

OS Circular: Internet Client for Reference

Kuniyasu Suzaki, Toshiki Yagi, Kengo Iijima, and Nguyen Anh Quynh
– National Institute of Advanced Industrial Science and Technology, Japan

ABSTRACT

OS Circular is a framework for Internet Disk Image Distribution of software for virtual machines, those which offer a “virtualized” common PC environment on any PC. OS images are obtained via the stackable virtual disk “Trusted HTTP-FUSE CLOOP”.

The system is designed to utilize Mirror servers and Proxies for highly-scalable worldwide deployment. OS Circular easily and efficiently handles both partial and periodic OS updates, including a rollback facility to ease experimentation with new OS images that might not be ready for production. This paper describes the design of OS Circular and the techniques to reduce the network traffic for quick downloading and booting.

Introduction

We have developed “OS Circular” [1, 2], a technique for booting any OS on an anonymous PC over the internet. It enables selection of OS images and can facilitate a reference installation. This means that an OS image can be “installed” to ease testing and feasibility testing of new functionality before it is installed on the local hard drive for production use. The previous (reference) version of an OS image stores old application software and enables opening of files in any previously supported format. In a world of frequent security updates, this environment dramatically enhances the ease of testing OS service packs.

Virtual machines host a common PC environment on a standard PC. Software to host virtual machines is easy to obtain and use, with open source packages QEMU, KQEMU, KVM, and Xen in addition to free versions of VMware and Virtual Box. Recent performance of virtual machines has quite low overhead and enables use of guest OSes without high resource utilization. Furthermore, newer x86 architecture CPUs have a virtualization extension (Intel’s VT or AMD’s SVM) that promotes even better virtual machine software (including a mode for trapping sensitive instructions). Both KVM and Xen-HVM use this virtualization extension and offer full virtualization.

One current challenge for virtual machines is the scheme used to share virtual disks on the Internet [3, 4]. These virtual disks represent both OSes and application software and should be updated periodically (e.g., for security). The previous disk image should be archived efficiently and might be used for replaying the old OS image when an update is unsatisfactory. New master disk images could be shared with a potentially massive number of users, and efficiently doing that is one of the primary goals of OS Circular.

OS Circular is designed to utilize existing infrastructures in order to enable global deployment and

minimize maintenance cost. As a client-centric system, OS Circular reduces server load by limiting server functionality to file distribution. OS Circular uses HTTP for file distribution because of the ease of accessing Web hosting services (including exploitation of mirror servers and cache proxies).

The client PC checks data for validity during downloads (which are provided by Trusted HTTP-FUSE CLOOP, a stackable virtual disk). Trusted HTTP-FUSE CLOOP includes support for update of virtual disks. OS Circular uses full virtualization (to avoid exploits due to the kernel of the guest OS being insecure). It also includes an automatic security update service.

This paper details OS Circular’s design and implementation paradigm. We mention related work, then describe the virtual machine as an abstraction layer. The next section describes the requirement of virtual disks. Subsequently, we discuss details of the Trusted HTTP-FUSE CLOOP and the current implementation and performance of OS Circular. Finally, we discuss future plans and conclude.

Related Work

Virtual disks are a popular means for distribution of ready-to-use OS images for virtual machines. The OS Zoo project [5] distributes many virtual disk files for QEMU (an open source machine emulator and virtualizer). This eases experimentation with various operating systems since installation is relatively simple. However, virtual disks often must be treated in some ways as a single – potentially extremely large – file. Downloading hundreds of megabytes can take quite a while. Furthermore, even the update of a single bit requires the (time-consuming) reconstruction of the entire virtual disk, a real detriment when an urgent security update is required.

FLOZ (Free Live OS Zoo) [6] is an interesting derivative work of OS Zoo project. It enables booting

of many OSES via a Web browser. FLOZ runs a QEMU virtual machine on the server and transfers the visual console of the QEMU to a Web browser on some client. While very innovative for OS testing, performance and scalability are limited due to its server-centric design which also suffers greatly as Internet network latency (for file access) increases. Furthermore, OSES hosted on FLOZ do not allow Internet connections due to server network resource and security concerns.

“Collective” [3] and “Ventana” [4] also propose client-centric systems. They run virtual machines on a client and download “diffs” of updated OS images.

- Collective uses VMware’s COW (Copy On Write) feature for partial update. This is a disk block level mapping and thus handles any filesystem format. Updated data is saved to a file, making it easy to deal with. Unfortunately, it is difficult to map many COW images. The developers established the company called Moka5 to offer LivePC [7], an improved version of Collective.
- Ventana is a virtualization-aware filesystem with versioning, access control, and disconnected operation. It is a management system for virtual disks and offers a customized view of a disk image for each user. Unfortunately, it is based on NFS and it is not designed for a massive number of anonymous users on the Internet.

Virtual Machines

Full virtualization offers a sort of abstraction layer for OSES that provides a common virtualized PC environment on an anonymous PC. We can install a guest OS with its normal installer, update the OS with its usual package management software, and migrate it to other PCs via virtual disks. Various virtualizers do this in different ways: VMware is used on SoulPad [8], VAT (Virtual Appliance Transceiver) of Collective [3], and Internet Suspend/Resume [9].

Device Model

It is critical that full virtualization offer the same device model that a guest OS expects. When this virtualization is present, the guest OS need only prepare drivers for abstracted devices – not for any specific model or type of disk.

QEMU-DM (Device Model) is becoming popular in the open source community. It assumes RealTek RTL8029 for NIC, Cirrus Logic GD5446 for Video Card and a few others. Both Xen and KVM also support QEMU-DM and guest OS only have to support them.

The differences among IA32 architectures cause no problems for the emulators because recent OSES offer common i386 packages for IA32 which run on Pentium Pro or later IA32 CPUs. QEMU, KQEMU and KVM offer the Pentium II architecture for guest

OSES but Xen-HVM offers the same architecture as the real CPU upon which it is running. These differences are mitigated by the “universal” packages of OSES.

VM Loader

OS Circular requires both a virtual machine and a stackable virtual disk. For convenience, we have developed and offer a 1CD Linux installation which includes both of these.

This host OS supports the drivers for real devices. SoulPad and VAT of Collective use KNOPPIX as the host OS, because KNOPPIX has an “AutoConfig” function that automatically detects available devices at boot time and loads the appropriate drivers. The combination of AutoConfig and full virtualization of the host OS acts as a virtual machine loader.

