Seamless Interaction Across Roles

Chris Long
Sophie Kim
Chris Hill
Next Century
Falls Church, VA 22042, USA
chris.long@nextcentury.com
sophie.kim@nextcentury.com
chris.hill@nextcentury.com

Wole Omitowoju
Kyle Drumm
Kevin Aranyi
Next Century
Annapolis Junction, MD 20701,
USA
wole.omitowoju@nextcentury.com
kyle.drumm@nextcentury.com
kevin.aranyi@nextcentury.com

Abstract

The SAVIOR system demonstrates a virtual desktop environment that partitions applications and data into isolated Roles for security, which run in separate virtual machines in the cloud. SAVIOR addresses two usability issues with such an architecture: management of and interaction with applications scattered across VMs in the cloud, and secure information sharing across Roles. First, its custom Desktop application unifies presentation and interaction across Roles and provides a user experience almost like running the applications locally. The Desktop also provides a single interface for starting and stopping installed applications. Second, SAVIOR allows sharing data across Roles with familiar clipboard operations and file sharing with single sign-on to standard file servers, as permitted by the administrator. For security, user credentials are never shared with applications themselves or even the VMs they reside in.

Author Keywords

hcisec; usability; cloud computing; malware; virtual machines

ACM Classification Keywords

• Security and privacy~Usability in security and privacy • Security and privacy~Malware and its mitigation • Security and privacy~Virtualization and security • Human-centered computing~Interactive systems and

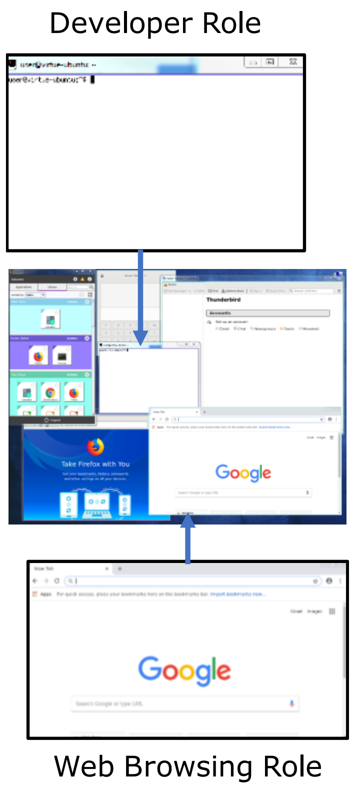

Figure 1 Example SAVIOR Session. Switching among Roles is easy. For example, the terminal from the Developer Role and web browser from the Web Browser Role are visible simultaneously.

tools • Computer systems organization~Cloud computing

Introduction

Desktop computer security is still a big problem. Malware, phishing, and intruders are still prevalent. One reason for their success is that all applications run with access to all file and applications the user has access to. We have developed the SAVIOR system (Secure Applications in Virtual Instantiations of Roles) to leverage virtualization and cloud computing to provide users with a more secure computing environment that is still easy to use.

SAVIOR allow users to easily run unmodified applications that have restricted access to resources, including files, networking, and other applications. Unlike other similar solutions, SAVIOR partitions applications using separate virtual machines running in the cloud, specifically Amazon Web Services (AWS) Elastic Computing Cloud (EC2). The separation of applications into roles limits their access to only the resources available in that role. For example, an Email Role that had only an email client installed (e.g., Thunderbird, Outlook) would not have access to the web or to the user's sensitive documents. As others have discussed, while not a panacea, this type of compartmentalization will thwart many types of malware and other threats.

The SAVIOR Desktop application provides a unified interface for launching and seamlessly interacting with applications running on multiple virtual machines and operating systems, with a user experience nearly identical to running applications locally (see Figure 1).

One challenge for partitioned systems such as this is that users sometimes do need to share information across these boundaries. To support this, SAVIOR allows sharing across roles via SMB file shares (e.g., Windows, Samba) and clipboard copy & paste.

Related Work

SAVIOR was motivated in part by prior research showing the need for security to be usable (e.g., [18]). It also adheres to many of the principals for usability security espoused by [18] and [2], such as: you can't retrofit security, tools aren't solutions, and mind the upper layers.

There are many mechanisms available today for restricting application execution. These include AppArmor [1] and SELinux [12], which both allow policies at the level of specific programs and files. Virtualization is also used for security, with systems such as VirtualBox [23], VMWare [20], and Xen [4].

Several previous systems were designed to increase security by limiting application actions while providing a unified interface for interacting with them. Chameleon [7] and Polaris [19], for example, run applications as different users to restrict applications. WindowBox [3] and Janus [6] used process-level protections. Bromium uses "micro-virtual machines" [5]. Apiary [15] uses containerization provided by the operating system and a custom virtual layered file system. Qubes OS [17] uses Xen domains. Unlike SAVIOR, none of these work with cloud-based virtual machines, and only Qubes OS supports multiple operating systems.

SAVIOR

SAVIOR was designed with the following goals:

- Easy to use, specifically by being as similar to the standard desktop experience as possible.
- Support for existing Linux and Windows applications, including Outlook and Microsoft Office.
- Secure by design, via role-based partitioning.
- Instrumented with sensors at multiple levels, to provide data for security analytics.
- Integrated with enterprise infrastructure: Active Directory authentication, single-sign-on access to file shares and printers.

