchover

The following syslog messages are created:

00:08:31: %HA_EM-2-LOG: memory-demo:
Memory exhausted; current available memory is 4484196 bytes
00:08:31: %HA_EM-6-FMS_SWITCH_HARDWARE:
fh_io_msg: Policy has requested a hardware switchover

... a switch-over is forced
Embedded Event Manager (EEM) 2.x Architecture
Embedded Event Manager (EEM) 2.0

- Introduced the following event detectors

  **Application-Specific**—The Application-Specific ED allows any EEM policy to publish an event

  **Counter**—The Counter ED publishes an event when a named counter crosses a specified threshold

  **Interface Counter**—The Interface Counter ED publishes an event when a generic Cisco IOS Software interface counter for a specified interface crosses a defined threshold

  **Timer**—The Timer ED publishes events for the following four different types of timers; absolute-time-of-day, countdown, watchdog, and CRON

  **Watchdog**—The Cisco IOS Watchdog ED publishes an event when CPU or memory utilization for a Cisco IOS Software process crosses a threshold
• Added ability to
  - Publish an application-specific event
  - Modify of a named counter
  - Generate an SNMP trap
Embedded Event Manager (EEM) 2.1

- Added support for user written TCL-based policies
- Introduced the following event detectors:
  - **CLI**—The CLI ED screens Command-Line Interface (CLI) commands for a regular expression match
  - **None**—The None ED publishes an event when the Cisco IOS Software event manager run CLI command executes an EEM policy
  - **OIR**—The Online Insertion and Removal (OIR) ED publishes an event when a particular HW insertion or removal event occur
• Introduced the following actions
  Executing a Cisco IOS Command-Line Interface (CLI) command
  Requesting system information when an event occurs
  Sending a short e-mail
  Manually running an EEM policy
• Permits multiple concurrent policies to be run using the new event manager scheduler script command
Embedded Event Manager (EEM) 2.1.5

• Introduced the following event detectors

**GOLD** — The Generic Online Diagnostic (GOLD) ED publishes an event when a GOLD failure event is detected.

**Process** — The Process ED publishes an event when a Cisco IOS Software Modularity process starts or stops.

**System Manager** — The System Manager ED generates events for Cisco IOS Software Modularity process start, normal or abnormal stop, and restart events.

The events generated by the system manager allows policies to change the default behavior of the process restart.
Embedded Event Manager 2.1.5 (Cont.)

- Introduced the following event detectors (Cont.)
  
  **Watchdog** (Cisco IOS Software Modularity)—The Cisco IOS Software Modularity Watchdog System Monitor (WDSYSMON) ED detects infinite loops, deadlocks, and memory leaks in Cisco IOS Software Modularity processes

- Introduced the ability to display EEM reliability metric data for processes
Embedded Event Manager (EEM) 2.2

• Introduced the following ED’s:

  **Enhanced Object Tracking (EOT)**—The EOT ED publishes an event when the tracked object changes

  **Resource** —The Resource ED publishes an event when the **Embedded Resource Manager (ERM)** Introduced in Release 12.3(14)T, reports an event for the specified policy

  **RF** —The Redundancy Framework ED publishes an event when one or more RF events occur during synchronization in a dual Route Processor (RP) system; The RF event detector can also detect an event when a dual RP system continuously switches from one RP to another RP (referred to as a ping-pong situation)
Embedded Event Manager 2.2 (Cont.)

- Introduced the following actions
  - Reading the state of a tracked object
  - Setting the state of a tracked object
Cisco IOS Watchdog ED Example (EEM v2.1)

- Monitor the IP SNMP process every 10 seconds: if CPU exceeds 50%, publish an application-specific event on the well-known user subsystem, and send an SNMP trap

```plaintext
event manager applet IPSNMPWD
  event ioswdsysmon sub1 cpu-proc taskname "SNMP ENGINE" op ge val 50 period 10
  action 1.0 publish-event sub-system 798 type 1 arg1 "IP SNMP" arg2 "50"
  action 2.0 snmp-trap intdata1 50 strdata "IP SNMP Process above 50% within 10 seconds"

snmp-server enable traps event-manager
```
EEM Policies and TCL

- EEM policies can be written in TCL
- TCL can do everything that applets can do, and more!
- TCL permits global variables—called environment variables—to be defined for use within an EEM policy
  - User-defined
  - Cisco-defined for a specific sample policy
  - Cisco system-defined
- Cisco provides built-in TCL namespaces and libraries to facilitate in creating EEM policies
EEM Policies and TCL (Cont.)

• Policies should be arranged in the following format
  
  Event register keyword **REQUIRED**
  
  Environment “must defines”
  
  Namespace import
  
  Entry status
  
  Body **REQUIRED**
  
  Exit status
Enabling Policies

Router#mkdir disk0:/policies
Router#copy tftp://10.10.10.10/syslog_policy.tcl disk0:/policies/syslog_policy.tcl

event manager directory user policy disk0:/policies
event manager policy syslog_policy.tcl type user

• Three system policies come with Cisco IOS Software
  
  sl_intf_down.tcl—run CLI and send email on reception of a configurable syslog message
  
  tm_cli_cmd.tcl—run CLI and send email at a certain time
  
  tm_crash_reporter.tcl—triggers 5 sec after bootup and sends crashinfo, if relevant, to the specified URL

• Can be enabled using the following commands
  
  event manager policy sl_intf_down.tcl
  event manager policy tm_cli_cmd.tcl
  event manager policy tm_crash_reporter.tcl
EEM Environment Variables

• Variables that are referenced within policies that can be set in the config
• Can be used to customize policies
• Example:

```
Router#config t
Router(config)#event manager environment _email_server
email.cisco.com
Router(config)#event manager environment _email_from
soandso@somecompany.com
```

Note: environment variable names that start with the underscore character are reserved for Cisco use only
EEM Namespaces

::cisco::lib Namespace

<table>
<thead>
<tr>
<th>Library</th>
<th>Procedure</th>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMTP Library</td>
<td>smtp_subst</td>
<td>email_template_file</td>
<td>Substitute global email variables in a template file</td>
</tr>
<tr>
<td></td>
<td>smtp_send_email</td>
<td>email_text</td>
<td>Send email</td>
</tr>
</tbody>
</table>

Required Variables for Email Templates

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>_email_server</td>
<td>A Simple Mail Transfer Protocol (SMTP) server</td>
<td>smtp.mydomain.com</td>
</tr>
<tr>
<td>_email_from</td>
<td>Address from which mail is sent</td>
<td><a href="mailto:admin@mydomain.com">admin@mydomain.com</a></td>
</tr>
<tr>
<td>_email_cc (optional)</td>
<td>Address to which email is copied</td>
<td><a href="mailto:mgr@mydomain.com">mgr@mydomain.com</a></td>
</tr>
<tr>
<td>_email_to</td>
<td>The email recipient address</td>
<td><a href="mailto:user@mydomain.com">user@mydomain.com</a></td>
</tr>
</tbody>
</table>
EEM Namespaces (Cont.)