We offer VMKNOPPIX, a collection of virtual machine software on KNOPPIX, as a VM Loader. It includes QEMU, KQEMU, KVM, and Xen-HVM, each of which offers full virtualization. Xen-HVM and KVM require a CPU with the virtualization extension (Intel VT or AMD-SVM) but QEMU and KQEMU run on any IA32 architecture. KNOPPIX acts as the host OS and has drivers for almost any PC.

Requirements for Virtual Disks

Virtual disks should offer features such as versioning, globalization, and security; see [4] for the main requirements. These requirements motivate the design for Trusted HTTP-FUSE LOOP.

Versioning

Versioning of virtual disks is very desirable:

- non-persistent versioning is used for “undo” of operations
- while persistent versioning is used for “rollback” of OS image

Versioning is also important for sharing and customization of virtual disks.

Some virtual machine software supports the “undo” function, including the “non-persistent” mode of VMware and “CopyOnWrite” of QEMU and User-ModeLinux. Xen uses DeviceMapper of Linux for non-persistent versioning. Virtual Disks should offer persistent versioning.

Trusted HTTP-FUSE CLOOP has the same versioning scheme as Venti [10], Plan9’s archival storage system that permanently stores data blocks (blocks which comprise the data of a filesystem – the filesystem’s structure is independent of the block storage). Each block is stored in a file whose name is its SHA1 hash. The system enforces a write-once policy since no other data block should ever have the same hash. Duplicate data is easily identified (since it has the same hash) and a given data block is stored only once. Data blocks cannot be removed, making the system ideal for permanent or backup storage. Unfortunately, Venti requires a special protocol for accessing the filesystem, and this limits its scalability.

Trusted HTTP-FUSE CLOOP saves data blocks in much the same way as Plan9 and utilizes HTTP to distribute them.

Globalization

One goal of OS Circular is the provision of virtual disks that can be shared via Internet. This does not connote the complete downloading of a virtual disk file but rather access to any part of a disk as requested.

The “Remote Block Device” scheme meets this requirement and is found in many implementations, including iSCSI, AOE (ATA over Ethernet), iFCP, and more. Unfortunately, all of these use special protocols and require special daemons on the server. Most of them are designed for use on a high speed LAN and aren’t as useful on the slower Internet. Trusted HTTP-FUSE CLOOP re-constructs a virtual disk with the partial block files which are downloadable via HTTP.

Virtual disks also require a “disconnect operation” to achieve Mobile Computing. The “AFS” and “Coda” filesystems deal with the disconnect operation but also require special protocol and daemons for servers. Stateless Linux [11] offers a disconnect operation for a Thin Client PC. Stateless Linux runs with network storage or snapshot image saved local storage. Block files of Trusted HTTP-FUSE CLOOP are also saved to a local storage and re-usable (when whole block files are saved in a local storage, a network connection is not required).

Security

Basically, security management is independent of the virtual disk implementation. Security of the kernel and applications should be managed by security software or package manager. A virtual disk makes only

the commitment to keep the integrity of its contents. Probably the biggest part of the security for a network virtual disk is the prevention of intrusion. One way to prevent such an intrusion is the use of secure communication, but this requires a fixed server which limits scalability. Trusted HTTP-FUSE CLOOP adopts a client-centric contents validation mechanism with a driver that checks the validity of block contents when mapped to the virtual disk. It allows distributing block data by anonymous servers.

Trusted HTTP-FUSE CLOOP

KNOPPIX’s CLOOP (Compressed Loop back device) spawned the big picture idea for HTTP-FUSE CLOOP’s filesystem. KNOPPIX saves block device data to a file in order to reduce disk consumption (though a CLOOP file is still big). Traditional CD-KNOPPIX requires about 700 MB and must be treated as one file, slowing downloads. When even just a single bit is updated, a big CLOOP file must be rebuilt. Furthermore, CLOOP has no security protection.

To mitigate these problems, we adopted the block management style of Venti [10]. Data on a block device is partitioned into data blocks of a fixed size, compressed, and saved. Block files are treated as network transparent between local and remote machines, with local storage acting as a cache. The downloaded block files are validated by their SHA1 hash value. This yields these features:

- A block file is made of split, compressed block device blocks (default size is 256 KB) blocks (The original CLOOP’s split block size was 64 KB which was too small and created too many files).

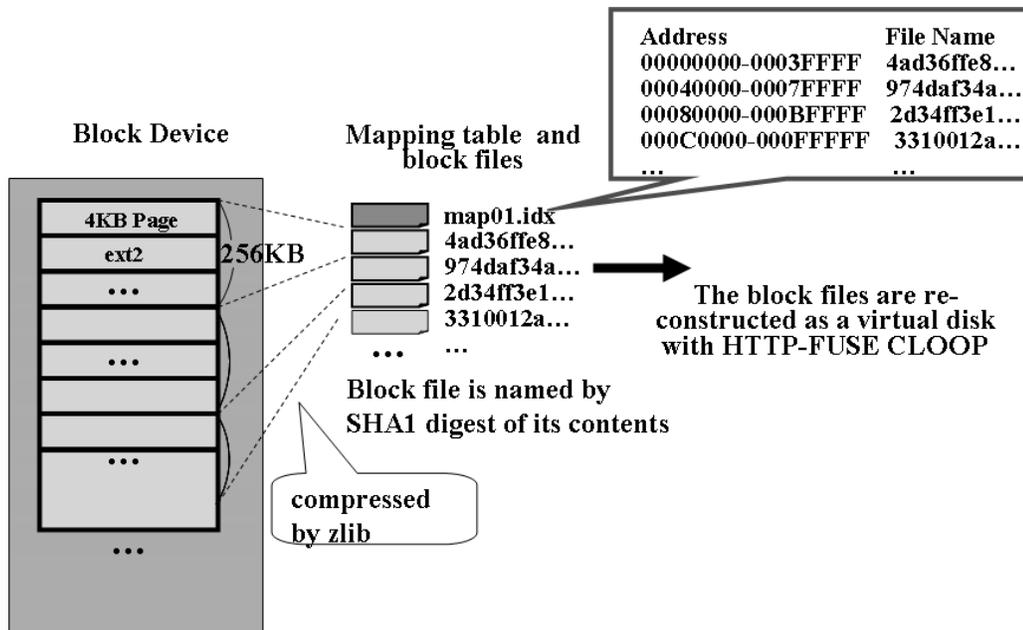


Figure 1: Creation of block files from OS image.

