What Makes It Tick: The Linux Boot Process

on IBM System z

John Jolly
Senior Software Engineer SUSE Linux Enterprise Server
jjolly@suse.de
Session #09897

2011-08-08

SUSE® Linux Enterprise Server for System z
10 years on the Mainframe

http://www.novell.com/partners/ibm/mainframe/img/timeline_lores.pdf

- The first deployments on Linux for the mainframe were file and print servers.
- The first piece of software that became popular was Samba.
- The first large commercial customer for SUSE Linux Enterprise Server for S/390 was Telia, the largest telecommunications company in Sweden.
- Today, companies are running their mission-critical workloads on top of SUSE Linux Enterprise Server for System z.

Why Customers Prefer SUSE® Linux Enterprise Server for System z

The optimized version for IBM System z:

- **SUSE Linux Enterprise Server is #1**

in mainframe Linux market (80%+ share)
in SAP-on-Linux market (75% share)
in High Performance Computing (6 of top 10)

- **SUSE Linux Enterprise Server for System z:**

Fully supported by IBM – supports all benefits of the mainframe

10 years of expertise (available five years ahead of competition)

Ideal for workload consolidation, providing major cost savings

New features specific to System z

Hosting of Subscription Management Tool on System z

More than 1,700 certified applications available

Differentiators

- SLES for System z: enterprise-class and fully supported operating system tailored for mainframe available to customers 5 years ahead of competition. Market-share leader for zLinux with 80+ percent.

- Close partnership between IBM – SAP – Novell

IBM's System z, SAP's and Novell's Linux core engineering and L3-support teams

Fastest response and resolution times for any type of support-, consulting- or

- Build Service Advantage

Reduces production problems

Consolidates IT skills across disparate systems

Delivers critical updates in hours – not days or weeks

Differentiators - Unique tools for SUSE Linux Enterprise Server for System z

Yast and Integrated Systems Management

Configure every aspect of the server

- **Subscription Management Tool Hosting**
  Subscription and patch management made easy
- **High Availability Extension for SLES**
  Included in SLES for System z
- **.NET Applications on Linux: Mono**
  Migrate existing .NET applications to Linux without having to rewrite code
- **Starter System for System z**
  Starter System for System z is a pre-built installation server

SUSE® Linux Enterprise Server for System z
Starter System for System z

Historically, one of the biggest hurdles to implementing Linux on the mainframe has been gaining network access to the installation media from the mainframe (Installation routine cannot access built-in DVD reader, Firewall rule changes needed)

SUSE Linux Enterprise Server Starter System for System z is a pre-built installation server — facilitates installation of SUSE Linux Enterprise Server for System z on a z/VM system

Eliminates the network access hurdle to try out Linux on the mainframe — **gaining network access** to the installation media from the mainframe

Allows **customers** with little or no Linux or z/VM experience to initiate evaluations of SUSE Linux Enterprise Server for System z

http://www.novell.com/partners/ibm/mainframe/starterpack.html

On To The Subject At Hand: The Boot Process

How It All Starts: zIPL

- Part of the s390-tools package
- Much like lilo or grub on lesser systems
  - or are they like us?
- zIPL has three purposes
  - Loading and executing the Linux Kernel in memory
  - Creating a system dump on a DASD partition, tape device or SCSI partition
  - Launching a Linux Kernel from NSS

Installing zIPL

- zIPL is part of the s390-tools package
- Installs on many different device types
  - DASD
  - zFCP
  - Classic and Compatible ECKD DASD
  - Even Tape!
    - This is generally the slowest option, for obvious reasons
- Installation can be done from command line or zipl.conf
  - Only one configuration can be specified from the command line
  - Multiple configurations can be specified via the zipl.conf

Oh, The Flexibility!
The zipl.conf file

- Multiple configuration can be specified within zipl.conf
- Each configuration can be assign a numeric value
  - Up to 62 values available for DASD devices, 30 for SCSI
- Two section types within this configuration file
  - Configuration sections: Titles are surround with square brackets
  - Menu sections: Titles are preceded with a colon character
- Special configuration section titled [defaultboot]
- Has two settings allowed
  - default=<configuration name>
  - defaultmenu=<menu name>

