New Kernel Configuration Mechanism for HP-UX 11iv2

Steven Roth

Software Architect
Hewlett-Packard Company
Enterprise Systems Group

HP WORLD 2003
Solutions and Technology Conference & Expo

New Kernel Configuration Mechanism for HP-UX 11iv2

- The traditional HP-UX kernel configuration mechanism requires compiling kernel code (conf.c), relinking the kernel (vmunix), and rebooting after every change
  - Outdated mechanism inherited from BSD
  - Not appropriate for highly available, dynamic systems
- Mechanism was greatly complicated by addition of features such as Dynamic Tunables and Dynamically Loadable Kernel Modules (DLKMs)
- For HP-UX 11iv2, the kernel configuration mechanism has been completely redesigned and rewritten.

Features of New Mechanism

- All KC tasks performed through kcweb GUI or new CLI
- Kernel configurations can be saved, restored, and moved between systems
- Admins can save any number of kernel configurations, and switch between them at will – often without reboot
- Running kernel configuration is automatically backed up before each configuration change (if desired)
- System maintains log of all configuration changes
- Tools provide descriptions of kernel modules and tunable parameters. Tunable parameters have on-line documentation.
- CLI gives both user-friendly and script-friendly output.

Summary of Changes

- New kernel configuration commands: kconfig, kcmodule, kctune, kclog, kcpath
- Revised kcweb (SAM) GUI
- Obsolete commands: config, kmadmin, kminstall, kmmodreg, kmsystem, kmupdate
- Temporary transition stubs: kmpath, kmtune, mk_kernel
- No more master files, space files, or .h files in /usr/conf
- Kernel configurations are directories in /stand
- Kernel configuration changes take effect immediately unless a reboot is required

What is a Kernel Configuration?

Logically:

- A set of kernel modules, each with a desired state
- A set of kernel tunable parameter value assignments
- A primary swap device specification
- A set of dump device specifications
- A set of bindings of devices to device drivers
- A name and description

Physically (in a directory in /stand):

- An HP-UX kernel executable (vmunix)
- A set of HP-UX kernel module files
- A kernel registry database, containing the above settings
- A system file, with the above settings in human-readable form

Kernel Configuration Commands

kconfig	Manages whole kernel configurations
kcmodule	Manages module state settings
kctune	Manages tunable parameter settings
kclog	Manages the kernel configuration log file
kcpath	Gives locations of configuration files

Temporary:

kmpath	Stub to call kcpath
kmtune	Stub to call kctune
mk_kernel	Stub to call kconfig

Kernel Configuration GUI: kcweb

- Web-based and user-friendly graphical user interface
- Monitors and modifies kernel tunables
- Accesses the man pages for tunables
- Adds, modifies, and removes tunable usage alarms
- Modifies kernel module states
- Provides command preview

Accessed from:

- Command line (using kcweb command)
- HP Service Control Manager
- Kernel Config area of SAM
- Any web browser (if kcweb server has been started)

Managing Kernel Modules

- The HP-UX kernel is built from a number of modules:
  - device drivers
  - kernel subsystems
  - etc.
- A typical kernel configuration has 200-300 modules
- Modules can be managed using the kcmodule command or the kcweb GUI

Kernel Module Information

- Name
- Description
- Version ([3E36E5FA] or 0.1.0)
- Current state and what caused it
- State at next boot and what caused it
- Supported states
- Dependencies
- Exported interfaces

Kernel Module States

unused	Module is not in use in the configuration
static	Module is statically bound into the kernel executable (vmunix)
loaded	Module is dynamically loaded during boot
auto	Module is dynamically loaded when first needed
best	The “best” of the above states as chosen by the module developer

Kernel Module Causes

explicit	Admin explicitly requested this state
best	Admin requested best state
auto	Admin requested auto state; module was needed and has now been loaded
required	Module is required
depend	Module is needed to satisfy dependencies

Changing Module States

1. Using kcmodule:

   kcmodule cdfs=loaded
   kcmodule cdfs=best
   kcmodule cdfs=unused

2. Using kcweb
3. Using system files (described later)

- Changes take effect immediately if possible, otherwise are held for next boot. -h holds changes for next boot.

Managing Tunable Parameters

- Tunable parameters are integer variables that control the operation of kernel modules:
  - control resource allocations
  - control security policies
  - enable optional kernel behavior
  - etc.
- A typical kernel configuration has 150-200 tunables
- Admins can create their own tunables if desired
- Tunables can be managed using the kctune command or the kcweb GUI

Tunable Information

- Name
- Description
- Module
- Current value and expression
- Next boot value and expression
- Last boot value
- Default value (can change over time)
- Constraints on the value
- When the value can be changed

Default Values and Automatic Tuning

- Some tunables have algorithms that recompute their default value periodically based on system conditions (available memory, load on the system, hardware configuration, etc.).
- When one of these tunables is set to “Default”, its value will change whenever the default value is recomputed. These are “automatic” (a.k.a. “self-tuning”) tunables.
- Setting a tunable to anything other than “Default” disables automatic tuning.
  - This includes setting the tunable to the default value reported by kctune, instead of setting it to “Default”.
- If a non-automatic tunable is set to “Default”, it will receive the benefit of future HP changes to the default.

Changing Tunable Values

1. Using kctune:

   kctune nproc=4300
   kctune nproc=
   kctune nproc=Default
   kctune 'nkthread=nproc*2+100'
   kctune nproc+=100
   kctune ‘nproc>=5000’

2. Using kcweb
3. Using system files (described later)

- Changes take effect immediately if possible, otherwise are held for next boot. -h holds changes for next boot.