- Block files are mapped to a loopback device with a mapping table file.
- This mapping for block files is performed when a relevant read request is issued. After mapping, the block file is erasable from local (cache) storage, since it can be downloaded anew any time it is needed.
- The name of a given block file is the value of its SHA1 hash with all the good properties listed above.
- Block files are downloaded from the HTTP server because HTTP is expected to be strong file delivery infrastructure as demonstrated by mirror servers and proxy servers.
- A proxy server has a size limitation on cached file sizes so block files should be smaller than this size.
- When mapping a block file to the loopback device, the block's contents are hashed into a SHA1 file name which is listed in the mapping table file.
- The block device is Partially Updatable: When an application is updated on the original block device, relevant block files and the mapping file are renewed; cache block files related to the non-updated block are reusable.
- "FUSE" (File system in USEr-space) [12] is used to implement the virtual loopback device.

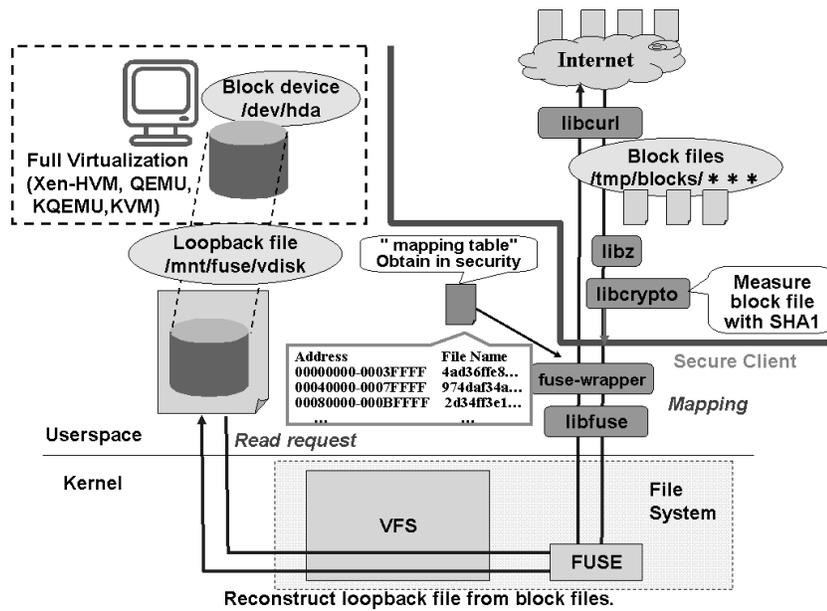


Figure 2: Diagram of Trusted HTTP-FUSE CLOOP.

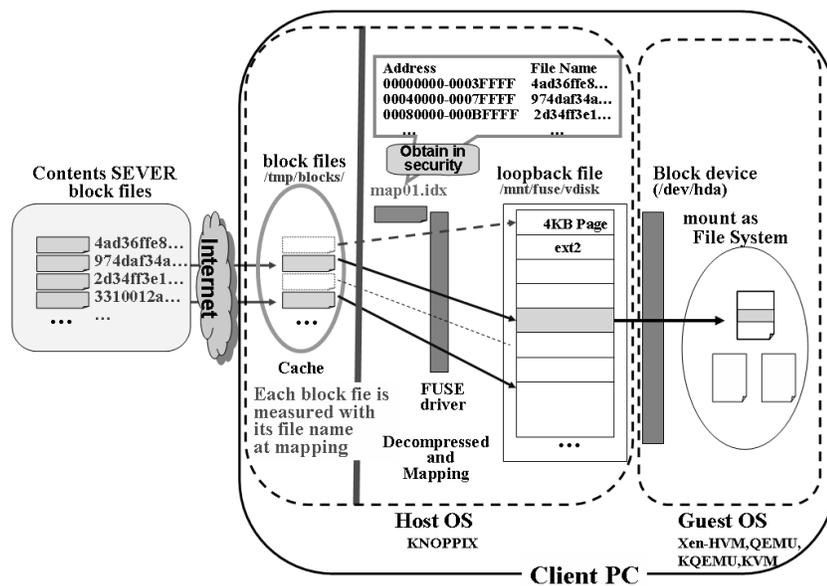


Figure 3: Block mapping of Trusted HTTP-FUSE CLOOP.

Figure 1 depicts the creation of block files and the mapping table file “map01.idx” which is also made from a block device which includes root filesystem. Each block file is named by its SHA1 hash.

Figure 2 shows the block diagram of Trusted HTTP-FUSE CLOOP. The loopback device is mapped as a normal block device on a virtual machine. The main program is implemented as a part of FUSE wrapper program. A maintains the validity of the virtual block and must be distributed in secure way. The mapping table file is used to set up Trusted HTTP-FUSE CLOOP. When a read request is issued, Trusted HTTP-FUSE CLOOP driver searches the relevant block file using the mapping table. When the relevant file exists on a local storage, that file is used; otherwise, the file is downloaded from Internet.

Block file elements are downloaded from an HTTP server with “libcurl”, the Client for URLs library. Each downloaded file is decompressed by “libz”, hashed by “libcrypto”, and logged into /var/log/fs_wrapper_PID.log. Invalid file transfers (due to network errors or security breaches) are detected using the SHA1 hash. Figure 4 shows an example that detects a defective block file. The downloaded block file is first stored at local storage. If the storage space is insufficient (more than 80% used), previously downloaded files are removed by LIFO (a water mark algorithm). Figure 3 shows Trusted HTTP-FUSE CLOOP from the viewpoint of block mapping.

Partial Update By Adding Block Files

The mapping table handles block addressing. Any update of HTTP-FUSE CLOOP is performed by adding updates both to the block files and to the mapping table.

Unchanged (local) block files become reusable for caching. To achieve this function, the HTTP-FUSE CLOOP filesystem treats block-unit update as an EXT2 filesystem. The “iso9660” filesystem turns to be unsuitable for HTTP-FUSE because partial update of an iso9660 file changes the location of subsequent blocks. As usual, updated blocks are saved to a file named by the SHA1 hash of the block. Collision of file name is extremely rare (this being the goal of the SHA1 hash). In the unusual event of a collision, we can check and repair the problem before uploading the block files.

Figure 5 shows an example of an HTTP-FUSE CLOOP update which creates a new mapping table file “map02.idx” and associated block files. This is particularly useful when updating KNOPPIX applications, especially in the case of security updates. Furthermore, we can rollback to an old filesystem by reinstating the previous mapping table “map01.idx” and block files.

Optimization for Download

Trusted HTTP-FUSE CLOOP is sensitive to network latency because small block files are downloaded in the order the blocks are requested.

Our original implementation suffered performance problems at boot time until we implemented two new functions: “netselect” and “DLAHEAD” (download ahead).

