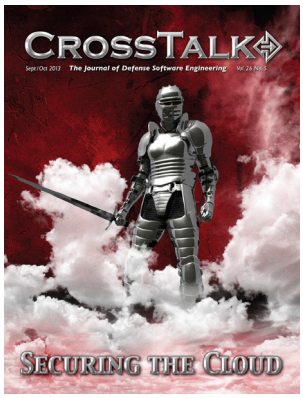
CrossTalk

Sept/Oct 2013 The Journal of Defense Software Engineering Vol. 26 No. 5

Securing the Cloud

Departments



Cover Design by Kent Bingham

Securing The Cloud

CrossTalk

NAVAIR Jeff Schwalb
DHS Joe Jarzombek
309 SMXG Karl Rogers

Publisher Justin T. Hill
Article Coordinator Lynne Wade
Managing Director David Erickson
Technical Program Lead Thayne M. Hill
Managing Editor Brandon Ellis
Associate Editor Colin Kelly
Art Director Kevin Kiernan

Phone 801-777-9828
E-mail Crosstalk.Articles@hill.af.mil
CrossTalk Online www.crosstalkonline.org

CrossTalk, The Journal of Defense Software Engineering is co-sponsored by the U.S. Navy (USN); U.S. Air Force (USAF); and the U.S. Department of Homeland Security (DHS). USN co-sponsor: Naval Air Systems Command. USAF co-sponsor: Ogden-ALC 309 SMXG. DHS co-sponsor: Office of Cybersecurity and Communications in the National Protection and Programs Directorate

The USAF Software Technology Support Center (STSC) is the publisher of **CrossTalk** providing both editorial oversight and technical review of the journal. **CrossTalk's** mission is to encourage the engineering development of software to improve the reliability, sustainability, and responsiveness of our warfighting capability.

Subscriptions: Visit <www.crosstalkonline.org/subscribe> to receive an e-mail notification when each new issue is published online or to subscribe to an RSS notification feed.

Article Submissions: We welcome articles of interest to the defense software community. Articles must be approved by the **CrossTalk** editorial board prior to publication. Please follow the Author Guidelines, available at <www.crosstalkonline.org/submission-guidelines>. **CrossTalk** does not pay for submissions. Published articles remain the property of the authors and may be submitted to other publications. Security agency releases, clearances, and public affairs office approvals are the sole responsibility of the authors and their organizations.

Reprints: Permission to reprint or post articles must be requested from the author or the copyright holder and coordinated with **CrossTalk.**

Trademarks and Endorsements: CrossTalk is an authorized publication for members of the DoD. Contents of **CrossTalk** are not necessarily the official views of, or endorsed by, the U.S. government, the DoD, the co-sponsors, or the STSC. All product names referenced in this issue are trademarks of their companies.

CrossTalk Online Services:
For questions or concerns about crosstalkonline.org web content or functionality contact the **CrossTalk** webmaster at 801-417-3000 or webmaster@luminpublishing.com.

Back Issues Available: Please phone or e-mail us to see if back issues are available free of charge.

CrossTalk is published six times a year by the U.S. Air Force STSC in concert with Lumin Publishing <luminpublishing.com>.
ISSN 2160-1577 (print); ISSN 2160-1593 (online)

CrossTalk would like to thank DHS for sponsoring this issue.

Moving to the cloud is the latest irresistible force to sweep the C-suite and Main Street. The opportunities for flexibility, savings, reduced sustainment, and ubiquity have made cloud solutions compelling for Information Technology (IT) managers around the world.

Those who consider the benefits of cloud computing often cite potential improvements in efficiency, agility, and innovation. These individuals often indicate that existing computing facilities have some degree of duplication, are difficult to manage, and operate at less than optimum capacity. A major benefit of the cloud would be the ability to rapidly meet new demand for capacity and services due to the elastic capacity of cloud providers. Moreover, the cloud also reduces the need for asset management of rapidly evolving technology and enables the use of innovative solutions.

However, there are security challenges unique to cloud architectures, including: dynamic provisioning of platforms of unknown or dubious origin, global access by mobile (and largely insecure) devices, eroded trust boundaries, and the possibility of malevolent neighbors in your public cloud. The acquirer of services must be aware that there are variances in cloud provider security capabilities. Service Level Agreements (SLAs) provide important coverage, including properly articulated security and resiliency expectations, but might not offer a comprehensive solution.

The good news is that the processes, practices, tools, and techniques from traditional IT can be applied to address many cybersecurity concerns. As savvy consumers, we can employ established software and supply chain assurance methods when acquiring cloud-based services, as long as we recognize the new risks and challenges presented by this new technology. Cloud computing has the potential to improve our security capabilities and services. As agencies and departments consider various cloud architectures, more stringent security requirements will encourage cloud service providers to build cloud services with significantly improved security.

To help ensure the U.S. Government adopts best practice methods as we move to the cloud, DHS has coordinated with NIST and other federal agencies in standardizing expectations for IT security for cloud services. Moreover, DHS has provided technical assistance to the Federal Risk and Authorization Management Program (FedRAMP). Housed in the General Services Administration (GSA), the FedRAMP provides a security certification and authorization process that applies consistency and transparency across Federal departments and agencies for cloud implementation and security. This program builds security into the government-wide solicitations from the beginning, while enabling agencies to retain their responsibility and authority to meet their unique network security needs. For providers, the FedRAMP performs oversight of continuous monitoring, and allows vendors to participate in a single risk management process, share compatible requirements, and a consistent assessment process.

As we look to the promise of FedRAMP and other secure services delivering capabilities on which our nation depends, our cybersecurity processes and procedures will continue to evolve, and NIST Special Publications, such as SP 800-53 “Security and Privacy Controls for Federal Information Systems and Organizations,” will remain in the forefront of security guidance. Although there will always be more to be done to achieve a safe and cyber-secure cloud, we must embrace a shared strategy for cybersecurity and work together to reap the benefits we all envision from this new and dynamic technology.

Roberta “Bobbie” Stempfley
Acting Assistant Secretary
Office of Cybersecurity and Communications
Department of Homeland Security

What Does the Future Hold for Cloud Computing?

David Zage, Sandia National Laboratories
Dustin Franklin, Sandia National Laboratories
Vincent Urias, Sandia National Laboratories

Abstract. The topic of cloud computing continues to generate significant interest among information technology and government decision makers even as they hesitate to adopt cloud solutions due to security concerns. The authors define the risks associated with various cloud deployment models and identify key solutions that can create easy-to-use, secure cloud deployments.

1. Introduction

Even though recent reports have begun to forecast diminished interest in cloud computing [16], large numbers of services are still migrating to the cloud and further infrastructure is being dedicated to platforms and solutions. While a large amount of cloud research has focused on utilizing the power, flexibility, and potential cost savings of cloud computing platforms, reports such as the Department of Homeland Security Roadmap for Cybersecurity Research [9] and previous research [4, 13] have expressed the explicit need for continued security analysis of cloud computing solutions. In polls, over 70% of government decision-makers [2] and 80% of IT executives [3, 14] identify security and ease of deployment as the primary obstacles to cloud computing adoption.

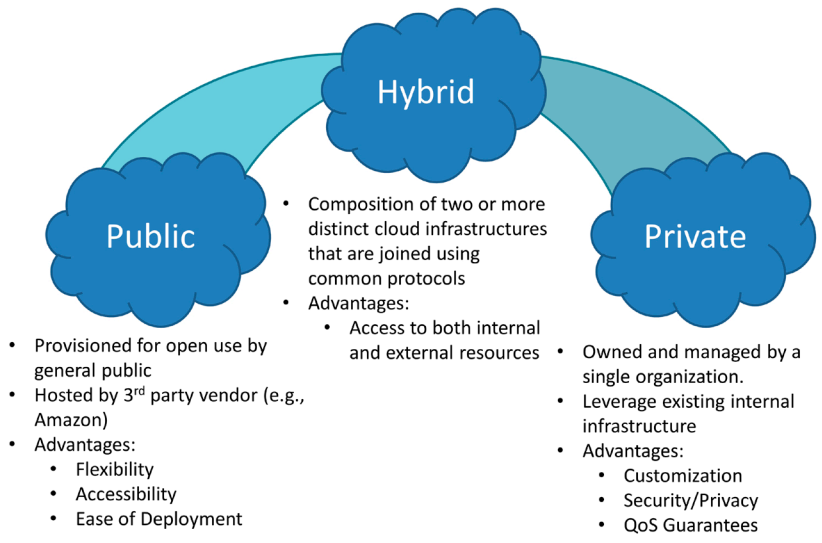


Figure 1 - Three deployment models for cloud computing. General descriptions and the advantages of each type are listed below each model.

Cloud services deal with amounts of data, users, and service heterogeneity that have never been seen before. These issues combine with the desired ubiquity of cloud accessibility to create a field that is ripe for vulnerabilities and a potential playground for adversaries. There has been work towards securing cloud deployments, but security is still typically an afterthought as companies focus on maintaining service availability. Many security solutions are ports of classic paradigms such as firewalls to web services. These ports enhance the security of services running on the cloud, but they do not increase the intrinsic security of the cloud service itself. This article analyzes the various deficiencies that continue to hinder cloud deployments and presents three key areas of work fundamental to improving cloud services. The discussion of these threats and solutions is colored by the authors experience in creating and using large-scale cloud infrastructures.

Cloud Deployment Models

In the field of cloud computing, three cloud deployment models seen in Figure 1 have emerged: 1) public, 2) private, and 3) hybrid clouds. A public cloud deployment is typified by the hardware and cloud components being hosted by a third party provider and the cloud being used by multiple users. A user has no control over the hardware layer and varying levels of control over other components of the cloud stack seen in Figure 2. Public cloud deployments were developed to optimize the cost of computation and storage and allow massive computing jobs to be performed for a fraction of their former costs. This cost savings typically comes at the expense of security; thus causing increasing interest in private cloud deployments. Private cloud deployments are owned and managed by a single organization. For example, a company's IT organization may choose to deploy a Infrastructure-as-a-Service (IaaS) cloud usable by anyone in the company. A private deployment enables the owners (e.g., corporate IT) and users to have control over the full stack of cloud system components.

The next logical evolution in cloud deployments leverages both the cost savings of public clouds and the potential security gains of private clouds by combining them into a hybrid cloud service. We believe this is the future of cloud computing, but will highlight issues that must be taken into account before deploying a hybrid cloud. While these models are designed to handle many of the same tasks and thus share a common set of threats, there are also security challenges unique to each due to the exposure faced by components of their respective infrastructure. We now look at threats common to all cloud deployments.

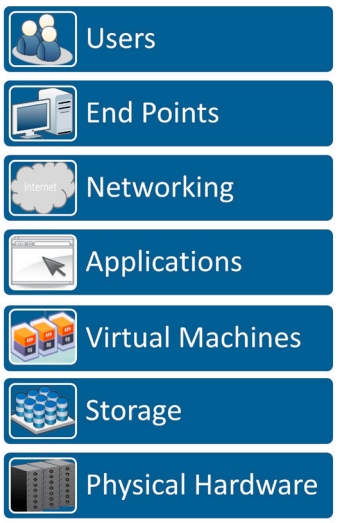


Figure 2 - The operational stack for typical cloud deployments.

2. Common Threats to Cloud Computing

The utility of cloud computing must be weighed against the threats it faces, which fall into three categories:

a. Failure to maintain security
b. Loss of availability
c. Reduction in usability

Security issues typically result from an adversary attempting to acquire data, knowledge, or persistent access to a system. Losses in availability stem from an adversary trying to deny legitimate user access to a cloud for the purposes of annoyance, delaying progress of work, or interfering with real-time responses to critical situations. Reductions in usability can affect the continued viability of a cloud service and the cloud paradigm as a whole. An adversary can cause systematic faults in a cloud service and ultimately diminish a user's faith in that cloud service. The threats affect the three deployment models universally and no one model stands out as being inherently better than the others.

Failure to Maintain Security

Security failures are the most conspicuous threats to cloud computing. Journalists regularly detail significant losses of data integrity and confidentiality originating from targeted attacks. Kapersky Lab reported that in 2013, 35% of businesses have lost data due to flawed system security [6]. The widespread availability of internet access has made hacking a global enterprise, allowing adversaries to work in areas where they face minimal penalties and have significant incentives.

Cloud deployments are complex systems of networked components that must work seamlessly in order to secure the large amounts of information they contain. A single misconfiguration or an unpatched vulnerability is sufficient to lead to exploitable security holes and allow an adversary access to the entire infrastructure. For example, in 2011, Sony Entertainment was the victim of a series of attacks on various pieces of their infrastructure in which personal information, including credit card information, was stolen from millions of accounts [11]. Exact details of the attack vector are unpublished, but the seriousness of the breach became apparent when the attackers released stolen data that indicated that Sony had been storing user information and passwords in plain text [8]. For their lack of security in credit card processing, Sony faced many repercussions, including a £250,000 fine by UK's Information Commissioner's Office. Although it is easy to blame the attackers, companies with valuable information need to be cognizant of the threats they face.

Including users into the chain of trust in cloud deployments has caused many security vulnerabilities. Good user security practices such as enforcing strong password selection, avoiding spearfishing, and testing web interfaces for cross site scripting attacks are necessary. While many of these vulnerabilities and practices are well known, it is important to make note of them as they continue to impact cloud deployments.

Loss of Availability

No matter the type of cloud, a user wants data to be accessible at any time and place. Availability is reduced as networks and machines fail, poorly deployed cloud solutions run into bottlenecks, and cloud services face distributed denial of service (DDoS) attacks. This is a particularly pressing issue for public clouds as they provide Service Level Agreements (SLAs) that are based on their contractual ability to provide computing power, storage, and services. Large sums of money stand to be lost when providers fail to meet their SLAs.

A concern that bridges both security and availability is the capability of monitoring the state of data during its lifetime in the cloud. Problems in data provenance include determining if data resides in locations that follow the same regulations a business must enforce (e.g., HIPAA), if the cloud service has stored the entirety of the data, if the data remains uncorrupted, and if the data is truly removed once it has been deleted by the user. A user that uploads his or her data to a third party may be forfeiting inherent rights to the control of their data, which could then be changed or viewed without notifying the user [5]. While this concern may appear easier to manage in private cloud deployments, cloud solutions are often adopted without understanding the full risk profile. IT typically lacks the tools necessary to understand how the complex pieces fit together and can provide minimal assurances for the end-user [14].

Reduction in Usability

If the expected performance of a cloud service is not up to a user's expectations or its SLA guarantees, the user may change or discontinue usage. Repeated negative experiences result in the user losing faith in the cloud computing paradigm. Additionally, corporations desire to have the ability to quickly deploy private and hybrid clouds with minimal effort (e.g., testbed-as-a-service), but the technology for doing this in an efficient, repeatable manner is still not mature enough for this to be a reality.

3. Threats to the Different Cloud Deployment Types

Given the general threats discussed in Section II, this section looks at cloud deployment-specific vulnerabilities.

Public Cloud Deployments

Common to all public cloud systems is the lack of control over the physical storage of data. In clouds which give the user minimal access to the cloud stack, such as software-as-a-service (SaaS) clouds, security of the data is reliant almost entirely on the practices used by the cloud provider, over which the user has little purview. In less restrictive systems, such as IaaS clouds, users are allowed to create and deploy virtual machines, which provide greater user customization and control of data storage. Several cloud service providers provide preconfigured virtual machines for their users to minimize user effort and eliminate obvious security flaws.

While virtualization is a great enabling technology, there have been actual attacks demonstrated where a virtual machine can be compromised and used to bypass system protections, enabling attacks on the rest of the cloud infrastructure [7, 18]. Although cloud providers do not provide information on other clients running on the same physical hardware, adversarial virtual machines can be used to find and attack specific services [15]. It should also be noted that cloud providers have minimal incentive to provide secure virtual machines. One could imagine cases

in which unscrupulous providers may distribute virtual machines containing flaws (e.g., networking issues) that result in extra computation and network usage, simultaneously earning money for the provider while costing the customer.

There are also large marketplaces where third-party companies and users exchange virtual machines. Users need to be wary of these virtual machines as they are often poorly secured [12, 1]. These preconfigured solutions have been found to contain unpatched code, share credentials with other virtual machines, and, worst of all, even Trojan software. Another less obvious problem with virtual machines that users must be aware of is they often lack the necessary randomness to create truly secure cryptographic keys, especially when the virtual machines are running on the same hardware.

Users of public clouds may suffer from attacks that are not directed at them. Any network outage that affects the cloud will restrict the cloud usability. Clouds regularly face DDoS attacks attempting to bring down a single service hosted on the cloud. When these attacks succeed, they have the side effect of affecting all of the other services hosted by the cloud provider. Innocent users of the cloud can expect that a successful attack will cause downtime for their services even though they are not the target. One potential solution to dealing with such issues is through the use of hybrid solutions, enabling at least partial data access even during downtime.

Private Cloud Deployments

The primary driver behind private cloud adoption is concern over the sensitivity of the data to be stored and processed. Businesses desire cloud solutions that can leverage excess internal capacity while minimizing the potential for data leakage. Additionally, internally run clouds can have significant advantages in availability and accessibility over public deployments.

Many companies and governmental groups are constructing private cloud infrastructures inside their network perimeter. This setup can often be easier (and more comforting) for them to deploy as it uses many traditional system security mechanisms. For example, existing web security (e.g., firewalls) and permission management infrastructure can be used to secure the system. While these mechanisms provide protection from outside an organization, they are not configured to protect resources from mismanagement and malicious insiders. It is extremely unlikely that every piece of data should be available to every department and all people. Relying entirely on access control at the network perimeter can lead to dissemination of data to an adversary that has entered the network through another route.

Private cloud deployments are also vulnerable to attacks designed at interrupting availability, such as DDoS attacks. Public cloud providers can prepare for such attacks by investing in redundant capacity that scales to handle excessive traffic as needed. A private cloud will not have the same growth capability or mitigation techniques and the system may fail when targeted, leading to unavailability and system downtime.

Ultimately, the security of a private cloud depends on the capabilities of the organization deploying it. Currently, the deployment procedures for clouds are opaque, with multiple services running and numerous layers of abstraction between each of the services and between the services and their administrative layers. Often, when configuring and using services like OpenStack, one of the numerous web services and authentications will fail silently. While there has been significant work done by the community to improve the deployment and administration of tools like OpenStack (such as using common configuration management/deployments tools like Puppet), there are really no standard solutions. Differences in networks (such as topology, IP Space, VLANs, etc.), in hardware (vendor, raids, etc.) and underlying virtualization tools all provide complexity and variation from the norm. When a failure does occur, determining where the failure occurred and why is similar to finding a needle in a haystack. Typically, a system administrator will embark on a debugging mission that might result in a functioning system or attempt to start over with alternative configurations. Currently, there are no tools that can give an administrator the data and insight into where/what might have failed.

Hybrid Cloud Deployments

Hybrid cloud deployments have the potential to offer many of the positive aspects of both public and private cloud deployments in a single service, but they also face unique challenges. A primary concern is understanding the composability and resulting security posture of the hybrid system. Given a secure private and public cloud deployment, (provably) aggregating these together to create a secure service is currently an open problem. Most hybrid solutions are joined by easier-to-understand higher level protocols (e.g., user programs) and not at lower levels (e.g., a cross-cloud database). Clearly identifying the interfaces and connectivity patterns between the public and private components is a critical first step towards creating a secure service. Not only must the security of the system be analyzed, mitigation plans for availability or security issues affecting either portion of the cloud must be in place.

Another major concern not present in other deployment scenarios is the accurate disseminate and tracking of data between the multiple components of the cloud. While it might be attractive to use the private portion of the cloud to store HIPAA data and the public for non-sensitive data, it must be ensured that the data will not commingle. Having a write-once, read-only cloud is of little use.

A final challenge that must be addressed in the creation of hybrid solutions is configuration management. While the potential to have heterogeneous solutions is beneficial in reducing dependence on any one piece of software, the cloud services must be easy to set up. This necessitates the fusing of data from both the public and private cloud to create a common interface for deployment and management.

4. Enhancing the Security of Cloud Deployments

In order to mitigate some of the security issues discussed, we present three promising solutions: A) enhanced deployment techniques that are automated and repeatable, B) full stack cloud introspection, and C) enhanced cloud storage solutions leveraging multiple providers. Each of the solutions mitigates a distinct security vulnerability that is found in the cloud infrastructure. They can provide much higher levels of confidence

in the security posture of the infrastructure at the cost of some additional management challenges.

Enhanced Deployment

If an organization deploys any of the cloud models discussed previously (e.g., a private cloud for internal use or a public cloud for commoditization and profit), the organization must understand how to construct and administer the entire cloud stack. A very basic cloud service install which an administrator would have to create might resemble Figure 3, with installation occurring from left to right. Creating such a procedure is difficult and must be streamlined and instrumented with greater amounts of understanding/introspection into the install process. Not only will this enable greater cloud adoption, it will also give administrators the ability to quickly set up and tear-down cloud deployments for research and testing. We have created a set of installers for OpenStack that coalesce an immense amount of logging (from the system, network, and applications) to a central location. We have developed tools for automated install analysis as well as a platform to begin understanding how and why the system fails.

Our research points to the creation of a deployment process resembling object-oriented design patterns, in which the interactions and required functionality between each phase of the install are predefined. This way, a component in any step can be simply exchanged for another which provides the desired functionality. For instance, we have created automated installers for the major hypervisors that can be easily interchanged. In this manner, we can construct standard cloud configurations and take the uncertainties out of deployment. This allows for the creation of standard secure builds that can be vetted, tested, and guaranteed to produce repeatable results. Using this work, we can go from the bare metal to the fully operational application stack that can be re-provisioned in under an hour on tens of different hardware variants.

Cloud Analysis

While clouds are complex, one of the potential advantages of cloud-based computing is that it opens the possibility of understanding the entire infrastructure. This understanding comes down to intelligently gathering and processing a myriad of system packets and logs. Each component of the cloud stack, from the bare metal operating system, to the virtual machine manager, to the application being hosted by the virtual machine synthesizes logs that need to be analyzed. If this flood of information can be efficiently aggregated and correlated, this enables an administrator to understand the context of the applications, develop situational awareness, and leverage this awareness for detection and prevention. One potential avenue currently being explored for creating analyzable cloud infrastructure is by instrumenting logs in each level of the system and capturing them in a security information and event management (SIEM) solution such as Splunk. The SIEM solution will interrogate each service and aggregate the information, allowing for easy visualization of data and trends. Figure 4 is an example of the analysis framework we are investigating. With this framework, we have been able to quickly triage hardware and application failures as well as provide a record of the events on the system.

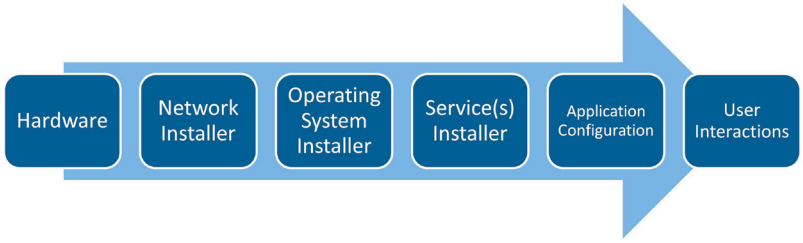


Figure 3 - Example cloud installation stack. As indicated by the arrow, the installation process flows from the hardware on the left to the final step of setting up the system for user interaction on the right.

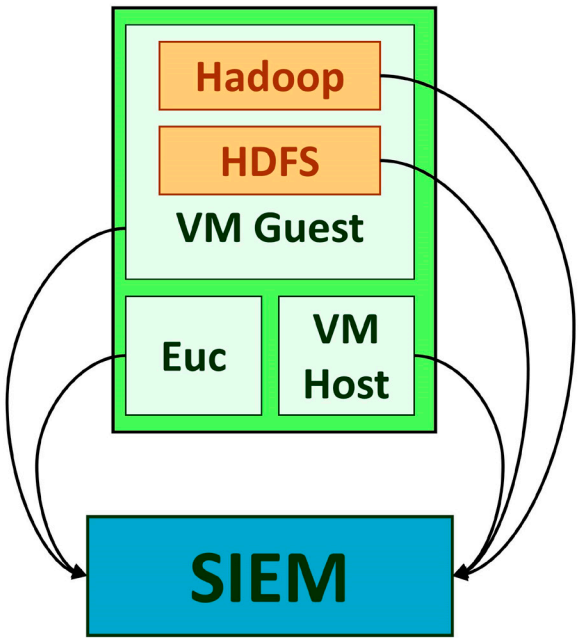


Figure 4 - System diagram of a full cloud analysis solution leveraging security information and event management solutions.

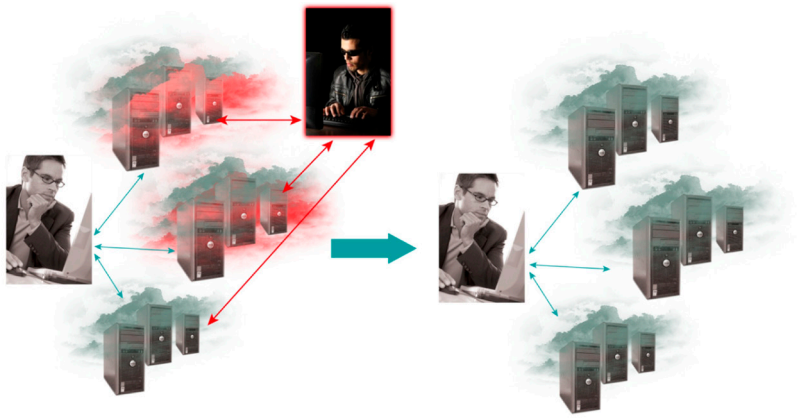


Figure 5 - The security of the cloud service should render the cloud usable to a legitimate user even when under attack.

