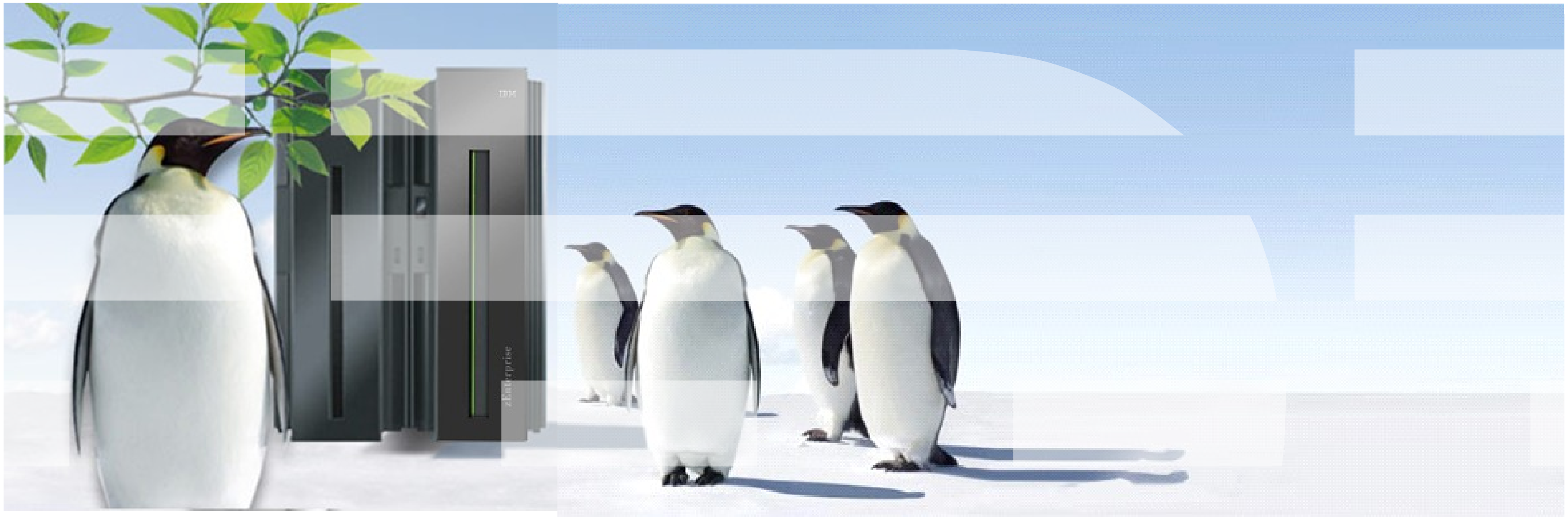
What's New in Linux on System z

Martin Schwidefsky
IBM Lab Böblingen, Germany
Aug 14 2013
Session 13530



Trademarks & Disclaimer

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries. For a complete list of IBM Trademarks, see www.ibm.com/legal/copytrade.shtml:

IBM, the IBM logo, BladeCenter, Calibrated Vectored Cooling, ClusterProven, Cool Blue, POWER, PowerExecutive, Predictive Failure Analysis, ServerProven, System p, System Storage, System x , System z, WebSphere, DB2 and Tivoli are trademarks of IBM Corporation in the United States and/or other countries. For a list of additional IBM trademarks, please see http://ibm.com/legal/copytrade.shtml.

The following are trademarks or registered trademarks of other companies: Java and all Java based trademarks and logos are trademarks of Sun Microsystems, Inc., in the United States and other countries or both Microsoft, Windows,Windows NT and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both. Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
UNIX is a registered trademark of The Open Group in the United States and other countries or both. Linux is a trademark of Linus Torvalds in the United States, other countries, or both. Cell Broadband Engine is a trademark of Sony Computer Entertainment Inc. InfiniBand is a trademark of the InfiniBand Trade Association.
Other company, product, or service names may be trademarks or service marks of others.

NOTES: Linux penguin image courtesy of Larry Ewing (lewing@isc.tamu.edu) and The GIMP

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Users of this document should verify the applicable data for their specific environment. IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.
Information is provided “AS IS” without warranty of any kind. All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

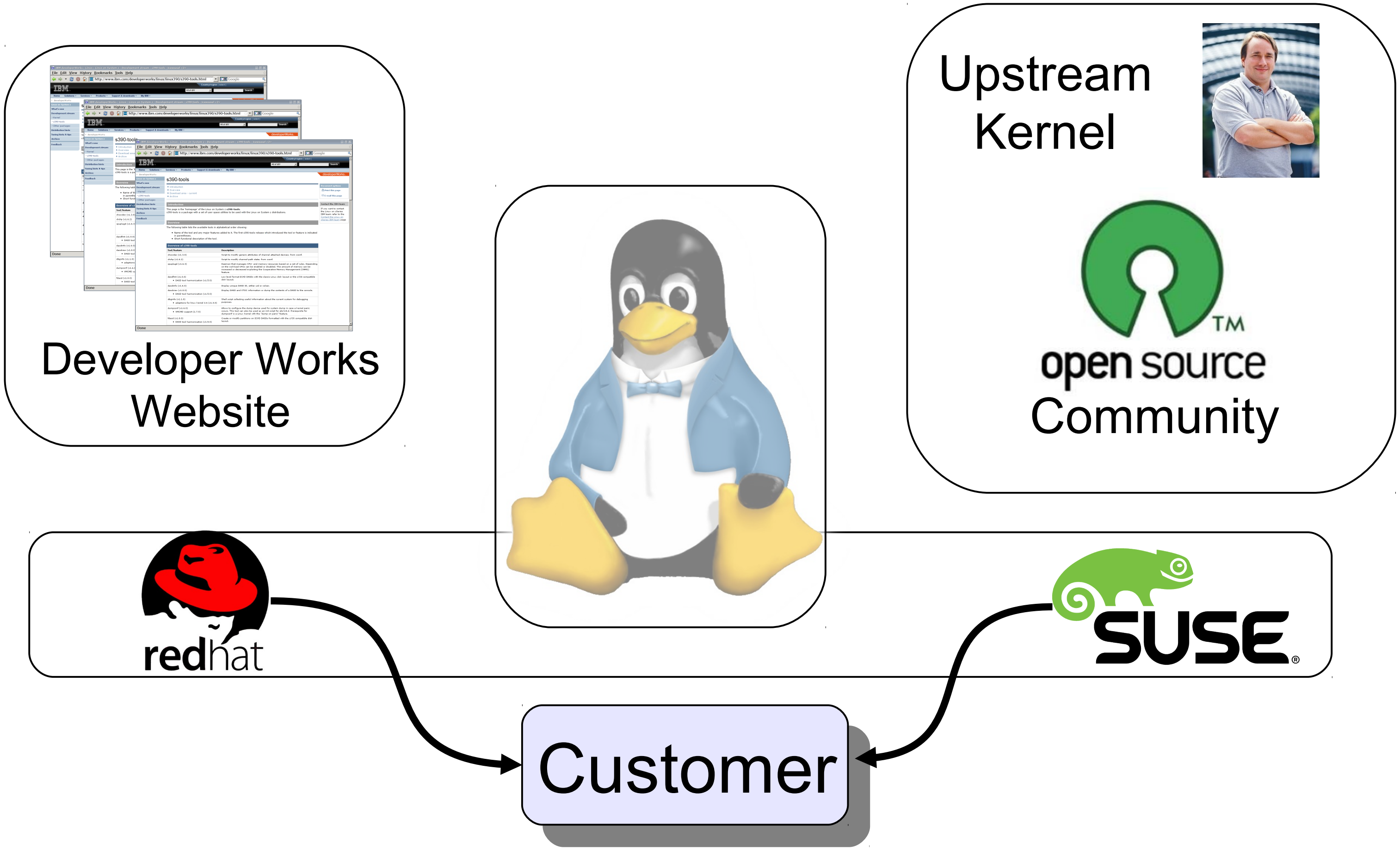
Prices are suggested US list prices and are subject to change without notice. Starting price may not include a hard drive, operating system or other features.
Contact your IBM representative or Business Partner for the most current pricing in your geography. Any proposed use of claims in this presentation outside of the United States must be reviewed by local IBM country counsel prior to such use. The information could include technical inaccuracies or typographical errors.
Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any

How Linux on System z is developed

How does the “community” work.