The overall architecture of SAVIOR is shown in Figure 2. Each user may be given access to any number of Roles and may run as many of these at a time as desired. Each Role can be run any number of Linux and/or Windows applications. Each Role is instantiated as a Xen hypervisor running on Amazon Elastic Computing Cloud (EC2) and, if the Role includes Windows applications, Windows and Windows Display Server instances, also (see Figure 3). While a user's applications are running, a CIFS Proxy allows authorized Roles to access files on standard Windows file servers. The Virtue Manager handles user authentication to the system, user access to Roles, and startup and shutdown of EC2 instances. Its features are exposed as REST services, which can be secured with https. Sensors run continuously and report many types of events to the Sensor Servers, including file access, running processes, and network activity. The level of detail of their reports can be dynamically varied.

All network connections in SAVIOR are secured. A few types of connections have their own security mechanisms, such as with Active Directory. Those that do not, such as Xpra, are secured by tunneling via ssh.

Desktop

The Desktop is the application users run to authenticate to SAVIOR, start and stop applications, and interact with running applications. As part of the overall SAVIOR ease-of-use design goal, the Desktop presents each application in its own top-level window, as if it were running locally, similar to "seamless" mode in VirtualBox or "unity" mode in VMWare Workstation.

This goal significantly restricted our choice of remote protocol, because most remote protocols only support full desktop remoting (e.g., VNC [16]). For Linux applications, these constrained our choices to X [14] and Xpra [24]. We chose Xpra because unlike X, it is designed to work well over wide area networks. The Desktop natively implements the Xpra protocol.

For Windows, only Microsoft's Remote Desktop Protocol (RDP) [10] with the remote application extension [11] fit our requirements. Although the server is not open source, there are multiple open source clients (e.g., [21], [22], [25]). We considered either adding RDP support directly to the SAVIOR Desktop, or adding an intermediate computer running an RDP client and Xpra server. We chose the RDP+Xpra approach for two reasons. First, it required substantially less engineering. Second, by using an intermediary, the Windows Display Server, Windows applications remain running if there is a network problem or the Desktop exits. The WDS runs an Xpra server to which the Desktop connects and an RDP client, connected to the Windows Application VM, for each Windows application that is running.

CIFS Proxy

The CIFS (Common Internet File System [8]) Proxy exists to satisfy two apparently contradictory goals: 1.

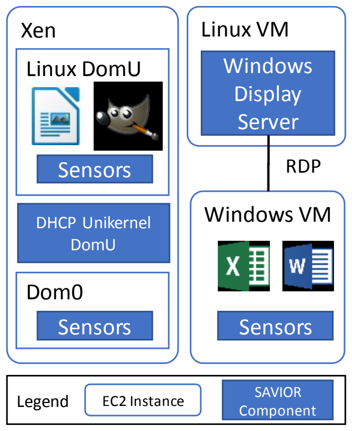

Figure 3 Example SAVIOR Role. Applications for both Linux (LibreOffice, Gimp) and Windows (Microsoft Excel and Word) are available.

to allow users access files shared with them on existing remote file servers using single-sign-on, and 2. to deny the applications that use those files access to the user's credentials for those files, thus limiting them to access granted to their Role. To resolve this conundrum, the CIFS Proxy acts as an intermediary between virtual machines running user applications and the file servers hosting the shared files. It connects to the file server on the user's behalf and mounts the file share (i.e., a specific set of shared files). It then exposes this share only to virtual machines in roles that are allowed to access the share, potentially with different permissions (as configured by the system administrator). There is exactly one Proxy for each logged-in user.

First, the Virtue Manager uses the username [9] and password it received from the Desktop to obtain a Kerberos [13] Ticket Granting Ticket (TGT) for the user and from it a service ticket for the Proxy. It passes that service ticket to the Proxy with a request to grant a role access to a file share. The Proxy uses the service ticket with Kerberos constrained delegation and Service for User to Proxy (S4U2Proxy) [9] to get a service ticket for the desired file server. The CIFS proxy is then able to mount the file share. To provide access to the files to the authorized virtual machines, the Proxy runs a Samba server. To control access on a per-role basis, the Proxy generates a unique username/password combination for each role and sets permissions on the exported Samba share to read only or read-write, as specified by the system administrator.

Limitations

SAVIOR is a research prototype and so has several limitations. One is the approximately five minutes it takes to start a new Role instance. We are exploring keeping hot spares and other avenues to reduce this time. Also, operating on files across Roles is not as convenient as it could be. It is possible with shared file servers and via the clipboard, but direct access [7] [19] would be better.

SAVIOR was designed specifically to run on AWS EC2. Some components, such as the Desktop, are not sensitive to the specific cloud technology, but others, such as the Virtue Manager, have substantial ties to virtual machine management. Creating new Roles involves the manual construction of virtual machine images that can be instantiated by the Virtue Manager. We are developing tools to ease this process.