While a common log area and visualizations are of great utility to an administrator, these are only a first step to better understand a cloud. Automated analysis techniques such as outlier identification and (un)supervised machine learning techniques can be built on top of this data, allowing for near real-time, less manually intensive identification of problems and security concerns. Also, much of this information would be useful to the end-user of the cloud. A critical research challenge for the future is enabling an end-user to leverage these logs in a secure manner.

Improved Cloud Storage

The third area we see as critical to the continued success of cloud computing is the continued development of improved data storage protocols. The concept of the cloud has been great for monetizing computing capabilities and a provider's return on investment has been tightly coupled with availability. This has typically left security as an afterthought or the burden has been placed on the user to ensure the confidentiality and integrity of their data. As seen in Figure 5, users need storage solutions that can seamlessly integrate multiple heterogeneous storage services (e.g., a local cloud storage service and Amazon S3) while providing the security the user needs and expects. Even with the system under attack, the user should be able to experience it as if the environment was benign.

One of the areas we are currently exploring is the use of wheat and chaff (W&C) storage. W&C uses multiple algebraic operations of linear subspaces to encode and replicate data. By encoding data in large finite fields, we create solutions which offer the end-user provable data confidentiality and integrity and provide lightweight checks on the user's data-related service level agreement. As part of this research, a completely non-preferential dynamic partitioning system was developed utilizing online codes [10] that allows for maximal robustness when splitting data between multiple cloud providers. This total system can provide the end-user with informed trade-offs between cost, performance, and security. This functionality is critical for the continued growth of hybrid solutions. For more information, see [17].

5. Looking Towards Continued Adoption

We believe the future of cloud computing rests in the opportunities and challenges present in hybrid cloud deployments. These allow organizations to have better resiliency to failure, establish data models for multiple types of data (i.e., increased privacy for data that remains in a private infrastructure), and optimize cost and resource usage by utilizing the appropriate cloud offerings. The solutions we present and continued work in the areas of creating automated cloud deployments, improved full cloud management, and secure storage will mitigate new cloud challenges before they become problematic.

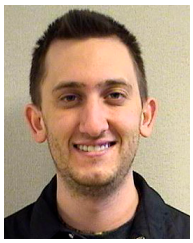
Disclaimers:

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000

ABOUT THE AUTHORS

David Zage is a Principle Member of Technical Staff at Sandia National Laboratories. He received his Ph.D. in computer science from Purdue University. He has worked on various research topics including computer security, distributed systems, fault-tolerant protocols, routing, and wireless mesh networks.

E-mail: djzage@sandia.gov



Dustin Franklin is a Graduate Student Intern at Sandia National Laboratories while he studies as a Ph.D. candidate in the computer science department at The University of New Mexico. His interests include distributed systems and adaptive intrusion response systems.

E-mail: drfrank@sandia.gov

Vincent Urias is a Principle Member of Technical Staff at Sandia National Laboratories, where he has spent the last ten years conducting cyber security research and development. His research areas include cyber testbedding, cyber modeling and simulation, as well as cyber analytics, cloud computing, and networking.

E-mail: veuria@sandia.gov

REFERENCES

1. Nelson Elhage. Virtunoid: Breaking out of kvm. Technical report, Black Hat USA, 2011.
2. Federal Computer Week. Cloud computing snapshot. Technical report, Federal Computer Week, 2010.
3. Frank Gens. New IDC IT cloud services survey: Top benefits and challenges. Technical report, IDC eXchange, 2009.
4. Nils Gruschka and Meiko Jensen. Attack surfaces: A taxonomy for attacks on cloud services. In Proceedings of IEEE CLOUD, 2010.
5. Jim Dempsey. <https://www.cdt.org/personnel/jim-dempsey> 2012.
6. Kaspersky Labs. Global it security risks: 2012. <http://www.kaspersky.com/downloads/pdf/kaspersky_global_it-security-risks-survey_report_eng_final.pdf> 2012.
7. Kostya Kortchinsky. Cloudburst: A vmware guest to host escape story. Technical report, Black Hat USA, 2009.
8. Mathew Schwartz. Sony hacked again, 1 million passwords exposed. <http://www.informationweek.com/security/attacks/sony-hacked-again-1-million-passwords-ex/229900111> 2011.
9. Doug Maughan et al. A roadmap for cybersecurity research. Technical report, Department of Homeland Security Science and Technology Directorate, 2009.
10. Petar Maymounkov. Online codes. Technical report, Technical report, New York University, 2002.
11. R McMillan. Sony cuts off sony online entertainment service after hack. Computer World, 2011.
12. Haroon Meer, Nicholas Arvanitis, and Marco Slaviero. Clobbering the cloud. In Black Hat USA, 2009.
13. David Molnar and Stuart Schechter. Self hosting vs. cloud hosting: Accounting for the security impact of hosting in the cloud. In Proceedings of WEIS, 2010.
14. Intel Research. Whatï¿½s holding back the cloud? Technical report, Intel, 2012.
15. Thomas Ristenpart, Eran Tromer, Hovav Shacham, and Stefan Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In Proceedings of ACM CCS, pages 199–212. ACM, 2009.
16. David Mitchell Smith. Hype cycle for cloud computing. Technical report, Gartner's, 2011.
17. David Zage and James Obert. Utilizing linear subspaces to improve cloud security. In Proceedings of IEEE/IFIP DSN-W, 2012.
18. Yinqian Zhang, Ari Juels, Michael Reiter, and Thomas Ristenpart. Cross-vm side channels and their use to extract private keys. In Proceedings of ACM CCS, pages 305–316. ACM, 2012.

The Art of Cyber Bank Robbery

Stealing Your Money Through Insidious Attacks

Aditya K. Sood, Michigan State University
Richard J. Enbody, Michigan State University

Abstract. Cyber criminals are using advanced attacks to exploit online banking systems and services to covertly steal money. This paper describes the tactics currently used by cyber criminals to conduct cyber bank robbery.

Introduction

Cyber criminals use botnets (malware) for a wide range of cyber crimes, and these attacks are increasing. The economics of e-crime and the related underground market have been studied which reveal a significant increase in online fraud [1]. Internet banking (e-banking) has transformed the economic and financial culture of the world. Over time, banks have strengthened the security of their servers to the point that attackers now target end-user systems. Server-side defenses are easier for banks because the banks have control over their servers. As client computers are outside of the banks' control, this makes it harder for the banks to subvert insidious attacks conducted on end-user systems. Due to this reason, Internet-based threats are posing security challenges to online banking. Given the increasing sophistication of attacks on the client side, it is imperative to build robust protection mechanisms on the client side that can be managed from the server side.

Necessity is the mother of invention. This aphorism applies to the current creativity of cyber criminals. Ever more sophisticated defenses have spurred attackers to develop more advanced attacks. The resulting innovative system-exploitation tactics exfiltrate data from infected clients around the world. The web browser is the primary user interface to the Internet and thus is a centralized target for attacks. The attackers design sophisticated client side malicious code that subverts a browser's functionality to harvest credentials and to perform money transfers on-the-fly in a hidden manner. The fact that these attacks are designed and structured around browsers shows how critical it has become to secure browser software.

Today, the most common platform for broad attacks on banking is via botnets. Those attacks are causing significant losses both in fraud and in defensive costs. Selling and renting botnet frameworks are an integral part of the underground economy's revenue model. Hundreds of millions of dollars are earned by cyber criminals, and billions of dollars are expended keeping those losses in check.

In 2009, Cormac and Dinei [2] conducted a study on the economics of the underground economy and estimated that a botnet herder earns approximately $0.50 per machine per year. For a botnet of 50,000 machines, a botnet herder could earn approximately $25,000. Recent botnets such as Zeus, SpyEye, and Citadel have infected millions of machines. If the same formula is applied, potential earnings are in millions of dollars every year. Some income comes from renting out the infected machines, but there are also Pay Per Infection (PPI) services where bot herders charge customers to distribute malware for a fee across their botnet. PPI rates vary significantly depending on where targeted machines are located. For example, $130 to $150 is charged per 1,000 machines to load malware on computers located in the U.S., but the rate is as low as $3 to $5 for locations in Asian countries such as China. In either case, providers of PPI services can earn millions of dollars annually.

On the defensive side, Anderson et al. in their study of cyber crime [3] pointed out that botnet mitigations cost $ 3.2 billion for anti-virus software alone. Globally, the study estimated that companies spend roughly $10 billion annually to provide defenses against cyber crimes. In addition, they projected that total global law enforcement expenditures were approximately $400 million for cyber crime. The study also concluded that global online banking fraud losses were close to $300 million, and to prevent additional frauds, banks spent approximately $1 billion. Florencio and Herley of Microsoft Research [21] found that credentials are offered in the underground market at $0.05 on the dollar value of the account. It leads them to observe that converting credentials to cash is the hard part and only a few stolen credentials result in actual theft. They analyze that the biggest cost comes from defensive costs and Anderson's data supports that conclusion.

In this paper, we present the cyber bank robbery model that is used by cyber criminals to conduct online frauds using automated exploitation frameworks such as botnets. This model is used for attacking end-user systems and mobile platforms.

Overview and Threat Model

Skilled cyber criminals are responsible for the majority of online bank fraud. The attack process can be outlined as follows:

- **Infection Entry Point and Exploitation:** A cyber criminal begins by co-opting a high-volume website to host an automated exploitation framework. That framework exploits browsers having vulnerable components using what is known as a drive-by download. The users are coerced to visit the infected website using techniques such as phishing. In addition, malicious applications can also be installed on mobile devices to control communication.
- **Data Exfiltration:** A bot is installed on the infected system that connects back to a C&C computer. For example, if the cyber criminal wants to attack Bank of America (BofA) sessions, it commands the bot to download the appropriate plugin. The bot hijacks (hooks) the communication channel initiated by the browser with the BofA website to steal account information, credentials, registered email addresses, etc. The key point is that the attack exploits client-side software, the browser in particular. Apart from that, the bots can

simply send phishing emails that exploit brand reputation of online websites and trick users to provide sensitive information. In mobile devices, apart from HTTP, SMS is used as a carrier for exfiltrating data.

- **Fraud:** Once the data is exfiltrated from the user machine, cyber criminals either sell it in the underground community or use it themselves. In advanced attacks, malicious code can execute fraudulent transactions directly from the infected systems. All these features depend on the design of bots.

This paper presents a model of cyber bank robbery structured into four phases. Phase 1 describes malware design. Phase 2 presents strategies to get malware onto users' computers and mobile devices. Phase 3 chronicles the exfiltration of sensitive data and automated transactions. Phase 4 covers the transformation of data to money. To conclude, we discuss different security mechanisms deployed by banks to combat online fraud and their shortcomings.

We use the following terminology: malware refers to any malicious code that modifies the behavior of target components. A bot is an automated malware that communicates with a remote server and performs multiple tasks in an infected system in a stealthy manner.

1. Phase 1: Malware Design

Botnets play a critical role in widespread infections on the Internet. A botnet is a network of compromised machines that are infected with bots. Bots steal sensitive information such as banking credentials from target users and have the ability to perform other nefarious tasks. The bots are sophisticated and implement advanced techniques to bypass antivirus engines and other host-based protection software [4].

Present-day bots have the capability to co-opt the communication flow in browsers through Man-in-the-Browser (MitB) attacks. These attacks enable the bots to harvest credentials using techniques such as form grabbing and web injects (explained later in this paper). In addition, the MitB attack allows the bots to make automated fraudulent transactions by exploiting the active session with the banks. Because these attacks are executed from the infected system, they are mostly hidden from the banks. MitB functionality has revolutionized the design of third-generation botnets. Since a browser is a user's window to the Internet, it is the target of attackers: controlling the browser controls the interaction. As operating systems have become hardened, attackers find attacking applications such as browsers to be easier. A detailed browser-malware taxonomy [5] exists that discusses the various classes of browser-based malware. Understanding browser-based malware is necessary to comprehend the strategies opted by malware authors to conduct stealthy attacks on the end user systems.

On similar benchmark, Man-in-the-Mobile (MitMo) attacks are conducted in mobile devices to manipulate and hijack the functionalities of installed applications. In these attacks, malicious applications use a camouflaging trick to hide their identity and trick users to believe them as authentic ones.

The cyber criminals are designing malicious code for computer systems as well as mobile platforms. The most prominent malware designs that are used in online banking frauds are discussed next.

1.1 Man-in-the-Browser (MitB) Agents

The evolution of MitB [6] attacks has given birth to advanced client-side attacks. MitB attacks are similar to Man-in-the-Middle (MitM) attacks, but exist within the operating systems to exploit browsers. MitB agents can be thought of as userland rootkits that subvert the integrity of browsers by hooking [7] selective Dynamic Link Libraries (DLL) to control the execution flow of various browser functions. When the browser calls a communication function, the hook diverts control to malicious code. This approach allows cyber criminals to conduct stealth attacks by manipulating the communication channel between browsers and the remote servers.

Hooking is an integral to many operating systems and is used frequently in Windows. In the context of browser exploits, hooking allows running processes to alter the behavior of various components in the system by intercepting the interprocess communication channel. The latest bots use inline function hooking [8] which is hard to detect because it uses hot patching and late binding, that is, the hook is actually executed during runtime. MitB agents are capable of stealing data, manipulating content and automating the critical operations without the intervention of users. Web injects and form grabbing are the two most widely used MitB techniques that implement hooking to control browser operations. These are discussed in the next sections.

1.2 Browser Rootkits

Browser rootkits [9] are defined as advanced levels of malware that hide inside browsers and perform unauthorized operations without users' knowledge. The concept of a browser rootkit originated from system rootkits that are capable of hiding and covertly interacting with the system components. Browser rootkits are malicious extensions (add ons) that use JavaScript to manipulate the content of web pages. In addition, browser rootkits can easily alter the look and feel of the web pages to fool users and trick them into performing illegitimate operations. These are also capable of altering information [10] in active sessions, account profiles, online transactions, etc. after the user successfully authenticates to an online banking website. The browser rootkits are primarily designed to execute fraudulent transactions when a user activates a session with an end server.

1.3 Man-in-the-Mobile (MitMo) Agents

With the advent of mobile technologies, cyber criminals have started targeting smart phones. Mobile platforms such as Android have been the target of cyber criminals. In the last few years, a number of mobile-based botnets have been revealed that subverted the integrity of mobile platforms to conduct attacks and exfiltrate sensitive information. For example: the existence of mobile variants of Zeus and SpyEye i.e. Zitmo and Spitmo [25] respectively show that the design of botnets is evolving with new technologies. Mobile botnets

[26] are similar to standard botnets but they aim specifically to exploit mobile architectures. Mobile bots are termed as MitMo agents that are malicious applications installed to thwart the security model of the mobile device and exfiltrate data accordingly. These are designed to control the communication channel initiated by legitimate applications with legitimate servers in a stealthy manner.

Malicious mobile applications work in conjunction with traditional botnets to subvert the multi channel protection mechanisms such as two-factor authentication (TFA) [29]. Malicious applications are designed to conduct piggybacking attacks [27] to monitor the state of target application (such as banks) and stealing information during transmission. Fake applications can also be forced to be installed on mobile devices that trick the users to provide sensitive information. On Android [30], apart from exploiting vulnerabilities, malware authors use infection techniques such as stealthy assets, infected boot images, time specific code execution, etc. to hide malicious codes. Android being open source is the preferred choice of cyber criminals. Because Blackberry and Apple use closed source operating systems, the ratio of mobile malware attacking these platforms is less than on Android.

1.4 Automated Phishing Bots

Apart from browser-based exploitation, bots are also designed to trigger phishing attacks. End users are tricked to visit illegitimate domains hosting fake web pages that appear similar to legitimate bank sites. Bots can send thousands of phishing emails at a time to a large set of users on the Internet. Honeynet [22] talks about how bots can be used to send phishing emails directly from infected computers and also from C&C panels. The phishing attacks are not new and have been in existence for years. But, the amazing part is that these attacks still exist and play a significant role in data exfiltration today. No stealthy technique is deployed during these attacks because phishing is based on social engineering to exploit the trust and knowledge of users. Botnets such as Grum and Festi [24] are specifically designed for conducting phishing attacks including spamming. On the contrary, Spamhaus [23] is an effort that is used to track botnets that send spam.

2. Phase 2: Malware Distribution

The following section is an examination of tactics chosen by cyber criminals to widely infect systems. Broad-based attacks (mass infections) have evolved over time and currently a popular technique is to drive victims to websites where they will be served malware or redirected to sites that serve malware. A target website is often a legitimate website that has been corrupted (e.g., injected with a malicious iframe) to send visitors to a malicious site. Some of the most-widely used malware distribution strategies are discussed below:

- Phishing is used to drive users to sites hosting a drive-by download attack [11]. A drive-by download attack silently exploits vulnerabilities in browser components to download malware without user action. This malware is capable of executing MitB attacks to perform fraudulent transactions and data exfiltration from the infected system. To automate the exploitation, cyber criminals have designed Browser Exploit Packs (BEPs) such as BlackHole. A browser exploit pack fingerprints the user's browser to identify vulnerabilities and then load the appropriate exploit. BEPs are sold as a crimeware service that charges buyers using a PPI model as discussed earlier.
- The popularity of Online Social Networks (OSNs) makes them attractive targets for attackers to distribute malware by exploiting trust among users. The attackers use the social network platforms and trust among "friends" to direct "friends" to malicious websites. For example, Likejacking attacks cause users to inadvertently "like" a malicious site that tricks a user to download malware.
- Bots are also distributed in traditional ways such as in warez or freeware that are downloaded from the illegitimate websites on the Internet carrying malware. Also, fake antivirus and other phony tools are still used to trick users to download malicious code.
- Bots have a built-in functionality of spreading using which they infect peripheral devices such as USBs to transmit themselves to different machines. In addition, spreaders can also infect Instant Messaging (IM) software and OSNs.
- Mobile bots and malicious applications are distributed as repackaged applications that mean the malicious code is hidden inside a legitimate application. The repackaged applications are distributed on alternate markets. Existence of vulnerabilities present in legitimate market stores also allows the attackers to host malicious applications. Other carriers include Over-the-Air (OTA) installation, mobile malvertising, etc.

Together these methods are sufficiently effective in distributing bots. The resulting zombie machines (infected systems) are managed remotely through a centralized C&C server that is owned and operated by a botmaster (or bot herder). Once a cyber criminal has controlled a set of infected computers, the next step in financial fraud is to collect credentials or conduct automated transactions.

3. Phase 3: Data Exfiltration and Stealthy Operations

Data exfiltration refers to transferring sensitive data from an infected machine to a remote C&C server. Multiple techniques exist; the most widely deployed data exfiltration and automated injection techniques used by banking malware are discussed below.

3.1 Form grabbing and Keylogging

Form grabbing is an impressive technique for extracting data present in web forms. This technique is more advanced than keylogging—a tool that results in a lot of irrelevant data that must be sifted through to find desired information such as credentials. In contrast, form grabbing grabs only the HTTP Post data sent as a part of form submission request. In particular, form grabbing greatly simplifies and automates the extraction of banking credentials making this process available for the less sophisticated criminal. However, with recent

botnets such as Citadel, both keylogging and form grabbing techniques are deployed for assurance purposes.

Form grabbing works on forms that users fill out and submit to a bank—especially forms used for logging and online transactions. As the browser is already hooked (MitB), a bot agent can easily snoop the communication channel between the client and the server. As soon as the user submits the form, the bot agent extracts the data present in the forms, generates a socket in the system and transmits the data back to a C&C server. Data in all the HTTP POST requests can be exfiltrated from the system without a user's knowledge [12].

```
set_url https://www.wellsfargo.com/* G
data_before
<span class="mozcloak"><input type="password"*</span>
data_end
data_inject
<br><strong><label for="atmpin">ATM PIN</label>:</strong> <br />
<span class="mozcloak"><input type="password" accesskey="A" id="atmpin" name="USpass"
size="13" maxlength="14" style="width:147px" tabindex="2" /></span>
data_end
data_after
data_end
```

Listing 1 - WI rule written against Wells Fargo Bank

3.2 Web Injects

Web Injects (WI) is an advanced technique of content injection. When a user submits a form and waits for a response from a web server, a bot agent is activated and starts injecting illegitimate content into the incoming HTTP responses. This process tricks the user into believing the web server has sent all of the content. WI is effective in coercing users to provide information that is otherwise not easy to attain. For example, an attacker could request a PIN, a Social Security number, or a second-channel SMS number. This attack is a variant of a MitB attack because it hooks various read/write functions in browser libraries to inject data. This technique is implemented as follows:

- Cyber criminals have to design specific rules for a bot agent to perform WI. A bot agent reads various rules from a static file and then uses hooking to apply those rules to modify incoming HTTP responses. Rules are tied to specific web pages, e.g., the login page of a bank.
- It is crucial that the rules are structured properly because inappropriate WI rules can seriously disrupt the web page layout and the dynamic execution of JavaScripts. Wild modification of the web stream will be obvious and hence ineffective. For successful WI, the injected content has to work inline without any display of errors or notifications to the users.
- Cyber criminals are required to define several parameters to write different WI rules. The WI rules are written explicitly for every GET and POST request with a dedicated URL. There are two specific parts of the WI rule. First, it is required to define the target URL (bank website, etc.) whose content is to be hooked and modified. Second, in every rule it is required to define the layout of the web pages, e.g. specify a portion of the webpage in which the content is to be injected in order to render the content appropriately in the browser.

Listing 1 shows a WI rule extracted from an infected machine. The rule injects additional input asking for a user's ATM PIN. It is an unusual request from a bank, but since the page is otherwise legitimate, trust compels a user to enter the information. This injection is placed before the password input box (specified by the data_before tag)—injecting inline as the web page enters the browser. The details of the parameters used to write a WI rule are discussed in [13]. As WI is a problem at the client side, banks currently have no robust protection against this attack. In addition, cyber criminals can inject sophisticated JavaScripts to perform online transactions automatically. For example, a bot injects malicious JavaScript during an active session with the server. The JavaScript interacts with the server and initiates a transfer from the user's account to an offshore institution. When the server sends a notification about a change in balance in the account, the incoming data (balance amount) is manipulated to reflect a different number. The user is tricked to believe that the account balance is intact. A bot can also generate unauthorized messages on behalf of the server.

3.3 Custom Plugins

Modern botnets implement a plug-in framework for executing a variety of attacks. The plug-in framework extends the capability of botnets by allowing the cyber criminals to write custom code that can be easily incorporated into running botnets. During our analysis of the SpyEye botnet [15], we came across interesting plug-ins that are used for data exfiltration. These are as follows:

- A browser certificate-grabber plug-in captures information about various certificates that are present in the browser storage repository and are used to verify the integrity of communicating parties.
- A credit card-grabber plug-in that is designed specifically to extract credit card information during an active session with a bank's server.
- A screenshot stealer and video grabber plug-ins that capture screenshots and videos of the browser when a user performs online banking. In addition, cyber criminals configure plug-ins in such a manner that a screenshot is captured based on the movements of the mouse cursor.
- Cyber criminals can also design plug-ins specific to a bank's website. For example, the SpyEye botnet has built-in information stealing plug-in that is designed specifically for BofA.

3.4 Mobile Platforms: SMS and HTTP as Data Carriers

Most of the mobile platforms are smart phones these days that provide the same functionality as standard computers, so data exfiltration models remains the same. The mobile bots and malicious applications can perform keylogging and monitoring of data that is transmitted through the device. Generally, mobile bots can communicate over HTTP and control the communication flow. The primary addition in the data exfiltration process apart from standard protocols is the use of SMS as a carrier for transmitting data. It means the mobile bots can steal sensitive information and use the SMS capability

of the device to send data to a backend domain managed by the cyber criminal. Mobile bots can perform piggybacking on legitimate applications and steal data by controlling specific events such as when the applications send data to a banking server. As discussed earlier, mobile bots can also circumvent the TFA process that uses SMS (mobile) as a second channel. Zitmo and Spitmo are the examples of mobile malware that support this fact.

3.5 Phished Web Pages

As discussed in the malware design section, automated bots are used for sending phishing emails with luring links. The phishing emails are constructed in a sophisticated manner that it becomes easy to force the users to visit the phished website. Once the user clicks the embedded link, the browser opens the phished website, which contains web forms that ask specific information from the users. Since the web pages look legitimate, users provide sensitive information such as credentials, credit card numbers, etc. This is an old-school trick, but works neatly in exfiltrating data from infected end user machines.

4. Phase 4: Underground Business

At some point, stolen data must be converted to cash, and for that we turn to the underground economy. In the underground market, there are three basic players: sellers, buyers and money mules. Sellers sell the data, buyers purchase the data, and money mules convert data to cash.

4.1 Underground Forums and IRC Channels as Business Platforms

Internet Relay Chat (IRC) [19] channels are used as the primary business platform in the underground economy because it allows cyber criminals to remain anonymous. Cyber criminals use Virtual Private Network (VPN) to initiate connections to IRC servers for registering communication channels. With the existence of invisible IRC, the communication channels are unreadable, encrypted and untraceable.

Once data is successfully stolen from infected machines, cyber criminals need to sell it. During our study, we analyzed underground forums that advertise various IRC channels used by cyber criminals to sell sensitive information. Automated MIRC scripts regularly advertise updates and availability of the stolen data. Sellers advertise a unique ICQ code with an IRC channel that a buyer can use to connect directly so the buyer is unable to identify the seller.