The IBM Linux development process

IBM Linux on System z development contribues in the following areas: Kernel, s390-tools, open source tools (e.g. eclipse, ooprofile), gcc, glibc, binutils



Developer Works Website

Upstream Kernel

Community

Customer

Facts on Linux

- Linux kernel 1.0.0 was released with 176,250 lines of code
  How many lines of code has the kernel version 3.10 ?

  16,961,959 lines of code (+1,963,222 since v3.2)
- How many of the world's top 500 supercomputers run Linux (Jun 2013)?

  476 / 95.2%
- What percentage of web servers run Linux (Jul 2013) ?

  65.7% run Unix, of those 47.5% run Linux (50.6% unknown) = 31.2%
- What percentage of desktop clients run Linux (Jun 2013) ?

  2.89% (+1.14 since Dec 2012), 5.01% via Android
- What is the architecture with the larges amount of core changes?

  ARM, for v3.10 ~112 KLOC, followed by mips, powerpc and x86 ~25 KLOC,
  System z (alias s390) had ~8 KLOCS

- Linux is Linux, but …features, properties and quality differ dependent on your platform and your use case

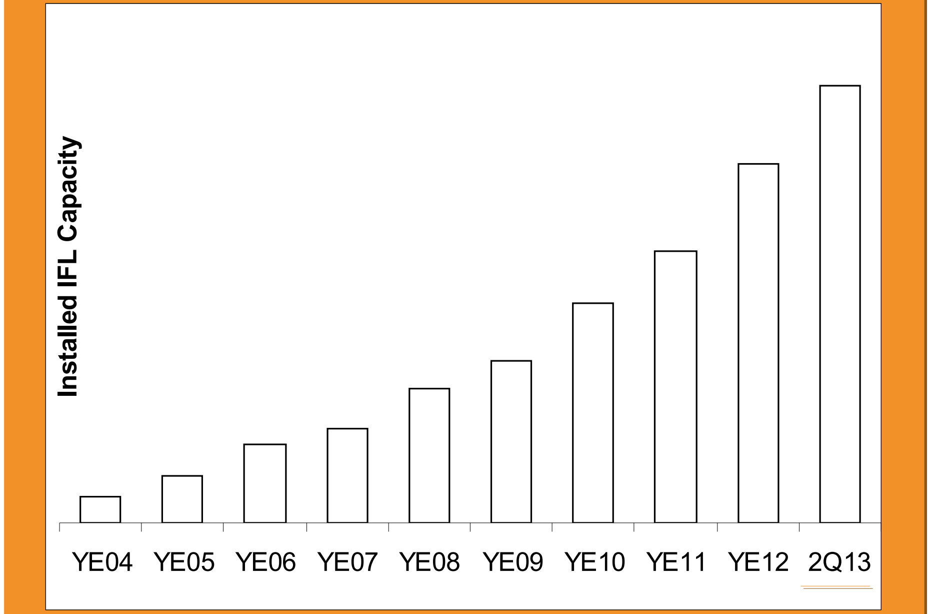
Source: http://kernel.org
http://top500.org/statistics
http://w3techs.com
http://www.w3counter.com
http://googleblog.blogspot.com/2011/05/android-momentum-mobile-and-more-at.html

Linux on IBM System z in 2Q2013

*Installed Linux MIPS at 48% CAGR**

- 25.1% of Total installed MIPS run Linux as of 2Q13
- Installed IFL MIPS increased 40% from 2Q12 to 2Q13
- 37% of System z Customers have IFL's installed as of 1Q13
- 76 of the top 100 System z Customers are running Linux on the mainframe as of 2Q13 **
- 33% of all System z servers have IFLs