The “netselect” functionality searches for the lowest latency download site among candidates (through judicious use of “ping”). We arranged the HTTP sites to be dispersed across the global Internet, a spread that also foster load-balancing of HTTP services.

```
1150452051.109: #00000000(845b31ded38e15c1fa8febf97fe0781f23af98c3) :missed.
1150452051.112: #00000000(845b31ded38e15c1fa8febf97fe0781f23af98c3) :hits.
1150452051.112: #00000001(166cbaedbb1cc836e7c95d7d9943efde5a53829e) :missed.
1150452051.113: #00000002(29c4e363dbad648072751ca1f856e5780dd2981d) :missed.
1150452051.114: #00000003(fa8ad05b713a9cf8a701636ca6c353dc58fd6b6fd) :missed.
1150452051.114: #00000004(1f82a543fa9310c44eff6a13618beca3cacffc12) :missed.
1150452051.128: #00000004(1f82a543fa9310c44eff6a13618beca3cacffc12) :hits.
1150452051.128: #00000005(916f62a6e2caedc1279a0a74975a406ddb60ec25) :missed.
1150452051.129: #00000006(19111dfc877a4fe241e125d10176d85a99b4bb86) :missed.
1150452051.130: #00000007(950c1d7623b374f8e03309a93041f5adfa3af80f) :missed.
1150452051.130: #00000008(486472b0ee27157d755bd59d623179cfc0034747) :missed.
```

Figure 4a: Correct downloading of block files.

```
1150452375.989: #00000000(845b31ded38e15c1fa8febf97fe0781f23af98c3) :missed.
1150452375.993: #00000000(845b31ded38e15c1fa8febf97fe0781f23af98c3) :hits.
1150452375.993: #00000001(166cbaedbb1cc836e7c95d7d9943efde5a53829e) :missed.
1150452375.994: #00000002(29c4e363dbad648072751ca1f856e5780dd2981d) :missed.
1150452375.995: #00000003(fa8ad05b713a9cf8a701636ca6c353dc58fd6b6fd) :missed.
1150452375.996: #00000004(1f82a543fa9310c44eff6a13618beca3cacffc12) :missed.
1150452375.997: #00000004(1f82a543fa9310c44eff6a13618beca3cacffc12) :hits.
1150452375.997: #00000005(916f62a6e2caedc1279a0a74975a406ddb60ec25) :missed.
1150452375.998: #00000006(19111dfc877a4fe241e125d10176d85a99b4bb86) :missed.
E: can't validate block.
```

Figure 4b: Invalid block file is detected.

Figure 4: Log of Trusted HTTP-FUSE CLOOP (/var/log/fs_wrapper_PID.log). The “missed” tag indicates a block requires downloading; “hits” indicates the block file is in local storage.

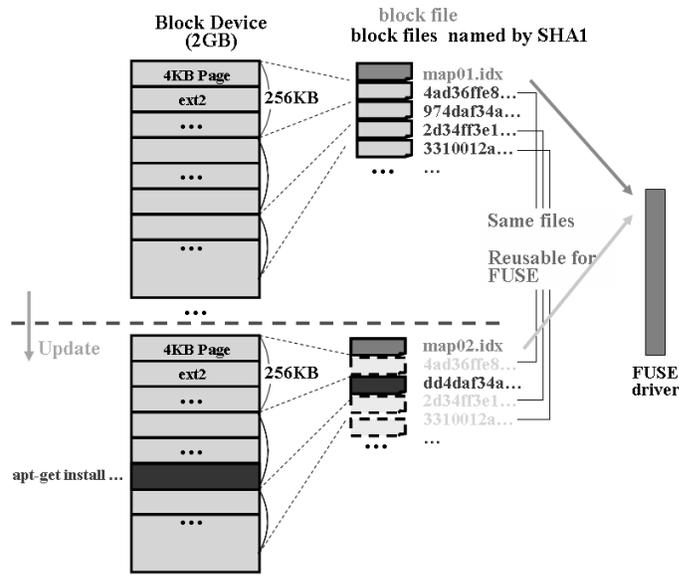


Figure 5: Update of HTTP-FUSE CLOOP.

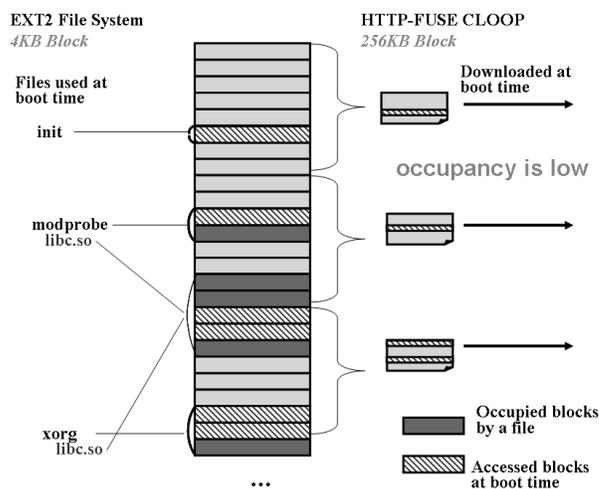


Figure 6: Ext2 filesystem is translated to block files of HTTP-FUSE CLOOP and the occupancy of accessed 4 KB data blocks at boot time.

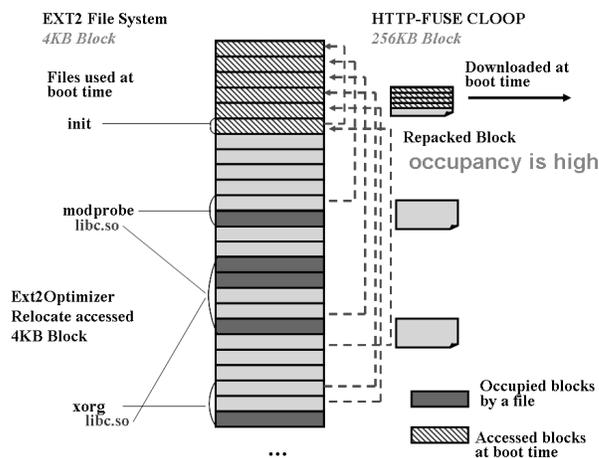


Figure 7: Ext2optimizer repacks the data backs to be formed in line.

DLAHEAD downloads the block files necessary for boot, potentially before they are requested, from a list of required block files contained in the boot profile. While this profile depends on the particular client PC, the difference among varying boot sequences is small with the greatest variation caused by the device drivers themselves. DLAHEAD establishes multiple (default: four) connections in order to download block files in parallel and thus reduces apparent network latency. Downloaded block files are saved to a local storage and work as a cache.