Data is sold in the form of dumps as shown in Figure 1 that are sent to the buyer once the seller receives payment.

Sellers require money in the form of Liberty-Reserve, Western Union, Money Gram, etc., which are e-currencies that can be converted into Euros, dollars or pounds. E-currency involves an intermediate third-party who does not reveal the identity of the buyer or the seller to maintain anonymity. The underground business is based on an implicit trust between the buyer and the seller that the seller will release purchased data upon receiving payment—there is no third party to turn to for resolving disputes.

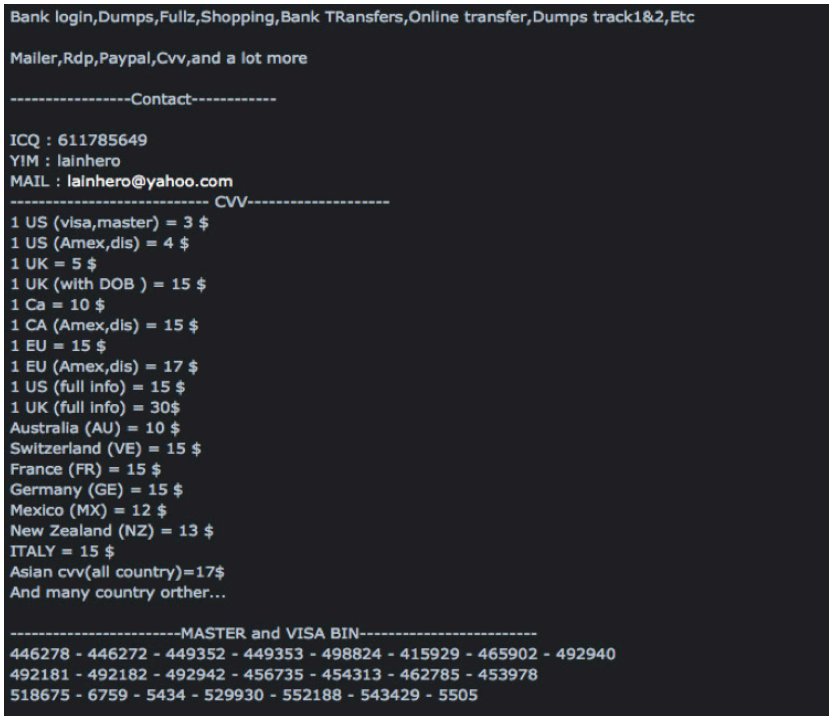


Figure 1 - Advertising Dumps of the Stolen Bank Data (Source: Underground Forum <http://madtrade.org/>)

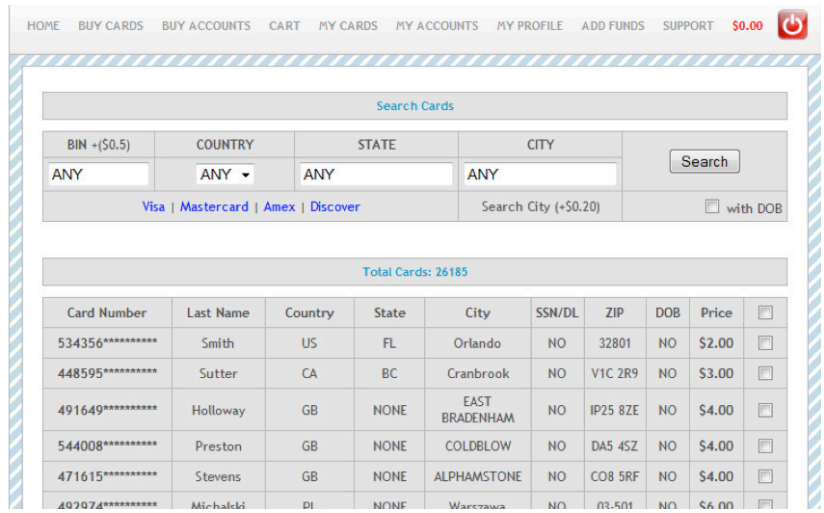


Figure 2 - Credit Card Shop in Action

4.2 Credit Card (Plastique) Shops

Credit card shops are e-shops that exist in the underground market to sell stolen credit card information. The credit card shops are similar to regular e-commerce websites. The buyer visits various underground websites to find information about the credit card sellers, and obtain the address of the credit card shops from various IRC channels and underground forums. The buyer then has to register with the shop. Once the registration is complete, the buyer can easily navigate the credit card shop and select credit cards for purchasing. Currently, the stolen credit card information is sold at very cheap rates ranging from $2 to $20. Figure 2 shows the layout of one current credit card shop we found during penetration testing of domains associated with malware.

Figure 3- Service Advertisements for Offshore Money Transfers (Source: Underground Forum http://madtrade.org/)

4.3 Money Mules

Money mules [18] are transfer agents hired to convert data into cash. For a fee, money mules use credentials (data) to extract money from a bank and then transfer the money to offshore accounts, often as e-currency. For bank transactions, money mules must usually have accounts in the banks that are targeted by cyber criminals for transferring funds—a requirement that puts mules at risk.

Most banks have strong security measures for transferring money outside a bank, but little security for transfers within a bank so it is common to transfer within a bank. We assume that credentials have been collected using techniques such as form grabbing as described above.

- Sometimes additional information is needed such as the user's account including registered email and password. It can be easily collected using techniques such as Form-grabbing or Web Injects as described above. If the bank uses TFA, the associated information such as an SMS number can be gathered in the same way. Hijacking sessions while in progress as outlined above can circumvent one-time passwords.
- With that information, the buyer needs to enlist a mule so the buyer needs the mule's name, account number, and routing number. Given restrictions on transfer amounts, multiple transactions or multiple mules may be needed.
- A buyer can use account credentials to transfer money to a mule or a mule's credentials can be provided back to the seller to build transactions into a victim's live session. The seller can use WI to inject the mule's credentials into web pages using JavaScript. The script causes a fraudulent transaction during a user's session to transfer money directly to the mule's account.
- Once money has been transferred to a mule's account, the buyer sends a confirmation to the money mule, e.g., a screenshot. Upon receiving the confirmation, the money mule moves the money outside the bank. The transfer may be to cash, to an overseas account, to merchandise, or to e-currency. Upon transferring the money, the mule will extract a fee for their services. The fee can vary significantly depending on the complexity of service provided, but we have observed fees ranging from 2% to 10%. Figure 3 shows an advertisement for this kind of service in the underground market. Money mules are prevalent in regions that currently lack strong cyber laws: Eastern Europe, Russia, Middle East, etc.
- An optional fourth actor may be present—a bank insider who can be thought of as a type of money mule. A bank employee can facilitate overseas transfers, especially large transfers. An overseas transfer needs another money mule at the other end to complete the transaction.

Underground markets facilitate the buying and selling of the stolen data without revealing the identity of the players.

5. Existing Countermeasures and Defensive Mechanisms

Banks are deploying several interesting techniques to combat online fraud. Several of them are discussed as follows:

- The majority of banks implement SSL that protects customers from network layer attacks by encrypting the channel between end points. While worthwhile, this practice is not suffice to combat browser-based data exfiltration attacks conducted by MitB agents. By working within the browser, the attack is done before SSL encrypts the data.
- Banks also deploy multi-factor authentication systems using multiple channels to authenticate clients. A popular one is TFA. Display tokens such as RSA Secure ID, Safenet's e-token and Vasco secure tokens use either time-based or sequence-based algorithms to generate unique tokens for authentication or digital transaction signing. The user possesses a small device that generates tokens at regular intervals. The token is used as a second factor in authentication. For example, HSBC bank uses RSA Secure ID, and BofA uses Safe Pass.
- In a variation on TFA, some banks use one-time passwords [20] for authentication. Banks store the information about users' computers including IP address, browser, geo IP location, etc. If the bank's server finds that the information has changed, it activates the OTP scheme. The bank will have on file either an email address or mobile number for receiving the OTP. Using this second channel, the OTP is sent to the user. JP Morgan Chase bank is an example of a bank that implements this procedure.
- Banks have also implemented site-key authentication to thwart phishing attacks. During account registration, the user selects an image with a key for additional verification. The legitimate account login page includes this site key which assures the user of the authenticity of the website. Typically, a complete site key consists of an image, selected text and challenge questions. Generally, the challenge questions are asked when the connected computer is not recognized. BofA and HDFC bank are examples of banks that incorporate this functionality. Note that this technique does not help prevent MitB attacks.
- Some banks recommend third party monitoring solutions such as Trusteer Rapport [17]. It is an active fraud prevention and account takeover detection solution, and users are advised to install it before using banking websites. Companies like Netqin [28] provide mobile anti-malware solutions to protect the integrity of mobile devices.
- Banks have also built a protection against keylogging attacks in the form of virtual keyboards using JavaScript. This technique prevents keylogging but fails to protect against form grabbing. A few banks are using client-side password encryption to defend against the reuse of stolen credentials. The State Bank of India (SBI) is following this practice.
- Apart from technical solutions, banks also perform forensic investigative analysis of money fraud problems reported by users. This includes analyzing the anomalies that persist in transactions. The anti-fraud teams collaborate with government agencies to unmask the players behind these frauds.

Banks are taking a variety of steps to fight against a variety of cyber crime, but none prevent current MitB attacks. TFA is an effective defense against the use of stolen credentials, but WI can allow criminals to collect information on the second channel. TFA raises the bar and WI provides a work-around, but it is a difficult work-around.

6. State of Cyber Laws

Nations with advanced economies such as the U.S. or the UK have begun to implement cyber laws. The biggest problem in eradicating cyber crime globally is the lack of centralized cyber laws. The proposed cyber laws are country specific and cannot be enforced across borders (except to a limited extent through existing treaties). Quite naturally, countries are most concerned with cyber crimes that impact their own institutions, so law enforcement agencies are more interested in investigating or prosecuting cyber criminals that exploit the integrity of their own country's critical infrastructure. Contributing to the problem is the international nature of cyber crime. Cyberspace has no borders so cyber criminals can work anywhere. Many countries have still not implemented strong cyber laws and that is a problem for managing cyber crime internationally. The laws that have been implemented vary considerably—the crimes are too new to have developed widespread standards. The U.S. is one of the leaders in making and implementing cyber laws [16] but those laws cannot be enforced globally. As an example, U.S. cyber law 18 USC 1030 deals with crimes that are conducted through compromised (unauthorized access) computers and further using them to execute identity fraud against financial institutions. A convicted person can get five to 10 years in prison. Clearly, more needs to be done and countries are working to build a robust approach against cyber crime. The efforts must be international, if we are to build a secure cyberspace.

Conclusion

In this paper, we presented attack methods for conducting online bank fraud. To carry out fraud, cyber criminals have created sophisticated methods of malware distribution, infection, and data exfiltration. One important trend is toward infecting users' systems rather than attacking banks' servers. The criminals coerce users to visit malicious domains where drive-by downloads use browser vulnerabilities to download malware. The malware hooks browser functions to allow form data (credentials) to be grabbed from banking sessions. On mobile devices, malicious applications are installed that perform piggybacking, hijacking communication channels of other legitimate applications and transmitting data using HTTP or SMS to remote servers. The sensitive information is sent to cyber criminals who convert data to cash using different channels. Some banks have implemented OTP and TFA—and these authentication systems work well against some attacks—but they fail to provide adequate protection against MitB and MitMo attacks. As a result, cyber bank fraud has become a critical problem on the Internet. To secure online banking, multilayer defenses including user education are needed.

ABOUT THE AUTHORS

Aditya K. Sood is a senior security researcher/consultant and PhD candidate at Michigan State University. His research interests include web security, malware analysis, mobile security, and penetration testing. Sood has an MS in cyber law and information security from the Indian Institute of Information Technology, India. He is regular speaker at industry wide security conference and have contributed to number of security magazines and journals.

30 Newport Parkway, Apt 2111
Jersey City, NJ -07310, USA
Phone: 517-755-9911
E-mail: soodadit@cse.msu.edu

Richard J. Enbody, Ph.D., is associate professor in the Department of Computer Science and Engineering at Michigan State University (USA) where he joined the faculty in 1987. Enbody has served as acting and associate chair of the department and as director of the computer engineering undergraduate program. His research interests include computer security; computer architecture; web-based distance education; and parallel processing.

Department of Computer Science and Engineering
Michigan State University
3115 Engineering Building
East Lansing, Michigan 48824
Phone: 517-353-3389
Fax: 517-432-1061
E-mail: enbody@cse.msu.edu

REFERENCES

1. V. Garg, C. Kanich and L Camp, Analysis of eCrime in Crowd-sourced Labor Markets: Mechanical Turk vs. Freelancer, 11th Workshop on the Economics of Information Security (WEIS), 2012
2. C. Herley and D. Florencio, Nobody Sells Gold for the Price of Silver: Dishonesty, Uncertainty and the Underground Economy, In Proceedings (online) of the Workshop on Economics of Information Security, June 2009.
3. R. Anderson, C. Barton, R. Bohme, R. Clayton, M. Eeten, M. Levi, T. Moore and S. Savage , Measuring the Cost of Cybercrime, 11th Workshop on the Economics of Information Security (WEIS), 2012
4. G. Ollman, Serial Variant Evasion Tactics. Damballa Whitepaper. <http://www.damballa.com/downloads/r_pubs/WP_SerialVariantEvasionTactics.pdf>
5. A. Sood and R. Enbody, A Browser Malware Taxonomy, Virus Bulletin Magazine, June 2011 <http://secniche.org/released/VB_BRW_MAL_TAX_AKS_RJE.pdf>
6. K. Curran and T. Dougan. Man in the Browser Attacks. International Journal of Ambient Computing and Intelligence. Vol (4) -1, 2012
7. N. Harbour, Win at Reversing - API Tracing and Sandboxing through Inline Hooking, In 17th Annual DEFCON Conference, 2009
8. J. Butler and P. Silberman, RAIDE - Rootkit Analysis Identification and Elimination, In BlackHat Security Conference, 2006.
9. C. Devaux and J. Lenoir, Browser Rootkits, Hack Luxembourg Conference, 2008 <http://archive.hack.lu/2008/rootkits-navigateurs.pdf>
10. C. Jackson, D. Boneh and J. Mitchell, Transaction Generators - Rootkits For Web, In Usenix HotSec Conference, 2007
11. M. Cova, C. Kruegel and G. Vigna. Detection and analysis of drive-by-download attacks and malicious JavaScript code. In Proceedings of the 19th international conference on World wide web, 2010
12. Malware-at-Stake Blog, (SpyEye & Zeus) Web Injects - Parameters, <http://secniche.blogspot.com/2011/07/spyeye-zeus-web-injects-parameters-and.html>
13. A. Sood, R. Enbody and R. Bansal, The Art of Stealing Banking Information - Form Grabbing on Fire, Virus Bulleting Magazine, November, 2001.
14. Chase - Malware and Virus, Do Not Fill Out Pop-Up Windows Like This <https://www.chase.com/index.jsp?pg_name=ccpmapp/privacy_security/fraud/page/virus_malware_examples>
15. A. Sood, R. Enbody and R. Bansal, Dissecting SpyEye – Understanding the design of third generation botnets, Elsevier Computer Networks Journal, Online Print, August 2012
16. A. Rees, Cybercrime Laws of the United States, October, 2006. <http://www.oas.org/juridico/spanish/us_cyb_laws.pdf>
17. Trusteer Rapport, User Guide, <http://www.trusteer.com/support/user-guide/3.5.1201/Rapport_UG_3_5_1201_4.pdf>
18. M. Aston, S. McCombie, B. Reardon, and P. Watters. A Preliminary Profiling of Internet Money Mules: An Australian Perspective. In Proceedings of the 2009 Symposia and Workshops on Ubiquitous, Auto nomic and Trusted Computing, 2009.
19. C. Mazzariello. IRC Traffic Analysis for Botnet Detection. In Proceedings of The Fourth International Conference on Information Assurance and Security, 2008
20. A. Rubin, Independent one-time passwords. In Proceedings of the 5th conference on USENIX UNIX Security Symposium, 1995
21. Dinei Florencio, Cormac Herley, "Is Everything We Know about Password Stealing Wrong?" IEEE Security & Privacy, vol. 10, no. 6, pp. 63-69, Nov.-Dec., 2012
22. Honeynet Project, Phishing Using Botnets, <http://www.honeynet.org/node/92>
23. Spamhaus, <http://www.spamhaus.org>
24. T. Morrison, Spam botnets: The fall of Grum and the rise of Festi, <http://www.spamhaus.org/news/article/685/spam-botnets-the-fall-of-grum-and-the-rise-of-festi>
25. C. Castillo, Spitmo vs. Zitmo: Banking Trojans Target Android, <http://blogs.mcafee.com/mcafee-labs/spitmo-vs-zitmo-banking-trojans-target-android>
26. Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution, <http://www.csc.ncsu.edu/faculty/jiang/pubs/OAKLAND12.pdf>
27. W. Zhou, Y. Zhou, M. Grace, X. Jiang and S. Zou, "Fast, Scalable Detection of `Piggybacked' Mobile Applications, <http://www.csc.ncsu.edu/faculty/jiang/pubs/CODASPY13.pdf>
28. NetQin, NetQin Mobile Security, <http://www.netqin.com/en/antivirus>
29. SecNiche Security Blog, <http://secniche.blogspot.com/2012/08/digital-forensics-magazine-dismantling.html>
30. A. Sood, ToorCon 14 (2012): Malandroid - The Crux of Android Infections, <http://zeroknock.blogspot.com/2013/05/toorcon-14-2012-malandroid-crux-of.html>

Digital Forensics in the Cloud

Shams Zawoad, University of Alabama at Birmingham
Ragib Hasan, University of Alabama at Birmingham

Abstract. Today's cloud computing architectures often lack support for computer forensic investigations. Besides this, the existing digital forensics tools cannot cope with the dynamic nature of the cloud. This paper explores the challenges of digital forensics in the cloud, possible attacks on cloud-evidence, and mitigation strategies against those challenges.

Introduction

Cloud computing offers immense opportunities for business and IT organizations by providing highly scalable infrastructure resources, pay-as-you-go service, and low-cost on-demand computing. While clouds attract diverse organizations, the security and trustworthiness of cloud infrastructure has become a rising concern. Clouds can be a target of attacks or can be used as a tool to launch attacks. Malicious individuals can easily exploit the power of cloud computing and can perform attacks from machines inside the cloud. Many of these attacks are novel and unique to clouds.

To illustrate the use of clouds for malicious purpose, we consider the following hypothetical scenario:

Bob is a successful businessman who runs a shopping website in the cloud. The site serves a number of customers every day and his organization generates a significant amount of profit from it. Therefore, if the site is down even for a few minutes, it will seriously hamper not only their profit but also the goodwill. Mallory, a malicious attacker, decided to attack Bob's shopping website. She rented some machines in a cloud and launched a Distributed Denial of Service attack to the shopping website using those rented machines. As a result, the site was down for an hour, which had quite a negative impact on Bob's business. Consequently, Bob asked a forensic investigator to investigate the case. The investigator found that Bob's website records each visiting customer's IP address. Analyzing the visiting customer records, the investigator found that Bob's website was flooded by some IP addresses which are owned by a cloud service provider. Eventually, the investigator issued a subpoena to the corresponding cloud provider to provide him the network logs for those particular IP addresses. On the other hand, Mallory managed to collude with the cloud provider after the attack. Therefore, while providing the logs to the investigator, the cloud provider supplied a tampered log to the investigator, who had no way to verify the correctness of the logs. Under this circumstance, Mallory will remain undetected. Even if the cloud provider was honest, Mallory could terminate her rented machines and leave no trace of the attack. Hence, the cloud provider could not give any useful logs to the investigator.

Fig. 1: Process Flow of Digital Forensics

To identify the actual attacker in the above attack scenario, we need to execute digital forensics procedures in clouds. Currently, extensive research is going on to protect clouds from external or internal attackers. However, in case of an attack, we need to investigate the incident. Besides protecting the cloud, it is important to focus on this issue. Unfortunately, cloud forensics is not yet a popular research topic and there has been little research on adapting digital forensics for use in cloud environments. In this paper, we address the problems of cloud forensics and some mitigation strategies, which have significant real-life implications in investigating cloud-based cyber-crime and terrorism.

Understanding Cloud Forensics

NIST defines digital forensics as an applied science for "the identification, collection, examination, and analysis of data while preserving the integrity of the information and maintaining a strict chain of custody for the data" [1]. Figure 1 illustrates the process flow of digital forensics. Cloud forensics can be defined as applying all the processes of digital forensics in the cloud environment. Ruan et al. defined cloud forensics as a subset of network forensics [2], because cloud computing is based on extensive network access, and network forensics handles forensic investigation in private and public networks. However, cloud forensics also includes investigating file systems, process, cash, and registry history. Different steps of digital forensics shown in Figure 1 vary according to the service and deployment model of cloud computing. For example, the evidence collection procedure of Software-as-a-Service (SaaS) and Infrastructure-as-a-Service (IaaS) will be different. For SaaS, we solely depend on the Cloud Service Provider (CSP) to get the application log. In contrast, in IaaS, we can acquire the virtual machine image from customers and can initiate the examination and analysis phase. In the public deployment model, we rarely can get physical access to the evidence, but this is guaranteed in the private cloud deployment model.

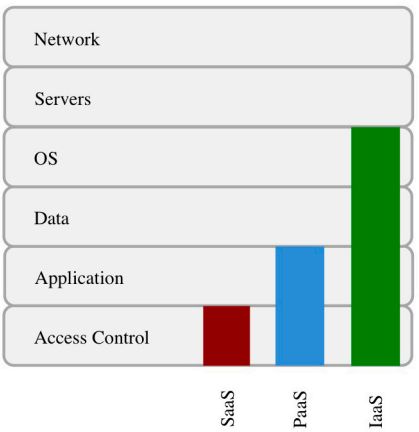


Fig. 2: Customers' control over different layers in different service model

Why Are Clouds Not Forensics Friendly?

Several characteristics of cloud computing complicate the process of cloud forensics. As the storage system is no longer local, law enforcement agents cannot confiscate the suspect's computer and get access to the digital evidence even with a subpoena. In a cloud, each server contains files from many users. Hence, it is not feasible to seize servers from a data center without violating the privacy of many other benign users. Moreover, even if the data belonging to a particular suspect is identified, separating it from other users' data is difficult. The trustworthiness of the evidence is also questionable, because other than the cloud provider's word, there is no usual way to link a given evidence to a particular suspect. The following issues make cloud forensics challenging.

• In traditional computer forensics, investigators have full control over the evidence (e.g., router logs, process logs, and hard disks). Unfortunately, in a cloud, the control over data varies in different service models. Figure 2 shows the control of customers in different layers for the three different service models – IaaS, PaaS, and SaaS. Cloud users have highest control in IaaS and least control in SaaS. This physical inaccessibility of the evidence and lack of control over the system make evidence acquisition a challenging task in the cloud. For example, in SaaS, customers do not get a log of their system, unless the CSP provides the logs. In PaaS, it is only possible to get the application log from the customers. To get the network log, database log, or operating system log we need to depend on the CSP. In IaaS, customers can only get the operating system logs, they do not have access to network or process logs. For example, Amazon does not provide load balancer logs to the customers [3], and it is not possible to get MySql log data from Amazon's Relational Database Service [4].

• Cloud computing is a multi-tenant system, while traditional computing is a single owner system. To give an analogy, the cloud can be compared to a motel, while the other can be compared to a personal house. In a cloud, multiple Virtual Machines (VM) can share the same physical infrastructure, i.e., data for multiple customers can be co-located. An alleged suspect may claim that the evidence contains information of other users, not her. In this case, the investigator needs to prove to the court that the provided evidence actually belongs to the suspect. Conversely, in traditional computing systems, a suspect is solely responsible for all the digital evidence located in her computing system. Moreover, in the cloud, we need to preserve the privacy of other tenants. The multi-tenancy characteristic also brings novel side-channel attacks [5] that are difficult to investigate.

• Volatile data cannot sustain without power. Data residing in a VM are volatile, as after terminating a VM, all the data will be lost. In order to provide the on demand computational and storage service, CSPs do not provide persistent storage to VM instances. There is, though, a way to preserve VM data by storing an image of the VM instance. An attacker can exploit this vulnerability in the following way: after doing some malicious activity (e.g., launch DoS attack, send spam mail), an adversary can terminate her VM that will lead to a complete loss of the evidence and make the forensic investigation almost impossible. A malicious user can also fraudulently claim that her instance was compromised by someone else who had launched a malicious activity. In the absence of any evidence, it will be difficult to prove her claim as false via a forensic investigation [6].

• Chain of custody is one of the most vital issues in traditional digital forensic investigation. Chain of custody should clearly depict how the evidence was collected, analyzed, and preserved in order to be presented as admissible evidence in court [7]. In traditional forensic procedure, it is trivial to maintain an access history of time, location, and person to access the computer, hard disk, etc. of a suspect. On the other hand, in a cloud, we do not even know where a VM is physically located. Also, investigators can acquire a VM image from any workstation connected with the internet. The Investigator's location and a VM's physical location can be in different time zones. Hence, maintaining a proper chain of custody is challenging in clouds.

• Currently, investigators are completely dependent on CSPs for acquiring cloud evidence. However, the employee of a cloud provider, who collects data on behalf of investigators, is most likely not a licensed forensics investigator and it is not possible to guarantee his integrity in a court of law. A dishonest employee of a CSP can collude with a malicious user to hide important evidence or to inject invalid evidence to prove the malicious user is innocent. On the other hand, a dishonest investigator can also collude with an attacker. Even if CSPs provide valid evidence to investigators, a dishonest investigator can remove some crucial evidence before presenting it to the court or can provide some fake evidence to the court to frame an honest cloud user. In traditional storage systems, only the suspect and the investigator can collude. The three-way collusion in the cloud certainly increases the attack surface and makes cloud forensics more challenging.