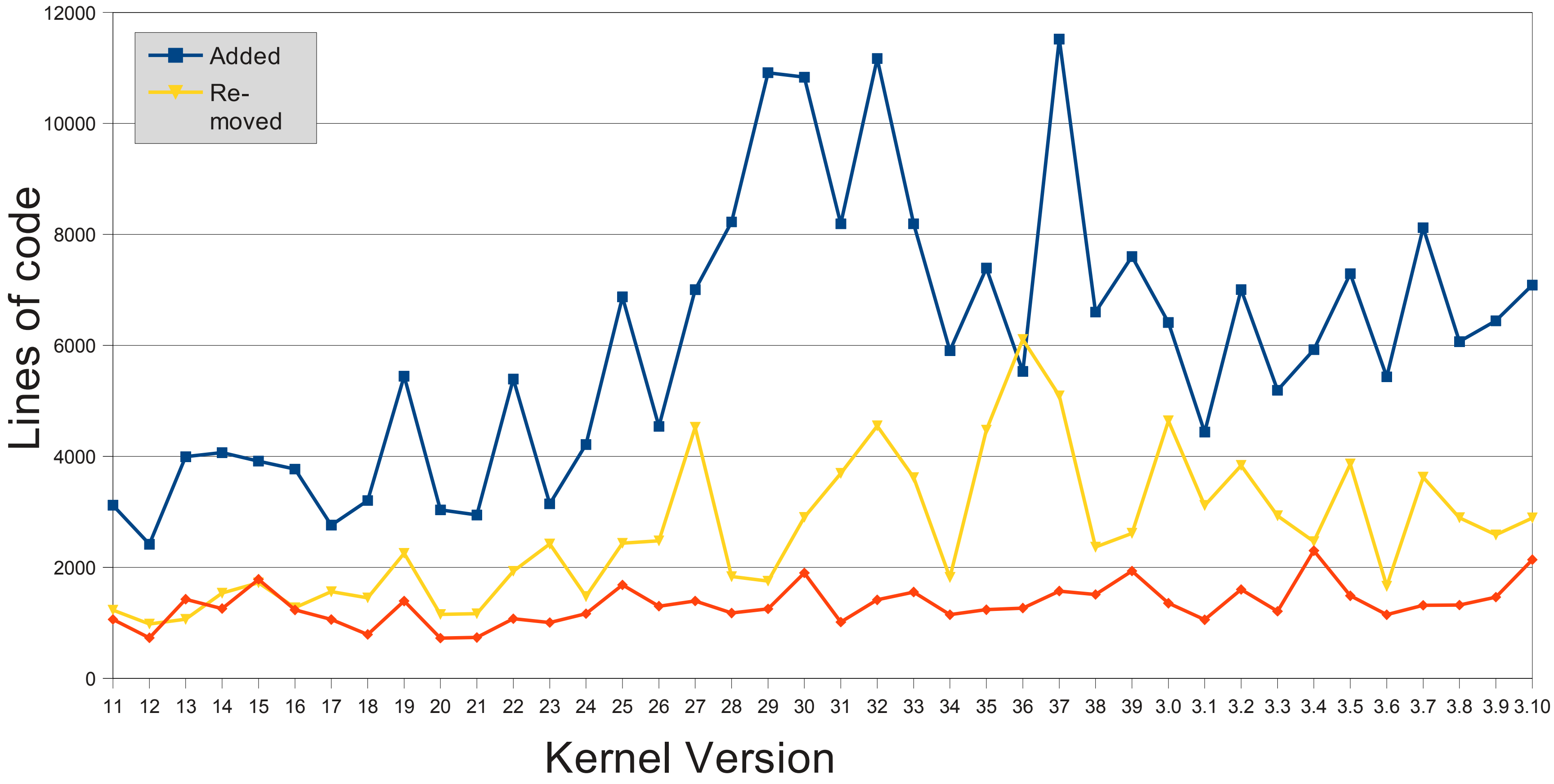
Installed Capacity Over Time



* Based on YE 2003 to 2Q 2013
**Top 100 is based on total installed MIPS

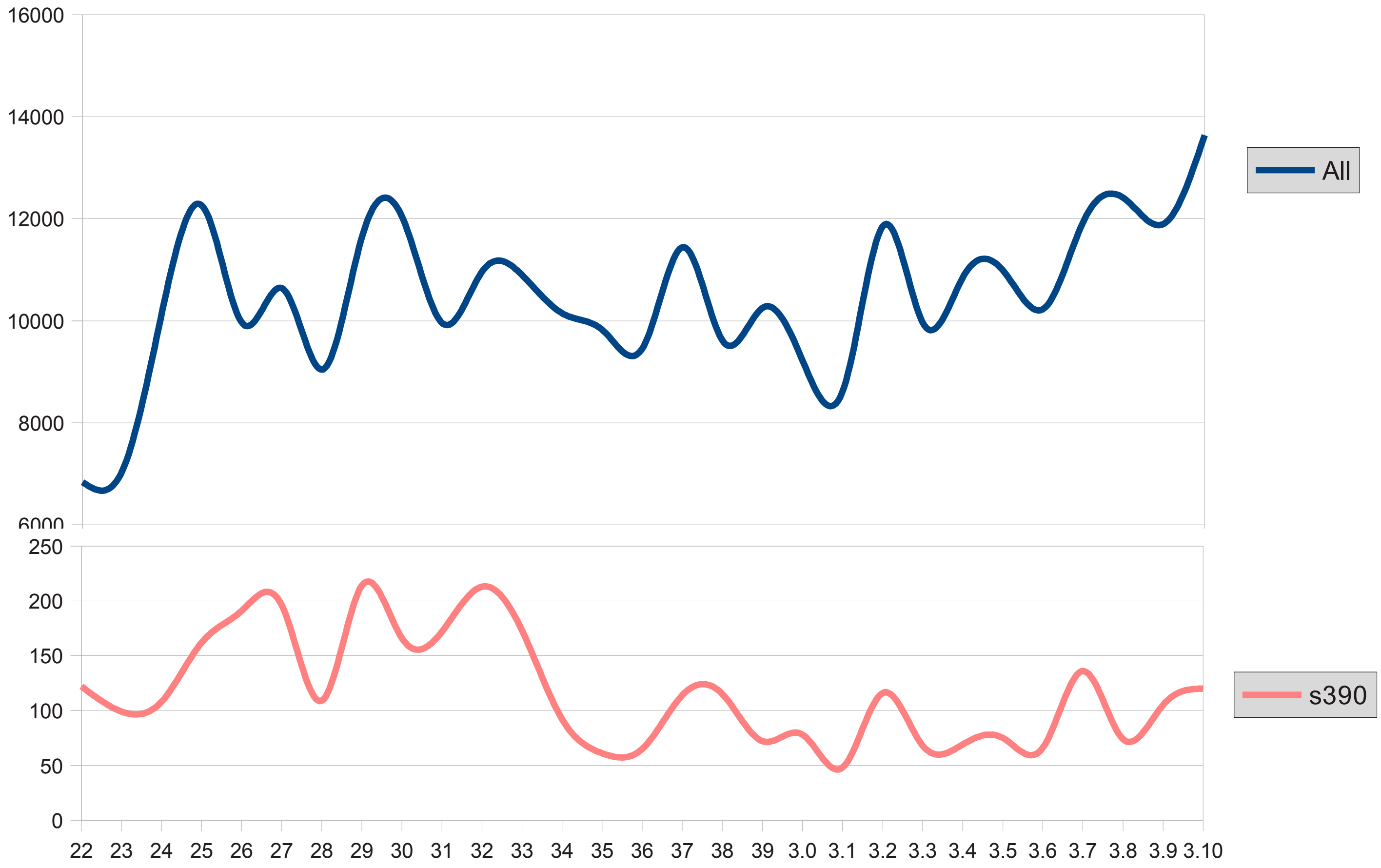
Linux kernel development: rate of change

Average for the last 8 1/2 years (without renames): 78 days per release, 5990 lines added, 2633 lines removed and 1304 lines modified **per day**



Linux kernel development: System z contributions

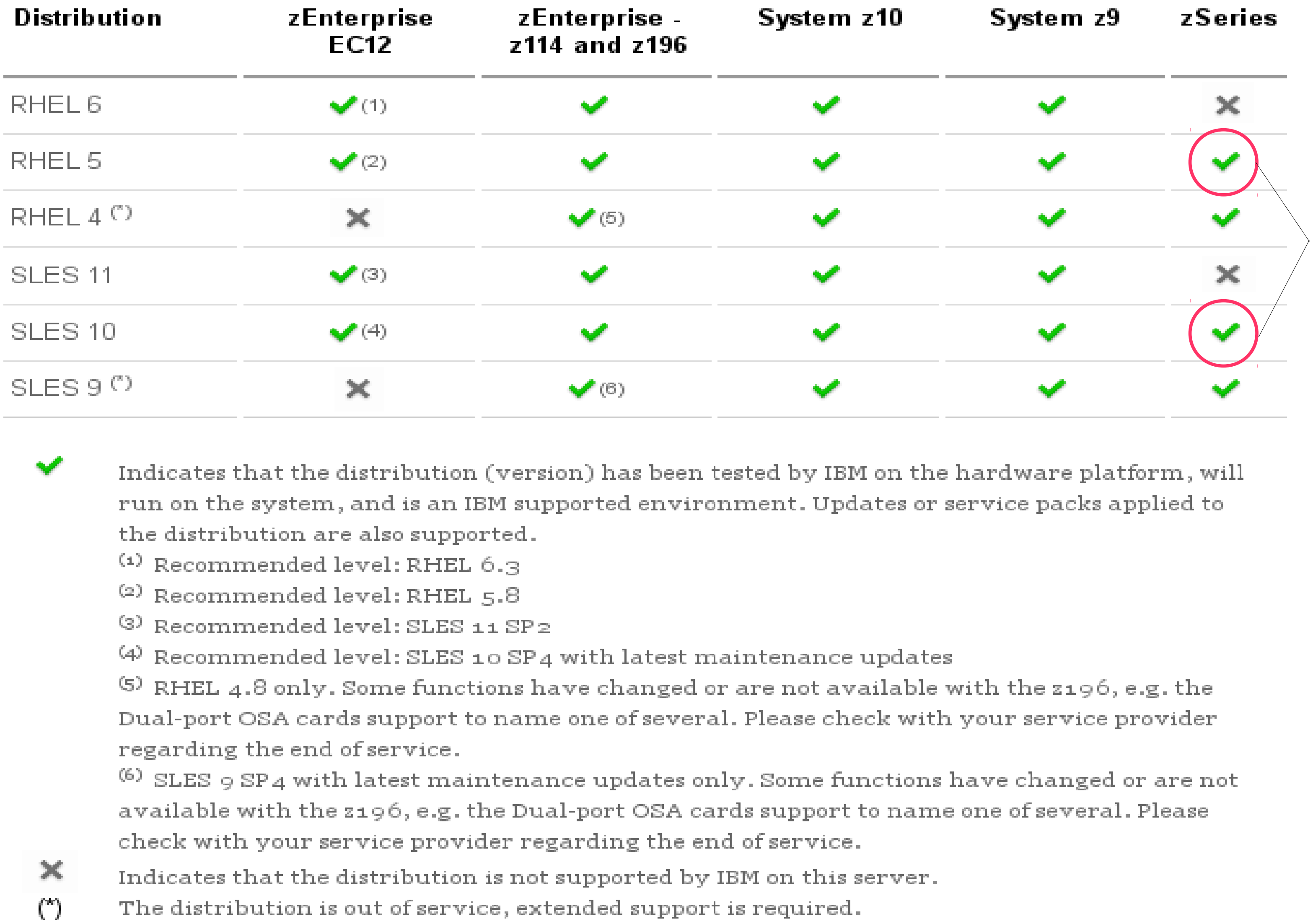
Changesets per 2.6.x / 3.x kernel release



Linux on System z distributions (Kernel 2.6 based)

- **SUSE Linux Enterprise Server 9 (GA 08/2004)**
  - Kernel 2.6.5, GCC 3.3.3, Service Pack 4 (GA 12/2007), end of regular life cycle
- **SUSE Linux Enterprise Server 10 (GA 07/2006)**
  - Kernel 2.6.16, GCC 4.1.0, Service Pack 5 (GA 05/2012)
- **SUSE Linux Enterprise Server 11 (GA 03/2009)**
  - Kernel 2.6.27, GCC 4.3.3, Service Pack 1 (GA 06/2010), Kernel 2.6.32
  - Kernel 3.0, GCC 4.3.4, Service Pack 3 (GA 06/2013)
- **Red Hat Enterprise Linux AS 4 (GA 02/2005)**
  - Kernel 2.6.9, GCC 3.4.3, Update 9 (GA 02/2011), end of regular life cycle
- **Red Hat Enterprise Linux AS 5 (GA 03/2007)**
  - Kernel 2.6.18, GCC 4.1.0, Update 9 (GA 01/2013)
- **Red Hat Enterprise Linux AS 6 (GA 11/2010)**
  - Kernel 2.6.32, GCC 4.4.0 Update 4 (GA 02/2013)
- **Others**
  - Debian, Slackware,
  - Support may be available by some third party

Supported Linux Distributions



Distribution	zEnterprise EC12	zEnterprise - z114 and z196	System z10	System z9	zSeries
RHEL 6	✓ (1)	✓	✓	✓	✗
RHEL 5	✓ (2)	✓	✓	✓	✓
RHEL 4 (*)	✗	✓ (5)	✓	✓	✓
SLES 11	✓ (3)	✓	✓	✓	✗
SLES 10	✓ (4)	✓	✓	✓	✓
SLES 9 (*)	✗	✓ (6)	✓	✓	✓

✓ Indicates that the distribution (version) has been tested by IBM on the hardware platform, will run on the system, and is an IBM supported environment. Updates or service packs applied to the distribution are also supported.