Setting Tunable Alarms

- Some tunable parameters represent kernel resources whose usage can be monitored.
- For these tunables, you can set alarms to notify you when the usage passes a threshold you specify.
- Notification is via your choice of: console, opcmsg, syslog, textlog, email, snmp, tcp, or udp.

Persistence of Changes

- All changes are applied immediately to the currently running system if possible. If not possible (or –h specified), changes are held for next boot.
- All changes made during a single KC command invocation are applied together. If one is held for next boot, they all are.
- Changes being held for next boot can be listed using kconfig –D and discarded using kconfig –H. Changes are also discarded when subsequent changes override them.
- Once a change is applied, it remains in force across reboots.

Saved Configurations

- Admins can have as many saved kernel configurations as desired (subject to available space in /stand).
- Each configuration has a name and an optional title.
- kctune and kcmodule can act on a saved configuration instead of the currently running configuration when given the –c *configname* flag.
- Saved configurations are managed using the kconfig command.

Saved Configurations

kconfig	List saved configurations
kconfig –c	Copy a configuration
kconfig –d	Delete a configuration
kconfig –e	Export a system file describing a config
kconfig –i	Import a system file to a configuration
kconfig –l	Load a configuration (make it the currently running configuration)
kconfig –n	Mark a configuration for use at next boot
kconfig –r	Rename a configuration
kconfig –s	Save the currently running configuration
kconfig –t	Set the title of a configuration
kconfig –w	Which configuration is running?
kconfig –H	Discard pending changes

Booting a Saved Configuration

- The easiest way to boot a saved configuration is to mark it for use at next boot before shutting down:

  $ kconfig –n configname
  $ shutdown

- You can also choose a saved configuration at the boot loader command line:

  HPUX> boot *configname*

- When you boot a saved configuration (either way), the previous configuration is discarded. If you want to retain it, save it using kconfig –s before rebooting.

Using System Files

- Historically, admins have made configuration changes by editing /stand/system and then running mk_kernel. This still works.
- Each configuration has a system file, a text file that contains all of the configuration settings. This is located at /stand/system for the currently running configuration, and /stand/*configname*/system for saved configurations.
- System files can also be created using kconfig –e.
- Configurations can be changed by editing the system file and running kconfig –i to import it. (mk_kernel is a temporary alias to kconfig –i for convenience.)

Using System Files (2)

- The format of a system file is enhanced to support module states and other new features. Entries added to a system file by an admin may be in either the old or new form; they will be automatically converted to the new form.
- System files are rewritten by the system after each configuration change. Do not put comments in system files. (Put them in the KC log file instead.)
- System files are useful when making lots of changes to be applied at once.
- System files are useful for moving configurations from one machine to another.

System Files vs. Commands

System File Entry	Command
modulename	kcmodule *modulename*=best
module *modulename state* [*version*]	kcmodule *modulename*=*state*
(no entry for *modulename*)	kcmodule *modulename*=unused
tunablename value	kctune *tunablename*=*value*
tunable *tunablename value*	kctune *tunablename*=*value*
(no entry for *tunablename*)	kctune *tunablename*=Default
swap *swapdevice*	(no equivalent)
dump *dumpdevice*	(no equivalent)
driver *devicename drivername*	(no equivalent)

Kernel Configuration Log File

- The KC commands maintain a log of all configuration changes at /var/adm/kc.log.
- The log can be read manually, or searched using the kclog command.
- All KC commands that make changes take a –C option, allowing the caller to specify a comment that will be put in the log entry for the change.
- Changes made through kcweb are reflected in the log since kcweb uses the KC commands to do its work.
- The log can be viewed from within kcweb.

Command Output Formats

The KC commands produce output in three formats:

- Table Output (Default)
  Human-readable summary; one line for each item being described; variable width columns.
- Verbose Output (–v)
  Human-readable detailed output; multiple lines for each item; items separated by a blank line; variable width columns.
- Parsable Output (–P)
  Script-friendly detailed output; multiple lines for each item; items separated by a blank line; tab-separated columns.

Parsable Output Format

- When using the parsable output format, the caller specifies which fields of information are desired for each item, e.g.

```
$ kctune –P name,current nproc
name      nproc
current   4200
```

- HP will support backward compatibility across releases for the –P parsable output format. (See full documentation for details.)
- Scripts and applications parsing the output of the KC commands must use the –P parsable output format. HP reserves the right to change the other output formats without notice.

Backup Configuration

- The system maintains a saved configuration called “backup”.
- The currently running configuration is saved to “backup” *before* each configuration change, so loading the “backup” configuration acts as an “undo”.
- At the first configuration change after each boot, the system requests confirmation before overwriting “backup”, in case you want to preserve the “backup” from the previous boot.
- The default behavior can be overridden:
  –B forces the “backup” to be saved,
  –K forces the “backup” to be left unchanged

For More Information

- This talk is a summary of the “Managing Kernel Configurations in HP-UX 11i Version 2” white paper, available on HP’s web site:

  Visit http://www.hp.com/go/hpux11iv2
  Click on “information library” in the grey box
  Look under “hp-ux 11iv2 white papers”

- Kernel module developers will want to read the Kernel Configuration and DLKM chapters of the 11iv2 Device Driver Guide, to be published soon at

  http://www.hp.com/dspp

  These chapters give details of providing kernel modules and tunables in the new mechanism.

Interex, Encompass and HP bring you a powerful new HP World.