Requirements For Forensics-Enabled Cloud

To mitigate the challenges that we discussed above, we identified the following characteristics that a forensics-enabled cloud should have:

• As CSPs do not provide persistent storage to VMs, turning off or rebooting a VM will eventually lose all the data residing in that VM. Data that are volatile in nature must be stored in persistent databases so that even if a malicious user terminates her virtual machine, we can still gather the evidence. One possible solution to this problem is that CSPs will provide a continuous synchronization API to customers. Using this API, customers can preserve the synchronized data to any cloud storage e.g., Amazon S3, or to their local storage. However, if the adversary is the owner of a VM, this mechanism will not work. Trivially, she will not be interested in synchronizing her malicious VM. To overcome this issue, CSPs by themselves can integrate the synchronization mechanism with every VM and preserve the data within their infrastructure. CSPs can constantly monitor all the running VMs and store the volatile data in a persistent storage. The volatile data can be network logs, operating system logs, and registry logs. When a VM is in active state, CSPs can track which data belongs to which VM. Hence, while preserving the data, CSPs can take care of segregating the data according to VM owner. In this way, multiple VM owners' data will not be co-mingled.

• After preserving all the evidence, CSPs need to ensure the integrity of the evidence in order to prevent collusion between CSPs, investigators, and cloud users. Without integrity preservation, the validity of the evidence will be questionable and the defense and the jury can object about it. Generating a digital signature on the collected evidence and then checking the signature later is one way to validate the integrity. Another way is preserving the proofs of past data possession [8]. Preserving the proofs of files can significantly decrease the continuous synchronization cost and at the same time ensure the integrity and confidentiality of cloud evidence. Trusted Platform Module (TPM) can also protect the integrity of cloud evidence. By using a TPM, we can get machine authentication, hardware encryption, signing, secure key storage, and attestation. It can provide the integrity of the running virtual instance, trusted logs, and trusted deletion of data to customers.

• Besides preserving the integrity of evidence, CSPs also need to provide proper chain of custody information. As provenance provides the history of an object, by implementing cloud provenance, CSPs can provide the chronological access history of evidence, how it was analyzed, and preserved, which can ensure the chain of custody for cloud forensics. However, as all the evidence and the access histories are under the control of CSPs, they can always tamper with the provenance record. Moreover, from the provenance data in the cloud, an attacker can learn confidential information about the data stored in the cloud. To protect provenance information from these types of attack, we need a secure provenance scheme [9].

• Considering CSPs are preserving all the evidence, investigators will be still dependent on CSPs to collect evidence, as all the cloud evidence resides in the providers' data center. CSPs can play a vital role in this step by providing a web-based management console or providing secure API to law enforcement agencies. Using web console or API, customers as well as investigators can collect network, process, database logs, and other digital evidence as well as the provenance records of those evidence.

Moving Towards Regulatory Compliant Cloud

As cloud computing does not provide the facility of proper forensics investigations, it cannot be used to store healthcare, business, or national security-related data, which require audit and regulatory compliance. Auditability is a vital issue to make the cloud compliant with the regulatory acts, e.g., The Sarbanes Oxley (SOX) Act [10] or The Health Insurance Portability and Accountability Act (HIPAA) [11]. According to SOX, financial information must reside in auditable storage that the CSPs cannot provide currently. Business organizations cannot move their financial information to a cloud, as it does not comply with the SOX act. As cloud infrastructures do not comply with HIPAA's forensic investigation requirement, hospitals also cannot move their patients' confidential medical records to cloud storage. A forensics-enabled cloud architecture that satisfies all the requirements stated in the previous section will definitely increase the auditability of a cloud environment. By deploying such an architecture, we will be able to store and provide the types of evidence from which we can get all the activities of cloud users.

Business and healthcare organizations are the two most data consuming sectors. Hence, cloud computing cannot reach its goal without including these two sectors. These sectors are spending extensively to make their own regulatory-compliant infrastructure. A regulatory-compliant cloud can save this huge investment. We need to solve the audit compliance issue to bring more customers into the cloud world. Implementing an architecture that allows cloud forensics investigations will make clouds more compliant with such regulations, leading to widespread adoption of clouds by major businesses and healthcare organizations.

Conclusion

In this paper, we discussed the technical challenges of executing digital forensic investigations in a cloud environment and presented the requirements to make clouds forensics-friendly. Collecting trustworthy evidence from a cloud is challenging as we have very little control over clouds compared to traditional computing systems. For now, investigators need to depend on the CSP to collect evidence from a cloud. To make the situation even worse, there is no way to verify whether the CSP is providing correct evidence to the investigators, or the investigators are presenting valid evidence to the court. Thus, we need to build a trust model to preserve the trustworthiness of evidence. For forensics data acquisition, CSPs can shift their responsibility by providing a robust API or management console to acquire evidence. However, the CSPs need to come forward to resolve most of these issues. Creating a secure model for cloud forensics is very important as it will lead to more trustworthy clouds, allowing their adoption in sensitive application domains such as defense, business, and healthcare.

Acknowledgement:

This research was supported by a Google Faculty Research Award, the Office of Naval Research Grant #N000141210217, the Department of Homeland Security Grant #FA8750-12-2- 0254, and by the National Science Foundation under Grant #0937060 to the Computing Research Association for the CI Fellows Project.

REFERENCES

1. K. Kent, S. Chevalier, T. Grance, and H. Dang, "Guide to integrating forensic techniques into incident response," NIST Special Publication, pp. 800–86, 2006.
2. K. Ruan, J. Carthy, T. Kechadi, and M. Crosbie, "Cloud forensics: An overview," in proceedings of the 7th IFIP International Conference on Digital Forensics, 2011.
3. AWS, "Amazon web services," <http://aws.amazon.com>, [Accessed July 5th, 2012].
4. R. Marty, "Cloud application logging for forensics," in In proceedings of the 2011 ACM Symposium on Applied Computing. ACM, 2011, pp. 178–184.
5. T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds," in Proceedings of the 16th ACM conference on Computer and communications security. ACM, 2009, pp. 199–212.
6. D. Birk and C. Wegener, "Technical issues of forensic investigations in cloud computing environments," Systematic Approaches to Digital Forensic Engineering, 2011.
7. J. Vacca, Computer forensics: computer crime scene investigation. Delmar Thomson Learning, 2005, vol. 1.
8. S. Zawoad and R. Hasan, "Towards building proofs of past data possession in cloud forensics," ASE Science Journal, 2012.
9. R. Hasan, R. Sion, and M. Winslett, "Preventing history forgery with secure provenance," ACM Transactions on Storage (TOS), vol. 5, no. 4, p. 12, 2009.
10. Congress of the United States, "Sarbanes-Oxley Act," <http://thomas.loc.gov>, 2002, [Accessed July 5th, 2012].
11. Centers for Medicare and Medicaid Services, "The health insurance portability and accountability act of 1996 (hipaa)," <http://www.cms.hhs. gov/hipaa/>, 1996, [Accessed July 5th, 2012].

ABOUT THE AUTHORS

Shams Zawoad is working as a graduate research assistant in SECuRE and Trustworthy Computing Lab (SECRETLab) and a Ph.D. student at the University of Alabama at Birmingham (UAB). His research interest is in cloud security especially in cloud forensics, and in location provenance. He received his B.Sc. in Computer Science and Engineering from Bangladesh University of Engineering and Technology in 2008. Before joining UAB, Zawoad had been working in software industry and developed authentication and authorization framework for several critical business applications, including a online payment system of Bangladesh Post Office.

Phone: 205-915-4262
E-mail: zawoad@cis.uab.edu

Ragib Hasan, Ph.D. is a tenure-track Assistant Professor at the Department of Computer and Information Sciences at the University of Alabama at Birmingham. With a key focus on practical computer security problems, Hasan explores research on cloud security, mobile malware security, secure provenance, and database security. Hasan is the founder of the SECuRE and Trustworthy Computing Lab (SECRETLab) at UAB (http://secret.cis.uab.edu). He is also a member of the UAB Center for Information Assurance and Joint Forensics Research. Prior to joining UAB in Fall 2011, Hasan was an NSF/CRA Computing Innovation Fellow and Assistant Research Scientist at the Department of Computer Science, Johns Hopkins University. He received his Ph.D. and M.S. in Computer Science from the University of Illinois at Urbana Champaign in October, 2009, and December, 2005, respectively. Before that, he received a B.Sc. in Computer Science and Engineering and graduated summa cum laude from Bangladesh University of Engineering and Technology in 2003. He is a recipient of a 2013 Google RISE Award, a 2012 Google Faculty Research Award, the 2009 NSF Computing Innovation Fellowship and the 2003 Chancellor Award and Gold Medal from Bangladesh University of Engineering and Technology. Dr. Hasan's research is funded by the Department of Homeland Security, the Office of Naval Research, and Google. Hasan is also the founder of Shikkhok.com – a grassroots movement and platform for open content e-learning in South Asia.

Phone: 205-934-8643
E-mail: ragib@cis.uab.edu

User-Centric Identity Management
A Future Vision for IdM

Marc Novakouski, Carnegie Mellon Software Engineering Institute

Abstract. Identity management (IdM) is the complex and constantly evolving practice of identifying individuals and controlling their access to a network and connected resources. IdM research focuses primarily on making systems secure while the quality of the user experience is largely ignored. This article explores reasons why creating a user-centric IdM paradigm has become necessary, discusses existing efforts to make IdM more user centric, and presents one possible implementation of user-centric IdM that, in theory, could leverage mobile devices.

Introduction

Computer users are becoming increasingly aware of the dangers engendered by the Internet. Breaches of personal privacy and identity theft have created an overwhelming need for security. To address these challenges, researchers are engaged in the growing field of identity management (IdM), which involves strategies for identifying individuals in a network and controlling their access to its resources. To date, IdM research has largely focused on system security while often ignoring the quality of the user experience. As a result, IdM practices have made systems more secure but harder to use.

There is growing recognition of a need to address the imbalance between security and usability. For example, in April 2011 the White House gave NIST a mandate to partner with the private sector to make online transactions both easier and safer by establishing the Identity Ecosystem [1]. Collaborators envision this ecosystem as a user-centric environment that will support identity authentication in ways that are convenient for users. The goal is to provide developers with mechanisms to build systems that are both secure and user friendly. This article contributes to this effort by exploring what a user-centric IdM paradigm might look like, how it could be implemented, and the implications of that implementation.

IdM research provides significant value to the U.S. DoD. Establishing a viable identity ecosystem will encourage the commercial and industrial sectors to invest in improved identity management and security mechanisms. The DoD could then adopt these tools, techniques, and processes to improve the security of DoD identities and systems.

The Present State of IdM

One symptom of the heightened need for security is the growing number of passwords that users must keep track of. A few years ago, most users had only a handful of passwords to remember; a naïve user might have kept the same password for all systems. Today, casual computer users need a dozen passwords, and sophisticated users need several dozen. Some users might need to manage over 100 passwords [2].

However, computer users have bigger worries. Password theft plays a less significant role in identity theft than phishing and keylogging [3], and viruses, worms, malware, and other malicious software continue to increase [4]. Aspiring thieves who do not have the technical skills to perform attacks themselves can buy malware that others have created [4, 5]. And the Stuxnet worm has shown that cyber-attacks can be powerful enough to be used as weapons of international espionage or even war [6, 7]. Despite lack of expertise in security and IdM, users are often the first (and sometimes the only) line of defense against ever more dangerous forms of attack, such as:

- **Wifi hacking [8, 9]**
- **Compromised personal devices and infrastructure systems [10, 11]**
- **Social engineering [2, 3, 12]**
- **Cookie sniffing [13, 14]**
- **Timing attacks [15]**
- **Man-in-the-middle (MITM) attacks [16, 17]**
- **Insecure websites [2]**
- **Broken encryption attributed to GPU (graphics processing unit) [18] and quantum computing attacks [19]**

Most users find these problems too complex to manage [17]. As a result, many ignore security advice or engage in poor practices such as using simplistic passwords or writing passwords on pieces of paper near their computers [20, 21]. While the security community belittles users for these approaches [14], some security experts state that this behavior is not only predictable but also rational, given the overwhelming amount of security advice that users receive [3, 10, 22, 23].

Clearly, average users lack the knowledge and skills needed to manage their own security. To resolve this dilemma, a radical shift must take place in IdM research. New directions in IdM research must meet the challenge for improved security by addressing a growing number of threats while reducing security demands on the user. The user-centric IdM model proposed in this article provides a potential solution.

Changing the Game with User Centricity

Because many users lack sufficient knowledge to manage their own online identity, any viable IdM strategy has three goals:

1. **Improved Threat Resilience: Increase the capability of users to resist threats.**
2. **Improved Credential Management: Improve the capability of users to manage an arbitrary number of credentials.**
3. **Reduced User Load: Decrease the knowledge and effort required of users to resist threats.**

Unfortunately, these goals tend to be contradictory. The typical approach to improving the security of a system addresses threats individually, which tends to increase system complexity and restrictions on user access and require more skills and knowledge of the user to ensure safe behaviors. Moreover, each system addresses problems in different ways, which leads to unique IdM requirements for each system with which a user interacts. Thus, the goal of increasing threat resilience overrides the user-based goals of improving credential management and reducing user load.

As a result, users cope with increased complexity by relying on bad security practices [24], such as reusing passwords [25, 26], practicing bad password construction [25, 27, 28], and neglecting basic mobile device security practices such as setting an access PIN [29].

Contradictions among IdM goals can potentially be resolved through a user-centric IdM paradigm. Because there is no universal definition of what user-centric IdM entails, here is a simple definition:

A user-centric IdM system places the user priorities of improved credential management and reduced user load at the same importance as the system owner priority of improved threat resilience.

This means that improvements to threat resilience must not increase user requirements or degrade credential management capabilities. The IdM field will need novel and technologically advanced solutions to eliminate threats without making users' lives harder. For example, eliminating passwords and replacing them with rolling PINs combined with multifactor or multichannel authentication reduces both demands on the user and threats such as brute-force attacks, replay attacks, and MITM attacks. These methods would prevent many poor user practices because complex passwords would no longer be required for using networked services. Therefore, user-centric IdM paradigms with advanced security mechanisms could result in more secure systems that are also easier to use.

Despite the perceived advantages of a user-centric approach, for many reasons user-centric IdM systems have not become ubiquitous. The rest of this article examines some of these issues and identifies existing and potential solutions.

The User-Centric IdM Paradigm

To theorize about the user-centric IdM system of the future, it is important to examine existing work. This section also presents a vision of a user-centric IdM system and discusses the associated risks, benefits, shortcomings, and opportunities.

History

Many groups have tried to improve IdM in a way that embraces usability as a key driver along with security. Two of the most prominent developments are the Web single sign-on (SSO) solutions of OpenID and Microsoft InfoCard [24], which have provided an initial model for how a single login for all Internet services might work. The Kantara Initiative [30] and the Burton Group [31] have explored a wide range of technology and policy solutions. Similarly, identity experts such as Dick Hardt have contributed to the concept of "Identity 2.0" [32]. This set of principles and practices demonstrates how a next-generation IdM system could work and provides the foundation for many qualities of the proposed user-centric IdM system. Finally, developers of password management mechanisms such as Mozilla BrowserID [33] and manager programs such as Billeo have begun to instantiate basic implementations of user centralization [2, 24, 34]. Unfortunately, none of the commercial approaches have been widely adopted. The DoD Common Access Card (CAC) program represents the only wide-scale single-identity solution that can be considered successful. However, like most DoD programs, the tailoring to the military domain limits the applicability to more general domains.

The Vision: Building on Existing Models

There are several reasons why existing user-centric IdM efforts have failed to reach critical mass. However, the causes for failure are not intrinsic to the user-centric IdM paradigm and could be addressed in a way that creates a viable model for IdM while retaining the essential characteristics of user centricity. Therefore, the proposed vision of IdM's future builds on several fundamental characteristics of existing web-SSO systems and the Identity 2.0 paradigm. These characteristics include:

- **Centralization of all identity decisions to a single point or portal**
- **Ubiquity across all network-enabled services**
- **Elimination of multiple credentials**

These characteristics are necessary but not sufficient to meet the user IdM goals. Four additional characteristics are necessary for a refined vision of IdM:

- **Elimination of passwords as the primary credential**
- **Use of advanced security mechanisms such as digital certificates, rolling PINs, and multifactor and multichannel authorization**
- **Leverage of mobile devices for centralization**
- **Security managed by qualified security professionals, rather than by users**

Implementing a user-centric IdM system with these characteristics would increase both security and ease of use. First, threat resilience would improve through migration to non-password-based mechanisms and offloading of security maintenance to qualified professionals. Second, credential management would improve through the removal of passwords and the simplification to a single credential. Finally, user load would be reduced due to password elimination, single-credential centralization of access, and offloading the need for deep technical knowledge to trained security professionals.

While this vision would effectively meet both the user and the system goals of IdM, it must also address several additional issues for the proposed model to be viable.

Issue 1: Centralization

There is no question that centralization is a key risk to accepting and implementing the vision of user-centric IdM. Centralizing all identity decisions creates a single point of failure for accessing services, a single point of access for malicious users to steal credentials [35, 36], and a single point of vulnerability to innocent mistakes (e.g., misplacing a keychain) [29].

However, the benefits may now outweigh the risks. As previously discussed, the reality is that users are not qualified to handle their own security [3, 29]. Centralization makes it possible for users to leverage trained security professionals to keep their identity secure across all of the network services they use. This is the only proven way to manage the complex threat environment that now exists [34, 37]. By centralizing all identity decisions to a single point controlled by a single organization, a user's entire identity landscape can be protected in a consistent and comprehensive way. Centralization can also simplify the problem of credential management and help reduce security risks induced by poor user practices. It can also streamline

To drive sufficient demand, the proposed vision must meet user IdM goals with a high degree of success. However, providing a system that is sufficiently attractive to users to drive a critical mass of demand does not result in an economically viable organization. Infrastructure costs and salaries of the security professionals would be significant burdens to support.

recovery from compromise because the individual user maps to a single identity location that can be aggressively monitored, leading to improvements in compromise detection and notification. Centralization also reduces user load significantly.

The risks associated with the highly distributed IdM solutions of today that place so much responsibility on users have reached a tipping point. Without centralization, users must manage their own credentials. And as the number of credentials managed by the average user continues to increase, users will begin to look at the burden of credential management as a limiting factor to the services that they are willing to use.

Issue 2: Proper Security Mechanisms to Protect the Identity Store

The complexity of securing a centralized access point for identity is considerable, but the security community generally agrees that password schemes do not provide a sufficient level of security and must be phased out [3, 12, 22, 25, 38–41]. Existing mechanisms that can replace password schemes or augment non-password schemes include:

- **Certificates, such as those used for wireless access by the DoD CAC [42];**
- **Rolling PIN numbers, such as those used by RSA SecurID devices [43];**
- **Two-factor authentication [13, 44, 45];**
- **Automatic update devices [34];**
- **Modern encryption technologies [35];**
- **Disposable accounts and similar privacy approaches [2, 10, 46]; and**
- **Privacy-enhancing software [47–49]**

However, given the scope of the proposed vision and the complex mosaic of security technologies, these mechanisms are not sufficient to guarantee secure centralization. The greatest threat to any centralized identity store is that many systems used by many users could be compromised by a single failure. Therefore, security of the centralized data store will likely require the development of additional security mechanisms.

Issue 3: Organizational Viability

The proposed vision focuses on the idea of having central points of access for all user identity decisions. Combined with the idea that trained security professionals manage the security of this point or portal, it is clear that the vision requires creating one or more organizations that manage access for users to the network services they use. An obvious question involves how this differs from the web-SSO solutions discussed above, such as Open ID or InfoCard. The answer is that the proposed vision has the potential of widespread adoption, unlike existing web-SSO solutions. To understand why, it is necessary to examine the reasons that systems such as OpenID have not attained widespread adoption.

As the DoD CAC has demonstrated, strong incentives are required to enforce cross-organizational use of a single identity. For the CAC, the DoD mandate provided these incentives. In the commercial world, the only incentive with sufficient strength to make organizations work together is financial. Because web-SSO approaches such as OpenID and InfoCard are noncommercial, there is little to no incentive for service providers to integrate themselves into the existing web-SSO systems [50].

In fact, the only incentives for using the available web-SSO solutions are managerial in nature (a reduction in overhead of user management), but there are also strong disincentives for using these services. For example, with existing SSO solutions, service providers assume all of the liability for compromise even though they do not control login data; they lose access to user data, which has proven valuable for advertising purposes; and they must rely on external services over which they have no control. Therefore, it has been exceedingly difficult to establish business agreements between service providers and web-SSO providers because the incentives to do so are overwhelmingly negative [24, 31].

This means that to meet the goal of ubiquity across all networked services, the proposed vision of a user-centric IdM system would have to be implemented by a company that could establish strong incentives for integrating with their system. The only incentive proven to be sufficient to drive service providers to integrate with an external provider to manage identity is user demand [24]. So to drive sufficient demand, the proposed vision must meet user IdM goals with a high degree of success. However, providing a system that is sufficiently attractive to users to drive a critical mass of demand does not result in an economically viable organization. Infrastructure costs and salaries of the security professionals would be significant burdens to support.

Implementing User-Centric IdM

In this proposed vision of IdM's future, an organization, nominally called a personal identity provider (PIdP), could deliver services in a way that is consistent with the proposed user-centric model. The responsibilities of such a company would be to provide users with personalized control of their identity landscape, mediate access between the user and any service that the user wishes to access, ensure that the user's identity is secure, and prepare for and respond to compromise.

Delivering Identity Services

To deliver services in a way that satisfies the user's needs, the PIdP company of the future would have to meet several requirements. First, it would be staffed by security professionals qualified to manage the persistent threats to users and to internal systems. Second, the company would have close agreements with the majority of service providers, as well as competing PIdP companies, to ensure that the connections are reliable and secure. Third,

the company would have customer service and legal support that are sufficient to handle inevitable compromises. If the company met these requirements, it would provide secure identity services, offer access to a wide range of services, and accept the burden of responsibility for securing user data and privacy.

Using Mobile Devices

As many daily tasks become folded into the capabilities of modern smartphones—such as calendar management, email, and financial transactions—the lives of technology-enabled users are increasingly centered on their mobile devices. In this proposed vision of IdM's future, a smartphone or similar mobile device could host an application that would allow users to see, manage, and understand their personal identity landscapes. Such an application would offer a centralized, simplified IdM experience for users.

The mobile device also provides the necessary hardware platform to support improved security technologies, such as two-factor authentication, single-use passwords, and disposable accounts. While efforts to bring these technologies to mobile devices are not yet mature, prototypes are likely to be fielded within the next few years, especially considering the extensive efforts that the DoD has made to leverage smartphones for common use [42, 51–55] and even for processing classified data [56]. Work on the mobile CAC reader, in particular, demonstrates that a secure, non-password-based credential can reside either on or near a smartphone and provide user-centric authentication [42].

While such devices are not yet pervasive, the incredible growth of cell phones [57] bolsters the argument that generalized mobile device use is inevitable in the near future. It is therefore not unrealistic to expect that in the future an average user will have access to a mobile device that supports the proposed system.

Creating a Viable Economic Model

Presenting a viable business case for the type of organization proposed is beyond the scope of this paper. However, it is possible to speculate about how such an organization might operate. One way to think about the PIdP model is as a type of insurance company. Just as an insurance company does, the PIdP organization would accept the risk of compromise for its customers, provide a centralized service, guarantee it to be safe as long as it is used correctly, proactively manage security as new threats emerge, and receive a periodic fee for doing so.

To have sufficient control over user identity to make such an arrangement possible, the PIdP would implement the infrastructure that provides users with the centralized point of access. This would involve technology infrastructure for managing identities, interfaces (web and/or mobile), and agreements with service providers. The necessary capabilities and policies would likely emerge as user-centric PIdP services evolve.

This simplified, personalized, unified IdM experience—built on the necessary infrastructure and offered by a trusted company that is willing to take financial and legal responsibility for security compromise—would attract sufficient customers to create a viable business model. Just as people are willing to purchase health insurance because they cannot foresee and control health events, it is possible that a large number of users would pay to resist unforeseen online events and possible negative consequences to their privacy, credit, and finances. Many people are already paying for services that provide offline identity protection, such as LifeLock [58].

In addition to the standard insurance company approach of a monthly fee, other strategies could encourage adoption of the service until it becomes ubiquitous. One alternative is to allow users to explicitly sell personal information to the PIdP company—based on their login activity—in exchange for free service. Given the profit potential of tracking online activity for advertising purposes [59], this option could ensure business viability. If the PIdP is explicit about the tradeoff of free service for less privacy, the privacy concerns that users typically raise when service providers track their activity are unlikely to be a problem. As demonstrated by the lukewarm reception to Google Buzz [60] versus the much warmer response to Google+ [61], users are far more willing to share and expose personal data when they feel that they control the sharing.

Summary

Current IdM systems are unsustainable. People have dozens of accounts, each accessible through unique passwords of ever-increasing complexity, while security threats come from every direction. It is time to change the game. The emergence of personal IdM companies that adopt user-centric identity management models in an age of ubiquitous, generalized mobile devices could set the stage for such a revolution. The computer security industry should take this bold step forward and embrace a new paradigm of identity management. There is no question that changing the game will be difficult; however, there is also no question that the game we are currently playing has already been lost.