(1) Recommended level: RHEL 6.3

(2) Recommended level: RHEL 5.8

(3) Recommended level: SLES 11 SP2

(4) Recommended level: SLES 10 SP4 with latest maintenance updates

(5) RHEL 4.8 only. Some functions have changed or are not available with the z196, e.g. the Dual-port OSA cards support to name one of several. Please check with your service provider regarding the end of service.

(6) SLES 9 SP4 with latest maintenance updates only. Some functions have changed or are not available with the z196, e.g. the Dual-port OSA cards support to name one of several. Please check with your service provider regarding the end of service.

✗ Indicates that the distribution is not supported by IBM on this server.

(*) The distribution is out of service, extended support is required.

Two options for zSeries machines

source: http://www.ibm.com/systems/z/os/linux/resources/testedplatforms.html

Current Linux on System z Technology

Features & Functionality developed in the **past 2 years**
contained in the SUSE & Red Hat Distributions

Linux on System z features – zEC12 support

- **Transactional execution (kernel 3.7)**

  6.4 11.3

  - Also known as hardware transactional memory
  - CPU features that allows to execute a group of instructions atomically
  - Optimistic execution, if a transaction conflicts a rollback to a saved state is done

- **Storage class memory – Flash Express (kernel 3.7)**

  6.4 11.3

  - Internal Flash Solid State Disk (SSD)
  - Accessed via Extended Asynchronous Data Mover (EADM) sub-channels
  - Up to four pairs of cards with a maximum capacity of 4x1.4TB = 5.6TB
  - Support for concurrent MCL updates with kernel version 3.8

- **Support for Crypto Express 4S cards (kernel 3.7)**

  6.4 11.3

  - New generation of crypto adapters plug-able into the I/O drawer
  - New type 10 which uses a bit field to indicate capabilities of the crypto card

System zEC12 features – Transactional Execution

11.3

6.4

- **Transactional execution is a concurrency mechanism of the CPU comparable to database transactions**
  - Several reads and stores from/to memory logically occur at the same time
  - Improves performance for fine-grained serialization
  - Useful for lock-less data structures and speculative compiler optimizations
- **Two types of transactions: constraint and non-constraint**
- **Conflicting memory accesses will cause the transaction to abort**
  - Transaction abort is rather expensive
  - Constraint transaction will automatically restart
  - Ratio of successful vs. aborted transaction is important for performance
- **Kernel support is required to enable user programs to use transactional execution**
  - Control registers setup
  - Debugging support for additional PER controls via ptrace

System zEC12 features – Transactional Execution

11.3

6.4

- **Example of a list_add operation**

```
struct spinlock_t list_lock;
struct list_head list_head;
void list_add(struct list_head *new)
{
    spin_lock(&list_lock, 0, 1);
    list_add(new, &list_head);
    spin_unlock(&list_lock, 1, 0);
}
```

Typical pattern:
1) lock, 2) a short operation, 3) unlock

```
    Traditional code:                           Transactional code
          # spin_lock                           # begin transaction
          larl    %r3,list_lock                 tbeginc 0,0
          lhi     %r1,1
    lock: lhi     %r0,0
          cs      %r0,%r1,0(%r3)
          ltr     %r0,%r0
          jne     lock
          # list_add                            # list_add
          larl    %r4,list_head                 larl    %r4,list_head
          lg      %r5,0(%r4)                    lg      %r5,0(%r4)
          stg     %r4,0(%r2)                    stg     %r4,0(%r2)
          stg     %r5,8(%r2)                    stg     %r5,8(%r2)
          stg     %r2,0(%r5)                    stg     %r2,0(%r5)
          stg     %r2,8(%r4)                    stg     %r2,8(%r4)
          # spin_unlock                         # end transaction
          cs      %r1,%r0,0(%r3)                tend
          br      %r14                          br      %r14
```

Linux on System z features – Core kernel

- **Jump label support (kernel 3.0)**

  11.2

  – Branch optimization for conditions that are rarely toggled, e.g. tracepoints

- **Two stage dumper / kdump support (kernel 3.2, s390-tools-1.17.0)**

  6.3 11.3

  – Use a Linux kernel to create a system dump
    - Use a preloaded crash-kernel to run in case of a system failure
    - Can be triggered either as panic action or by the stand-alone dumper, integrated into the shutdown actions framework

  – Pro
    - Enhanced dump support that is able to reduce dump size, shared disk space, dump to network, dump to a file-system etc.
    - The makedumpfile tool can be used to filter the memory of the crashed system
    - Dump disk space sharing is possible for server farms using network dump

  – Con
    - kdump is not as reliable as the stand-alone dump tools
    - kdump cannot dump a z/VM named saved system (NSS)
    - For systems running in LPAR kdump consumes memory

Liux on System z features – two stage dumper / kdump support

6.3 11.3

- Add a crashkernel= to the kernel command line

```
crashkernel=<size>@<offset>
```

- Boot your system and check the reservation

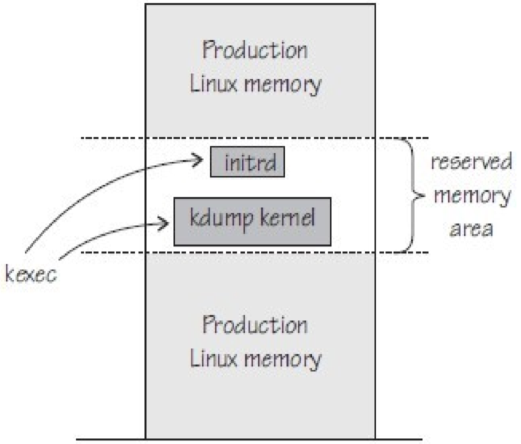
```
#cat /proc/iomem
00000000-3fffffff : System RAM
00000000-005f1143 : Kernel code
005f1144-00966497 : Kernel data
00b66000-014c4e9f : Kernel bss
40000000-47ffffff : Crash kernel
48000000-7fffffff : System RAM
```

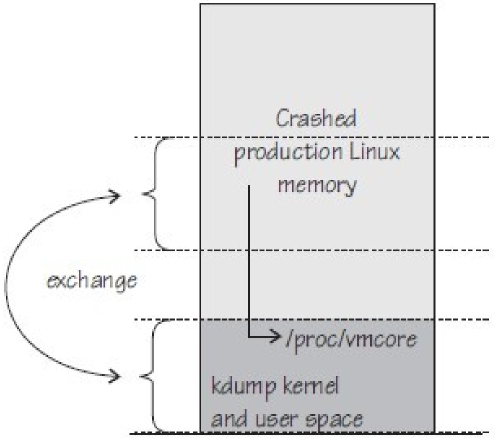
- Load the kdump kernel with kexec

```
# kexec -p kdump.image –initrd kdump.initrd
--command-line=”dasd=1234 root=/dev/ram0”
```

- Manually trigger for kdump under z/VM

```
#cp system restart
```





Linux on System z features – Core kernel

- **Allow to compare dump system with boot system**

  11.3

  - With z/VM 6.2 Single-System-Image it is possible to move active Linux instances between different z/VM instances
  - To aid debugging a log of past live-guest-relocations is made available in both the live system and in the dump of a system

- **Fuzzy Live Dump (kernel 3.5)**

  11.3

  - Add the capability to generate a dump of a live system.
  - Not all data structures will be consistent but the dump may still be useful.