File System Optimization

Any block size mismatch between filesystem and virtual block device causes fragmentation. For example, EXT2 filesystems generally assume a 4 KB blocksize while HTTP-FUSE CLOOP assumes blocks of size 256KB. Even if a single block (4 KB) of a local file is requested, HTTP-FUSE CLOOP downloads the relevant (compressed) 256KB block file and decompresses it. If the next read request isn't included in some already downloaded block files, HTTP-FUSE CLOOP must download and decompress the another block file.

A typical boot sequence and subsequent system operation requires much smaller size blocks than the downloaded block files. This level of "occupancy" depends on the filesystem type, the size of the entire filesystem, the block size, and the stored data. Reference [13] reports the occupancy of block files was 30% for running KNOPPIX 3.8.2 (when the filesystem was EXT2, the volume was 2 GB, and the block size was 256KB).

We applied "ext2optimizer" [13] to the filesystem for HTTP-FUSE CLOOP to create a profile of the actually accessed physical blocks at boot time. We then repacked those blocks into physical blocks located at the "front" of the file store. The ext2optimizer keeps the structure of ext2 filesystem even when the physical blocks rearranged, so the filesystem works the same as before.

Figures 6 and 7 show the effect of ext2optimizer. Figure 6 is the image in which the normal ext2filesystem is translated to block files of HTTP-FUSE CLOOP with accessed data blocks scattered throughout various block files. This requires download of extra data (most 4K blocks are not used in the boot process) and thus increases boot time. Figure 7 shows the effect of ext2optimizer. The accessed data blocks are repacked to increase the efficiency of downloads and reduce boot time.

From the filesystem point of view, ext2optimizer causes fragmentation but reduces download of block files and makes for a quick boot.

Implementation of OS Circular

The guest OS images are distributed by Trusted HTTP-FUSE CLOOP. VMKNOPPIX boots the host OS on a client PC and runs a virtual machine. In the current implementation, Debian GNU/Linux and FreeBSD are bootable on QEMU, KQEMU, KVM and Xen-HVM.

Security Updates

Debian GNU/Linux is supported by a strong community and offers a network update scheme [14] called the APT (Advanced Package Tool) which uses a list of repositories and available packages. If a newer package appears in the repository's list, APT downloads the package and hands the process over to "dpkg" (a medium-level package manager for Debian). The package manager checks the integrity of the package, updates the software, and manages security of the total contents.

Figure 8 details the image of an update on OS circular. When the master OS image on a virtual machine is updated by the "apt-get" command, both a new mapping table file and block files are created. These files are uploaded onto HTTP Servers. The Client PC subsequently sees a list of mapping table files and selects one. Older mapping table files represent less updated OS images. When we want to use an older OS image, the old mapping table file is selected.

We monitored upgrades of Debian packages from 07/Dec/2006 to 11/Dec/2006. A total of 854 packages and a 4 GB virtual disk were translated to 14523 block files (1.9 GB) on 07/Dec/2006. The 104 packages were updated with 3420 block files (335 MB) created on 11/Dec/2006. The new block files and mapping table file were added to HTTP servers and the clients used new OS image by rebooting with the new mapping table file.

Download Sites for OS Circular

Figure 9 shows the global location of OS Circular download sites. We dispersed these sites across the globe to prevent intercontinental connections and provide reasonable latency for downloading block files in US, Europe, and North Asia. Most of these sites are commercial Web hosting services with reasonable prices and a high level of maintenance.

OS Circular has a name resolver which tries to supply close mirror servers. The mirror servers are renamed to the unique sub-domain name which is registered on our DNS, so each client uses DNS lookup at our central site. This DNS is operated by DNS-Balance [15] which resolves the sub-domain name as the nearest mirror server (by using routing information offered by RADB.net, the Routing ASSET Database). It is a server-side solution instead of "netselect" on the client. Figure 10 shows an example of DNS-Balance. The degree of accuracy depends on RADB.net, but intercontinental download is almost always avoided.

DNS-Balance is also used for load balancing and fault-tolerance because it can replace a server at the timeout of keep alive of HTTP1.1.

Performance

We measured the boot time of OS Circular on an IBM/Lenovo Think PAD T60 with a 2 GHz Core2 Duo T7200 and 2 GB Memory. The network latency

was synthesized with a “dumynet” to monitor its effect. The synthesized network delay was tested at both 0 and 30 msec to emulate LAN and ISP environments. We also checked the effect of ext2optimizer and DLA-HEAD, both of which aimed to reduce total traffic and reduce the effect of network latency.

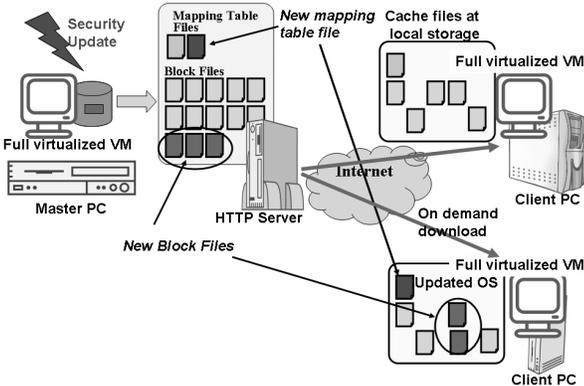


Figure 8: Security update of Trusted HTTP-FUSE CLOOP on OS Circular.

Release	Pkgs	Block Files	Size
Orig. 07Dec2006	884	14,523	1.9 GB
Update 11Dec2006	104	3,430	335 MB

Table 1: The difference of updated blocks.



Figure 9: Download sites for OS circular.

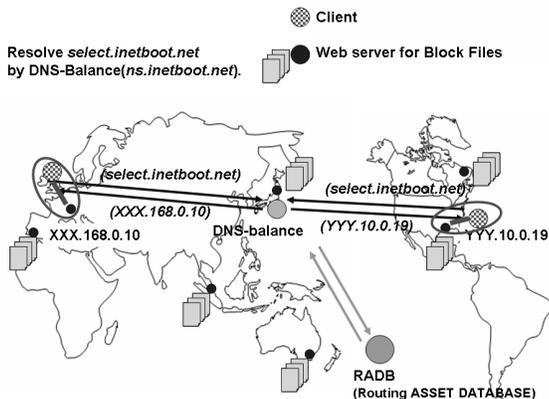


Figure 10: Searching the nearest download site by DNS-Balance.

We compared the boot time on Xen-HVM and KQEMU. Xen-HVM requires the CPU virtualization extension but KQEMU runs on any normal IA32 CPU. The Debian Etch on OS Circular was measured until the gdm (GNOME Display Manager) appeared.