ABOUT THE AUTHOR

Marc Novakouski is a member of the technical staff at the Software Engineering Institute. He has over 10 years of experience in software engineering, ranging from software development in the commercial and defense fields to consulting and academic research on topics like migration of service-oriented architecture, identity management, and e-Government interoperability. He is a member of IEEE and is currently pursuing work on identity management, DoD software acquisition policy, and battlefield contextual awareness.

SEI
Carnegie Mellon University
Pittsburgh, USA
E-mail: novakom@sei.cmu.edu
Phone: 412-268-4274

REFERENCES

The full set of references can be found at <http://www.sei.cmu.edu/uls/References.cfm>.

1. National Strategy for Trusted Identities in Cyberspace. Washington, DC: National Institute of Standards and Technology, Apr. 2011. <http://www.whitehouse.gov/sites/default/files/rss_viewer/NSTICstrategy_041511.pdf>
2. Tynan, Dan. "Seven Lessons Learned from the Gawker and MacDonalds Hack Attacks: Email Addresses and Passwords for Millions of Gawker and McDonalds Fans Have Been Exposed–and Yours May Be Among Them." ITworld. 13 Dec. 2010. <http://www.itworld.com/internet/130546/seven-lessons-learned-gawker-and-mcdonalds-hack-attacks>
3. Herley, Cormac. "So Long, and No Thanks for the Externalities: The Rational Rejection of Security Advice by Users." Proceedings of the 2009 New Security Paradigms Workshop (NSPW '09). Ed. Anil Somayaji and Richard Ford. New York: ACM, 2009. 133-44. <http://doi.acm.org/10.1145/1719030.1719050>
4. Keefe, Mari. "A Short History of Hacks, Worms, and Cyberterror." ComputerWorld. 27 Apr. 2009. https://www.computerworld.com/s/article/9131924/A_short_history_of_hacks_worms_and_cyberterror
5. Constantin, Lucian. "World's Most Sophisticated Rootkit Is Being Overhauled." ITworld. 21 Oct. 2011. <http://www.itworld.com/security/215495/worlds-most-sophisticated-rootkit-being-overhauled>
6. Leyden, John. "Bastard Child of SpyEye/ZeuS Merger Appears Online." The Register. 25 Jan. 2011. <http://www.theregister.co.uk/2011/01/25/spyeye_zeus_merger>
7. Keizel, Gregg. "Is Stuxnet the 'Best' Malware Ever?" Infoworld. 16 Sept. 2010. <https://www.infoworld.com/print/137598>
8. SecurityWeek News. "Man Pleads Guilty for Hacking Neighbor's Wireless, Sending Threats against Vice President." SecurityWeek. 21 Dec. 2010. <http://www.securityweek.com/man-pleads-guilty-hacking-neighbors-wireless-sending-threats-against-vice-president>
9. Hogben, Giles, and Marnix Dekker. Smartphones: Information Security Risks, Opportunities and Recommendations for Users. Heraklion, Crete, Greece: European Network and Information Security Agency, Dec. 2010. <http://www.enisa.europa.eu/act/it/risks-and-data-breaches/smartphones-information-security-risks-opportunities-and-recommendations-for-users>
10. Smith, Peter. "The Case for Lousy Passwords." ITworld. 16 Dec. 2010. <http://www.itworld.com/personal-tech/131005/the-case-lousy-passwords>
11. Pauli, Darren. "AusCERT: Cisco IP Phones Prone to Hackers." SC Magazine. 12 May 2011. <http://www.scmagazine.com.au/News/257265,auscert-cisco-ip-phones-prone-to-hackers.aspx>
12. Kirk, Jeremy. "Will 2011 Be the Year of Mobile Malware?" Network World. 21 Dec. 2010. <http://www.networkworld.com/news/2010/122110-will-2011-be-the-year.html?hpg1=bn>
13. Garfinkel, Simson. "Facebook Wants to Supply Your Internet Driver's License." Technology Today. 5 Jan. 2011. <http://www.technologyreview.com/web/27027/?p1=A1&a=f>
14. LosHuertos, Gary. "Herding Firesheep in New York City." Technology Sufficiently Advanced. 27 Oct. 2010. <http://technologysufficientlyadvanced.blogspot.com/2010/10/herding-firesheep-in-new-york-city.html>
15. McMillan, Robert. "Researchers: Authentication Crack Could Affect Millions." ComputerWorld. 15 July 2010. <http://www.computerworld.com/s/article/9179224/Researchers_Authentication_crack_could_affect_millions>
16. Naone, Erica. "Car Theft by Antenna." Technology Review. 6 Jan. 2011. <http://www.technologyreview.com/computing/27037/?p1=A1&a=f>
17. Gedda, Rodney. "Days of Individual Security Over, Says IIA Chief." CIO. 29 Mar.2011. <http://www.cio.com.au/article/381359/days_individual_security_over_says_aiia_chief>
18. Kingsley-Hughes, Adrian. "Cheap GPUs Are Rendering Strong Passwords Useless." ZDNet. 1 June 2011. <http://www.zdnet.com/blog/hardware/cheap-gpus-are-rendering-strong-passwords-useless/13125>
19. Wood, Lamont. "The Clock Is Ticking on Encryption. Today's Secure Cipher-Text May Be Tomorrow's Open Book." ComputerWorld. 17 Dec. 2010. <http://www.computerworld.com/s/article/9201281/The_clock_is_ticking_on_encryption>
20. Florencio, D., and C. Herley. "A Large-Scale Study of Web Password Habits." WWW '07: Proceedings of the 16th International Conference on the World Wide Web. New York: ACM, 2007. 657-66.
21. Gaw, S., and E. W. Felten. "Password Management Strategies for Online Accounts." Proceedings of the Second Symposium on Usable Privacy and Security. New York: ACM, 2006. 44-55.
22. Stross, Randall. "A Strong Password Isn't the Strongest Security." New York Times. 4 Sept. 2010. <https://www.nytimes.com/2010/09/05/business/05digi.html?_r=1>
23. Mirick, James R. "The User Cost of Internet Security." My Take on Everything. 2 Feb. 2010. <https://jamesmirick.wordpress.com/2010/02/02/the-user-cost-of-internet-security>
24. Sun, San-Tsai, Yazan Boshmaf, Kirstie Hawkey, and Konstantin Beznosov. "A Billion Keys, but Few Locks: The Crisis of Web Single Sign-on." Proceedings of the 2010 Workshop on New Security Paradigms (NSPW '10). Ed. Angelos Keromytis and Sean Peisert. New York: ACM, 2010. 61-72. <http://doi.acm.org/10.1145/1900546.1900556>
25. Help Net Security. "Passwords Are the Weakest Link in Online Security." Help Net Security. 22 Dec. 2010. <http://www.net-security.org/secworld.php?id=10353>
26. SecurityWeek News. "Study Reveals 75% of Individuals Use Same Password for Social Networking and Email." Security Week. 16 Aug. 2010. <http://www.securityweek.com/study-reveals-75-percent-individuals-use-same-password-social-networking-and-email>
27. Help Net Security. "Analysis of 32 Million Breached Passwords." Help Net Security. 21 Jan. 2010. <http://www.net-security.org/secworld.php?id=8742>
28. National Institute of Standards and Technology. The National Strategy for Trusted Identities in Cyberspace: Why We Need It. Gaithersburg, MD: NIST, 2011. <http://www.nist.gov/nstic/NSTIC-Why-We-Need-It.pdf>
29. Sousa, João Pedro. "Challenges and Architectural Approaches for Authenticating Mobile Users." Proceedings of the 1st International Workshop on Software Architectures and Mobility (SAM '08). Ed. Rami Bahsoon, Licia Capra, Wolfgang Emmerich, and Mohamed E. Fayad. New York: ACM, 2008. 15-20. <http://dl.acm.org/citation.cfm?id=1370893>
30. Wikipedia. "Kantara Initiative." Wikipedia. 1 Aug. 2011. <http://en.wikipedia.org/wiki/Identity_Assurance_Framework>
31. Fontana, John. "Evolution of Federation." Ping Identity. 2 Aug. 2010. <http://www.pingidentity.com/blogs/pingtalk/index.cfm/2010/8/2/Evolution-of-federation>
32. Mike Nowak. "One Login to Bind Them All." Wired. 1 Aug. 2005. <http://www.wired.com/politics/security/news/2005/08/68329>
33. Mozilla. "Introducing BrowserID: A Better Way to Sign In." Identity at Mozilla. 14 Aug. 2011. <http://identity.mozilla.com/post/7616727542/introducing-browserid-a-better-way-to-sign-in>
34. Grossman, Jeremiah. "Lessons Learned from the Gawker Hack." Threatpost. 15 Dec. 2010. <http://threatpost.com/en_us/blogs/lessons-learned-gawker-hack-121510>
35. Beckman, Mel. "Your Laptop Data Is Not Safe. So Fix It." InfoWorld. 19 Jan. 2009. <http://www.infoworld.com/d/security-central/your-laptop-data-not-safe-so-fix-it-553>
36. Badger, Emily. "The Government Internet ID Proposal's Pros and Cons." Miller-McCune. 19 Apr. 2011. <http://www.miller-mccune.com/politics/the-government-internet-id-proposals-pros-and-cons-30448>
37. Evans, Karen, and Franklin Reeder. A Human Capital Crisis in Cybersecurity. Washington, DC: Center for Strategic and International Studies, 2010. <http://csis.org/publication/prepublication-a-human-capital-crisis-in-cybersecurity>
38. Teeghman, David. "Online Passwords Could Be a Map." Discovery News. 28 Sept. 2010. <http://news.discovery.com/tech/online-passwords-could-be-a-map.html>
39. Marlinspike, Moxie. "SSL and the Future of Authenticity." Thoughtcrime Labs. 11 Apr. 2011. <http://blog.thoughtcrime.org/ssl-and-the-future-of-authenticity>
40. Goss, Grant. "White House Releases Trusted Internet ID Plan." ComputerWorld. 16 Apr. 2011. <http://www.computerworld.com.au/article/383473/white_house_releases_trusted_internet_id_plan>
41. Johansen, Tor Anders, Ivar Jørstad, and Do van Thanh. "Identity Management in Mobile Ubiquitous Environments." Proceedings of the 3rd International Conference on Internet Monitoring and Protection (ICIMP '08). Ed. Seppo Heikkinen, Ivar Jørstad, and Nicolae Tapus. Piscataway, NJ: IEEE Computer Society Press, 2008. 178-83. <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4561345>

REFERENCES

42. Hernon, Mike. "Going Mobile: BlackBerries and CACs Get Connected." Business Library. Oct.-Dec. 2007. <http://findarticles.com/p/articles/mi_m0OBA/is_4_25/ai_n21130433/
43. RSA. "Securing Your Future with Two-Factor Authentication." RSA Secure ID. 2012 <http://www.emc.com/security/rsa-securid.htm>
44. Donohue, Brian. "Internal Memo Outlines Gawker's Security Plan." Threatpost. 20 Dec. 2010. <https://threatpost.com/en_us/blogs/internal-memo-outlines-gawker-s-security-plan-122010>
45. Perez, Juan Carlos. "Facebook Tightens Log-In Verification." ComputerWorld. 13 May 2011. <http://www.computerworld.com.au/article/386405/facebook_tightens_log-in_verification>
46. Romenesko, Jim. "Memo: Gawker Tech Team Didn't Adequately Secure Our Platform." Poynter. 17 Dec. 2010. <http://www.poynter.org/latest-news/mediawire/111549/gawker-tech-team-didnt-adequately-secure-our-platform>
47. McMillan, Robert. "EFF: Forget Cookies, Your Browser Has Fingertips." Network World. 18 May 2010. <http://www.networkworld.com/news/2010/051810-eff-forget-cookies-your-browser.html>
48. Wikipedia. "Tor (Anonymity Network)." Wikipedia. 18 Jan. 2012. <https://secure.wikimedia.org/wikipedia/en/wiki/Tor_(anonymity_network)>
49. Murdoch, Steven J., and G. Danezis. "Low-Cost Traffic Analysis of Tor." Proceedings of the 2005 IEEE Symposium on Security and Privacy. Piscataway, NJ: IEEE Computer Society Press, 2005. 183-95. <http://www.cl.cam.ac.uk/~sjm217/papers/oakland05torta.pdf>
50. Anderson, Ross. "Can We Fix the Security Economics of Federated Authentication?" Workshop on the Economics of Information Security. Ed. James A. Malcolm. New York: Springer, 2011. 28-30. <http://www.mendeley.com/research/we-fix-security-economics-federated-authentication>
51. Lister, John. "US Army Smartphone War Draws Closer." BLORGE.com. 14 Dec. 2010. <http://mobile.blorge.com/2010/12/14/us-army-smartphone-war-draws-closer>
52. Lister, John. "Military Working on Smartphone Encryption." BLORGE.com. 14 Apr. 2011. <http://mobile.blorge.com/2011/04/14/military-working-on-smartphone-encryption>
53. Oltsik, Jon. "Apple and Google Make the Department of Defense Jump Through Hoops for Mobile Device Security." Network World. 9 Dec. 2010. <http://www.networkworld.com/community/blog/apple-and-google-make-department-defense-jump>
54. Bray, Tim. "Phones and Soldiers." Ongoing. 11 Dec. 2010. <http://www.tbray.org/ongoing/When/201x/2010/12/11/DoD-and-Mobile-Phones>
55. Strategy Page. "Bringing Internet Access to the Battle." StrategyWorld. 3 Jan. 2011. <http://www.strategypage.com/htmw/htiw/20110103.aspx>
56. LifeLock. "Frequently Asked Questions." 24 Jan. 2012. <http://www.lifelock.com/how-it-works/faq>
57. Kenyon, Henry. "Secure Android Kernel Could Make for 'Classified' Smart Phones." Government Computer News. 13 Oct. 2011. <http://gcn.com/Articles/2011/10/11/AUSA-secure-andriod-kernel-technology.aspx#>
58. CBS News. "Number of Cell Phones Worldwide Hits 4.6B." CBS Money Watch. 18 Feb. 2010. <http://www.cbsnews.com/stories/2010/02/15/business/main6209772.shtml>
59. Google. "Google Adwords Help." Google. 2012. <https://adwords.google.com/support/aw/bin/static.py?hl=en&topic=21905&guide=21899&page=guide.cs&answer=146309>
60. Canadian Broadcasting Corporation. "Privacy Commissioner Reviewing Google Buzz." CBC News. 16 Feb. 2010. <http://www.cbc.ca/technology/story/2010/02/16/google-buzz-privacy.html>
61. Audrey Watters. "Will Google+ Replace Twitter or Facebook for Teachers?" Mindshift. 11 July 2011. <http://mindshift.kqed.org/2011/07/will-google-replace-twitter-or-facebook-for-teachers/#more-13562>

Building a Resilient Service-Oriented Architecture Environment

Quyen L. Nguyen, International Cyber Center and Department of Computer Science
Arun Sood, George Mason University and SCIT Labs

Abstract. In a Service-Oriented Architecture (SOA) system, services contain operations with openly defined input and output parameters. While satisfying functional requirements, a service also exposes its attack surface via published operations, open protocols, and accessible data as an adverse side effect, which makes it susceptible to exploitation by malicious actors. With this context, it is a challenge to build an SOA environment such that it is resilient in the face of hostile attacks. In this paper, we propose an approach to design services such that their attackability can be controlled and intrusion tolerance guaranteed despite the exposed attack surface. Our approach relies on Self-Cleansing Intrusion Tolerance (SCIT), a recovery-based intrusion tolerance architecture combined with service-oriented programming constructs.

1. Introduction

SOA is based on loosely coupled services. Services such as infrastructure and common services are atomic. Other services are called composite services because they are composed of atomic services or even other composite services. A service is designed with clearly defined functionalities that other services and applications can use. Along with the functional requirements, a service must satisfy non-functional requirements, among which security quality is a crucial one. Indeed, due to the inherent distributed nature of services communicating with each other via networks and open protocols, security quality is critical in SOA in order to ensure availability, integrity, and confidentiality of services and thus make them usable. On the other hand, since security attacks have become more and more sophisticated, a system cannot rely solely on intrusion prevention and detection for its security protection. Therefore, intrusion tolerance systems should be part of the solution for securing computer information systems.

The challenge of making a service resilient is that it is bound to expose resources via open interfaces as a side effect while fulfilling functional and business requirements. Each resource is likely to have vulnerabilities, and these together constitute the service's attack surface, which includes operations, data items accessible to read and/or write, and communication channels over which services operate [1]. The more operations and data a service provides, the more its attack surface increases. Thus, limiting the attack surface does not always work. Moreover, if COTS is used to realize a service, then the service's attack surface is pretty much predefined, leaving little room for reducing its exposure via configuration.

In this paper, we will present an approach to build a resilient SOA environment, which has four characteristics: a) to withstand malicious attacks; b) to rapidly recover from compromises; c) to provide service continuity even in the case of attacks; and d) to adapt to changing operational environment. The characteristics a) and b) can be quantitatively represented by the QoS parameters of Mean Time to Security Failure (MTTSF), and Mean Time to Repair (MTTR) respectively. Our approach consists of compensating the undesired expansion of a service's attack surface due to enhancements by utilizing Self-Cleansing Intrusion Tolerance (SCIT) and restricting the exposure time [2]. Characteristic c) of service continuity can be mitigated by having redundant live nodes, with the pair of primary/backup and diversity in the SCIT environment. We will also show how SCIT allows services to adapt to changing environment while maintaining their promised resilience. Based on timed-recovery mechanisms provided by SCIT, software architects are offered additional tools to design a service so that its attackability can be controlled, and intrusion tolerance QoS guaranteed despite the exposed attack surface. Moreover, our approach leverages service composition constructs that allow fronting an important service with another service well controlled by SCIT to reinforce the resilience of the service to be protected.

2. System Architecture

SCIT Mechanism

The goal of SCIT is to make applications and services resilient in the face of malicious attacks. SCIT's recovery-based mechanism consists of automatic and periodic cleansing of the servers running on a virtualization layer, which allows the instantiation of multiple servers with possible options of having different guest operating systems on a single host machine. Moreover, the utilization of virtual machines enables rapid reloading and reactivation of the servers. SCIT's pattern operates on two major components as depicted in Figure 1:

a. Central Controller managing and controlling all the nodes to be protected. Diversity of these nodes can be employed to further reduce the likelihood of malicious exploitations. For example, we can have a group of web servers, one running on Linux, another on Windows, and a third one on Mac OS.

b. Cluster of nodes providing the same applications and services. Managed by the Central Controller, each node continuously goes through the following state sequence:

- Live Spare state, in which the node is pristine but offline;
- Active state for the duration W_o, (called exposure window), where the node is online to serve incoming requests;
- Grace Period state with pre-configured duration, where the node stops accepting new transaction requests, but completes processing of already queued requests;
- Cleansing state, where the node is offline and undergoes the full restoration to a known pristine state.

Given that the cleansing time depends on the specific service, we have obtained the number of redundant nodes in a cluster to perform the rotation cycle of duration W_o [2]. Figure 1 shows that Node 1 is in Active state, Node 2 in Live Spare state, and Node n in Cleansing state.

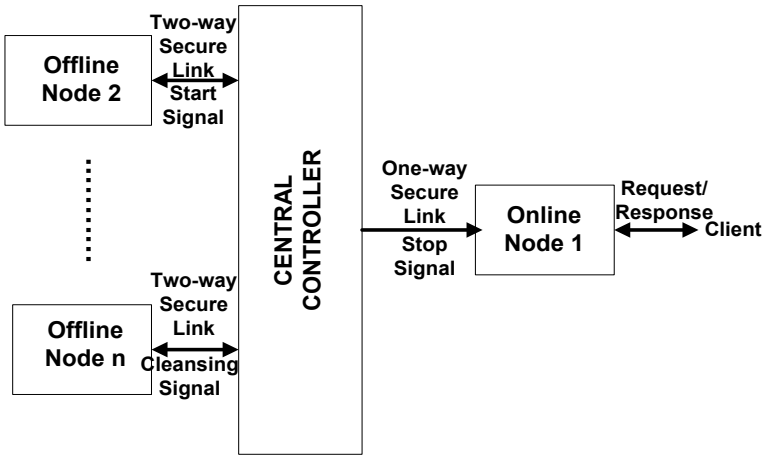


Figure 1. SCIT Architecture Components.

In terms of deployment, it is advisable to host the Controller physically separated from the nodes in order to avoid a data path between the nodes and the Controller, thus eliminating potential malware propagation to the Controller. Furthermore, in the current SCIT pattern, the link from the Controller to the online node is unidirectional.

The SCIT product has been developed by SCIT Labs under the direction of Dr. Sood. Experimentation was performed on the implementation of SCIT and reported in [3], where results demonstrated that the performance and computing resource usage impacted by SCIT is minimal. Recently, SCIT Labs was selected for a Small Business Innovation Research award that focuses on Scalable Moving Target Defense based on SCIT technology. Moreover, they plan to perform more tests as to the application of SCIT in various domains—defense, civil, and commercial. Such tests would assess the performance and the security of the implementation.

Cleansing Mode

The main functionality of SCIT is node cleansing, which consists of bringing a node back to a pristine state based on a virtual image stored and maintained in a safe place. The virtual image of a service can be created on an offline machine to ensure its unreachability by potential hackers. There are two cleansing modes to accommodate two types of services:

Full cleansing mode is used for services with transactional operations. Usually, those transactions are short requests so that they can be implemented as stateless RESTFul web services. In this basic cleansing mode, the nodes will undergo full erasure to make room for the clean image. This image is static because it does not contain any in-flight data. In fact, since the service operations are stateless, there is no need to capture state information of the node before performing the cleansing.

Partial cleansing mode is used for services with long running operations. Scientific computation falls into this category. Since state information may exist when a node enters the cleansing phase, using full cleansing mode may unintentionally cause loss of process state information, hence corrupting the on-going computation. This partial cleansing mode includes the following steps:

- Step 1. Capture the state of the service running in the node running the virtual machine, and create a snapshot dynamic image. The running state of the service is comprised of resources such as memory, file descriptors, buffer content, program counter, stack pointers, registers, etc. used by the service in the virtual machine. There has been research done in this area. Aroma [4] is a Java compatible VM allowing the capture of state and threads. "Process Introspection" was proposed by Ferrari to capture process state and perform its recovery [5].
- Step 2. Perform cleansing.
- Step 3. Migrate the snapshot containing the service's state to the node scheduled to be turned online.

3. Resilient Service Design

In this section, we will analyze the relationship between SCIT exposure window and the resilience of services expressed by the QoS parameters MTTSF (Mean Time To Security Failure) and MTTR (Mean Time To Repair), via the methodology presented in [6]. The analysis utilizes Semi-Markov Chain whose states capture the behaviors of both the attacker and the service being studied. As a result of the established correlation, designing a service's resilience is tantamount to computing the exposure window and the number of diverse and redundant services. To confirm this theoretically proven mechanism of improving resilience QoS parameters of MTTSF and MTTR, testing will be performed in addition to the experiments reported in [3].

Atomic Service

First, we start with an atomic service, i.e. one that does not need to depend on other services in the SOA framework. Figure 2 shows that service S protected by SCIT undergoes three states: Good (G), Attacked (A), and Failure (F). An atomic service starts with state G. In the case where the service is attacked by some malicious actor, it transitions to state A with attack probability P_A. Service S enters state F when it is compromised. Thanks to the SCIT periodic cleansing and recovery scheme, service S can recover to good state G from state A, with probability P_C. Similarly, service S can get out of state F to go to state G because of SCIT forced cleansing.

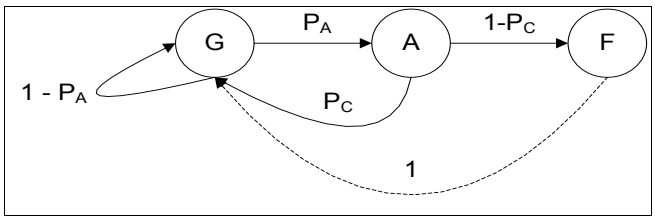


Figure 2. SCIT Atomic Service Transition Diagram.

P_A is factored by the service's attack surface and attack arrival. In [2], we have proved that the attack arrival and P_C can be controlled by exposure window W. Since MTTSF and MTTR are conditioned by P_A and P_C, it can be shown that MTTSF increases and MTTR decreases when W decreases.

Orchestration with Dependent Services

The notion of attack target and enabler is described in [7]; malicious intruders can exploit the enabler's attack surface to get access to and compromise the aimed target. With the SOA paradigm, we can have a service orchestration with enabler services and a target service. For example, let us consider the

case of an archive application that allows a public user to search for digital records preserved in the archive. The application is implemented as a composition of User Input Service, which validates all user input criteria in the search requests, and the Search Service performing the actual search. The Search Service can be based on a Free Open Source Software or COTS, such as Apache Solr, SeekQuarry, Autonomy, Vivisimo, Google Search Appliance, FAST, etc., whose attack surfaces have been pre-determined at release time. The key characteristic of the composite service with enabler as depicted by Figure 3 is that an attacker must compromise the enabler service S1 first before she can break into the target service S.

States G, A, and F of enabler service S_1 are the same as for an atomic service. The transition from state F to FA expresses the notion of enabler and target mentioned above. State FF happens when both services are compromised. Our discussion considers only one enabler service, but can be extended to the case where there are multiple enablers in the service orchestration.