More Choices: zipl.conf Configuration Sections

- target=<path>
  - Base directory for a configuration file bootmap
  - Device where path can be found will have zipl boot loader installed
- image=<path to kernel>[,<address>]
  - An address can be specified if you don’t want the default address of 0x10000
- ramdisk=<path to initrd>[,<address>]
  - Again, if you wish to chose an alternate to the default address
- parameters=<kernel boot parameters>
- parmfile=<path to parameter file>[,<address>]
  - Kernel boot parameters (we’ll discuss those later)

Even More Choices: zipl.conf Menu Sections

- target=<path>
  - Same as boot configurations (previous page)
- <number>=<configuration name>
  - Assigns the specific menu number to the specified configuration
  - DASD-based systems can use 1-62, 1-30 on SCSI systems
  - 0 is automatically set to the default setting, and is reserved
- default=<number>
  - Boots automatically after the timeout expires
- timeout=<seconds>
- prompt=<0/1>
  - A value of 1 allows for the interactive menu (DASD only), otherwise boot to default

Will The Choices Never End: zipl.conf NSS and Dump Configuration

- segment=<NSS name>
  - Created by a Class E user with the SAVESYS=<name> kernel boot option

- dumpto=<DASD or Tape partition device node>[,<size>]
- dumptofs=<SCSI partition device node>[,<size>]
- mvdump=<ECKD DASD device dump list>[,<size>]
  - Specify the device node (almost always under /dev)
  - Maximum allowed dump size can also be specified
  - SCSI option places file on the filesystem of the partition

Kernel Boot Options
System z Specific

- Can be passed to zIPL boot loader several ways
  - Through the zipl.conf parameters or parmfile keys
  - ipl <addr> parm <linux boot parameters>
  - Via the interactive menu (on DASD boot loader)

Linux Kernel Boot Option: cio_ignore=

- Allows the blacklist of device ids that should not be scanned by the CommonIO subsystem
- Ranges may also be specified
- Multiple ranges can be separated by a comma
  - Multiple ranges are scanned left-to-right
- The ‘all’ option will blacklist every device
- The exclamation point (!) can “include” a range
- Example
  - cio_ignore=all,!0.0.0009,!0.0.700-0.0.702,!fd00

Linux Kernel Boot Option: ccw_timeout_log=

- Activates log of device timeouts
- Very useful for finding device IDs (or ranges of IDs) to blacklist using cio_ignore

Linux Kernel Boot Option: dasd=

- Allows specification of features for one or more DASD
- The only documented feature currently is read-only (ro)
  - Although I suspect there are more (diag, raw, erplog, failfast)
- Example:
  - dasd=0180(ro),0190(ro)

Linux Kernel Boot Option: condev= and conmode=

- 3215 emulation options for specifying device and console mode parameters
- Not common IMO, but still used.

Linux Kernel Boot Option: hvc_iucv= and hvc_iucv_allow=

- hvc_iucv=<number>
  - Number of allowed z/VM IDs allowed access via IUCV
  - Value can be 0 to 8
- hvc_iucv_allow=<userIDs>
  - Comma-separated list of allowed user IDs for IUCV access
- Useful for emergency console access
  - And none of that mucking about in line-entry mode!
- More information can be found online:
  - http://www.vm.ibm.com/education/lvc/lvc1117c.pdf

Linux Kernel Boot Option: vm*=

- vmpanic=<CP command>
- vmreboot=<CP command>
- vmhalt=<CP command>
- vmpoff=<CP command>
  - Allows for a CP command to be executed.
    - vmpanic="cp ipl <dasd> loadparm 1" will restart the system after a panic

Linux Kernel Boot Option: savesys=

- savesys=<NSS ID>
- This option saves an image of the kernel at the NSS to be used at another time
- Only used once to create the NSS with the specified ID
- Must be a logged on as a Class E user to successfully launch with this switch
- Exceptional resource savings for large-scale systems
- NSS is a feature of z/VM


suse®