Linux on System z features – Crypto

- **CP ACF exploitation of System z196 (kernel 3.0)**

  6.2 11.2

  – Add support for new HW crypto modes:
  cipher feedback mode (CFB), output feedback mode (OFB),
  counter mode (CTR), Galois counter mode (GCM),
  XEX based Tweaked Code Book with Cipher Text Stealing (XTS),
  cipher based message authentication mode (CMAC),
  and counter with cipher block chaining message authentication (CCM)

- **New libica APIs for supported crypto modes (libica 2.1.1)**

  6.2 11.2

  – Provide a programmatic way to query for supported crypto ciphers, modes and key sizes.

  – Deliver information whether the cryptographic features are implemented in hardware or in software

- **AP adapter resiliency (kernel 3.7)**

  11.3

  – Improve RAS capabilities of the AP bus and the zcrypt devices.

  – External AP bus configuration changes are now handled correctly.

Linux on System z features – FICON

- **Extended DASD statistics (kernel 3.1)** 6.3
  - Add detailed per-device debugging of DASD I/Os via debugfs
  - Useful to analyze problems in particular for PAV and HPF

- **DASD sanity check to detect path connection errors (kernel 3.3)** 11.3
  - An incorrect physical connection between host and storage server which is not detected by hardware or microcode can lead to data corruption
  - Add a check in the DASD driver to make sure that each available channel path leads to the same storage server

- **FICON Express8S hardware data router support for FCP (kernel 3.2)** 11.3
  - Hardware data router support requires an adapted qdio request format.
  - Improves performance by reducing the path length for data.

Linux on System z features – Networking

- **Add OSA concurrent hardware trap (kernel 3.0)**

  6.3 11.2

  - To ease problem determination the qeth driver requests a hardware trace when the device driver or the hardware detect an error
  - Allows to correlate between OSA and Linux traces.

- **Support VEPA Mode (kernel 3.8, s390-tools 1.22)**

  11.3

  - Virtual Ethernet Port Aggregator (VEPA) mode support
  - All packages are sent to the networking switch to enable external routing
  - Reduces CPU overhead in the virtual machine
  - Uses the security, filtering, and management features of the physical switch

Linux on System z features – s390-tools

- **Improve memory ballooning with cpuplugd (s390-tools 1.15.0)**

  6.2 11.2

  - A number of improvements to the memory balloon part to cpuplugd to make it more useful.

- **Extend lscpu tool and add new chcpu tool (util-linux 2.21)**

  11.3

  - Improve the lscpu tool to display CPU topology and CPU state
  - Add the new chcpu tool to change CPU state, rescan CPUs and change the CPU dispatching mode (horizontal vs. vertical polarization)

lscpu & chcpu

11.3

- Query CPU information

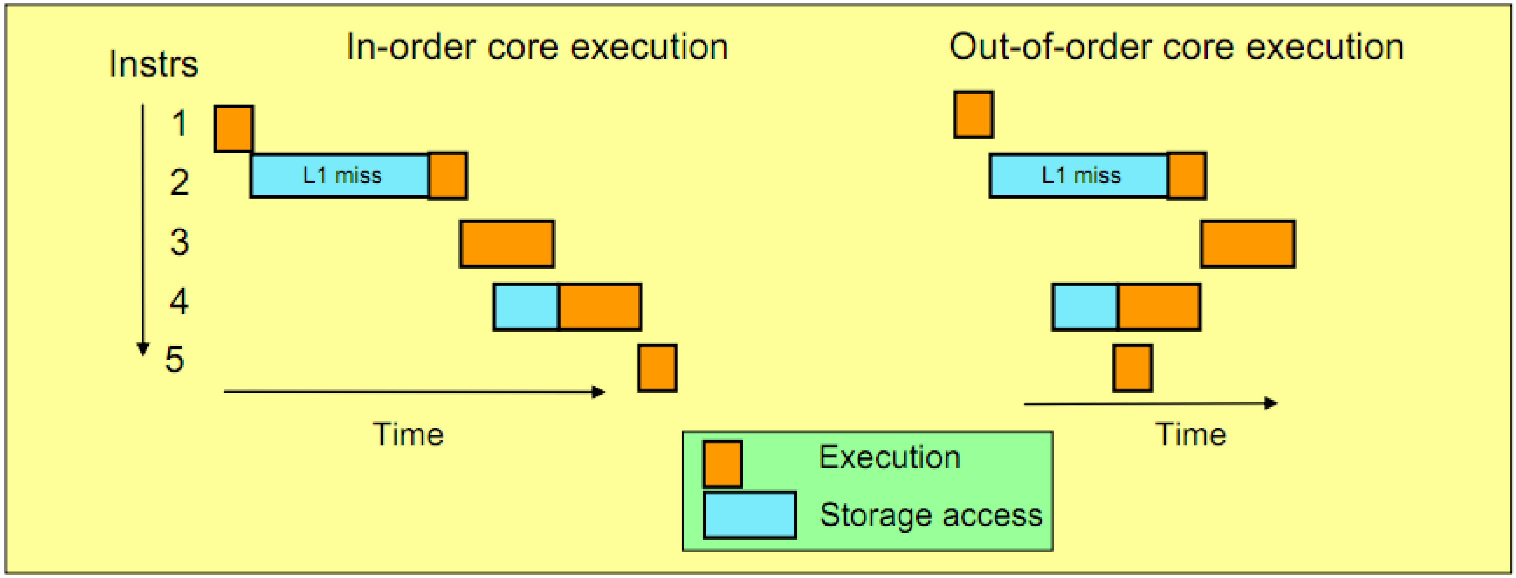
```
# lscpu -a
Architecture:           s390x
CPU op-mode(s):         32-bit, 64-bit
Byte Order:             Big Endian
CPU(s):                 3
On-line CPU(s) list:    0-2
Thread(s) per core:     1
Core(s) per socket:     1
Socket(s) per book:     1
Book(s):                3
Vendor ID:              IBM/S390
BogoMIPS:               18115.00
Hypervisor:             z/VM 6.2.0
Hypervisor vendor:      IBM
Virtualization type:    full
Dispatching mode:       horizontal
L1d cache:              96K
L1i cache:              64K
L2d cache:              1024K
L2i cache:              1024K
```

- Change CPU configuration, e.g. disable a CPU

```
# chcpu -d 2
CPU 2 disabled
```

Linux on System z features – Compiler toolchain

- **zEnterprise 196 exploitation (gcc 4.6)** 6.1 11.2
  - Use option -march=z196 to utilize the new instructions added with z196
  - Use -mtune=z196 to schedule the instruction appropriate for the new out-of-order pipeline of z196
  - Re-compiled code/apps get further performance gains through 110+ new instructions



Future Linux on System z Technology

Software which has already been developed
and integrated into the upstream Linux Kernel
- but is **not yet available** in any
Enterprise Linux Distribution

LNXHC – Linux Health Checker

- The Linux Health Checker is a command line tool for Linux.
- Its purpose is to identify potential problems before they impact your system's availability or cause outages.
- It collects and compares the active Linux settings and system status for a system with the values provided by health-check authors or defined by you. It produces output in the form of detailed messages, which provide information about potential problems and the suggested actions to take.
- The Linux Health Checker will run on any Linux platform which meets the software requirements. It can be easily extended by writing new health check plug-ins.
- The Linux Health Checker is an open source project sponsored by IBM. It is released under the Eclipse Public License v1.0
- http://lnxhc.sourceforge.net/

System z kernel features – Core kernel

- **Add support for physical memory > 4TB (kernel 3.3)**
  - Increase the maximum supported memory size from 4TB to 64TB.
  - Memory sizes large than 4TB require a 4-level page table
  - Makes memory accesses by the kernel slightly slower, the kernel will automatically use a 3-level page table for memory sizes <= 4TB
- **Add support for System z hardware counters (kernel 3.4)**
  - Enables hardware counters for Linux running in an LPAR
  - Available counters are: basic counter set, problem-state counter set, crypto-activity counter set, extended counter set with System z10
  - System zEC12 counter set with kernel version 3.7
- **BPF JIT compiler for System z (kernel 3.7)**
  - The Berkeley Packet Filter is an interface and a language definition that allows to pass a filter to the kernel to select network packets to send on a socket
  - The BPF JIT compiler in the kernel translates the interpreted BPF code to System z code.
  - A secondary use of the BFP language is system call filtering.