::cisco::eem Namespace Common Procedures

<table>
<thead>
<tr>
<th>Procedure</th>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>event_register_xxx</td>
<td>Variable</td>
<td>Register a specific type of event (ie: CLI, Syslog, OIR, SNMP, etc.)</td>
</tr>
<tr>
<td>event_reqinfo</td>
<td>None</td>
<td>Event-specific info</td>
</tr>
<tr>
<td>event_publish</td>
<td>comp_id id type type [arg1 arg] [arg2 arg] [arg3 arg] [arg4 arg]</td>
<td>Publish a subsystem-specific event (comp_id of 0x031e if reserved for user policies)</td>
</tr>
<tr>
<td>action_snmp_trap</td>
<td>[intdata1 data] [intdata2 data] [strdata data]</td>
<td>Send an event manager trap with the specified data</td>
</tr>
<tr>
<td>action_syslog</td>
<td>priority prio msg text</td>
<td>Send a syslog message with a specific severity and message body</td>
</tr>
</tbody>
</table>

- Many more procedures are available

  [Cisco Link]
::cisco::eem::event_register_syslog pattern ".*UPDOWN.*Serial0/0.* changed state to down"

# We don't require any global variables to be set.

namespace import ::cisco::eem::*
namespace import ::cisco::lib::*

array set arr_einfo [event_reqinfo]

if { $_cerrno != 0 } {
    set result [format "component=%s; subsys err=%s; posix err=%s:\n%s" 
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str] 
    error $result 
}

global slg_msg
set slg_msg $arr_einfo(msg)
set newmsg [format "Primary uplink has gone down: %s" $slg_msg]

action_syslog priority emerg msg $newmsg

if { $_cerrno != 0 } {
    set result [format "component=%s; subsys err=%s; posix err=%s:\n%s" 
        $_cerr_sub_num $_cerr_sub_err $_cerr_posix_err $_cerr_str] 
    error $result 
}
Embedded Event Manager (EEM) Debugging and Show Commands

• **Debug commands**

  ```
  debug event manager tcl cli_lib
  debug event manager tcl commands
  debug event manager tcl smtp_library
  ```

  Note: EEM delivers debug to syslog at the “debugging level”

• **Show commands**

  ```
  show event manager policy available
  show event manager directory user policy
  ```

• **User policies are run in Safe-Tcl which restricts certain commands**

• **USER POLICIES MUST BEGIN WITH AN event_register_XXX LINE!**
# Embedded Event Manager (EEM) Version Comparison

<table>
<thead>
<tr>
<th>Feature</th>
<th>EEM 1.0</th>
<th>EEM 2.0</th>
<th>EEM 2.1</th>
<th>EEM 2.1.5</th>
<th>EEM 2.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syslog, SNMP EDs</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Watchdog, Counter, Interface Counter, Timer, Application-Specific EDs</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>OIR, CLI EDs</td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Syslog, SNMP Actions</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Counter Modification, System Info, Email Actions</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>User and System TCL Policies</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>GOLD, System Manager, WDSysMon EDs</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Resource, RF, EOT EDs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

*Note: X indicates the feature is included in the version.*
EEM Built-in Actions

- An Embedded Event Manager Policy can:
  - Execute an IOS CLI command and receive the result
  - Send a CNS event
  - Increment or decrement an EEM counter
  - Force a switchover to the standby in a redundant configuration
  - Request system information
  - Send an e-mail
EEM Built-in Actions (Cont.)

- An Embedded Event Manager Policy can (cont.):
  - Cause another EEM policy to be executed
  - Publish an application specific EEM event
  - Reload the box
  - Send an SNMP trap with custom data
  - Log a message to Syslog
Embedded Event Manager (EEM)

Additional References

- Embedded Event Manager 1.0 guide:
  

- Embedded Event Manager 2.0 guide:
  

- Embedded Event Manager 2.1 guide:
  
Embedded Event Manager (EEM) Additional References (Cont.)

- Embedded Event Manager 2.2 guide:

- Writing Embedded Event Manager policies:
Embedded Syslog Manager
Embedded Syslog Manager (ESM)
Introduction

- Customizable framework for correlating, augmenting, filtering, routing Cisco IOS Software logger output
- Does not replace UDP logger (Syslog classic), but operates in parallel
Classic Syslog Implementation

- Configure four targets
- Configure facility, severity thresholds
Embedded Syslog Manager (ESM)
Why ESM?

