Sine Nomine Associates

Supercharged VM Startup: A System V-Style INIT Process for VM and Guests

David Boyes
Sine Nomine Associates
WAVV Colorado Springs 2005

Presentation Download

This presentation will be available for download from:

www.sinenomine.net

Agenda

- Introduction to SysVInit
- Construction
- Example of Use
- Conclusion
- Questions

Introduction to SysVInit

What Are We Doing?

- SysVInit is a systematic approach to service virtual machine management for z/VM and VM/ESA
- Freely available from

  www.sinenomine.net/vm/s5i

Problems to Solve:

- VM has no easily programmatically accessible standard method for adding and deleting items from the system startup processing
- VM has no way to indicate that a particular virtual machine startup depends on the availability of other virtual machine
- VM has no way to set up differing configurations of system services for different running conditions
- CP SHUTDOWN does not have any standard "graceful shutdown" processing

SysVInit: Introduction

- What is System V Init?
  - Method of starting services under Unix and Linux
    Startup scripts in /etc/init.d
    - Scripts invoked with standard functions, eg
      "start," "stop," "restart," "reload" et al.
  - Services grouped into "runlevels"
    - Each runlevel lives in /etc/init.d/rc*X*.d or /etc/rc*X*.d, where *X* is the runlevel name.
    - Symbolic links tie "Startup" and "Kill" and priority levels back to scripts in base directory

System V example of use

- Example
  - /etc/init.d/ssh starts the ssh daemon.
  - Symbolic links:
    - /etc/rc0.d/K20ssh, /etc/rc2.d/S20ssh, others
    - Starts at priority 20 in runlevels 2,3,4,5, and 6
    - Stops at priority 20 in runlevels 0 and 1

SysVInit: What Good Is It?

- This scheme allows Unix systems a way of specifying: “In runlevel *X*, start the following services in this order. When leaving runlevel *X*, shut them down in this order.”
- Tools like *insserv* and *chkconfig* allow for easy programmatic insertion of new services.
- Some distributions read comment structure in script to determine numeric runlevel priorities.

What Does z/VM Have?

- AUTOLOG1 PROFILE EXEC (on AUTOLOG1’s 191-disk)
  - Default configuration as shipped by IBM (controlled by SYSTEM CONFIG setting) starts AUTOLOG1 at IPL
  - Provides a “semi-standard” place to do ordered startup of service virtual machines
    - No pauses in between machines unless you insert them by hand, no error checking, no conditional processing (unless, you do it by hand)
    - Anything beyond simple list to start is completely ad-hoc and reinvented for each site.
    - Is not incorporated into shutdown processing

SysVInit for z/VM and VM/ESA

- What is it?
  - A easy way to specify multiple groups of virtual machines for specific purposes
  - Each runlevel contains an ordered set of services and their dependencies.
  - A mechanism for moving between runlevels (equivalent to Linux/Unix *init*/*telinit*) in a controlled fashion
  - Basic framework for consistent start, stop, status testing, configuration management process for human and program controlled management.
  - Demonstration of the power of built-in CMS functions to provide useful system management services for non-CMS environments.

SysVInit for z/VM and VM/ESA

- How is it different from Sys V init?
  - Does not implement Perl or any non-CMS scripting language
  - Runlevels can be any legal 8 char identifier
  - Dependencies are not assigned by means of special comments in the scripts
  - Dependencies are actually significant
  - No numeric priority
  - Centralized event dispatcher

Why isn’t it just init-for-VM?

- System V Init (the real thing, for Unix) suffers from some *really* annoying shortcomings
  - Doesn’t respect dependencies: S80bar will always run after S40foo, even if bar depends on foo and foo fails.
  - Single-digit runlevel names? Two-digit priorities?
  - Different Linux distributions (to say nothing of Unix systems!) don’t agree:
    - What each runlevel means
    - Commands to manipulate runlevel scripts
    - Existence of tools to determine numeric priority levels
    - Even where runlevel symlinks live

Construction

Construction

- The basic event handling loop is done with PROP combined with a set of REXX action routines to implement the commands.
  - PROP does not match regular expressions, so PROP just recognizes verbs and hands the full command to DISPATCH EXEC
  - DISPATCH EXEC does command parsing and in turn spawns RUNLEVEL EXEC or particular service EXECs, as well as manipulating global variables.