Since the composite service can transition from F to G thanks to cleansing, we can make the following (see Equation 1):

$$P_A = 1 - P_{C1}.$$

Equation 1:

The significance of this equation is quite interesting. Indeed, it reveals that probability of attack to the target can be controlled by means of the cleansing probability P_{C1} of the enabler service S_1. Overcoming S's attack surface, hence improving its resilience can be realized by increasing P_{C1}.

Orchestration with Independent Services

Services S_1 and S_2 are independent in terms of attackability, as shown by the paths $G_1 \rightarrow G_2 \rightarrow A_2$ and $F_1 \rightarrow G_2$ (Figure 4). The same would apply for parallel or conditional branches in a service orchestration. Consequently, in order to analyze these cases, we can apply the results for an atomic service to the services S_1 and S_2 separately.

Diversity

In the literature, diversity has been proposed as one of the mechanisms to increase the resilience of services. Let $\{s_i \mid i=1,..,N\}$ be the set of N different versions of the same service S, which can be obtained by using different operating systems, middleware packages, or implementations as proposed by Huang [8]. Assuming that a line of attack only works for a specific version of a service S, the attack probability P_S of the surface of S is bounded by the following (see Equation 2):

$$P_S \leq \frac{1}{N}\{\max_{i=1,..,N} P_{s_i}\},$$

Equation 2:

In Equation 2, the values P_{Si} are the attack probabilities of the different version of the same service S. This expression shows that the probability of attack on the service's surface will decrease if more versions are used.

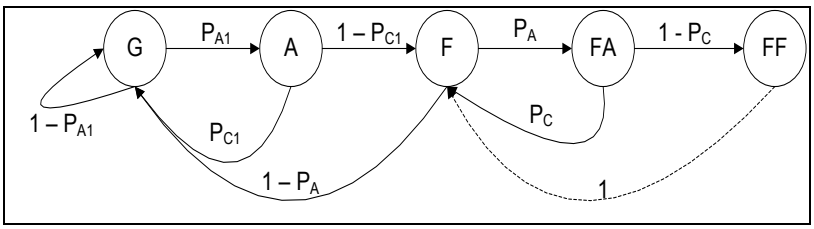


Figure 3. State Transition of Dependent Services.

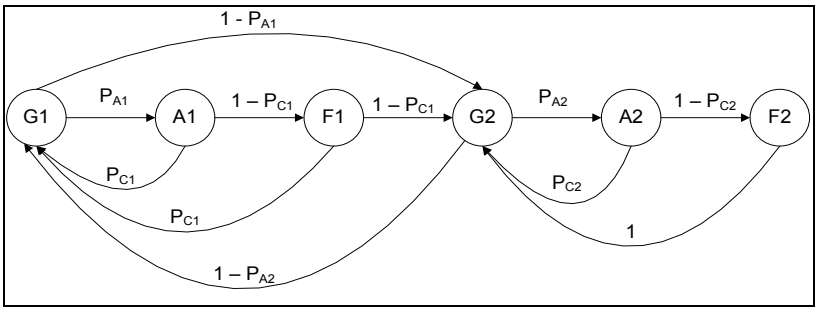


Figure 4. State Transition of Independent Services.

4. SCIT and SOA

Service Provider Architecture

For a single service provider, the SCIT architecture depicted in Figure 1 is enhanced with two additional components (Figure 5) as described in [9]:

- The Service Registry contains metadata about the services in the system including service operations, access points, and Intrusion Tolerance QoS (IT-QoS) characteristics. The notion of IT-QoS was introduced in [9] to capture QoS attributes that are related to an intrusion tolerant system, such as MTTSF, S-Reliability, MTTR, maximum intrusion number, etc. For scientific computation, Computational QoS needs to be considered [10]. The Central Controller queries the Service Registry about desired IT-QoS values for a service, and also updates the current operational values.
- The Monitor provides the Central Controller with information about the current operating environment based on analytics of logs, or reports provided by sensors such as IDS sensors.

In the SOA adaptation, a "node" in the SCIT architecture pattern described above becomes a Service Container, which can be implemented by an application server where services are deployed and activated. A Service Container will host services requiring the same level L of IT-QoS, which is tied to a value for the exposure window. In order to apply SCIT periodic cleansing, the system needs N replicas of similar service containers, i.e. containers with services of same IT-QoS level L. These N replicas form a Container Group associated with a specific level L.

A system providing differentiated IT-QoS levels will have multiple Container Groups. Figure 2 exhibits an example of a system with 4 services S1, S2, S3 and S4. There are only two levels of IT-QoS denoted by level 1 and level K. Due to the low level of level 1, Group 1 only needs 2 replicas of Service Containers, namely Containers 11 and 12. For the higher level K, 3 Service Containers are configured: Containers K1, K2 and K3. Note that services S1 and S3 are contained in all containers, since they operate at both levels 1 and K. But, S2 instances are deployed only in containers of level 1, while S4 instances only in containers of level K.

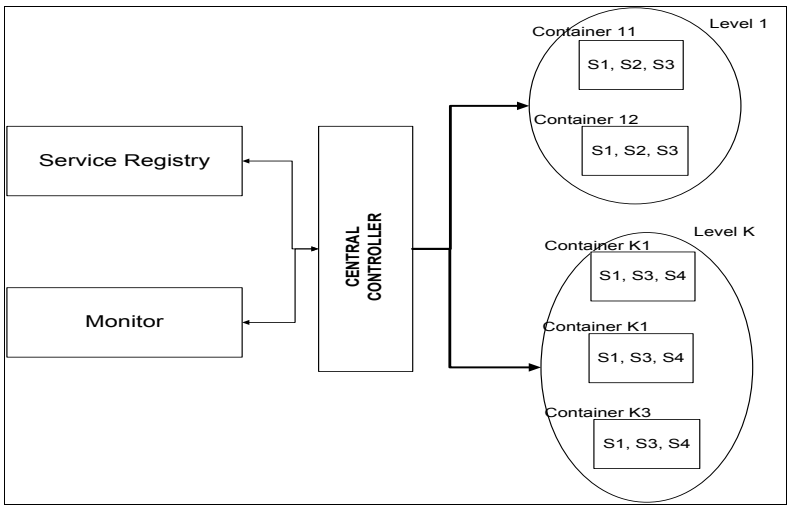


Figure 5. SCIT for a Single Service Provider's Services.

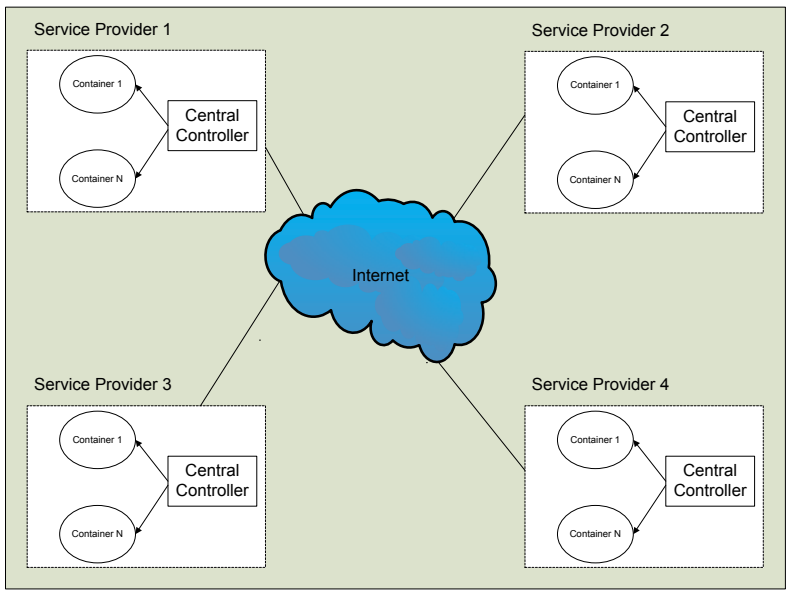


Figure 6. Resilient SOA Ecosystem.

Resilient SOA Ecosystem

In SOA, services can be provided by various providers. For instance, an application can utilize storage service from Amazon S3, and incorporate Google map service. If we can apply the above SCIT resilient architecture to the service providers involved in building an SOA application, then we can create a resilient SOA ecosystem. Figure 6 shows four service providers whose environments have implemented SCIT. Each service provider will have its own Central Controller to manage the cleansing of its service containers.

5. Conclusion

In this paper, we have shown that the SCIT architecture of Controller, Service Container, and cleansing algorithm based on exposure window contributes to build a resilient SOA environment. This is achieved by increasing each service's resilience in the SOA ecosystem by compensating the attack surface with a smaller exposure window, reducing the exposure window of Enabler Services in a service orchestration, or reducing the effective attack surface of a service with multiple diverse configurations of that service. Thanks to the simple parameters of exposure window and diversity number, software architects can specify resilience quality quantitatively during the service design.

REFERENCES

1. Pratyusa K. Manadhata and Jeannette M. Wing. "An Attack Surface Metric". IEEE Transactions on Software Engineering, May-June .2011.<http://testlab.sit.fraunhofer.de/downloads/Publications/heumann-quantifying_the_attack_surface_of_a_web_application-GI_Sicherheit_2010.pdf>
2. Quyen Nguyen and Arun Sood. "Quantitative Approach to Tuning of a Time-Based Intrusion-Tolerant System Architecture". WRAITS 2009, Lisbon, Portugal. <http://wraits09.di.fc.ul.pt/wraits09paper2.pdf>.
3. Anantha Bangalore and Arun Sood. "Securing Web Servers using Self Cleansing Intrusion Tolerance (SCIT)". Proceedings of Second International Conference on Dependability (DEPEND 2009), Athens, Greece. June 18-23, 2009.
4. Niranjan Suri , Jeffrey M. Bradshaw , Maggie R. Breedy , Kenneth M. Ford , Paul T. Groth , Gregory A. Hill , Raul Saavedra. "State capture and resource control for java: The design and implementation of the aroma virtual machine". In Proceedings of the Java Virtual Machine Research and Technology Symposium, 2001.
5. Adam Ferrari, Steve J. Chapin and Andrew Grimshaw. "Heterogeneous process state capture and recovery through Process Introspection". Journal of Cluster Computing, Volume 3, Issue 2, 2000.
6. Bharat B. Madan, Katerina Goseva-Popstojanova, Kalyanaraman Vaidyanathan, and Kishor S. Trivedi. "A Method for Modeling and Quantifying the Security Attributes of Intrusion Tolerant Systems". Dependable systems and networks-performance and dependability symposium (DSN-PDS) 2002.
7. Michael Howard, Jon Pincus, and Jeannette M. Wing. "Measuring Relative Attack Surfaces". <http://www.cs.cmu.edu/~wing/publications/Howard-Wing03.pdf>.
8. Yih Huang and Anup K. Ghosh. "Introducing Diversity and Uncertainty to Create Moving Attack Surfaces for Web Services". Advances in Information Security, 1, Volume 54, Moving Target Defense, Pages 131-151.
9. Quyen Nguyen and Arun Sood. "Realizing S-Reliability for Services via Recovery-driven Intrusion Tolerance Mechanism". 2010 International Conference on Dependable Systems and Networks Workshops (DSN-W), Chicago, Illinois. Jun 28-Jul 1, 2010.
10. Boyana Norris , Jaideep Ray , Rob Armstrong , Lois C. Mcinnes , Sameer Shende. "Computational Quality of Service for Scientific Components". Proceedings of the International Symposium on Component-based Software Engineering (CBSE7).

ABOUT THE AUTHORS

Quyen L. Nguyen is a system architect in the Systems Engineering Division of the Electronic Records Archives Program Management Office at the US National Archives and Records Administration. His research interests include system architecture, software modeling, and security architecture. Nguyen has a BS in computer and information sciences from the University of Delaware, an MS in computer science from the University of California at Berkeley, and is a computer science PhD candidate at George Mason University.

E-mail: qlnguyen@yahoo.com

Dr. Arun K. Sood is Professor of Computer Science in the Department of Computer Science, and Director of the International Cyber Center (ICC) at George Mason University, Fairfax, VA. His research interests are in security architectures; image and multimedia computing; performance modeling and evaluation; simulation, modeling, and optimization.

He and his team of faculty and students have developed a new approach to server security, called Self Cleansing Intrusion Tolerance (SCIT). We convert static servers into dynamic servers and reduce the exposure of the servers, while maintaining uninterrupted service. This research has been supported by US Army, NIST through the Critical Infrastructure Program, SUN, Lockheed Martin, Commonwealth of Virginia CTRF (in partnership with Northrop Grumman).

Recently SCIT technology was winner of the Global Security Challenge (GSC) sponsored Securities Technologies for Tomorrow Challenge. This technology has been awarded 3 patents and 3 additional patents are pending. SCIT Labs, a university start up, has been formed to commercialize SCIT technology – Dr. Sood is the founder and CEO of SCIT Labs.

Since 2009 Dr. Sood has directed an annual workshop on Cyber Security and Global Affairs with Office of Naval Research support – Oxford 2009, Zurich 2010, Budapest 2011 and Barcelona 2012.

Dr. Sood has held academic positions at Wayne State University, Detroit, MI, Louisiana State University, Baton Rouge, and IIT, Delhi. His has been supported by the Office of Naval Research, NIMA (now NGA), National Science Foundation, U.S. Army Belvoir RD&E Center, U. S. Army TACOM, U.S. Department of Transportation, and private industry.

He was awarded grants by NATO to organize and direct advance study institutes in relational database machine architecture and active perception and robot vision.

Dr. Sood received the B.Tech degree from the Indian Institute of Technology (IIT), Delhi, in 1966, and the M.S. and Ph.D. degrees in Electrical Engineering from Carnegie Mellon University, Pittsburgh, PA, in 1967 and 1971, respectively.

His research has resulted in more than 170 publications, two edited books, 8 patents, and his resume including publications list is available at http://cs.gmu.edu/~asood.

Professor, Computer Science & Director, International Cyber Center
George Mason University
MS 4A5, 4400 University Drive, Fairfax, VA 22030
E-mail: asood@gmu.edu

Founder and CEO SCIT Labs Inc
13834 Springstone Dr., Clifton, VA 20124
E-mail: asood@scitlabs.com

Applying Software Assurance Concepts to the Cloud

Randall Brooks, Raytheon
John Whited, Raytheon

Abstract. It was once said that the last time one had full control of their software was right before they released it. This is ever more important as organizations move applications and services into a public cloud to support a mobile lifestyle. Clouds have been described as "a safe and secure private cloud," "a semi-trusted partner cloud," or "a Wild West full and open public cloud." It is typically toward the latter in which the industry has been moving. Because of this, developers must understand their attack surface and threat environment to ensure that they have focused on "building security into" their applications.

Software Assurance

Software Assurance (SwA) is defined by the National Information Assurance Glossary, Committee on National Security Systems Instruction (CNSSI) No. 4009, as "the level of confidence that software is free from vulnerabilities, either intentionally designed into the software or accidentally inserted at anytime during its lifecycle, and that the software functions in the intended manner." SwA focuses on "building security in" to ensure the software is resilient in harsh operating environments such as those in which cloud applications operate. According to the DHS, SwA addresses:

- **Trustworthiness**
- **Predictable Execution**
- **Conformance**

In a cloud environment, Trustworthiness is key to providing confidence to the user that no exploitable vulnerabilities exist. With use of mobile and the cloud, users will demand Predictable Execution. Cloud applications that function as intended will avoid user frustration. Cloud applications must conform to appropriate secure APIs to ensure interoperability with other applications and services.

To deliver SwA, various principles can be applied such as determining the criticality of the application, defining the Attack Surface, developing misuse cases, and testing for unintended consequences. But these principles have a slightly different emphasis when applied to the cloud.

Determine Component Risk and Criticality

As with other complex problem spaces, understanding Attack Surface, the threat environment, and related elements of risk management in the cloud is best addressed by first decomposing the problem space into its constituent components. Stepping down in the hierarchy from "the cloud," we find the following:

- A cloud environment is composed of interacting systems.
- Each system may be comprised of multiple subsystems.
- Systems and subsystems are built from one or more components – at a software level, the host OS, most likely one or more guest operating systems, and the applications that are controlled by the OS and use OS services.

This decomposition can be extended arbitrarily to the required depth, until assessing the risk of the lowest level element (here, a "component") can be meaningfully addressed. At this lowest level, the risk associated with a component can be characterized by assessing the following:

- Weaknesses or vulnerabilities inherent in the component
- The likelihood that a weakness or vulnerability can be exploited
- The existence of a means to exploit the weakness or vulnerability
- The presence of a threat actor with the skills and access to launch an exploit
- The impact of a realized exploit should it occur

Once a risk profile is created using these factors for each component, a combined risk analysis can then be undertaken on subsystems and then systems while considering the operational importance to the organizational mission. This top-down decomposition, component risk assessment, and bottom-up reconstitution yields a structured means of assessing the risk associated with a complex cloud environment.

Threat Modeling: Define the Attack Surface and Develop Misuse Cases

To further understand component risk, one must understand the specific means by which a threat actor might be able to exploit a vulnerability. This is the component's Attack Surface. Once decomposition is accomplished, a good way to define one's composite Attack Surface is to perform Threat Modeling on each component. Microsoft teaches an application centric method called Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege (STRIDE). Aspects such as Denial of Service (DoS) and Elevation of Privilege have a more pronounced importance in cloud applications.

For example, due to the need of accessibility to data, an effect of a DoS attack may cause far worse consequences in a public cloud than a private cloud. A prolonged DoS attack may also affect user perception of how well an application operates.

Further, if one hosts their cloud application with a cloud provider, it is likely that their application will reside on a virtual machine (VM) running on a large server class system with other VMs, as many public cloud-based systems utilize hosted VMs. In this virtual environment, privilege escalation may have a far-reaching effect on one's hosted application and other applications hosted on the same system. As depicted in Figure 1, a successful privilege elevation attack of a hosted VM may put other hosted VMs in one's "digital neighborhood" at risk.

If the cloud application in "App B" has an exploitable vulnerability, successful privilege elevation to the "Host Operating System", may enable an attacker to mediate or disrupt "App A" or "App C."

Given the example above, threat modeling can help one think about boundaries to applications and the various interfaces that cross these boundaries. This effort can help one derive misuse cases and think about how an attacker could leverage these interfaces and discover flaws in one's design. Requirements and test cases can then be developed to focus on a cloud-based environment. These requirements and associated test cases may highlight bugs in the form of weaknesses as listed in the Common Weakness Enumeration (CWE), or in the form of exploitable vulnerabilities as listed in the Common Vulnerabilities and Exposures. These requirements can be verified and validated over time with periodic static and dynamic testing.

Static and Dynamic Testing

According to the Cloud Security Alliance (CSA) Research group, the top cloud threats are "Trust, Data Leakage, Insecure Cloud software, Malicious use of Cloud services, Account/Service Hijacking, Malicious Insiders, and other Cloud-specific attacks." Testing one's cloud application prior to deployment and early within the System Development Lifecycle is important to ensure SwA of the cloud application.

Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) are critical activities to ensuring secure coding practices are verified and that no latent vulnerability remains. For DAST, additional focus should be applied to simulating a cloud environment and to fuzz testing input and protocols.

SAST should focus on the CWEs most relevant to one's mission. With knowledge of one's component risk and criticality, the Common Weakness Risk Analysis Framework can be applied to prioritize weaknesses. For example a "Use of a One-Way Hash without a Salt (CWE-759)" may be rated a lower effective risk than "Double Free (CWE-415)." A Double Free can create corruption that causes the program to crash leading to service interruption.

DAST should focus on exercising and testing the Attack Surface that was identified during threat modeling. Fuzz testing of all inputs may reveal coding weaknesses that represent exploitable vulnerabilities. For example, consider a "Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') (CWE-120)" that exists within the application server. A malicious distributed client could be deployed by a malevolent actor to try multiple concurrent input attempts. Each input provides data which is accepted by the system, but which exceeds the expected size of the buffer receiving the input. The extraneous data can be crafted to give control of the application to the attacker.

DAST is particularly useful for testing backend systems such as Databases. One of the biggest risks to a cloud application is

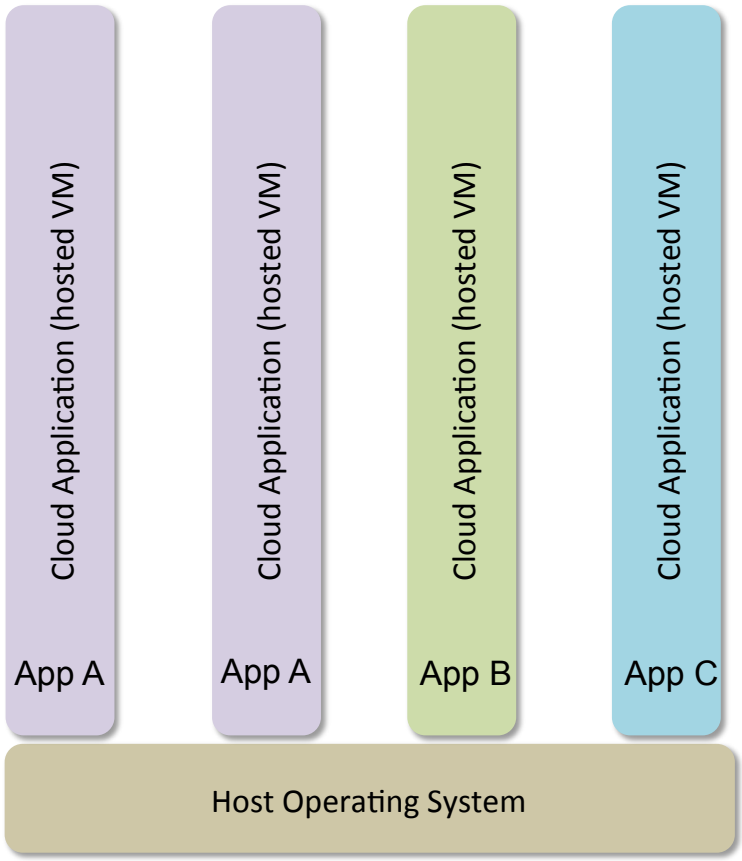


Figure 1. Digital Neighborhood

unsanitized input that is treated as a valid database SQL command. The Open Web Application Security Project (OWASP) number one "Application Security Risk" is entitled "A1 – Injection". OWASP states "Injection flaws, such as SQL, OS, and Lightweight Directory Access Protocol injection occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data." CWE/System Administration, Networking, and Security Institute Top 25 Most Dangerous Software Errors notes this as Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') (CWE-89). DAST can be leveraged to dynamically execute SQL commands injected into the backend database. This helps to locate non-parameterized queries prior to production release.

Since the kinds of applications under discussion will be in a threat environment that is fully open, 3rd party testing is more critical than ever. Internal development houses may have their "blinders" on and may lack the "Think Evil" gene required to appropriately test their application. A coder's ability to understand how one compromise might be leveraged to obtain something of greater value may be lacking. Although education of software writers over time is important, this shortcoming in software staff is currently prevalent and suggests that 3rd party testing is a key to successful application security testing in a cloud environment.

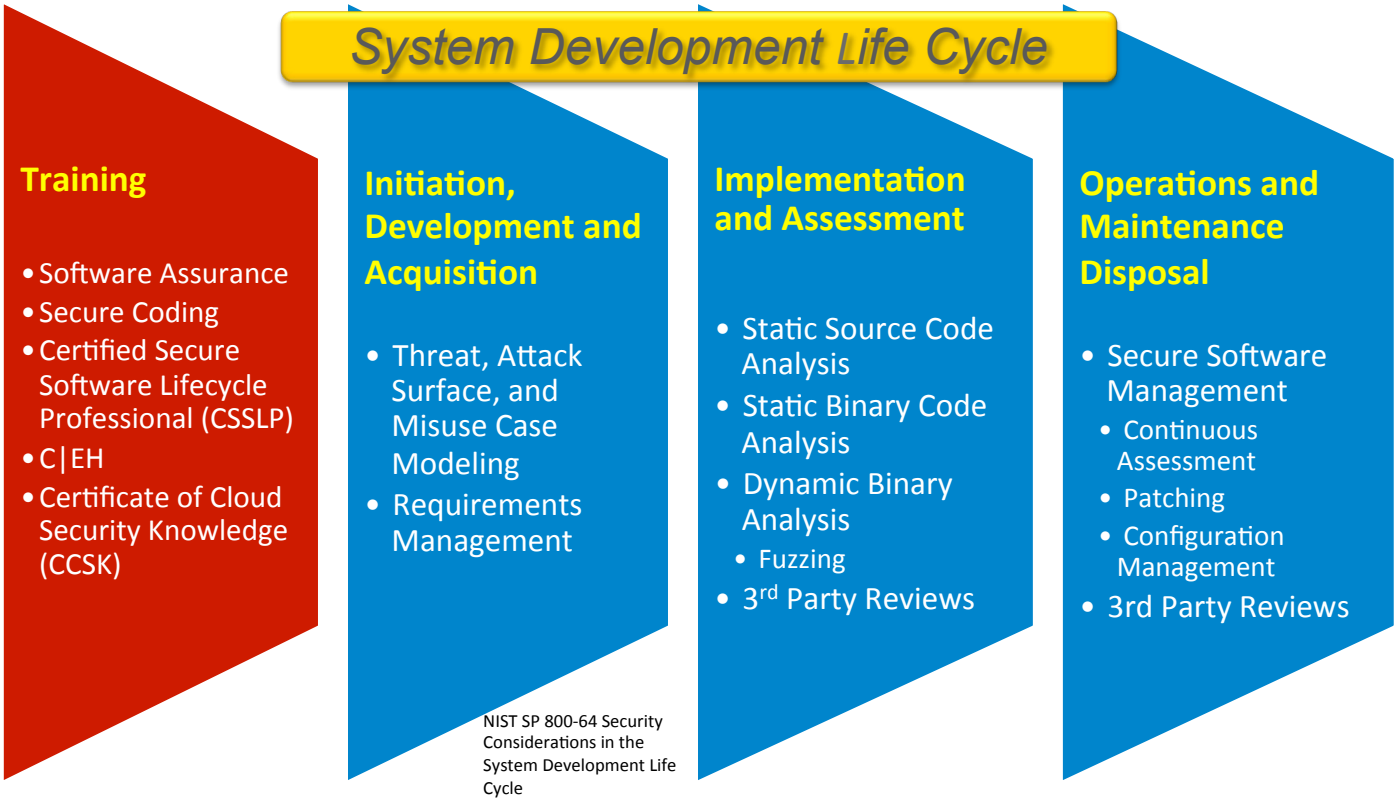


Figure 2. Software Assurance for the Cloud

Additional Items to Consider

As shown in Figure 2, the NIST Special Publication (SP) 800-64 Revision 2 entitled "Security Considerations in the System Development Life Cycle" covers the Initiation, Development/Acquisition, Implementation/Assessment, Operations and Maintenance, and Disposal phases of a development Life Cycle. The topics previously discussed within this article align with these phases, but can be further augmented with training:

- **Software Assurance – Training covering a general overview of SwA and its importance**
- **Secure Coding – Training covering secure coding practices to avoid common programming mistakes**
- **Certified Secure Software Lifecycle Professional (CSSLP) – A professional certification by the International Information Systems Security Certification Consortium (ISC)², which includes eight Common Body of Knowledge domains**
- **Certified Ethical Hacker (C|EH) – A professional certification provided by the International Council of E-Commerce Consultants (EC-Council)**
- **Certificate of Cloud Security Knowledge (CCSK) – A professional certification by the Cloud Security Alliance which covers 14 domains across 3 sections (Cloud Architecture, Governing in the Cloud, and Operating in the Cloud)**

Although not required when adequate 3rd party security test staff is available, training of both systems and software staff is important as it helps to address issues early in the Life Cycle.