• On-box intelligence for local event correlation
• Severity escalation
• Syslog message routing/distribution
• Alternate reliable transport/persistence
• Custom message formats/tagging
Embedded Syslog Manager (ESM) Design

- Post-process syslog messages with selected ESM filters (proactive rules-based analysis)
- User definable scripting (TCL)
- Available in images with TCL 8.3.4, in Releases 12.3(2)T, 12.2(25)S
- Syslog collector message stream can be divided into separate streams
ESM Filters

- TCL Scripts – located locally or remotely
- Pre-compiled or plain text
- Processed serially
- Configured, (re)ordered, and (re)loaded via CLI
- Arguments passed via CLI or filters may be edited directly
• Filters are passed all syslog message data elements as TCL global variables, including the original formatted message
• Filters operate on message and return desired message
• Filters can optionally change the “stream” variable to route the message to specific syslog servers
• Filters can optionally change the “severity” variable
• Filters can send messages directly to the output queue or down the filter chain
• ESM Filters can query status via CLI interface
• Where:
  
  `<URL>` is a IOS path to an ESM filter TCL script
  
  `[position]` is an optional order number
  
  (if multiple filters are defined)
  
  `[args]` are optional command line arguments to pass to the
  filter script

```
Router(config)#logging filter `<URL>` [position] [args args]
```

```
Router(config)#logging [console|monitor|buffer] filtered
Router(config)#logging host `<ip_address>` filtered
[stream_id]
```

`[stream_id]` can be set in the filter script to route certain
events to certain destinations
Embedded Syslog Manager (ESM) Example

- **Severity escalation**: messages that Cisco deemed low priority may be very important to some users
- Example: escalate syslog messages that contain the word ‘CONFIG_I’ to severity level of four (they are by default level five)

Router(config)# logging filter flash:/ABCTCL/escalate.tcl CONFIG_I 4
Router(config)# logging console filtered

Router#
*Nov 18 13:44:26.410: %SYS-4-CONFIG_I: Configured from console by console
Router#
# Embedded Syslog Manager, Severity Escalation Module
# ===================================================================
# Usage: Set CLI Args to "mnemonic new_severity"
# Namespace: global
# Check for null message

if { [string length $::orig_msg] == 0} {
    return ""
}

if { [info exists ::cli_args] } {
    set args [split $::cli_args]
    if { [ string compare -nocase [lindex $args 0] $::mnemonic ] == 0 } {
        set ::severity [lindex $args 1]
        set sev_index [ string first [lindex $args 0] $::orig_msg ]
        if { $sev_index >= 2 } {
            incr sev_index -2
            return [string replace $::orig_msg $sev_index $sev_index \[lindex $args 1]]
        }
    }
}

return $::orig_msg
• **Message routing**: categorize messages using criteria other than facility or severity
  
  Example: send all spanning tree messages to a separate syslog server

• **SMTP-based email alerts**: capability for notifications using TCP to external servers, such as TCP-based syslog collectors or Simple Mail Transfer Protocol (SMTP) servers
  
  Example: “configuration changes” sent to administrators via an email message

• **Your example… the possibilities are endless!**
Q: How do I tell if my image contains ESM?

A: From the CLI, type “show log”. The output will contain the status of the filter modules:

Router# show log
Syslog logging: enabled (10 messages dropped, 1 messages rate-limited, 0 flushes, 0 overruns, xml disabled, filtering disabled)
  Console logging: level debugging, 12 messages logged, xml disabled, filtering disabled
  Monitor logging: level debugging, 0 messages logged, xml disabled, filtering disabled
  Buffer logging: level debugging, 12 messages logged, xml disabled, filtering disabled
Logging Exception size (8192 bytes)
Count and timestamp logging messages: disabled

No active filter modules.
Router#show log
Syslog logging: enabled (10 messages dropped, 1 messages rate-limited, 0 flushes, 0 overruns, xml disabled, filtering enabled)
   Console logging: level debugging, 48 messages logged, xml disabled,
   filtering enabled
   Monitor logging: level debugging, 0 messages logged, xml disabled,
   filtering disabled
   Buffer logging: level debugging, 67 messages logged, xml disabled,
   filtering disabled
   Logging Exception size (8192 bytes)
   Count and timestamp logging messages: disabled

Filter modules:
   flash:/ABCTCL/escalate.tcl args CONFIG_I 4
   flash:escalate.tcl   - INVALID args CONFIG_I 4   - INVALID
Embedded Syslog Manager (ESM) Caveats

- No way to debug ESM filter scripts as they run
- ESM filters cannot be applied to SNMP history logs (i.e., filters will not be applied to messages logged from logging history or snmp-server enable traps syslog)
- All filters must be written in TCL
- Additional Reference:
  
Embedded Resource Manager
Embedded Resource Manager (ERM)

• Monitors system resource usage to better understand scalability needs by allowing you to configure threshold values for resources such as CPU, buffer, and memory

• The ERM framework provides a mechanism to send notifications whenever the specified threshold values are violated by any resource user

  Helps in reducing the CPU, buffer, and memory utilization issues
Embedded Resource Manager (Cont.)

- Introduced in Release 12.3(14)T
- Embedded Resource Manager guide:
  
ERM Concepts

• **Resource User (RU)**
  
  Entity or application that consumes one or more resources
  
  ie: BGP process

• **Resource Owner (RO)**
  
  Entity that allocates its resources to a RU
  
  ie: CPU, memory, buffer
ERM Concepts (Cont.)

• Notifications sent and actions taken
  
  RU registers with RO for threshold notifications
  
  RU is expected to change utilization of resource upon notification
  
  ie: if a process’s use of CPU exceeds a threshold, that process is expected to take action to limit use of the CPU
ERM Architecture

Diagram:

- Customer view
  - CLI
  - Common configuration and display mechanism for all resources
  - ERM framework
    - Common resource owner interface
      - Buffer
      - CPU
      - MEM
    - Resource Owners

- Common RU interface
  - BGP
  - Resource user

Diagram 127549
Types of Thresholds

• **System Global**
  All RU’s are notified when total resource utilization crosses a specified threshold value

• **User Local**
  A specific RU is notified when the resource utilization of that RU crosses a specified threshold value

• **Per User Global**
  A specific RU is notified when total resource utilization crosses a specified threshold value
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#resource policy
Router(config-erm)#policy policy1 type iosprocess
Router(config-erm-policy)#system
Router(config-policy-node)#buffer public
Router(config-owner-buffer)#critical rising 90 interval 12
falling 20 interval 10 global
Router(config-owner-buffer)#major rising 70 interval 12
falling 15 interval 10 global
Router(config-owner-buffer)#minor rising 60 interval 12
falling 10 interval 10 global

• If TOTAL buffer usage count rises above 90% at an interval of 12s, a critical Up notification is sent to the iosprocess RU
• If TOTAL buffer usage falls below 20% at an interval of 10s, a critical Down notification is sent to the iosprocess RU
Router#debug resource policy notification

When a threshold is violated:

*Mar  3 09:50:44.081: Owner: 'memory' initiated a notification:
*Mar  3 09:50:44.081: %SYS-4-RESMEMEXCEED: Resource user usrr1 has exceeded the Major memory threshold Pool: Processor Used: 42932864 Threshold :42932860
*Mar  3 09:50:46.081: Notification from Owner: 'memory' is dispatched for User: 'usrr1' (ID: 0x10000B9)
*Mar  3 09:50:46.081: %SYS-4-RESMEMEXCEED: Resource user usrr1 has exceeded the Major memory threshold Pool: Processor Used: 42932864 Threshold :42932860
Command Scheduler
Command Scheduler (Kron) Overview

• Allows EXEC commands to run periodically or at a specified time
• First introduced in Release 12.3(1)
• Runs commands in a fully-automated mode
• Interactive commands (ie: reload) are NOT supported
• Kron command scheduler guide:
• Configure a Kron policy to write the output of show interface to a TFTP server

```plaintext
kron policy-list writeshowint
cli show interface | redirect tftp://10.1.1.1/router.showint

kron occurrence showint-occur at 21:40 recurring
policy-list writeshowint
```

Note: a single occurrence can have multiple Policy-Lists
Kron Command Scheduler
Debugging and Show Commands

• **Debug commands**
  
  - `debug kron all` — show all kron debugging
  - `debug kron exec-cli` — debug CLI processing
  - `debug kron info` — show warnings and progress info
  - `debug kron major` — show all major Kron failures

• **Show commands**
  
  - `show kron schedule`
Kron Command Scheduler
Debugging and Show Commands (Cont.)

Sample Debug Output

Apr 12 01:54:07.479: Major 1, Minor 0
Apr 12 01:54:07.479: Timer Event showint-occur
Apr 12 01:54:07.479: Call parse_cmd 'show interface | redirect tftp://10.1.1.1/router.showint'
Apr 12 01:54:07.559: Kron CLI return 0

**CLI 'show interface | redirect tftp://10.1.1.1/router.showint':**

Apr 12 01:54:07.559: Major 4, Minor 7
Apr 12 01:54:07.559: Respond to end of CLI Process

Sample Show Command Output

Router#show kron schedule
Kron Occurrence Schedule
showint-occur inactive, will run again in 0 days 23:39:10 at 21:40 on

Note: One-Shot Policies will be removed from the config and the show Kron Schedule output after they run
Kron Command Scheduler
Caveats

- Interactive commands are not supported, and will fail at execution time
- NTP must be configured or the router clock must be authoritative
- Kron and TCL can run together since Release 12.4(4)T
Configuration Replace and Rollback
Configuration Replace and Rollback Overview

• Provides ability to replace current running config with a saved complete config
• Config rollback provides a way of replacing the current running config with any configuration file
• Hooks exist for comparing configs, and viewing context-sensitive diffs
• Questionable configuration changes can be evaluated and automatically backed out
• Rollbacks are done efficiently and safely by only reapplying commands that have changed
• Reference:

Configuration Replace and Rollback Availability

• Config replace and rollback was first introduced in Release 12.3(7)T

• The features were later integrated into Release 12.2(25)S

• Configuration locking support was integrated into Releases 12.3(14)T and 12.2(25)S
Configuration Archive
Configuration

Ad-hoc snapshots can also be taken

Router#archive config
Router#show archive
There are currently 3 archive configurations saved.
The next archive file will be named disk0:config-archive-3

<table>
<thead>
<tr>
<th>Archive #</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>disk0:config-archive-1</td>
</tr>
<tr>
<td>2</td>
<td>disk0:config-archive-2 &lt;- Most Recent</td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
</tr>
</tbody>
</table>
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#snmp-server community public ro
Router(config)#snmp-server community private rw
Router(config)#end
Router#config replace disk0:config-archive-1
This will apply all necessary additions and deletions to replace the current running configuration with the contents of the specified configuration file, which is assumed to be a complete configuration, not a partial configuration. Enter Y if you are sure you want to proceed. ? [no]: yes
Total number of passes: 0
Rollback Done
Configuration Rollback

Configuration Locking

- Starting in Release 12.3(14)T and 12.2(25)S the running config is locked during a rollback.
- No other changes can be made to the running configuration during this time.
- Use the no-lock argument to config replace to override this behavior.

```
Router#show config lock
Parser Configure Lock
---------------------
Owner PID    : 40
User         : mpalmero
TTY          : 2
Type         : EXCLUSIVE
State        : LOCKED
Class        : ROLLBACK
Count        : 1
Pending Requests : 0
User debug info : Rollback
```
Configuration Replace and Rollback Show and Debug Commands

- Viewing configurations in the archive
  show archive

- Debugging config archive operations
  debug archive versioning — debug all config archive operations
  debug archive config timestamp — show times and config sizes at all steps of a rollback

- Clearing configuration locks
  clear configuration lock
Sample Debugging Output

- Archive path pointing to misconfigured TFTP server

```plaintext
Router#archive config
TFTP: error code 1 received - 18025

Apr 23 21:26:16.114: backup_running_config
Apr 23 21:26:16.114: Current = 1
Apr 23 21:26:16.114: Writing backup file tftp://10.10.10.10/router-config-1
Apr 23 21:26:18.434: backup failed
```

- Archive path pointing to a correctly configured TFTP server

```plaintext
Router#archive config
!!!!
Router#
Apr 23 21:37:54.811: backup_running_config
Apr 23 21:37:54.811: Current = 1
Apr 23 21:37:54.811: Writing backup file tftp://10.10.10.10/router-config-1
Apr 23 21:37:56.059: backup worked
```
Configuration Replace and Rollback Caveats

- Free memory must be large enough to hold the current running configuration as well as the replacement configuration.
- Physical interface statements cannot be removed from a running configuration.
- Certain Cisco IOS Software configuration commands cannot be fully removed unless the router is reloaded.
- Archiving configurations via TFTP is a security risk; opt for FTP or RCP instead.
Contextual Configuration Diff Utility
Contextual Configuration Diff Utility