Command Authorization

- Most commands have to be authorized:
  - from the service machine the command affects
  - from a user listed in the ADMIN global variable (USER%HOST for remote admins)
- Some particularly dangerous commands must be double-authorized:
  - CMD (execute CP command as AUTOLOG1) has to come from user listed in SYSVINIT RTABLE
  - SHUTDOWN [REIPL] must come from ADMIN who is also in the RTABLE

SYSVINIT RTABLE (PROP)

Runlevels

- Each runlevel is implemented as a NAMES file
  - Runlevel NAMES files are not intended for human editing
  - Contain service and dependency information (which is runlevel-specific)
  - Maintained with LEVELMAINT command
  - Internally parsed into stem variable tree accessible from service EXEC

Service Local Variables

- Service variables are kept as CMS GLOBALVs, each service in its own group.
  - Service descriptions, start and stop timeouts.
  - Manipulated with LIST SERVICEVARS, GET SERVICEVAR *service variable*, SET SERVICEVAR *service variable value*
  - Value of service local variables preserved over IPLs by storage in LASTING GLOBALV on AUTOLOG1 191.

Global Variables

- Global vars affect the entire SYSVINIT server
  - Also stored as CMS GLOBALVs, group GLOBAL
  - Variables stored as global variables:
    - Default service startup/stop timeouts
    - List of allowable administrative users
    - Owner and address of configuration disk.

SVM Exec

- Service = Virtual Machine
- Userid() EXEC = controls virtual machine
- Template for all service manipulation.
  - Designed for one function per virtual machine.
  - Called with small set of standard “methods”
  - If defaults aren’t appropriate, can be modified to suit your needs.
- Initial setup for each virtual machine done with SERVICE *foo* ADD

SVM Methods

START	Starts a machine
STOP	Stop a machine
PROBE	Test status of machine
STATUS	Report status of machine
RESTART	Stop/start of machine
RELOAD	Reload config if needed
FORCERELOAD	Reload config by force

Example of Use

Example of Use

- Installation
  - Create AUTOLOG1 192
  - Unpack S5I VMARC onto it
  - Save and print AUTOLOG1 PROFILE EXEC
  - Copy over new PROFILE EXEC
  - Edit SYSVINIT RTABLE for local nodename and admin users
  - Start the service: it starts in runlevel DEFAULT with only DUMMY service enabled

Example of Use

- Add services you want
  - TELL AUTOLOG1 SERVICE VMSERVR ADD
  - TELL AUTOLOG1 SERVICE VMSERVU ADD …
- Put services in runlevel
  - TELL AUTOLOG1 LEVELMAINT DEFAULT VMSERVR …
  - TELL AUTOLOG1 LEVELMAINT DUMMY REMOVE

Example of Use

- Add dependency information
  - TELL AUTOLOG1 LEVELMAINT DEFAULT LINUX01 STARTAFTER TCPIP
  - TELL AUTOLOG1 LEVELMAINT DEFAULT TCPIP STOPAFTER LINUX01
- Change service variables
  - TELL AUTOLOG1 SET SERVICEVAR LINUX01 DESCR Debian Linux Sarge RC1
  - TELL AUTOLOG1 SET SERVICEVAR LINUX01 STARTTIMEOUT 45

Example of Use

- Monitor use
  - TELL AUTOLOG1 SHOW RUNLEVEL
  - TELL AUTOLOG1 SERVICE LINUX01 PROBE
  - TELL AUTOLOG1 SERVICE TCPIP RESTART
  - TELL AUTOLOG1 LIST GLOBALVARS
  - TELL AUTOLOG1 GET SERVICEVAR LINUX01 DESCR …

Live Demo

Conclusion

Conclusion

- SysVInit provides a flexible method for adding new services to a VM system.
- Far superior to traditional PROFILE EXEC autolog list
- Also superior to Unix System V Init
- Designed to be easily manipulated by product installation scripts

Future Enhancements

- Equivalents to *insserv* and *chkconfig* to allow:
  - Manipulating multiple runlevels at once
- Dependency list loop checking
  - Currently, if you misspell a dependency or have a circular dependency…your gun, your foot.

Questions

Contact Info

David Boyes
Sine Nomine Associates
dboyes@sinenomine.net
www.sinenomine.net