Figures 11 and 12 show the results on Xen-HVM and KQEMU. The three lines show the case of no optimization, ext2optimizer, and ext2optimizer coupled with DLAHEAD. Each line’s statistics commenced when GRUB read the kernel and initrd. “Boot” ends when the gdm appears.

Xen-HVM vs. KQEMU

The differences in boot times were not so big on Xen-HVM and KQEMU. In general, Xen-HVM is faster than KQEMU, perhaps twice as fast when the latency is 0 (with no optimization). This advantage was reduced, though, when both ext2optimizer and DLAHEAD were applied. For 30 msec latency, the difference was small and the boot time mostly depended on these optimizations. The results showed the importance of the access speed of virtual disk.

Magnitude of Traffic

The original block device was a 3 GB ext3 formatted disk and included 2 GB of Debian packages. The results show that the original filesystem contains blocks scattered through the filesystem. The ext2optimizer solved this problem.

Without optimization, downloaded disk (network) traffic measured 68 MB on Xen-HVM and 58 MB on KQEMU (the difference was caused the differing CPU architectures, Core2 Duo of Xen-HVM and Pentium-II of KQEMU) and the BIOS. The BIOS of Xen-HVM is not a fully supported function of QEMU.

We created a disk access profile for Xen-HVM, applied ext2optimizer, and recreated the block files. This reduced the downloaded block files for Xen-HVM to 40% of its original (from 68MB to 27MB) HVM and for KQEMU to 47% of its original (from 58MB to 27MB).

DLAHEAD downloads soon-to-be-accessed block files with four parallel downloading connections. DLA-HEAD usually downloads block files before they are requested but sometimes it can’t keep up. At that time, some block files are downloaded twice, increasing the traffic. Figures 11 and 12 showed these effects were small, less than 30 MB. The network latency reduction techniques were effective and enabled a quick boot.

Boot Time

Tables 2 and 3 show the boot times for Xen-HVM and KQEMU. To compare them, we added the boot time of the USB Memory (normal), which is normally installed Debian on the USB Memory, and the boot time of the cached block files on the USB memory (cache).

The difference between normal and cached booting is the effect of compressed block files. In Tables 2

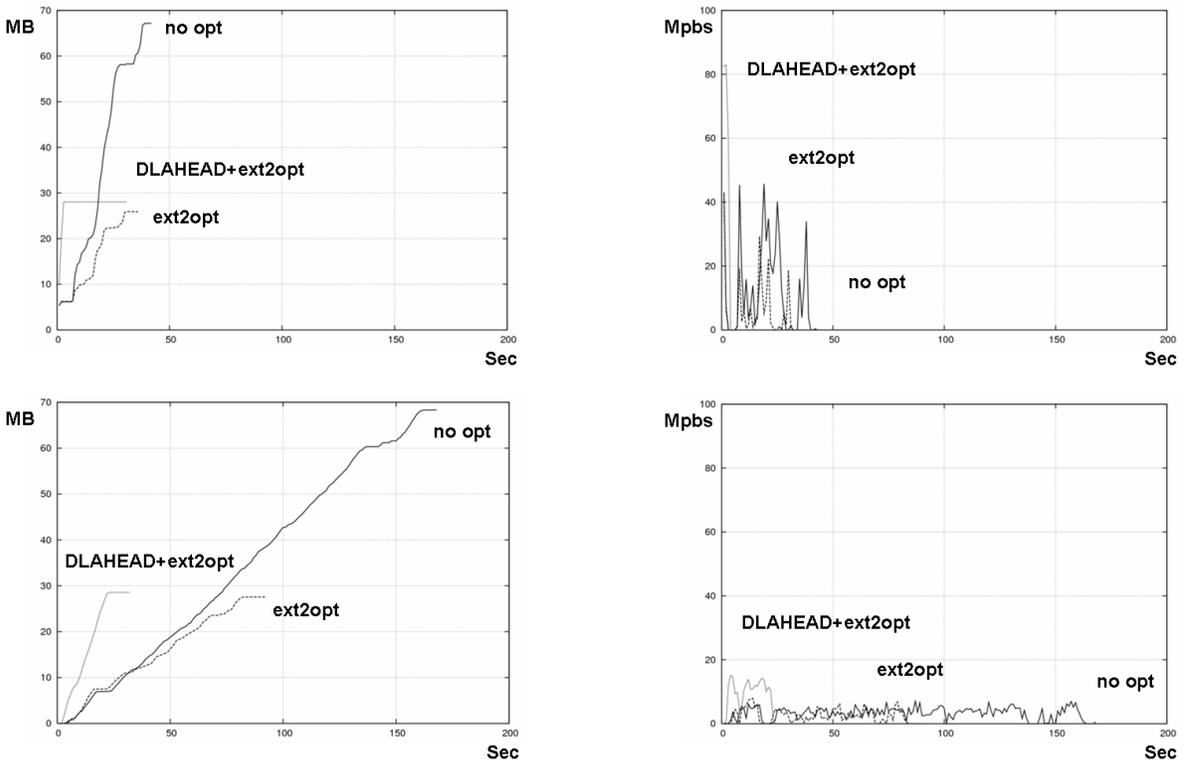


Figure 11: Amount of traffic (left) and throughput (right) at boot time on Xen-HVM under no latency (upper) and 30 msec latency (lower).