System z kernel features – Core kernel

- **Expose CPU cache topology in sysfs (kernel 3.7)**
  - Add an interface to expose the CPU cache topology to user space.
  - System z only provides information about CPU caches which are private to a CPU, information about shared caches is not exposed.

- **Add page table dumper (kernel 3.7)**
  - Add a sysfs interface to read the current layout of the kernel address space.
  - Useful information for the kernel developer.

- **Transparent huge page support (kernel 3.7)**
  - Make the common code transparent huge page support available for Linux on System z.
  - With THP 1MB pages will be used to back normal anonymous memory mappings.
  - Any application will benefit from using huge pages.
  - Not as effective as using the large pages directly, no memory savings for page tables due to huge page splitting.

System z kernel features – Core kernel

- **Kernel compile & disassemble support for zEC12 (kernel 3.8)**
  - Add new instructions to the kernel disassembler and allow compiling with -march=zEC12
- **Implement write protection based dirty page detection (kernel 3.8)**
  - Convert dirty page detection from the change-bit in the storage key to a fault based method. An unmodified page is now always mapped read-only.
  - Due to dirty page accounting for memory mappings no additional faults are necessary
  - Reduces the amount of storage key operations which improves performance
- **Add robustness against missing interrupts to non-path-grouped internal IO requests (kernel 3.8, s390-tools 1.22)**
  - Improve the Linux behavior in case of a missing interrupt during path grouping
- **Add channel ID sysfs attribute (kernel 3.10)**
  - Add an attribute to each channel-path description with the channel-ID of the path

System z kernel features – Storage FICON / FCP

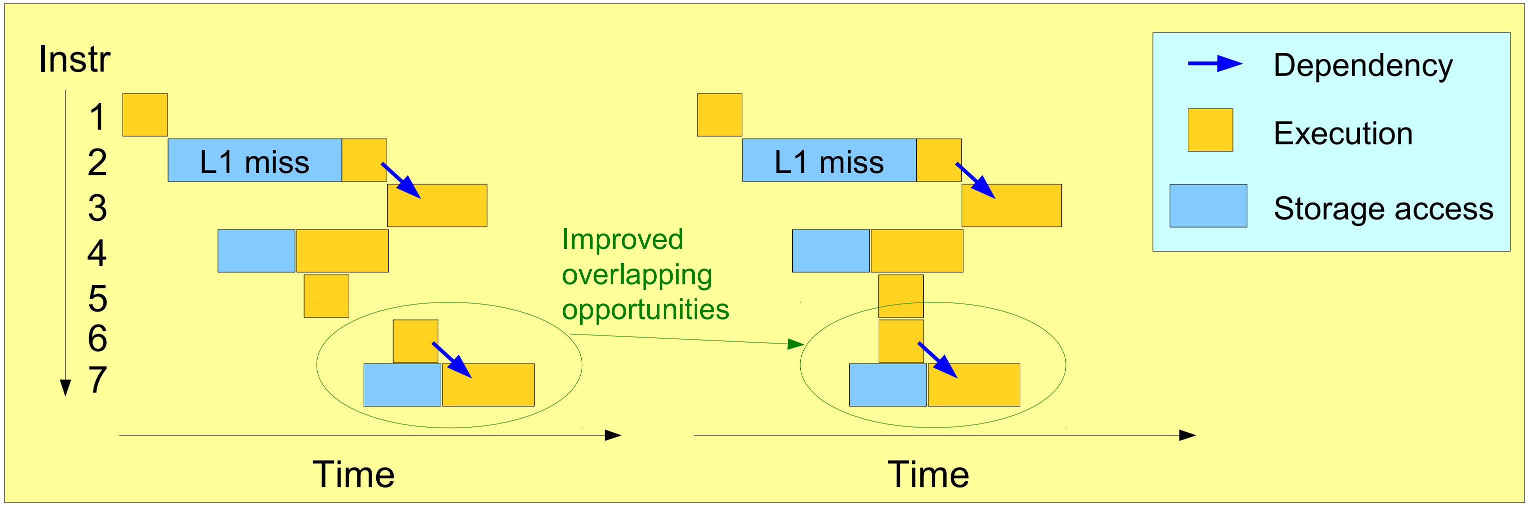
- **FCP support for DIF/DIX (kernel 3.2)**
  - End to end data checking (aka data integrity extension) is no longer experimental.
  - Can be used with either direct I/O with a file system that fully supports end-to-end data consistency checking. Currently XFS only.
- **No automatic port rescan on events (kernel 3.7)**
  - The rescan of a zfcp port following a fabric change event can cause high fabric traffic, especially when many Linux images share an FCP channel over multiple subchannels with NPIV enabled. This can lead to errors due to timeouts.
  - Ports are still scanned when the adapter is set online and on manual user triggered writes to the port_rescan sysfs attribute.
- **Safe offline interface for DASD devices (kernel 3.8, s390-tools 1.21)**
  - Gracefully complete all outstanding I/O requests before a DASD is set offline.
- **Improve speed of dasdfmt (kernel 3.10)**
  - Reorganize format I/O requests and enable usage of PAV.

System z kernel features – Networking

- **Add support for AF_IUCV HiperSockets transport (kernel 3.2)**
  - Use HiperSockets with completion queues as transport channel for AF_IUCV sockets
- **Allow multiple paths with netiucv between z/VM guests (kernel 3.3)**
  - Adds support for multiple point-to-point netiucv interfaces between guests
  - Speed up netiucv by using parallel IUCV paths.
- **Add query OSA address table support (kernel 3.4)**
  - The new qethqoat command queries the OSA address table and displays physical and logical device information.
  - Viewing the OSA address table is a low level diagnostic requirement.

Linux on System z features – Compiler toolchain

- **zEnterprise zEC12 exploitation (gcc 4.8)**
  - New compiler options in support of the zEC12 CPU
    - -march=zEC12 to utilize the instructions added with zEC12
    - -mtune=zEC12 to schedule the instructions appropriate for the pipeline of zEC12
  - zEC12 comes with new instructions:
    - Transactional Memory support
    - Improved branch instructions

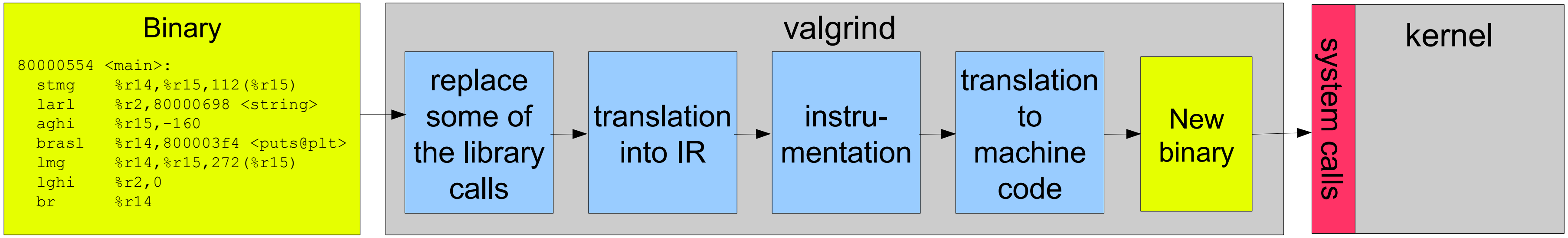


System z toolchain

- **Oprofile z196 hardware customer mode sampling (kernel 3.3)**
  - Extend the hardware sampling to support z196.
- **Oprofile zEC12 hardware sample support (kernel 3.10)**
  - Extend the hardware sampling to support zEC12
- **Valgrind System z support**
  - Valgrind is a generic framework for creating dynamic analysis tools and can be used for memory debugging, memory leak detection and profiling (e.g. cachegrind)
  - Valgrind is in essence a virtual machine using just-in-time (JIT) compilation techniques
  - Memory debugging is available with Valgrind version 3.7.0

Valgrind System z support

- Several tools, selected with --tool=xxx
  - Memcheck (default): detects memory and data flow problems
  - Cachegrind: cache profiling
  - Massif: heap profiling
  - Helgrind: thread debugging
  - DRD: thread debugging
  - None: no debugging (for valgrind JIT testing)
  - Callgrind: codeflow and profiling
- System z support since version 3.7 (SLES-11-SP2)
- Backports into 3.6 (SLES-10-SP4, RHEL6-U1)



```
80000554 <main>:
  stmg    %r14,%r15,112(%r15)
  larl    %r2,80000698 <string>
  aghi    %r15,-160
  brasl   %r14,800003f4 <puts@plt>
  lmg     %r14,%r15,272(%r15)
  lghi    %r2,0
  br      %r14
```

s390-tools package: what is it?