On the other end of the Life Cycle is Operations and Maintenance. One must consider the importance of availability in a cloud application. Patching vulnerabilities identified may have to take place while that system is online and operating, which may lead to data integrity and stability issues. In private clouds it may be permissible to schedule a maintenance window, and turn off portions of cloud application functionality. In the highly dynamic public cloud, where resilience is key, patching and operating in a redundant fashion will enable smoother transitions to upgraded versions.

With specific reference to cloud applications, the CSA publishes the Governance, Risk Management and Compliance (GRC) stack at <https://cloudsecurityalliance.org/research/grc-stack>. The GRC stack covers a suite of four integrated initiatives: the Cloud Audit, the Cloud Controls Matrix, the Consensus Assessments Initiative, and the Cloud Trust Protocol. By applying SwA concepts to one's cloud application built in compliance with the GRC stack, one's application should be far more resilient to a harsh cloud environment.

Conclusion

It is important that organizations understand the criticality of their application or service with respect to its organizational value. With this knowledge, one needs to understand their Attack Surface and what affect a threat may have to their critical program information. Performing Threat Modeling and focusing on architectures can help one derive testing requirements. Internal and external SAST and DAST testing can go a long way to ensure SwA of their cloud application. Always remember that the time spent testing the application internally for security flaws will be dwarfed by the time a determined attacker may be willing to spend attempting to exploit that application.

ABOUT THE AUTHORS

Randall Brooks, Engineering Fellow, Raytheon, has more than 15 years of experience in Cybersecurity with expertise in Software Assurance (SwA) and secure development life cycles (SDLC). He has been awarded three US patents on Intrusion Detection and Prevention, and three US and one UK patent(s) on Cross Domain solutions. He is also a CISSP, CSSLP, ISSEP, ISSAP and an ISSMP. He is a graduate of Purdue University with a Bachelors of Science from the School of Computer Science. He represents Raytheon within the U.S. International Committee for Information Technology Standards Cyber Security 1 (CS1).

Phone: 727-302-2189
E-mail: brooks@raytheon.com

John Whited, Principal Engineer, Raytheon, has 5 years of experience in Cybersecurity with expertise in Software Assurance (SwA) and secure development life cycles (SDLC). Prior to joining Raytheon, he was a software and a systems engineer in commercial telephony, holding five US patents on Intelligent Networks. He is also a CISSP and a CSSLP. He is a graduate of Texas Tech University with a Bachelors of Science and a Masters of Science in Electrical Engineering. He has made two joint presentations at the RSA Security Conference (2010 and 2012).

Phone: 972-205-4750
E-mail: john.whited@raytheon.com

Cloud Shifts the Burden of Security to Development

Arthur Hicken, Parasoft

Abstract. The move to the cloud brings a number of new security challenges, but the application remains your last line of defense. Engineers are extremely well poised to perform tasks critical for securing the application—provided that certain key obstacles are overcome.

Introduction

This paper explores three ways to help development bear the burden of security that the cloud places on them:

- Use penetration testing results to help engineers determine how to effectively "harden" the most vulnerable parts of the application.
- Apply the emerging practice of "service virtualization" to provide engineers the test environment access needed to exercise realistic security scenarios from the development environment.
- Implement policy-driven development to help engineers understand and satisfy management's security expectations.

New Risks, Same Vulnerability

Before the move to the cloud, few organizations lost sleep over application security because they assumed their internally controlled security infrastructure provided ample protection. With the move to cloud, security concerns are thrust into the forefront as organizations consider how much security control they are willing to relinquish to cloud service providers and what level of exposure they are willing to allow [1].

The fact of the matter is that with or without the cloud, failure to secure the application always is—and always has been—a dangerous proposition [2]. Even when the bulk of the network security rested under the organization's direct control, attackers still managed to successfully launch attacks via the application layer. From the 2002 breach at the Australian Taxation office where a hacker accessed tax details on 17,000 businesses [3], to the 2006 incident where Russian hackers stole credit card information from Rhode Island government systems [4], to the recent attack that brought down the NIST vulnerability database [5], it is clear that a deficiency in the application layer can the be one and only entry point an attacker needs.

Public cloud, private cloud, or no cloud at all, the application is your last line of defense and if you do not properly secure the application, you are putting the organization at risk [6]. Nevertheless, the move to the cloud does bring some significant changes to the application security front:

- Applications developed under the assumption of a bullet-proof security infrastructure might need to have their strategies for authorization, encryption, message exchange, and data storage re-envisioned for cloud-based deployment.
- The move to cloud architectures increases the attack surface area, potentially exposing more entry points for hackers. This attack surface area is compounded with more distributed computing technologies, such as mobile, web, and APIs.
- As applications shift from monolithic architectures to composite ones, there is a high degree of interconnectedness with 3rd party services—and a poorly engineered or malfunctioning dependency could raise the security risk of all connected components. For example, a recent attack on Yahoo exploited a vulnerability from a third-party application [7] The composite application is only as secure as its weakest link.
- As organizations push more (and more critical) functionality to the cloud, the potential impact of an attack or breach escalates from embarrassing to potentially devastating—in terms of safety, reputation, and liability.

With the move to the cloud placing more at stake, it is now more critical than ever to make application security a primary concern. The industry has long recognized that development can and should play a significant role in securing the application. This is underscored by the DoD's directive for certifications in the area of software development security (e.g., via CISSP) [8, 9]. Select organizations that have successfully adopted a secure application development initiative have achieved promising results [10]. However, such success stories still remain the exception rather than the rule.

Should Development Be Responsible for Application Security?

Due to software engineers' intimate familiarity with the application's architecture and functionality, they are extremely well-poised to accomplish the various tasks required to safeguard application security. Yet, a number of factors impede engineers' ability to shoulder the burden of security:

- The organization's security objectives are not effectively communicated to the development level.
- For engineers to determine whether a particular module they developed is secure, they need to access and configure dependent resources (e.g., partner services, mainframes, databases) for realistic security scenarios—and such access and configurability is not commonly available within the development environment.
- Management often overlooks security when defining non-functional requirements for engineers and planning development schedules; this oversight, paired with the myopic nature of coding new functionality, commonly reduces security concerns to an afterthought.
- Security tests are frequently started at the testing phase, when it is typically too late to make the necessary critical architectural changes.

In the following sections, we explore how strategies related to penetration testing, service virtualization, and policy-driven development can better prepare engineers to bear the heavy burden of security that accompanies the shift to the cloud.

Moving Beyond Penetration Testing: Divide and Conquer

Penetration testing is routinely used to barrage the application with attack scenarios and determine whether or not the ap-

plication can fend them off. When a simulated attack succeeds, you know for a fact that the application has a vulnerability which makes you susceptible to a particular breed of attacks. It alerts you to real vulnerabilities that can be exploited by known attack patterns—essentially sitting ducks in your applications. When a penetration attack succeeds, there is little need to discuss whether it needs to be repaired. It is not a matter of "if", but rather of "how" and "when."

The common reaction to a reported penetration failure is to have engineers patch the vulnerability as soon as possible, then move on. In some situations, taking the path of least resistance to eliminating a particular known vulnerability is a necessary evil. However, relying solely on a "whack a mole" strategy for application security leaves a considerable amount of valuable information on the table—information that could be critical for averting the next security crisis.

Switching to a non-software example for a moment, consider what happened when the U.S. Army realized how susceptible Humvees were to roadside bombs in the early 2000s. After initial ad-hoc attempts to improve security with one-off fixes (such as adding sandbags to floorboards and bolting miscellaneous metal to the sides of the vehicles), the Army devised add-on armor kits to address structural vulnerabilities and deployed them across the existing fleet [11]. In parallel with this effort, they also took steps to ensure that additional protection was built into new vehicles that were requisitioned from that point forward.

How does such as strategy play out in terms of software? The first step is recognizing that successful attacks—actual or simulated—are a valuable weapon in determining what parts of your application are the most susceptible to attack. For example, if the penetration tests run this week succeed in an area of the application where penetration tests have failed before—and this is also an area that you have already had to patch twice in response to actual attacks—this module is clearly suffering from some underlying security issues that probably will not be solved by yet another patch.

Divide ...

This is where "divide and conquer" comes into play. If you can zero in on your most vulnerable components, it is much simpler for the engineers tasked with securing the application to devise an effective attack plan.

The first task is to determine which parts of the application are most prone to security attacks. Since penetration testing is essentially an "outside in" testing strategy (e.g., launching attacks from the UI or API level and examining the response for indications of success or failure), this can be challenging—particular if your penetration tests are broad rather than targeted. In other words, penetration testing typically tells you that a certain type of attack succeeded, but fails to inform where or how the success was actually achieved.

One way to better isolate the precise point of failure is to have "runtime error detection" monitor the back-end of the application to report exactly where the attack succeeded. Another way is to use an emerging test environment simulation technique known as Service Virtualization to effectively isolate attacks on a component-by-component basis. This strategy will be discussed in more detail later in this paper (along with other uses of service virtualization for application security purposes).

...and Conquer

Once you have zeroed in on the application components where you want to focus your application security resources, it is time to conquer those areas by addressing security from the inside out. Since the development team is most familiar with the application internals, these tasks can and should fall under their purview.

Many organizations are accustomed to relying on penetration testing performed by performance specialists as their primary security strategy. However, as application security comes to the forefront with the move the cloud, it is no longer conscionable to assume that zero successful penetration test attacks indicates a secure application. It is important to recognize that there are two key drawbacks of penetration testing:

- It is reactive: Penetration testing is a reactive approach, checking whether the application resists a set of known attack patterns. As with anti-virus programs, you are constantly chasing after "flavor of the week" attacks rather than taking a proactive approach of identifying and removing root causes.
- It is incomplete: Penetration tests uncover problems only in the paths that the tests exercise. Considering that even a relatively small 10,000-line program has 100 million possible execution paths, the opportunities for overlooked vulnerabilities are staggering.

Fortunately, such drawbacks can be overcome by having development apply two complementary types of static analysis to the modules that your penetration tests identify as being the most vulnerable to security attacks.

Flow-based static analysis is perfectly suited for quickly exposing security vulnerabilities in large code bases without requiring the definition or execution of a single test case. You can hone in on vulnerabilities like the ones that your penetration tests exposed in this component, or you can scour the component for a broader scope of vulnerabilities that might be of interest.

Although flow-based static analysis undeniably checks a broader swath of the application than penetration testing feasibly could, it is important to realize that it is not a panacea. It too has some significant shortcomings. Most notably:

- A complex application has a virtually infinite number of paths [12], but flow-based static analysis can traverse only a finite number of paths using a finite set of data. As a result, flow-based analysis inevitably finds only a finite number of vulnerabilities.
- Flow-based static analysis identifies symptoms (where the vulnerability manifests itself) rather than root causes (the code that creates or allows the vulnerability).

This is where pattern-based static analysis comes in. Pattern-based static analysis exposes root causes rather than symptoms, and can reliably target every single instance of that root cause. For example, if you are relying on flow-based static analysis alone, you will probably find a few instances of SQL Injection vulnerabilities… but you will not find them all. However, if you enforce an input validation rule through pattern-based static analysis that requires wrapping input methods in validation methods—finding and fixing every instance where inputs are

not properly validated—you can guarantee that SQL Injection vulnerabilities will not occur because you no longer have a near-infinite number of paths to search.

Taking Security to the Next Level with Service Virtualization

One emerging trend that empowers engineers to perform additional security verification—from the early phases of the development process and from their traditional development environment—is Service Virtualization. Service virtualization is a method to emulate the behavior of specific components in heterogeneous component-based applications. It is used to provide development and testing teams access to dependent system components that are needed to exercise an application under test (AUT), but are unavailable or difficult-to-access for testing purposes. With the behavior of the dependent components "virtualized," testing can proceed without having to access the actual live components.

Service virtualization opens a number of opportunities for enabling engineers to perform earlier and more effective security testing.

Begin Security Testing Earlier

As applications move to the cloud, tests that validate the security of an end-to-end transaction must pass through more (and more complex) dependencies. Just testing the security of a single transaction often involves interacting with dependencies ranging from mobile applications, databases, mainframes, 3rd party services, internal services, and more. The common approach to testing in such an interconnected environment is to delay testing until all the necessary resources can be accessed at a single point in time. However, such complete access typically comes late…and sometimes never comes at all. With service virtualization emulating the behavior of unavailable or difficult-to-access components, security testing efforts can start significantly earlier—considerably reducing the cost, effort, and resources required to remediate each vulnerability of weakness detected.

Isolate Where Attacks Succeed

Service virtualization enables testing and validation to be applied at multiple depths and levels. By applying validations and assertions to the messages from the application under test, engineers can isolate and zero in on specific components. They could validate a service's response for any indication that its security controls are not working properly (or that additional ones should be implemented). For example, they can determine how the exact element they are working on handles an attempted attack such as an SQL injection. This direct validation provides significantly more visibility into the behavior of a particular component than could be obtained via the output of the larger integrated system.

Gain Predictable, Complete Environment Access from the Development Desktop

After a vulnerability is exposed during penetration testing (or in the field), the software engineers responsible for remediating the problem typically spend considerable effort trying to reproduce the problematic behavior in their own environment so that they can understand it, diagnose the root cause, then verify whether attempted fixes actually resolved the problem.

When tests are run vs. a simulated test environment, engineers are able to rapidly and accurately replicate the precise conditions that triggered the vulnerability. They can then debug the problem and validate their proposed fixes directly from their desktop. If this debugging and validation involves systems beyond development's scope of control (e.g., partner services, databases that are difficult to access for testing, mainframes, etc.), development can run vs. simulated instances of these systems so access limitations do not delay or compromise their ability to complete security remediation tasks.

Moreover, the anytime/anywhere access that service virtualization provides enables the development team to run continuous regression testing. This ensures that the team is alerted if previously remediated security vulnerabilities ever re-surface as the code base evolves.

Run Functional Tests vs. Realistic and Extensive Security Scenarios

With service virtualization, even team members who are not security experts can take their existing functional tests cases and execute them against a broad set of preconfigured security scenarios. Security specialists can pre-configure the environment's dependencies to emulate different security scenarios that would otherwise be difficult to set up and unfeasible to test against. For instance, with SSL, you could configure acceptable and unacceptable certificates. Or, you could emulate various behaviors related to authorization, authentication, and access controls. You could also configure the dependent system to deliver malicious payloads. This provides very granular control over the security behavior of the dependencies in your environment. Now, as a complement to penetration testing, the team's standard functional test scenarios can then be run against these different environment configurations.

Perform Stateful Security Testing

While standard penetration testing simulates attacks through an API in a non-stateful manner, service virtualization lets you emulate attacks at various levels of the system and at different points within a stateful process (e.g., at different points in a logical use case or workflow). This not only broadens your test coverage, but can also expose security vulnerabilities that are manifested only under a certain set of environment conditions—and that would not be apparent with non-stateful penetration testing.

Closing the Gap: Putting Policy in Place

A third component critical to helping development shoulder the burden of security is "policy-driven development." The goal here is to ensure that security requirements (and other non-functional requirements) are exposed and measured as aggressively as functional requirements. Adopting a policy-driven development process where expectations are enforced in

a standardized way across the development group is a low-hanging fruit for taking the necessary level of control over not only what software is being developed, but also over how that software is being developed.

The first step in policy-driven development adoption is to define what policies you want to implement. Policies can be formed around any aspect of the development process. To be effective, they must be definable, enforceable, measureable and auditable. You can define as many policies as necessary to help you achieve your goals, but you should start implementing them on a small scale. Introduce a few policies at a time, and as you become proficient with those policies, you can introduce more in small batches. One approach is to start with policies to ensure that new code is built in a way that prevents security weaknesses and vulnerabilities. Another common approach is to begin by focusing on eliminating the highest severity vulnerabilities in the existing code base.

Next, train software engineers on policies. Beyond documenting the how and the why of your policies, you also want to take steps to ensure that the connection between the two is clear. The absence of training is the Number 1 reason policies fail. If a policy requires code to be structured in a certain way, the engineer may not immediately see the potential for the bug that the structure is intended to prevent. If the engineer does not make the connection during this cycle or even the next few cycles, then the policy looks more like a guideline to him or her, leading typically to an (incorrect) declaration of "false positives." Thus, the code may not be properly structured before the product goes to market, and vulnerabilities may surface in the field. At this point, the implementation of the policy has failed.

Finally, use automation to drive a sustainable process. Automating policy monitoring, as well as the process for routinely notifying engineers of violations, ingrains policies into the day-to-day workflow. Without this level of automation, policies will quickly fade and expected behavior will degrade back into recommended rather than mandated behavior. A centralized infrastructure capable of managing policies will go a long way toward realizing the benefits of policy-driven development. Ideally, a single platform that monitors adherence to multiple types of policies and enables effective implementation will be in place to deliver the traceability required for certification and for audit purposes.

With such a policy-driven process in place, engineers not only know what is expected of them, but receive objective, immediate feedback on whether they are meeting expectations.

ABOUT THE AUTHOR

Arthur Hicken has been involved in automating various practices at Parasoft for almost 20 years. He has worked on projects including database development, the software development lifecycle, web publishing and monitoring, and integration with legacy systems. Arthur has worked with IT departments in companies such as Cisco, Vanguard, and Motorola to help improve their software development practices. He has taught at the College of DuPage in Illinois as well as developing and conducting numerous technical training courses at Parasoft. As an expert in his field, Arthur has been quoted in Business 2.0, Internet Week, and CNET news.com regarding Web site quality issues.

Phone: 626-275-2445
E-mail: ahicken@parasoft.com

REFERENCES

1. States, Lauren. "Tips for Building a Secure Cloud." Lauren States - IBM Cloud Computing. IBM DeveloperWorks, 1 Nov. 2010. Web. 03 May 2013.
2. "Top 10 2010-Main." OWASP. OWASP, n.d. Web. 03 May 2013.
3. Maslowski, Jakub. "10 Reasons Websites Get Hacked." Zone-H.org. Zone-H, 10 Oct. 2007. Web. 03 May 2013.
4. Maslowski, Jakub. "10 Reasons Websites Get Hacked." Zone-H.org. Zone-H, 10 Oct. 2007. Web. 03 May 2013.
5. Gross, Grant. "U.S. NIST's Vulnerability Database Hacked." Computerworld. Computer world, 14 Mar. 2013. Web. 03 May 2013.
6. Schwartz, Matthew. "InformationWeek: The Business Value of Technology." Information Week Security. InformationWeek Security, 7 Apr. 2011. Web. 03 May 2013.
7. Kovacs, Eduard. "Yahoo! Hack Demonstrates the Risks Posed by Third-Party Code in Cloud Computing." Softpedia. Softpedia, 30 Jan. 2013. Web. 03 May 2013.
8. Gilbertson, Daryl. DoD Directive (DoDD) 8570 & GIAC Certification. Publication. SANS, June 2011. Web. 3 May 2013.
9. Grimes, John G. Information Assurance Workforce Improvement Program. Publication. DoD, December 19, 2005, rev January 24, 2012. Web. 9 May 2013.
10. Maiffret, Marc. "Closing the Door on Hackers." New York Times. N.p., 4 Apr. 2013. Web. 3 May 2013.
11. Bowman, Tom. "U.S. Scrambles for Armored Cars as Troops Make Do With Sandbags."Baltimoresun.com. Baltimore Sun, 27 Mar. 2004. Web. 03 May 2013.
12. The underlying "path math" is
    [2n(L1*L2*…*Lx) (V1*V2*…*Vy)]!
    Where:N is the number of decisions (branches).
    Ln is the maximum number of times a loop can loop.
    x is the number of decisions which cause a loop (X≤N).
    Vn is the number of different values each input variable can assume.
    y is the number of input variables.
    The factorial (!) is there because the order of running the test cases counts.

The Active Shooter System

LTC Phillip G. Burns, U.S. Army

Abstract. The proposed Logical Active Shooter System is a referential architecture that guides the securing of data as DoD migrates to a Joint Information Environment. The goal is to secure critical information from malicious access or misuse that impacts mission accomplishment throughout the Joint phases of operation.

Introduction

The United States of America faces a challenge—cyber threats to national interests abound, while homegrown security professionals who are able to operate effectively in cyberspace do not. By all accounts, this lack of effective personnel is the weakest area in building and defending the network. A study by Frost and Sullivan supports this assertion [1]. To add to this, many of today's cybersecurity professionals have to read up on cyber threats, as many of them are digital immigrants and are not digital natives. In contrast, digital natives grew up with computers, video games and computer graphics. Automation is second nature to digital natives.

DoD organizations, such as U.S. Cyber Command (USCYBERCOM) and the NSA, must reach out to digital natives, recruiting and molding them to build, defend the military network and, at times, hunt for malicious intruders set on attacking the network. Of course, distinctions between United States Code (USC) Title 10 and USC Title 50 [2]—between operations and intelligence—may constrain how we build and defend the network, as well as hunting adversaries. According to one researcher, decisions to execute a defense against a cyber attack are often measured in seconds or milliseconds [3]. Operators placed in that position must have Title 10/50 authorities and the ability to make decisions locally, to apply operational effects necessary to protect or isolate the network. This decision making is a learned skill that adds to the challenge discussed at the outset.

As USCYBERCOM and NSA focus on building the bench of cybersecurity professionals, measures must be in place to protect information as the gap decreases between digital natives and digital immigrants. Until the bench is built, the focus must be to secure data, but not overly restrict the DoD users' access to data in a manner that prevents collaboration.

Within the scope of this discussion, the DoD is directing the consolidation of disparate data centers across the DoD network to a select set of core data centers. Efforts will lead to the integration of the Army's portion of the DoD network with the Joint Information Environment (JIE) at Figure 1. The JIE will provide a single network that is secure, standards-based, flexible and supports versatile mission sets [4].

Future Army network capabilities include chat services and software defined radios that, in accordance with the Unified Compliance Framework, will connect users at home or work with deployed enterprise users. As Figure 1 indicates, all is geared to ensure enterprise users have the "…information they need, when they need it, in any environment, to manage the Army Enterprise and enable Full-Spectrum Operations with our Joint, Coalition, and Interagency partners [5]." JIE will usher unprecedented access to information and a new era of collaboration and situational awareness that enables Mission Command, presenting a formidable foe to the Nation's enemies, with the technology and a network to back up its teeth.

While JIE will provide the standards and the common environment, the Services will employ technologies, such as Host Based Security System (HBSS), Public Key Infrastructure (PKI), Rights and Identity Management, to assure confidentiality, integrity and availability of information; however, these technologies alone may not foster a completely secure environment. The Deployed Environment and Defense Information Systems Network (DISN) clouds at Figure 1 typify one-to-many user interactions, which is hard to audit but not impossible.

This article focuses on logically and physically securing critical DoD information with limited impact to the user's experience and collaborative efforts to ensure situational awareness critical to Mission Command. This article explores a "Logical Active Shooter System" that ensures data is protected from unintentional or intentional spillage. The system must support Title 10/50 requirements, while simultaneous restricting the digital natives' ability to circumvent its controls. The Bradley Manning incident (i.e., "Wikileaks") is mentioned as a use case.