- View line-by-line comparison of two configuration revisions
- Compare order-sensitive data such as ACLs
- Output lists configuration lines that have been added, removed, or modified
- First introduced in Release 12.3(4)T and integrated into Release 12.2(25)S
- Reference:
  
Contextual Configuration Diffs
Example — Startup vs. Running Config

Router#show archive config differences nvram:startup-config system:running-config
Contextual Config Diffs:
+ip http server
+tacacs-server host 172.18.123.33
+tacacs-server directed-request
-no ip http server
-logging 192.168.0.254

Lines in running-config Not in startup-config

Lines in startup-config Not in running config
Agenda

• Introduction
• Embedded Management Tools
• Enhanced Device Interface
• Practical Applications
• Summary and Conclusion
Enhanced Device Interface (E-DI) Overview

• E-DI is
  An extension to the network device’s interface

• E-DI provides
  Enhanced Command Line Interface (CLI) to human users
  XML Programmatic Interface to management applications
  Scripting interface and platform for scripting applications

Interactive CLI Users
Scripting Applications
Management Applications

Various types of network devices

Syntax knowledge base of supported devices

NETCONF over SSH
E-DI as Mediation Gateway (Runs on Linux Based Appliance)
SNMP, Telnet/SSH, Syslog
Why E-DI?

- **Improve productivity** for:
  - Device configuration
  - Maintenance & troubleshooting
  - CatOS to IOS upgrades (planned)

- **Single point of access** to device configuration

- **Unified interface** across platforms & releases:
  - enhanced CLI
  - Perl integration
  - XML API
  (IETF NETCONF draft 5 compliant)

- Complementary to EMS and NMS

- Support existing and new Cisco devices
Design Approach

• Build and maintain a device command knowledge base – automatically learned from the device
• Maintain a list of managed devices and minimal inventory necessary for operations
• Provide a comprehensive configuration data model for supported devices
• For More Information:
  
## Enhanced Device Interface (E-DI) Device Support Packages

<table>
<thead>
<tr>
<th>Device Package</th>
<th>Device Package Version</th>
<th>Supported Software Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco 12000</td>
<td>1.1</td>
<td>Release 12.0(27)S5</td>
</tr>
<tr>
<td>Cisco 7600</td>
<td>1.1</td>
<td>Release 12.2(18)SXD4</td>
</tr>
<tr>
<td>Catalyst 6500</td>
<td>1.2</td>
<td>Releases 12.1(11b)E1, 12.2(17d)SXB6</td>
</tr>
<tr>
<td>Cisco 1700</td>
<td>1.2</td>
<td>Releases 12.2(15)T14, 12.3(8)T6</td>
</tr>
<tr>
<td>Cat 6500 Cat OS</td>
<td>1.1</td>
<td>7.6(6)</td>
</tr>
<tr>
<td>Cat 3550</td>
<td>1.2</td>
<td>Releases 12.1(14)EA1a, 12.1(22)EA2</td>
</tr>
<tr>
<td>Cat 4000</td>
<td>1.1</td>
<td>Release 12.1(19)EW1</td>
</tr>
<tr>
<td>Cisco 7200</td>
<td>1.1</td>
<td>Release 12.2(13)T14</td>
</tr>
<tr>
<td>Cat 2950</td>
<td>1.1</td>
<td>Release 12.1(13)EA1c</td>
</tr>
<tr>
<td>Cisco 2600</td>
<td>1.2</td>
<td>Releases 12.1(17), 12.2(24a)</td>
</tr>
<tr>
<td>Cisco 800</td>
<td>1.1</td>
<td>Release 12.3(8)T7</td>
</tr>
<tr>
<td>Cisco AP350 IOS</td>
<td>1.1</td>
<td>Release 12.3(2)JA2</td>
</tr>
<tr>
<td>Cat 3750</td>
<td>1.0</td>
<td>Release 12.1(19)EA1a</td>
</tr>
<tr>
<td>Cisco 3800</td>
<td>1.1</td>
<td>Release 12.3 (11)T</td>
</tr>
<tr>
<td>Cisco 1800</td>
<td>1.1</td>
<td>Release 12.3 (11)T</td>
</tr>
<tr>
<td>Cisco 3700</td>
<td>1.1</td>
<td>Release 12.3 (6)C</td>
</tr>
</tbody>
</table>
## Enhanced Device Interface (E-DI)
### Device Support Packages (Cont.)

<table>
<thead>
<tr>
<th>Device Package</th>
<th>Device Package Version</th>
<th>Supported Software Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco 2800</td>
<td>1.1</td>
<td>Release 12.3(11)T7</td>
</tr>
<tr>
<td>IAD 2400</td>
<td>1.1</td>
<td>Release 12.3 (8)T9</td>
</tr>
<tr>
<td>Cisco 2600</td>
<td>1.1</td>
<td>Release 12.3(10e)</td>
</tr>
<tr>
<td>Catalyst 6500</td>
<td>1.4</td>
<td>CatOS Version: 8.4</td>
</tr>
<tr>
<td>PIX Device</td>
<td>1.4</td>
<td>PIX OS Version 7.0(4)</td>
</tr>
<tr>
<td>Cat 6500 with CSM module</td>
<td>1.3</td>
<td>Release 12.2.(18)SXE3, IOS Version: Release 12.2.(18)SXF (With FSM Card)</td>
</tr>
</tbody>
</table>
Example: Network Virtualization

Access Point
- RAM
- Flash0

Switch
- RAM
- Flash0

Router
- RAM
- Flash0

Enhanced Device Interface – E-DI
- Configuration
- Exec Commands
- File System
- Events / Status
- SW Image
- Grouping
- Scheduling
- Authentication
- Admin
- Inventory

Engineer
Perl Script
NMS Application
Example: Network Virtualization

Access Point
- RAM
- Flash0

Radio-0
Radio-1
Eth-0

(A)

Switch
- RAM
- Flash0

Eth-0/0
Eth-0/1

(B)

Router
- RAM
- Flash0

Radio-0
Eth-0/0
Eth-0/1

(C)