- s390-tools is a package with a set of user space utilities to be used with the Linux on System z distributions.
  - It is **the** essential tool chain for Linux on System z
  - It contains everything from the boot loader to dump related tools for a system crash analysis .
- This software package is contained in all major (and IBM supported) enterprise Linux distributions which support s390
  - RedHat Enterprise Linux version 4, 5, and 6
  - SuSE Linux Enterprise Server version 9, 10, and 11
- Website: http://www.ibm.com/developerworks/linux/linux390/s390-tools.html
- Feedback: linux390@de.ibm.com

s390-tools package: the content

chccwdev
chchp
chreipl
chshut
chcrypt
chmem

CHANGE

lscss
lschp
lsdasd
lsluns
lsqeth
lsreipl
lsshut
lstape
lszcrypt
lszfcp
lsmem

DISPLAY

dasdfmt
dasdinfo
dasdstat
dasdview
fdasd
tunedasd

DASD

mon_fsstatd
mon_procd
ziomon
hyptop

MONITOR

ip_watcher
osasnmpd
qetharp
qethconf

NETWORK

tape390_display
tape390_crypt

TAPE

dbginfo
dumpconf
zfcpdump
zfcpdbf
zgetdump
scsi_logging_level

DUMP
&
DEBUG

vmconvert
vmcp
vmur
cms-fuse

z/VM

cpuplugd
iucvconn
iucvtty
ts-shell
ttyrun

MISC

zipl

BOOT

Kernel news – Common code

- **Linux version 3.5 (2012-07-21)**
  - Filesystem improvements: ext4 metadata checksums, btrfs latency reduction
  - Uprobes: userspace probes
  - Bufferbloat improvements: CoDel queue management
  - TCP: connection repair, early retransmit
  - SCSI over FireWire and USB

- **Linux version 3.6 (2012-09-30)**
  - Btrfs: subvolume quotas, quota groups, snapshot diff, cross-subvolume file clones
  - TCP Fast Open (client side)
  - Bufferbloat fight: TCP small queues
  - Safe swap over NFS/NBD

Kernel news – Common code

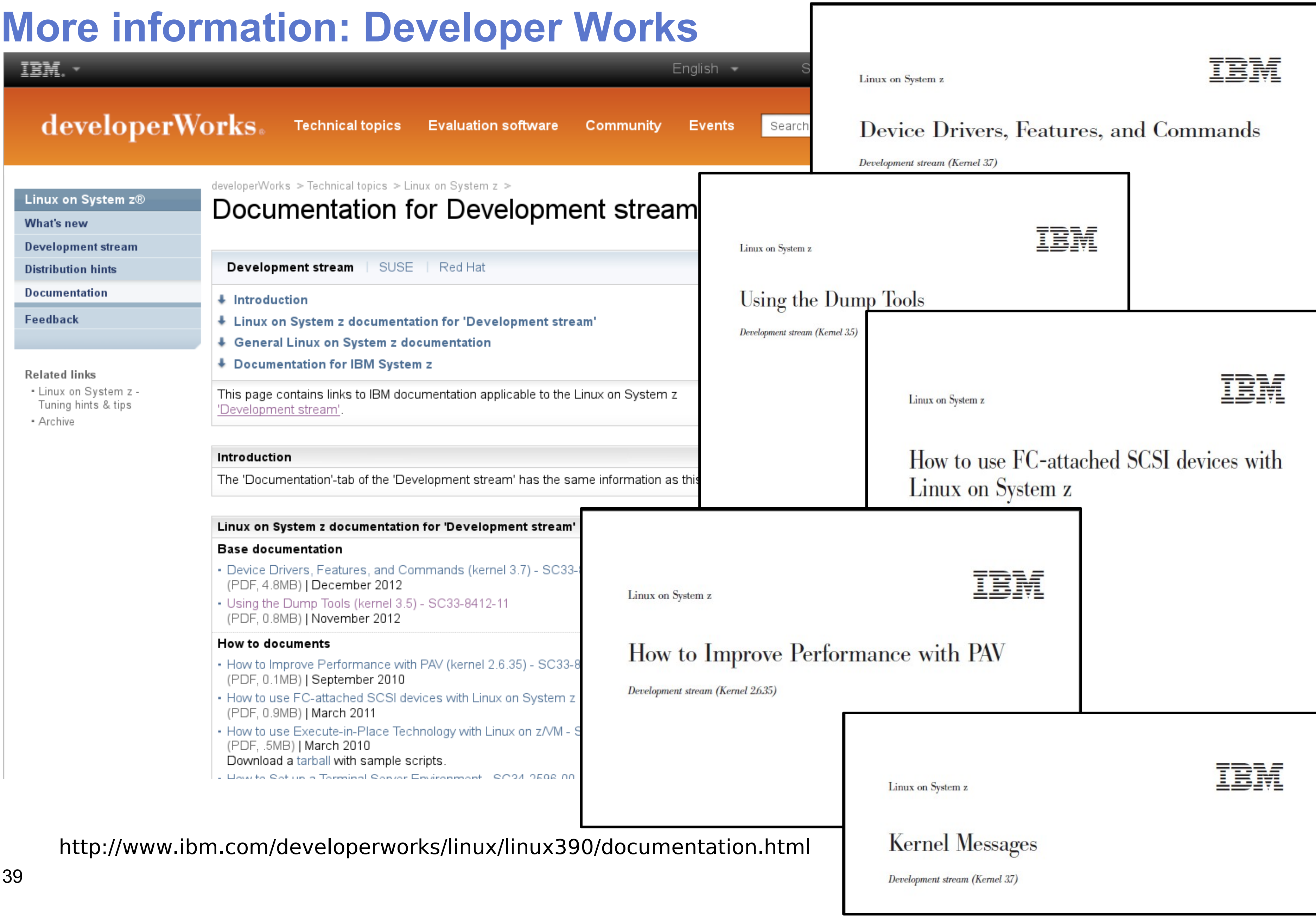
- **Linux version 3.7 (2012-12-10)**
  - ARM multi-platform support, 64-bit support
  - Btrfs updates: fsync speedups, hole punching, per-file no-cow support
  - TCP Fast Open (server side)
  - Experimental SMB 2 protocol support
- **Linux version 3.8 (2013-02-18)**
  - Ext4: embedding of very small files in the inode
  - Btrfs: fast device replacement
  - XFS: log checksums
  - User namespace support completed
  - Huge zero page support
  - Removal of support for 386 processors

Kernel news – Common code

- **Linux version 3.9 (2013-04-28)**
  - SSD cache devices
  - New architectures: Synopsys ARC 700 processors & Meta Imagination processors
  - KVM support in ARM architectures
  - New “suspend-freeze" suspend state
  - Removal of CONFIG_EXPERIMENTAL

- **Linux version 3.10 (2013-06-30)**
  - (Nearly) full tickless operation
  - Bcache, a block layer cache for SSD caching
  - SysV IPC scalability improvements
  - rwsem & mutex locking scalability improvements
  - ARM big.LITTLE support
  - MIPS KVM support

More information: Developer Works



Documentation for Development stream

Development stream | SUSE | Red Hat

- Introduction
- Linux on System z documentation for 'Development stream'
- General Linux on System z documentation
- Documentation for IBM System z

This page contains links to IBM documentation applicable to the Linux on System z 'Development stream'.