Conclusion

SAVIOR demonstrates a more secure virtual desktop environment, in which applications are partitioned from one another for security, but provide easy and familiar access to these applications and allow straightforward sharing of information among them, but under user control and subject to administrator oversight.

Acknowledgements

We thank our colleagues at Virginia Tech and Two Six Labs, and our sponsor, IARPA[1].

[1] The research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

References

[1] AppArmor. 2019. *AppArmor*.

[2] Dirk Balfanz, Glenn Durfee, Rebecca E. Grinter, and D. K. Smetters. 2004. In Search of Usable Security: Five Lessons from the Field. *IEEE Secur. Priv.* 2, 5 (September 2004), 19–24. DOI:https://doi.org/10.1109/MSP.2004.71

[3] Dirk Balfanz and Daniel R. Simon. 2000. WindowBox: a simple security model for the connected desktop. In *Proceedings of the 4th USENIX Windows Systems Symposium*, 37–48.

[4] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. 2003. Xen and the Art of Virtualization. In *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles* (SOSP '03), 164–177. DOI:https://doi.org/10.1145/945445.945462

[5] Bromium. 2019. *Bromium Secure Platform*.

[6] Ian Goldberg, David Wagner, Randi Thomas, and Eric A. Brewer. 1996. A Secure Environment for Untrusted Helper Applications Confining the Wily Hacker. In *Proceedings of the 6th Conference on USENIX Security Symposium, Focusing on Applications of Cryptography - Volume 6* (SSYM'96), 1–1. Retrieved from http://dl.acm.org/citation.cfm?id=1267569.1267570

[7] A. Chris Long and Courtney Moskowitz. 2005. Security and Usability: Designing Secure Systems That People Can Use. In Lorrie Faith Cranor and Simson Garfinkel (eds.). O'Reilly Media, Inc., 336–356.

[8] Microsoft. 2018. *[MS-CIFS]: Common Internet File System (CIFS) Protocol*.

[9] Microsoft. 2018. *[MS-SFU]: Kerberos Protocol Extensions: Service for User and Constrained Delegation Protocol*.

[10] Microsoft. *[MS-RDPBCGR]: Remote Desktop Protocol: Basic Connectivity and Graphics Remoting*. Retrieved April 8, 2019 from https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdpbcgr/5073f4ed-1e93-45e1-b039-6e30c385867c

[11] Microsoft. *[MS-RDPERP]: Remote Desktop Protocol: Remote Programs Virtual Channel Extension*. Retrieved April 8, 2019 from https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rdperp/83275957-2d0e-4c52-88d1-1b4c998c6bec

[12] National Security Agency. SELinux.

[13] Dr Clifford Neuman, Sam Hartman, Kenneth Raeburn, and Taylor Yu. 2005. *The Kerberos Network Authentication Service (V5)*. RFC Editor. DOI:https://doi.org/10.17487/RFC4120

[14] Adrian Nye. 1992. *X Protocol Volume 0 R6*. O'Reilly Media. Retrieved April 8, 2019 from http://shop.oreilly.com/product/9781565920835.do

[15] Shaya Potter and Jason Nieh. 2010. Apiary: easy-to-use desktop application fault containment on commodity operating systems. In *Proceedings of the 2010 USENIX conference on USENIX annual technical conference*, 8. DOI:https://doi.org/10.5555/1855840.1855848

[16] Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, and Andy Hopper. 1998. Virtual Network Computing. *IEEE Internet Comput.* 2, 1 (January 1998), 33–38. DOI:https://doi.org/10.1109/4236.656066

[17] Joanna Rutkowska and Rafal Wojtczuk. 2010. Qubes OS architecture. *Invis. Things Lab Tech Rep* 54, (2010). Retrieved from https://www.qubes-os.org/attachment/wiki/QubesArchitecture/arch-spec-0.3.pdf

[18] D. K. Smetters and R. E. Grinter. 2002. Moving from the design of usable security technologies to the design of useful secure applications. In *Proceedings of the 2002 workshop on New security paradigms*, 82–89. DOI:http://doi.acm.org/10.1145/844102.844117

[19] Marc Stiegler, Alan H. Karp, Ka-Ping Yee, Tyler Close, and Mark S. Miller. 2006. Polaris: Virus-safe Computing for Windows XP. *Commun ACM* 49, 9 (September 2006), 83–88. DOI:https://doi.org/10.1145/1151030.1151033

[20] VMware, Inc. 2003. *VMware*. Retrieved from http://www.vmware.com/

[21] *rdesktop is an open source UNIX client for connecting to Windows Remote Desktop Services.* rdesktop. Retrieved April 8, 2019 from https://github.com/rdesktop/rdesktop

[22] *FreeRDP is a free remote desktop protocol library and clients: FreeRDP/FreeRDP*. FreeRDP. Retrieved April 8, 2019 from https://github.com/FreeRDP/FreeRDP

[23] *Oracle VM VirtualBox*. Retrieved April 8, 2019 from https://www.virtualbox.org/

[24] *xpra home page*. Retrieved April 8, 2019 from https://xpra.org/

[25] *A feature rich Remote Desktop Application*. Retrieved April 8, 2019 from https://remmina.org/