Interfaces
- RAM
- Flash

- Radio-A0
- Radio-C0
- Eth-A0
- Eth-B0/0
- Eth-B0/1
- Eth-C0/0
- Eth-C0/1

E - CLI
Perl API
Netconf XML

Enhanced Device Interface – E-DI

Engineer
Perl Script
NMS Application
Main Features

• OS Parser Emulation
  CLI syntax checking & command context validation eliminates human errors

• Device Grouping
  Definition of administrative domains
  Access control per administrative domain level
  Simplifies configuration and administrative tasks

• Group least common denominator CLI
  Perform group operations without risk of generating unsupported command

• Context – Based CLI
  Simultaneously apply changes to one or more devices by selecting the context
  Single point for network configuration
Cisco IOS—Like CLI

• Real-time syntax validation and visual feedback
Programmatic Interface
Problem Definition

Customers expect structured, published and consistent interface

Management Applications

Un-published, un-structured, varying data models

Configuration (CLI) model-1
Operational (CLI) model-1
Events/alarms (Syslog) model-1
Configuration (CLI) model-2
Operational (CLI) model-2
Events/alarms (Syslog) model-2

Network Element-1
Network Element-2

Resulting in

• Longer analysis and design phase
• Longer implementation and validation phases
• Defects may be found towards the tail-end of the project or in the field
Programmatic Interface Use Case
How E-DI Helps?

Customers expect structured, published and consistent interface

Management Applications

Published, structured but varying configuration data models
(Operational models are targeted for future releases)

Cisco E-DI

Un-published, Un-structured, Varying Data Models

Network Element-1

Network Element-2
What Is in E-DI Generated Device Data Model?

- E-DI publishes data models as a collection of XML Schema Definition (XSD) files
- Structured model
- Named and hierarchical elements for predictability in parsing
  - Namespace to identify variations in data models
- Publishes
  - Data type (integers, strings, IP addresses, MAC addresses etc.)
  - Cardinality (min/max occurrences)
  - Constraints (ex: ranges)
  - Key fields (naming)
  - Order (sequence, choice)
  - Identification of negation logic

Diagram:
```
main.xsd
Cat3550/12.1(14)EA1a/1.2/
  └── Global Config
  └── Fast Ethernet
    └── SubscriberPolicy
    └── Other ....
```
Agenda

• Introduction
• Embedded Management Tools
• Enhanced Device Interface
• Practical Applications
• Summary and Conclusion
Problem

Editing Access-Lists on routers is time consuming and prone to errors
Working with Access-Lists

Editing Access-Lists the long way

1. Fetch Router Configuration
2. Edit ACL
3. Redeploy ACL
Working with Access-Lists

• Solution

Write an Access-List Editor in TCL that can run directly on the router
Working with Access-Lists

Using TCL to Edit the Access-List

```
2. Add access-list entry
3. Remove access-list entry
4. Quit and save changes
5. Quit without saving changes

Enter option: 1

1: access-list 113 permit udp any any eq isakmp
2: access-list 113 permit tcp any any eq 22
3: access-list 113 deny ip any any log
4: access-list 113 deny tcp any host 10.1.1.1 eq www
5: access-list 113 permit tcp any 0.0.0.0 255.255.255.0 eq www

Hit enter to continue...

Access-list Editor

Enter option: 
```
Router(tcl)#edit_acl 113
Access-list Editor
------------------
1. View access-list 113
2. Add access-list entry
3. Remove access-list entry
4. Quit and save changes
5. Quit without saving changes

Enter option: 1

1: access-list 113 permit udp any any eq isakmp
2: access-list 113 permit tcp any any eq 22
3: access-list 113 deny ip any any log
4: access-list 113 deny tcp any host 10.1.1.1 eq www
5: access-list 113 permit tcp any 0.0.0.0 255.255.255.0 eq www
TCL Access-List Editor

Adding an Access-List entry

Access-list Editor

1. View access-list 113
2. Add access-list entry
3. Remove access-list entry
4. Quit and save changes
5. Quit without saving changes

Enter option: 2

1: access-list 113 permit udp any any eq isakmp
2: access-list 113 permit tcp any any eq 22
3: access-list 113 deny ip any any log
4: access-list 113 deny tcp any host 10.1.1.1 eq www
5: access-list 113 permit tcp any 0.0.0.0 255.255.255.0 eq www

Insert before which line number: 3
Enter body of ACL rule to insert (without the access-list 113 portion):
permit tcp any any eq 80
Removing an Access-List Entry

Access-list Editor
-------------------

1. View access-list 113
2. Add access-list entry
3. Remove access-list entry
4. Quit and save changes
5. Quit without saving changes

Enter option: 3

1: access-list 113 permit udp any any eq isakmp
2: access-list 113 permit tcp any any eq 22
3: access-list 113 permit tcp any any eq 80
4: access-list 113 deny ip any any log
5: access-list 113 deny tcp any host 10.1.1.1 eq www
6: access-list 113 permit tcp any 0.0.0.0 255.255.255.0 eq www

Enter ACL entry number to delete: 1
Access-list Editor
------------------
1. View access-list 113
2. Add access-list entry
3. Remove access-list entry
4. Quit and save changes
5. Quit without saving changes

Enter option: 4
Access-list 113 was committed successfully.

Router(tcl)#show run | include ^access-list 113
access-list 113 permit tcp any any eq 22
access-list 113 permit tcp any any eq www
access-list 113 deny ip any any log
access-list 113 deny tcp any host 10.1.1.1 eq www
access-list 113 permit tcp any 0.0.0.0 255.255.255.0 eq www
Deploying Security Fixes
Exercise 2

• Problem

A New PSIRT Advisory has come out, and a workaround needs to be deployed and maintained on all of the routers
Deploying Security Fixes

• **SNMP Message Handling Vulnerability**

  The Cisco IOS “IP SNMP” process incorrectly attempts to process solicited SNMP requests on UDP port 162 as well as a random, high UDP port.

  A successful exploit of this vulnerability will result in a router reload.

  To workaround this problem, control-plane policing can be done to block SNMP requests without impacting packet-switching performance.