Introduction

The 'Documentation'-tab of the 'Development stream' has the same information as this

Linux on System z documentation for 'Development stream'

Base documentation

- Device Drivers, Features, and Commands (kernel 3.7) - SC33-
(PDF, 4.8MB) | December 2012
- Using the Dump Tools (kernel 3.5) - SC33-8412-11
(PDF, 0.8MB) | November 2012

How to documents

- How to Improve Performance with PAV (kernel 2.6.35) - SC33-8
(PDF, 0.1MB) | September 2010
- How to use FC-attached SCSI devices with Linux on System z
(PDF, 0.9MB) | March 2011
- How to use Execute-in-Place Technology with Linux on z/VM - S
(PDF, .5MB) | March 2010
Download a tarball with sample scripts.
- How to Set up a Terminal Server Environment - SC34-2596-00

Linux on System z
IBM
Device Drivers, Features, and Commands
Development stream (Kernel 3.7)

Linux on System z
IBM
Using the Dump Tools
Development stream (Kernel 3.5)

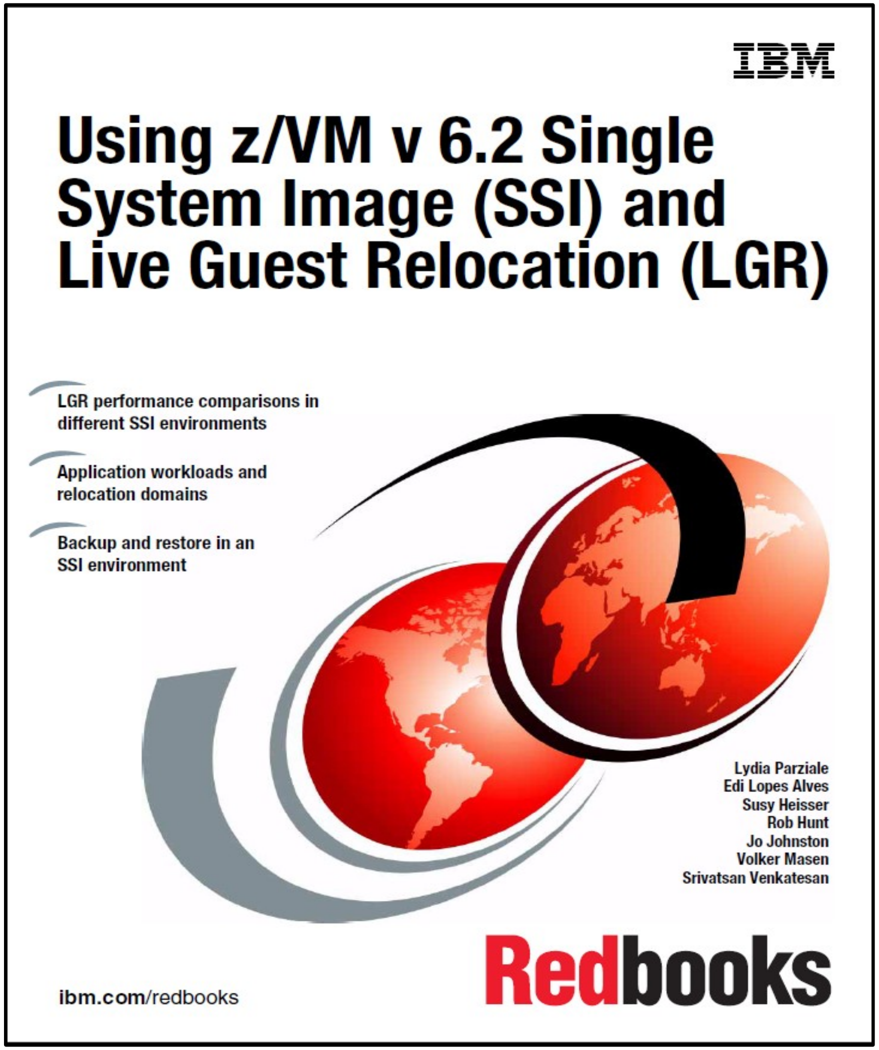
Linux on System z
IBM
How to use FC-attached SCSI devices with Linux on System z

Linux on System z
IBM
How to Improve Performance with PAV
Development stream (Kernel 2.6.35)

Linux on System z
IBM
Kernel Messages
Development stream (Kernel 3.7)

http://www.ibm.com/developerworks/linux/linux390/documentation.html

New Redbooks



IBM

Using z/VM v 6.2 Single System Image (SSI) and Live Guest Relocation (LGR)

LGR performance comparisons in different SSI environments

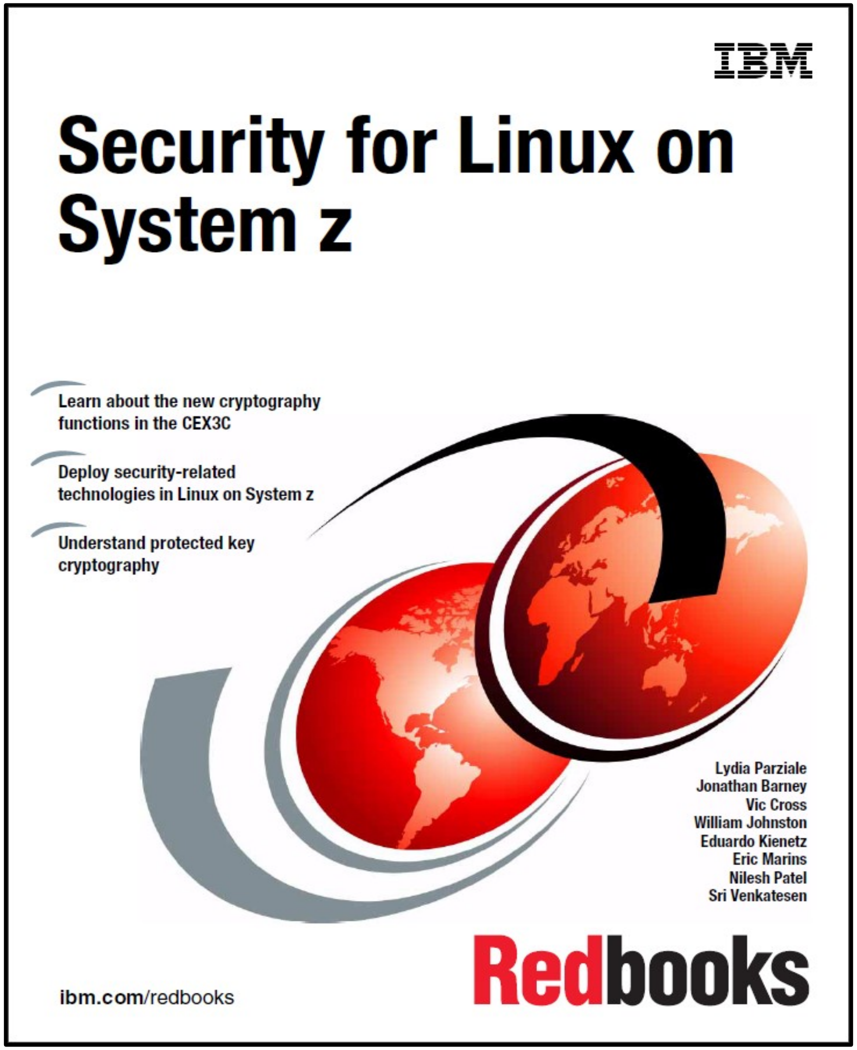
Application workloads and relocation domains

Backup and restore in an SSI environment

Lydia Parziale
Edi Lopes Alves
Susy Heisser
Rob Hunt
Jo Johnston
Volker Masen
Srivatsan Venkatesan

ibm.com/redbooks

Redbooks



IBM

Security for Linux on System z

Learn about the new cryptography functions in the CEX3C

Deploy security-related technologies in Linux on System z

Understand protected key cryptography

Lydia Parziale
Jonathan Barney
Vic Cross
William Johnston
Eduardo Kienetz
Eric Marins
Nilesh Patel
Sri Venkatesen

ibm.com/redbooks

Redbooks

Visit http://www.redbooks.ibm.com

Questions?

Martin Schwidefsky

Linux on System z
Development

IBM

Schönaicher Strasse 220
71032 Böblingen, Germany

Phone +49 (0)7031-16-2247
schwidefsky@de.ibm.com