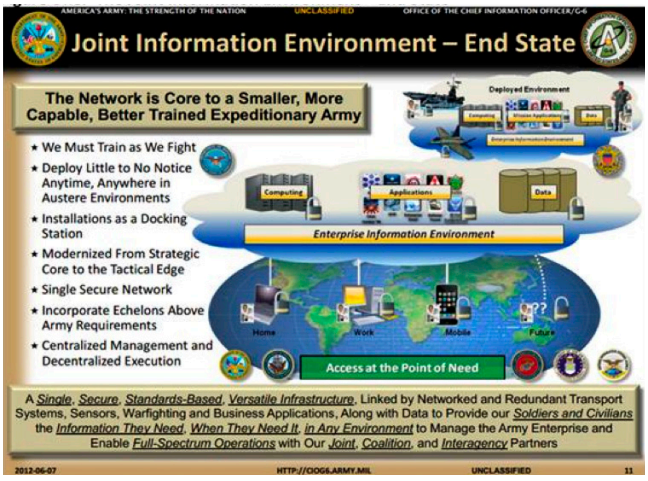


Figure 1. The Joint Information Environment – End State

The Logical Active Shooter System

U.S. Army Mission Command Center of Excellence's (CoE's) Requirement Governance Team, in coordination with U.S. Army Signal CoE's TRADOC Capability Manager for Global Network Enterprise, are developing an operational framework for a cloud-based computing network. Figure 2 illustrates [6] a proposed operational view underpinning the principles of this cloud-based computing network. Deployment of the Logical Active Shooter System would occur after the JIE end state as illustrated in Figure 1.

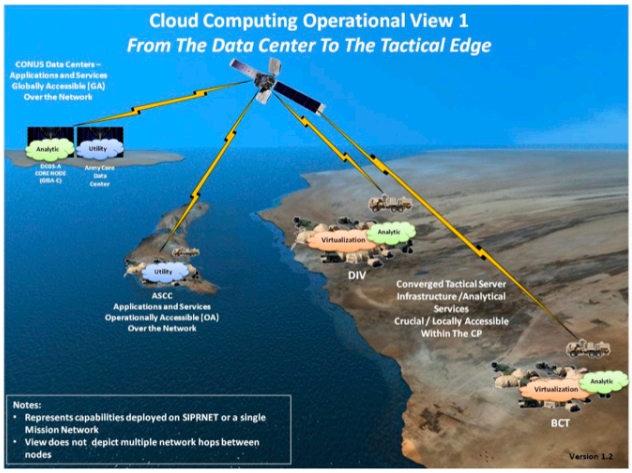


Figure 2. Cloud Computing Operational View

Security technologies, such as HBSS, PKI, and Rights and Identity Management, will be critical to the future network and engineered in the architecture from the start to ensure the end state of a "Single Secure Network." The DoD and Army cloud-based computing networks will leverage the NIST definition of cloud computing: "…a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [7]." The NIST definition implies that anonymous access to information is expected; however, shared concerns of mission security requirements, policy, and compliance considerations will be factored into instantiations of a cloud-based computing environment.

Within a deployed setting, loss of data or spillage of classified material is a real concern, and anonymous access is hard to monitor. It takes leadership and active participation of users to enable an environment where mission critical information is secured from unauthorized users and access.

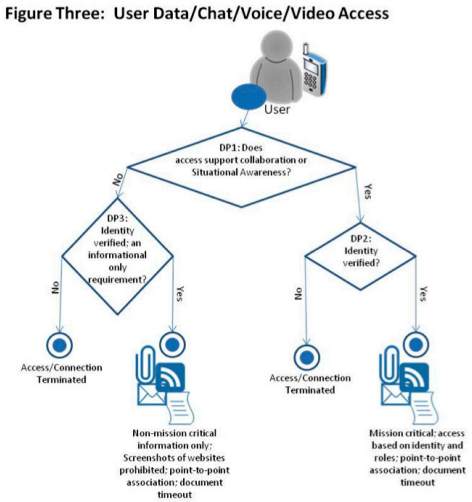
The Bradley Manning incident illustrates the complexity of preventing the spillage of classified material. The Bradley Manning incident is an excellent use case, as it serves as a stark reminder when lax involvement and security posture by leadership and users go awry. For uninitiated readers, Bradley Manning, a digital native who represented a class of insider threat (a disgruntled employee who displays some emotional distress), pled guilty to mishandling classified materials and uploading said information to WikiLeaks.org via his personal laptop. One researcher noted that to mitigate situations like this, "[a]n 'active shooter'-like stance or posture is needed. Technical controls are required and in some cases are implemented, but to what degrees of success are debatable [8]." The researcher goes on to note that, "Involvement of leadership helps to improve IT security, and a well-informed IT security staff helps to identify and correct situations [9]." (For the purpose of this article, taking an "active shooter"-like stance is to intercept the malicious attacker while he or she is in the progress of executing the attack on the network or information system. This taking action can be via involvement of leadership/fellow users or automated enforcement of rules and roles.)

Taking an "active shooter"-like stance alone will not in itself adequately protect the DoD network and mission critical information, because the distributed and open-access nature of cloud computing injects a level of risk that must be factored into risk assessments and technical controls. A roles- and rules-based system is needed to adjudicate or restrict access. Figure 3 illustrates a recommended capability that can secure critical information and logically establish an "active shooter" capability. For the purpose of this article, critical information is defined as information that enables situational awareness within a mission setting that includes classified or For Official Use Only information where its unintended release or leakage impacts a mission or strategic aims. Information releasable to the public is not defined as critical information that will be protected.

The first step is to adjudicate access based upon established roles- and rules-based policies, to which users can authenticate through technology such as Rights Management or PKI. The goal is to marry roles- and rules-based access to the specific platform where access was initially generated. This would be a goal at end state. This is decision point #1 (DP1) as illustrated in Figure 3. If access to information enables collaboration in support of mission informational and situational awareness requirements, then DP2 is enabled. If identity is not verified, then access to information is terminated. Levels of access to information under DP2 are determined by roles- and rules-based access requirements. Information can be in the form of voice, video, and data. Access to data files is time-limited and files are automatically shredded to keep information relevant and current. Timeframes for access to data file are determined by the data owners.

If the answer to DP1 is no, then DP3 is enacted, and the user's identity is verified. If the user's identity is verified, then the user has access to non-mission critical information only; screenshots of websites are prohibited; and data files are set to time out to ensure information is relevant and current. If the user's access under DP3 cannot be verified, then access to information under this category is terminated.

Figure 3. User Data/Chat/Voice/Video Access



As Figure 3 represents a recommended capability, there are several technologies that can enable the capability represented is Figure 3. While there are a number of solutions available, this article will discuss two: 1) Narus N10 and 2) Adobe's document security solution.

(For the purpose of this article, Narus N10 and Adobe's document security solutions represent desired characteristics critical to securing critical information as defined above, which includes lifecycle management of critical documents. The narrow scope of solution sets supports refinement of critical characteristics of the Logical Active Shooter System: role- and rule-based access; a virtual workplace where documents are shredded, encrypted, and interleaved upon termination of connection to the virtual workplace; supports bandwidth constrained environment.)

The first solution is the Narus N10. As a primer, Narus is a wholly owned subsidiary of Boeing. The N10 has the ability to authenticate access and contain access based upon roles and established rules. Updated information is continuously rendered to the user, and a dynamic auditing capability is enabled to scope future access based upon the informational needs of the user. Users do not directly access secured material. Users request access to a particular document and a secured, virtual workplace is created via a protected tunnel (see Figure 4). This virtual workplace facilitates tracking, queuing, and securing of document requests. In addition, the Narus N10 interfaces with a Mobile Synchronization Module, with which users can access current information every time documents are introduced to their workplace.

According to Narus, the N10 ensures that "[d]ocuments stored in the repository are shredded, encrypted, and interleaved with white noise before being scattered randomly throughout the storage environment [10]." Narus further touts that uploaded documents cease to exist in any "integral form" and, therefore, present no target for hackers to attack [11].

The second solution is Adobe's suite of document security software. Adobe's document security solutions support data encryption with symmetric, asymmetric, or hybrid keys, in order to ensure confidentiality. Adobe's solution supports IT security professionals or system administrators to set permissions, as well as support dynamic document control, expiration, and digital signatures. Adobe's document security product line includes Adobe Acrobat Family, Adobe Reader, Adobe LiveCycle Reader Extensions, Adobe LiveCycle Digital Signatures, and Adobe LiveCycle Rights Management [12]. Because Adobe supports integration with Lightweight Directory Access Protocol and Active Directory, roles- and rules-based access control is possible.

While Adobe can secure information in accordance with Figure 3, its products only secure the .pdf file type, and not the full breadth of file types in use across the DoD enterprise. In contrast, Narus N10 supports more of the file types found on the DoD Enterprise. Narus N10 may also have application in securing sensitive information on strategic networks; however, before use in a tactical setting, it must first be evaluated, as tactical users often access information within a bandwidth constrained environment.

While one goal of the JIE is to eventually virtualize Joint common services, tactical users must have access to critical information even while not connected to the DISN. Therefore, developing and maintaining a common operating picture in a disconnected environment is critical to the Warfighter and the Commander on the ground. Situational awareness data and collaborative services in support of missions must go unfettered throughout the Joint phases of the operation. Selected capabilities must support this critical need in addition to securing critical information from malicious exfiltration or willful disclosure of critical information.

In consideration of the Narus and Adobe capabilities, access validation through Rights Management, PKI, and Active Directory is a critical enabler to DP1. DP2 is divided into two sequels: DP2-A and DP2-B. DP2-A supports users in a bandwidth constrained environment, or users who will be adversely impacted if disconnected from the DISN. Therefore, DP2-A provides access to mission critical information with point-to-point association and document timeout to ensure information is current. DP2-B will support user's access to critical information when bandwidth and potential disconnection from the DISN is

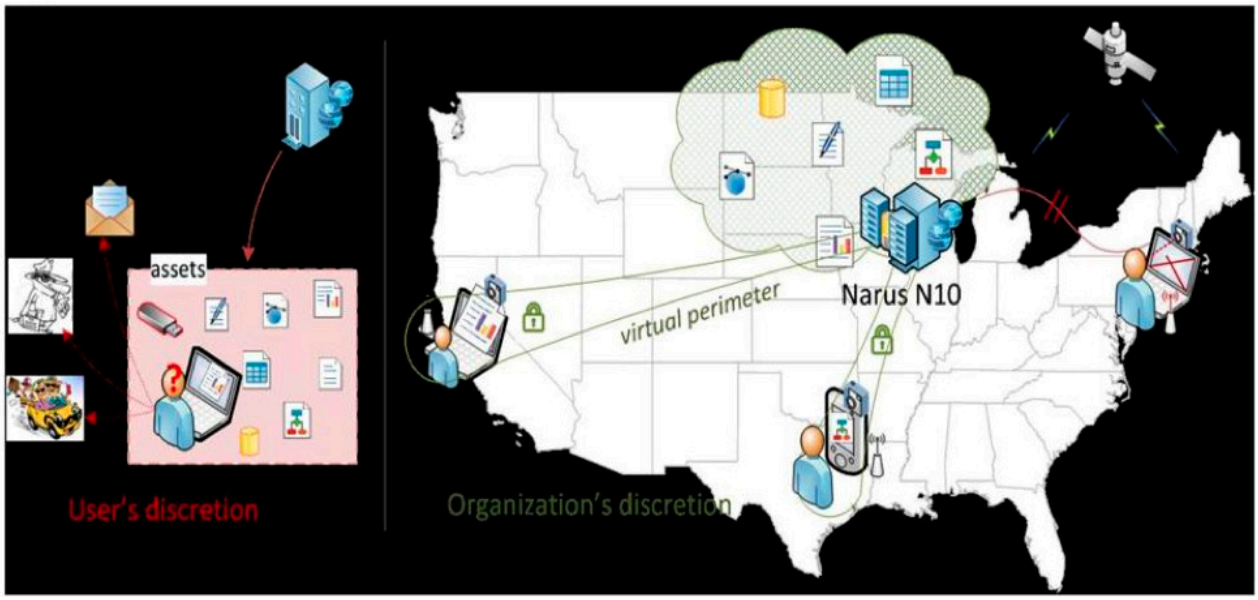


Figure 4. Narus N10

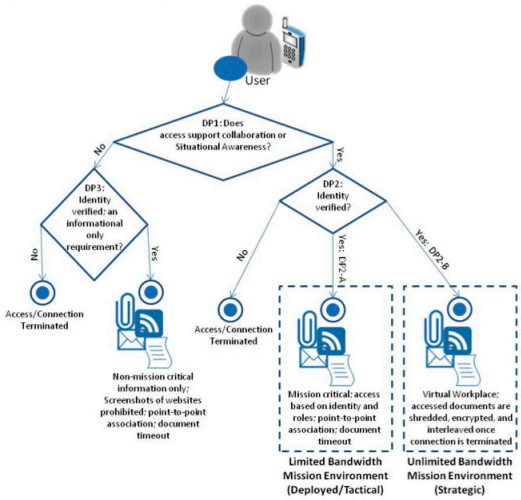


Figure 5. User Chat/Chat/Voice/Video Access

not an overarching concern. DP2-B provides a virtual workplace that facilitates access to mission critical information, in a manner similar to the Narus N10 capability mention above. Figure 5 provides an updated view of the proposed Logical Active Shooter System.

Conclusion

The two Logical Active Shooter System solutions described in this article are only the tip of the iceberg of capabilities that DoD can leverage. They provide a referential architecture that can support a secure cloud-based network. Both capabilities can go far to mitigate an insider threat like Bradley Manning. With the recent posturing and alleged hacking exploits by the Democratic People's Republic of Korea, the need to secure information against all threats becomes paramount as we develop and migrate to a Joint Information Environment. If an organization takes an appropriate "active shooter"-like stance, then the insider threat (intentional or unintentional) can be effectively mitigated. A logical means of bolstering this "active shooter"-like stance is needed to secure critical information and limit exploitation of critical information by insider and outsider threats.

Additional Reading

1. "Demystifying the Title 10-Title 50 Debate: Distinguishing Military Operations, Intelligence Activities & Covert Action" by Mr. Andru Wall, which can be found at <http://harvardnsj.org/wp-content/uploads/2012/01/Vol.-3_Wall1.pdf>. Mr. Wall is a former legal advisor for U.S. Special Operations Command Central (2007-2009). He provides a thorough synopsis of the Secretary of Defense's unique Title 10/50 responsibilities, as they pertain to unconventional and cyber threats. Within the document, Title 10/50 decisions in response to cyber threats are made in milliseconds and often by the same individual.

2. "Data Breach Investigations Report 2012" by Verizon's RISK Team, with cooperation from the Australian Federal Police, Dutch National High Tech Crime Unit, Irish Reporting and Information Security Service, Police Central e-Crime Unit, and the United States Secret Service. It can be found at <http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2012-ebk_en_xg.pdf>. The report provides statistical analysis of compromised records, and provides a analysis of cyber threats resulting in the compromise of records.

3. "NSA: Looking for a Few Good Cybersecurity Professionals" by Dirk Smith, which can be found at <http://www.networkworld.com/news/2012/111312-nsa-cybersecurity-264223.html>. Within the article, the reader is made aware of a current shortfall of 20,000 cybersecurity professionals, with a projected shortfall of 40,000. No concrete date was given for the aforementioned prediction. NSA is attempting to mitigate this by partnering with the Nation's service academies, colleges, and universities.

ABOUT THE AUTHOR

LTC Phillip G. Burns is a capability manager for the U.S. Army Signal Center of Excellence. Prior to that, he served as the Information Assurance Officer for the 2nd Infantry Division, Camp Red Cloud, South Korea. He holds a Master of Science in Computer Information Systems. In 2007, Burns graduated from the Information Systems Officer course at the U.S. Army's School of Information Technology at Fort Gordon, Georgia.

10810 Glenbarr DR
Johns Creek, GA 30097
Phone: 678- 548-9927
E-mail: Phillip.G.Burns.mil@mail.mil

REFERENCES

1. Frost and Sullivan, "The 2011 (ISC)2 Global Information Security Workforce Study." 13 May 2013 <https://www.isc2.org/uploadedFiles/Industry_Resources/FS_WP_ISC%20Study_020811_MLW_Web.pdf>
2. Wall, Andru. "Demystifying the Title 10-Title 50 Debate: Distinguishing Military Operations, Intelligence Activities & Covert Action." Harvard National Security Journal 3 (2011).
3. Ibid.
4. Lawrence, Susan. "Network Information Brief: Improving Network Security and Operational Effectiveness." 7 June 2012 <http://ciog6.army.mil/LinkClick.aspx?fileticket=04Ezkdq_fGU%3D&tabid=36>.
5. Ibid.
6. Mackert, Donald. "Cloud Computing Operational View 1 from the Data Center to the Tactical Edge." (email communication, 9 April 2013).
7. Mell, Peter, et. al. "The NIST Definition of Cloud Computing: Recommendations of the National Institute of Science and Technology." September 2011 < http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>.
8. Burns, Phillip. "IT Governance: Key to Security." Military Information Technology 15.9 (2011): 21.
9. Ibid.
10. Lockhart, David. "Technology Brief: Protection for a Mobile Workforce." (email communication, 24 February 2013).
11. Ibid.
12. Multiple Authors. "A Primer on Electronic Document Security: How Document Control and Digital Signatures Protect Electronic Documents." 9 April 2013 <http://www.adobe.com/security/pdfs/acrobat_livecycle_security_wp.pdf >.

Upcoming Events

Visit <http://www.crosstalkonline.org/events> for an up-to-date list of events.

APCOSEC 2013
9-11 September 2013
Yokohama, Japan
http://www.incose.org/newsevents/events/details.aspx?id=190

Defense Systems Acquisition Management Course
16-20 September 2013
Kansas City, MO
http://www.ndia.org/meetings/302E/Pages/default.aspx

Software and Supply Chain Assurance Forum
17-19 September 2013
McLean, VA
https://buildsecurityin.us-cert.gov/swa

(ISC)² Security Congress 2013
24-27 September 2013
Chicago, IL
https://www.isc2.org/congress2013/default.aspx

World Congress on Engineering and Computer Science
23-25 October 2013
San Francisco, CA
http://www.conferencealerts.com/show-event?id=112271

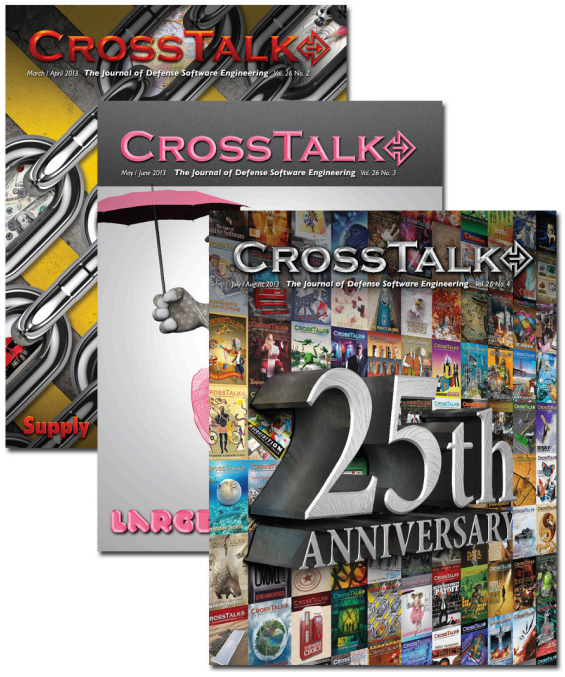
16th Annual Systems Engineering Conference
28-31 October 2013
Arlington, VA
http://www.ndia.org/meetings/4870/Pages/default.aspx

Technology Tools for Today (T3) Conference
3-5 November 2013
Rosemont, IL
http://2013t3enterprise-eorg.eventbrite.com

Aircraft Survivability Symposium 2013
5-7 November 2013
Monterey, CA
http://www.ndia.org/meetings/4940/Pages/default.aspx

2013 Homeland Security Symposium
7-8 November 2013
Washington, DC
http://www.ndia.org/meetings/4490/Pages/default.aspx

OWASP AppSec USA 2013
18-21 November 2013
New York, NY
http://www.sourcesecurity.com/events/free-event-listing/owasp-appsec-usa-2013.html



CALL FOR ARTICLES

If your experience or research has produced information that could be useful to others, CROSSTALK can get the word out. We are specifically looking for articles on software-related topics to supplement upcoming theme issues. Below is the submittal schedule for three areas of emphasis we are looking for:

Mitigating Risks of Counterfeit and Tainted Components
March/April 2014 Issue
Submission Deadline: Oct 10, 2013

The Immutable Laws of Software Development
May/June 2014 Issue
Submission Deadline: Dec 10, 2013

High Maturity Organizational Characteristics
July/August 2014 Issue
Submission Deadline: Feb 10, 2014

Please follow the Author Guidelines for CROSSTALK, available on the Internet at <www.crosstalkonline.org/submission-guidelines>. We accept article submissions on software-related topics at any time, along with Letters to the Editor and BackTalk. To see a list of themes for upcoming issues or to learn more about the types of articles we're looking for visit <www.crosstalkonline.org/theme-calendar>.

SUBSCRIBE TODAY!

To subscribe to CrossTalk, visit www.crosstalkonline.org and click on the subscribe button.

Excuse Me, But Do You Have the Time?

Back in 1982, I was a SSgt in the USAF, finishing my undergraduate degree at the University of Central Florida. Because of my course load, I was spending lots and lots of time in the computer center (I was taking Operating Systems, Compilers, and Assembly Language all in the same semester). The media I used for programming (punched cards) cost $0.94 for 500 cards. If you were paranoid (I was) and wanted a backup for security (I did), all you needed was an IBM 514 Duplicator, a deck of fresh cards, and about 2 minutes to duplicate 500 cards. My electronic footprint at that time, was three decks of cards (one for each class) so in 6 minutes, I could backup everything I needed.

However, I wearied of spending nights fighting for cardpunch machines, and then having to wait in line to run my job. I decided to surge forward with new technology. I bought a home computer—a Commodore SuperPet 9000—and it was awesome! It had not one, but two processors—a MOS 6502 (running Commodore OS with Commodore Basic and a Word Processing program), and a Motorola 6809 (running a Waterloo Programming Operating System, supporting APL, Fortran, COBOL, Pascal, Basic and Assembler). It had a blazing clock speed of 1 MHz. It was possibly the most technologically advanced small computer for the time (31 years ago).

My purchase included a Hayes 300 baud Smartmodem, and a dot-matrix printer for a total price of about $4,000. It did not come with a floppy disk unit, and I could not afford the higher-priced quad-density Commodore 8080, so I bought the cheaper Commodore 4040 dual disk drive—a single unit with two disk drives, each with a capacity of 340K. Before long, I migrated almost everything I had previously done manually (using my typewriter) to disk. I had disks for each class, disks with games, disks with lists of my VHS tapes … you get the idea. I had around 20 disks with "critical" information.

To backup all of my "critical" files, it took several boxes of disks, and about 2 – 4 minutes to copy each disk separately. As often happened, if disk had a single bad spot, you had to scrap the entire disk, and start with a new one. I could happily spend an entire night backing up 20 disks. I tried to remember to do it once a month, so a bad disk would never cost me more than 30 days of lost work. (I also learned to backup school data daily, sometimes hourly.) Come to find out, the backup took up quite a bit of my time. Sometimes, a full evening per month, with maybe an hour each couple of days for important programs, files, reports, etc.

Over the intervening years, I advanced from floppies to "firmies" (what else did you call those 3.5-inch floppies that were not very floppy on the outside?) to CDs, then USB drives, and now, the cloud.

The problem, of course, is that as the capacity for backup media increases, the amount of information that I need to backup increases. Every backup medium I used eventually became totally full from the vast amount of digital information I now considered important to keep backup copies of.

But the cloud? It is huge! For about $45 a month, I can have up to one terabyte of mostly-always-available storage. Is that too costly? Several commercial cloud providers provide five gigabytes totally free!

Advantages? It is online, always current, and virtually transparent to the user. Disadvantages? You have to be online. While the data synchronization and access occurs transparently, it does consume some bandwidth. You have a 5MB file that you want to access and modify? 5MB transfers pretty quickly. You want to access a 5GB database? Well, unless you have some mechanism in place to download or modify just a few records it might take you 1,000 times as long to download and then upload a modified version.

When I first started on the path to becoming a computer scientist, way back at the University of Central Florida in 1974 (it was called Florida Technological University back then), I had a class in data structures from an adjunct professor who, in his full-time job, designed and maintained large-scale databases. He taught me, "It is always about the tradeoff between time and space." Make something run faster, it probably takes up more storage. Reduce storage, and the application almost always takes longer to execute.

All I am trying to say is that nothing is free. Large-scale cloud usage requires increased access and file update time. Do not plan for real-time performance when bandwidth is congested, the Internet is down, or lots of users are all trying to access really large cloud files.

By the way, remember those punched cards that cost about $1 for 500 cards? Well, one card equaled about 160 bytes, so $1 bought me 80Kbytes of storage. This equates to about $12,500 per gigabyte of storage. Makes the cloud quite a bargain.

David A. Cook, Ph.D.
Stephen F. Austin State University
cookda@sfasu.edu

Homeland Security

Software and Supply Chain Assurance

Software and Supply Chain Assurance are essential to enabling the nation's critical infrastructure.
To ensure the integrity of that infrastructure, the software and the IT supply chain must be secure and resilient.

The Software and Supply Chain Assurance Community Resources and Information Clearinghouse provides corroboratively developed resources. Visit https://buildsecurityin.us-cert.gov/swa to learn more about relevant programs and how you can become involved.

Software and Supply Chain Assurance must be "built-in" and supported throughout the lifecycle.
Visit https://buildsecurityin.us-cert.gov/bsi to learn about the practices for developing and delivering software to provide the requisite assurance. Sign up to become a free subscriber and receive notices of updates.

The Department of Homeland Security provides the public-private collaboration framework for shifting the paradigm to software and supply chain assurance.

https://buildsecurityin.us-cert.gov/swa

CROSSTALK thanks the above organizations for providing their support.