  To do this effectively, the randomly chosen high UDP ports must first be determined on each router.
Deploying Security Fixes

Doing Things the Long Way

1. Determine the high UDP Port using show IP sockets
2. Create the control plane policing config
3. Deploy the New config commands

But, what if the router reloads…?
Deploying Security Fixes

• **Solution**

  Write a TCL Script to dynamically determine the High UDP port

  Automatically add the necessary control-plane policing commands

  Use Cisco IOS Software and do config command to make sure the script is run every time the router reloads
TCL SNMP Security Fix Script

• Running the script at Boot Time

  Copy the router’s startup configuration to a server, and add the following towards the end of the file

    do tclsh tftp://10.1.1.1/snmp_fix.tcl

  Copy the edited file directly into the router’s NVRAM

    copy tftp://10.1.1.1/router.cfg startup

  The router will load the script from the TFTP server each time it boots

    If the network takes a long time to converge, it might be better to copy the script to each router’s flash
Archiving Configurations
Example 3

• Problem

  Router configurations must be archived periodically to router flash as well as a network server for disaster recovery

  This process is time consuming and sometimes it is forgotten to be done
Archiving Configurations

• Backup the Config to Flash and to FTP

  Copy the current running configuration to flash

  `copy runn disk0:/config.bak`

  Archive the configuration on an anonymous FTP server

  `copy runn ftp://172.18.123.33/confgs/router.cfg`
Archiving Configurations

Doing Things the Long Way

```plaintext
copy runn disk0:/config.bak

copy runn ftp://172.18.123.33/router.cfg.20050429
```
Archiving Configurations

• Solution

Use Cisco IOS Software built-in configuration archive feature to backup the running config to flash

Next schedule a Kron policy to copy the current running configuration to the FTP server
Config Archive Configuration

• Archive the running configuration to flash every 1440 Minutes (ie: every day)

```plaintext
archive
path disk0:/config-archive
maximum 14
time-period 1440
```
FTP Kron Policy

Run the TCL Script to transfer the latest archive config to the anonymous FTP server every day

```bash
kron occurrence ftpconfig_occur in 1:0:0 recurring
policy-list ftpconfig
!
kron policy-list ftpconfig cli copy running-config ftp://172.18.123.33/configs/router.cfg
!
```
Archiving Configurations

Verifying the Router Configuration

Router#show archive
There are currently 4 archive configurations saved.
The next archive file will be named disk0:/config-archive-4

<table>
<thead>
<tr>
<th>Archive #</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>disk0:/config-archive-1</td>
</tr>
<tr>
<td>2</td>
<td>disk0:/config-archive-2</td>
</tr>
<tr>
<td>3</td>
<td>disk0:/config-archive-3 &lt;- Most Recent</td>
</tr>
</tbody>
</table>

...Router#show kron schedule
Kron Occurrence Schedule
ftpconfig_occur inactive, will run again in 0 days 23:54:17
Archiving Configurations

Verifying the FTP Server

```
file-server# cd /nms/ftp/configs
file-server# ls -l router.cfg
-rw-r----- 1 ftp ftp 6333 Apr 26 17:10 router.cfg
```
Applying Configuration Changes
Exercise 4

• Problem

Applying configuration changes to remote routers can cause lockouts.

When console access is not available, someone needs to reload these devices.
Applying Configuration Changes

• Applying a Change to a Remote Router

  Applying a configuration change such as an access-list could result in being locked out of the router

  If console access is available, configuration changes can be deployed using the console to avoid network problems

  If no out-of-band access is available, then a reload is necessary to undo the problematic config change
Applying Configuration Changes

Doing Things the Hard Way

Access-List Change

Now the Router must be reloaded
Applying Configuration Changes

• **Solution**
  
  Use Config Rollback to automatically backout the config change after a certain amount of time
Replace the running configuration with the latest good archive after two minutes unless the change being made is confirmed.

Router#show archive
There are currently 4 archive configurations saved.
The next archive file will be named disk0:/config-archive-4

<table>
<thead>
<tr>
<th>Archive #</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>disk0:/config-archive-1</td>
</tr>
<tr>
<td>2</td>
<td>disk0:/config-archive-2</td>
</tr>
</tbody>
</table>
| 3         | disk0:/config-archive-3       | <- Most Recent

Router#config replace disk0:/config-archive-3 time 120
Using Config Rollback

Apply the Potentially Problematic Configuration Change

Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int pos4/0
Router(config-if)#ip access-group 113 in
Using Config Rollback

• If the configuration was successful, apply the changes
  
  ```
  Router#config confirm
  ```

• If the config changes caused the user to be locked out, the router will automatically revert to the last saved archive configuration after two minutes, and connectivity will be restored
Agenda

• Introduction
• Embedded Management Tools
• Enhanced Device Interface
• Practical Applications
• Summary and Conclusion
Summary

• **TCL** can be used to build custom commands, automate device configuration and create other embedded tools

• **EEM** provides a way for the router to monitor itself for potential problems, and act accordingly

• **ESM** can filter and prioritize critical log messages

• **ERM** provides a way to monitor resource usage and set limits
• **Command Scheduler** is useful for scheduling automated tasks

• **Configuration Replace and Rollback** provides configuration archival and configuration replacement and rollback

• **Contextual Configuration Diff Utility** enables line-by-line comparison of two configuration versions

• **E-DI** provides an XML programmatic interface to Cisco devices as well as an enhanced CLI
Conclusion

• Cisco IOS Software has a lot of very powerful embedded tools

• Use the scripts and examples to build rich, customized tools that work for YOU
Appendix
TCL Access-List Editor

The Code

```tcl
proc get_acl { acl } {
  set command "show running-config | include ^access-list $acl"
  return [exec $command]
}
```