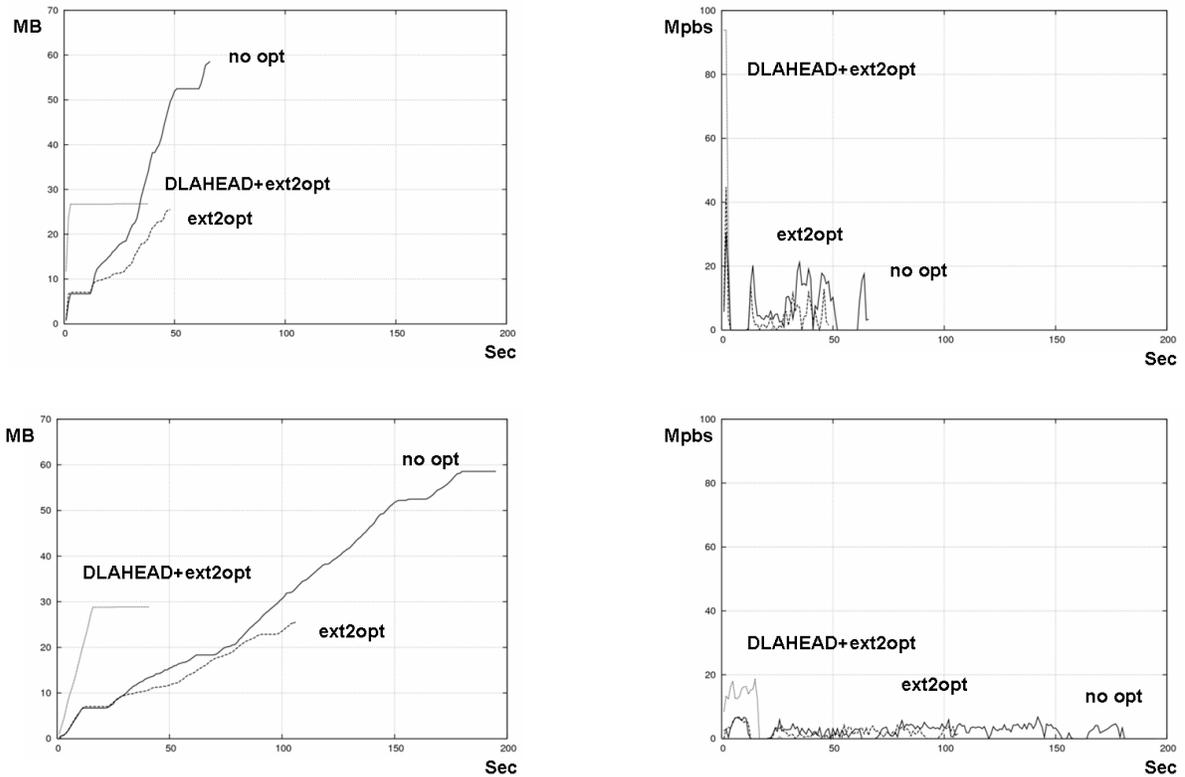


Figure 12: Amount of traffic (left) and throughput (right) at boot time on KQEMU under no latency (upper) and 30 msec latency (lower).

and 3, the difference didn't appear because most time was consumed by boot sequence itself.

Adding the optimizations improved ext2optimizer's boot times in all cases due to the reduction in downloaded block files. This effect grew as network latency was increased. The ratio of boot time with no optimization to that with ext2optimizer approaches the reduction of traffic (to 40% and 47% on Xen-HVM and KQEMU).

When block files were downloaded from the network, DLAHEAD became available. DLAHEAD starts even before a guest OS boots (usually by about five seconds), as soon as the virtual disk is set up for a virtual machine.

If DLAHEAD is not available, the tables show HTTP-FUSE CLOOP's performance is quite vulnerable to network latency. The boot times with 30 msec delay were more than 2x slower than 0 msec latency. The combination of ext2optimizer and DLAHEAD reduced the boot times to less than 41 seconds in each case – close to the case of cached optimized block files. These two types of optimization are necessary for reasonable performance of HTTP-FUSE CLOOP.

Throughput

Since download requests are usually sequential and the size of each download is small (average 100KB), network latency can dramatically reduce a system's performance.

DLAHEAD increased the network throughput to about 90 Mbps on both of Xen-HVM and QEMU when the network latency was 0 msec, finishing the download in five seconds. With 0 msec latency, though, its improvement was diminished. When the network latency was 30 msec, throughput was only 17 Mbps. However, the download finished in 25 seconds and the boot times were almost same as the cases of 0 msec network latency. These results showed the power of DLAHEAD for Internet-based servers.

Discussion

Linking Vulnerability Databases

OS Circular offers a framework for Internet clients. The disk image is updated by the package

manager (though there is no way to authenticate it). The disk image should be linked to Vulnerability Databases to check the contents.

The target vulnerability database is CVE (Common Vulnerabilities and Exposures) [16], operated by the MITRE corporation which includes vulnerability information for packages in each OS. The OVAL (Open Vulnerability Assessment Language) provides a uniform mechanism to report on and control security.

We will apply CVE and offer the information for anonymous users to aid OS selection decisions.

Trusted Boot to Detect Rootkit

The current implementation has to trust VMKNOPPIX. If VMKNOPPIX contains a rootkit or malware on the virtual machine, we have no way to detect it [17, 18].

The Trusted Computing Group (TCG) [19] promotes open standards for hardware-enabled trusted computing. It has released specifications for the Trusted Platform Module (TPM) chip which is often available on current PCs and is used for trusted boot.

We have a plan to integrate trusted boot (e.g., Trusted GRUB [20] and IMA Linux kernel [21, 22]) into VMKNOPPIX. Trusted GRUB and IMA (Integrity measurement Architecture) keep a log of equipped devices and opened files in the TPM. The log is sealed by the key of TPM and sent to a remote attestation server to certify the trusted boot.

Live Update

Some researchers have proposed live update of OSes on virtual machines, e.g., Intel vPro and LUCOS [23]. Virtual machines and the Guest OS communicate with each other and enable live update for security. Currently, OS Circular has no function for live update, but we hope to integrate it in the future.

PlayStation 3

OS Circular has been also been applied to other devices which run Linux. We have even applied OS Circular to a game machine "PlayStation 3"; we call it HTTP-FUSE PS3 Linux [24].

	No optimization	ext2optimiser	DLAHEAD+
Normal	30	28	–
Cache	31	28	–
0 msec latency	42	36	31(+5)
30 msec latency	168	92	32(+5)

Table 2: Boot time on Xen-HVM (sec).

	No optimization	ext2optimiser	DLAHEAD+
Normal	55	46	–
Cache	53	44	–
0 msec latency	66	48	38(+5)
30 msec latency	195	106	41(+5)

Table 3: Boot time on KQEMU (sec).

The most important features are the unified device model and preparation of the block device before booting. The device model of PlayStation3 is unified, and the boot loader, “kboot” [25], is stored in a 4 MB built-in Flash memory. kboot can download a kernel and a miniroot via HTTP from the Internet. This “miniroot” includes the driver for HTTP-FUSE CLOOP and the root filesystem is obtained by HTTP-FUSE CLOOP.

Conclusions

We have proposed OS Circular, a client-centric OS migration system. OS Circular utilizes easily obtained infrastructure, including Web hosting and security update service. OS images are maintained by a security management paradigm on the client, checking the validity of data blocks as they arrive. Performance improvements have enabled OS Circular to achieve performance almost as good as local disks. Future plans include the integration of Trusted Boot and Vulnerability Database checking.