Extract the desired Access-List from the running configuration
proc paginate { l } {
    set i 0
    while { $i < [llength $l] } {
        set new_page 1
        for { } { $i < [llength $l] && ($i == 0 || ($i % 24 != 0) || $new_page == 1) } { incr i } {
            set new_page 0
            set num [expr {$i+1}]
            set lentry [lindex $l $i]
            puts "$num: $lentry"
        }
        if { $i < [llength $l] } {
            puts -nonewline "Hit enter to continue..."
            flush stdout
            gets stdin key
            incr i
        }
    }
    return $i
}
proc add_acl { acl acllist } {
    puts "\n"
    set i [paginate $acllist]
    set aclend [expr {$i + 1}]

    puts -nonewline "Insert before which line number ($aclend to append): "
    flush stdout
    gets stdin choice

    if { $choice <= 0 || $choice > $aclend } {
        puts "Invalid line number, $choice.\n"
        return $acllist
    }

    puts -nonewline "Enter body of ACL rule to insert (without the access-list $acl portion): "
    flush stdout
    gets stdin body

    regsub -nocase {^access-list\s\[\s[^\s]+\]\s} $body "" body

    return [linsert $acllist [expr {$choice - 1}] "access-list $acl $body"]
}
proc delete_acl { acl acllist } {
    puts "\n"
    set i [paginate $acllist]

    puts -nonewline "Enter ACL entry number to delete: "
    flush stdout
    gets stdin lineno

    if { $lineno <= 0 || $lineno > $i } {
        puts "Invalid entry number, $lineno.\n"
        return $acllist
    }
    return [lreplace $acllist [expr {$lineno - 1}] [expr {$lineno - 1}]]
}

Delete a specific ACL entry
proc commit_acl { acl acllist orig_acllist } {
    ios_config "no access-list $acl"
    foreach line $acllist {
        if { [catch { ios_config $line } result] } {
            puts "Error committing access-list entry "$line" ($result)"
            puts "Re-adding the original access-list..."
            ios_config "no access-list $acl"
            foreach origline $orig_acllist {
                if { [catch { ios_config $origline } result] } {
                    puts "DANGER! Error committing original access-list entry "$origline" ($result)"
                    puts "Investigate this immediately!"
                    return
                }
            }
            return
        }
    }
    puts "Access-list $acl was committed successfully."
}
while { $done == 0 } {
    puts "Access-list Editor"
    puts "------------------"
    puts "1. View access-list $aclno"
    puts "2. Add access-list entry"
    puts "3. Remove access-list entry"
    puts "4. Quit and save changes"
    puts "5. Quit without saving changes"
    puts ""
    puts -nonewline "Enter option: ";
    flush stdout
    gets stdin choice

    switch $choice {
        1 { view_acl $acllist }
        2 { set acllist [add_acl $aclno $acllist] }
        3 { set acllist [delete_acl $aclno $acllist] }
        4 {
            set done 1
            set save 1
        }
        5 {
            set done 1
            set save 0
    }
}
Determine the High UDP port dynamically by inspecting the output of show ip sockets

proc snmp_fix { } {
    snmp_unfix
    set sockets [exec "show ip sockets"]
    set socket 0
    foreach line [split $sockets "\n"] {
        set line [string trim $line]
        if {{[regexp {^17\s+--listen--} $line] || [regexp {^17 0\..0\..0\..0} $line]}} {
            set tsocket [lindex $line 3]
            if {($tsocket > 49152 && $tsocket < 65535) { set socket $tsocket
                break
            }}
        }
    }
}
if {$socket > 0} {
    set myacl [find_acl 100 200]
    if {$myacl == 0} {
        set myacl [find_acl 2000 2700]
    }
    if {$myacl == 0} {
        puts "Failed to find a free access-list."
        return
    }
    ios_config "class-map match-all matchsnmp" "match access-group $myacl"
    ios_config "policy-map dropsnmp" "class matchsnmp" "drop"
    ios_config "access-list $myacl permit udp any any eq 162"
    ios_config "access-list $myacl permit udp any any eq $socket"
    ios_config "access-list $myacl deny ip any any"
    ios_config "control-plane" "service-policy input dropsnmp"

    puts "SNMP control plane access now denied to ports 162 and $socket"
    puts "using access-list $myacl. Use `snmp_unfix' to remove this"
    puts "configuration."
} else {
    puts "Failed to find a listening socket for SNMP."
}
Managing Syslog Events
Exercise 3

• Problem

Cisco Info Center is being used for general syslog and trap management

However, Resource Manager Essentials needs Config Change Syslog Messages for realtime configuration archival

Sending all the Syslog Messages to two destinations takes up network bandwidth, and puts unnecessary CPU load on RME
Managing Syslog Events

• Filter out unwanted syslog messages
  
  Configure a filter in RME to drop all syslog messages except the config change messages
  
  CONFIG_I
  
  CONFIG
  
  This reduces CPU overhead, but does not address the problem of increased network usage
Managing Syslog Events
Doing Things the Long Way

Router Sends OSPF-*-5-ADJCHG
Router Sends SYS-*-5-CONFIG_1 Messages

Define Message Filters in RME

Cisco Info Center

CiscoWorks
Managing Syslog Events

• **Solution**

Use Embedded Syslog Manager to filter out all but SYS-*-5-CONFIG_I and SYS-*-5-CONFIG Syslog Messages for transmission to the CiscoWorks Server

This approach does not require any syslog filters in RME, and keeps unwanted messages off of the network
ESM CONFIG Message Filter

Configuring the Router

- Define the ESM filter
  
  ```
  logging filter disk0:/config_chg.tcl
  ```

- Add the CiscoWorks server as a logging host, and pass all messages through the config change filter
  
  ```
  logging host 10.10.10.1 filtered
  ```

- Add the CIC server as an unfiltered logging host
  
  ```
  logging 10.10.10.2
  ```
Router#show logging

... Filter modules:
    disk0:/config_chg.tcl

    Trap logging: level debugging, 103 message lines logged
    Logging to 10.10.10.1 (udp port 514, audit disabled), 24
message lines logged, xml disabled,
    filtering disabled
    Logging to 10.10.10.2 (udp port 514, audit disabled), 3
message lines logged, xml disabled,
    filtering enabled
ESM CONFIG Message Filter
The Code

Only pass messages with a mnemonic of CONFIG_I or CONFIG on to the destination

```bash
# Embedded Syslog Manager, Only send CONFIG and CONFIG_I syslog messages
#
# Namespace: global

if {[string length $::orig_msg] == 0} {
    return ""
}

if {[string compare -nocase "CONFIG_I" $::mnemonic] == 0 ||
    [string compare -nocase "CONFIG" $::mnemonic] == 0} {
    return $::orig_msg
}

return ""
```