Author Biographies

Kuniyasu Suzaki received the B.Eng. degree and the M.Eng. degree in computer engineering from Tokyo University of Agriculture and Technology. He is a Senior Researcher at National Institute of Advanced Industrial Science and Technology, Japan. His current research interests include trusted computing and OS migration. Reach him electronically at k.suzaki@aist.go.jp.

Toshiki Yagi received the B.Sc. degree in Information Science from the University of Tsukuba. He is a Research Staff member of National Institute of Advanced Industrial Science and Technology, Japan. His current research interests include trusted computing and virtualization. Reach him electronically at yagitoshiki@aist.go.jp.

Kengo Iijima received the B.Sc. degree in Information Science from the University of Tsukuba. He is a Research Staff of National Institute of Advanced Industrial Science and Technology, Japan. His current research interests include OS migration and virtual stackable block device. Reach him electronically at k-ijima@aist.go.jp.

Nguyen Anh Quynh received the B.Sc. degree and the M.Sc in Information Technology from Vietnam National University and a Ph.D. in Computer Science from Keio University. He is a Post-Doctoral Research Scientist of Advanced Industrial Science and Technology, Japan. His current research interests include virtualization, trusted computing, and security. Reach him electronically at nguyen.anhquynh@aist.go.jp.

Bibliography

- [1] Suzaki, Kuniyasu, Toshiki Yagi, Kengo Iijima, and Nguyen Anh Quynh, “OS-Circular”: A Framework of Internet Client with Xen, Xen Summit

- 2007 Spring, York Town, NY, April, 2007, http://www.xensource.com/files/xensummit_4/XenSummit07Spring-Suzaki.pdf.
- [2] Suzaki, Kuniyasu, Toshiki Yagi, Kengo Iijima, Junichi Tsukamoto, Megumi Nakamura, and Seiji Munetoh, “OS Circulation Environment: Trusted HTTP-FUSE Xenoppix,” Linux Conference Australia 2007, Sydney, Australia, January, 2007, http://mirror.linux.org.au/linux.conf.au/2007/video/monday/monday_1450_Virtualisation.pdf.
- [3] Chandra, Ramesh, Nikolai Zeldovich, Constantine Sapuntzakis, and Monica S. Lam, “The Collective: A Cache Based System Management Architecture,” *Proceedings of the 2nd Symposium on Networked Systems Design and Implementation (NSDI’05)*, Boston, MA, May, 2005.
- [4] Pfaff, Ban, Tal Garfinkel, and Mendel Rosenblum, “Virtualization Aware File Systems: Getting Beyond the Limitations of Virtual Disks,” *Proceedings of the 3rd Symposium on Networked Systems Design and Implementation (NSDI’06)*, San Jose, CA, May, 2006.
- [5] OS Zoo, <http://www.oszoo.org/>.
- [6] FLOZ, http://www.oszoo.org/wiki/index.php/Free_Live_OS_Zoo.
- [7] Moka5’s LivePC, <http://www.moka5.com/products/>.
- [8] Cáceres, Ramón, Casey Carter, Chandra Narayanaswami, and Mandayam Raghunath, “Reincarnating PCs with Portable SoulPads,” *Proceedings on the 3rd Annual International Conference on Mobile Systems, Applications, and Services (MobiSys’05)*, Seattle, WA, June, 2005.
- [9] Kozuch, Michael, and Mahadev Satyanarayanan, “Internet Suspend/Resume,” *Proceedings on the Fourth IEEE Workshop on 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA’02)*, Washington, DC, June, 2002.
- [10] Quinlan, Sean and Sean Dorward, “Venti: A New Approach to Archival Storage,” *Proceedings of the FAST 2002 Conference on File and Storage Technologies*, Monterey, CA, January, 2002.
- [11] Stateless Linux, <http://fedoraproject.org/wiki/StatelessLinux>.
- [12] FUSE (File system in USEr-space), <http://fuse.sourceforge.net/>.
- [13] Kitagawa, Kenji, Tan Hideyuki, Daisuke Abe, Daisaku Chiba, Kuniyasu Suzaki, Kengo Iijima, and Toshiki Yagi, “File System (Ext2) Optimization for Compressed Loopback Device,” *Linux Kongress 2006*, http://www.linux-kongress.org/2006/abstracts.html#3_5_1.
- [14] Krafft, Martin F., *The Debian System: Concepts and Techniques*, Open Source Press GmbH, 2005.
- [15] Yokota, Hiroshi, Shigetomo Kimura, and Yoshihiko Ebihara, “A Proposal of DNS-Based Adaptive Load Balancing Method for Mirror Server Systems and Its Implementation,” *Proceedings*

on the 18th International Conference on Advanced Information Networking and Applications, March, Fukuoka, Japan, 2004.

- [16] *CVE*, <http://cve.mitre.org/>.
- [17] King, Samuel T., Peter M. Chen, Yi-Min Wang, Chad Verbowski, Helen J. Wang, and Jacob R. Lorch, "SubVirt: Implementing Malware With Virtual Machines," *Proceedings on the IEEE Symposium on Security and Privacy*, Berkeley, CA, May, 2006.
- [18] Garfinkel, Tal, Ben Pfaff, Jim Chow, Mendel Rosenblum, and Dan Boneh, "Terra: A Virtual Machine-Based Platform for Trusted Computing," *Proceedings on the 19th ACM Symposium on Operating Systems Principles (SOSP'03)*, NY, October, 2003.
- [19] Trusted Computing Group, <https://www.trusted-computinggroup.org>.
- [20] *Trusted GRUB*, <http://trousers.sourceforge.net/grub.html>.
- [21] *IMA (Integrity Measurement Architecture)*, http://domino.research.ibm.com/comm/research_people.nsf/pages/sailer.ima.html.
- [22] Sailer, Reiner, Xiaolan Zhang, Trent Jaeger, and Leendert van Doorn, "Design and Implementation of a TCG-based Integrity Measurement Architecture," *Proceedings on the 13th USENIX Security Symposium*, San Diego, CA, August, 2004.
- [23] Chen, Haibo, Rong Chen, Fengzhe Zhang, Binyu Zang, and Pen-Chung Yew, "Live Updating Operating Systems Using Virtualization," *Proceedings on the 2nd International Conference on Virtual Execution Environments*, Ottawa, Canada, June, 2006.
- [24] Yagi, Toshiki, and Kuniyasu Suzaki, "HTTP-FUSE PS3 Linux: An Internet Boot Framework With kboot," *CELF Embedded Linux Conference (ELC'07)*, Santa Clara, Ca, April, 2007, <http://www.celinux.org/elc2007/>.
- [25] *kboot*, <http://kboot.sourceforge.net/>.