Global Information Assurance Certification Paper

Moving Legacy Software and FOSS to the Cloud, Securely

GIAC (GSEC) Gold Certification

Author: Larry Llewellyn II, larry.llewellyn.ii@gmail.com
Advisor: Stephen Northcutt
Accepted: December 17th 2015

Abstract

As more organizations consider the benefits of cloud adoption and use of legacy or free and open source software (FOSS) to lower the cost of software engineering and information technology projects, the security implications of this model take center stage. Cloud is not a one-size-fits-all solution and any effort to adopt distributed data services must integrate information assurance early in the project. Correspondingly, transitioning legacy software or FOSS into a cloud environment has the added disadvantage of potentially injecting serious security challenges incurred when the code was originally developed. This paper investigates existing methods, and proposes a practical approach, for mitigating risks associated with transition of legacy software and FOSS to the cloud.

1. Introduction

1.1. The Challenge

Frequently, organizations inherit source code written by a development team, which has long since moved on to other projects. Without fail, business requirements drive software modifications due to market evolution and developing, competitive business strategies. One such example is the current trend by many industries and organizations realizing increased business value and decreased mission risk due to cloud adoption. In organizations which have determined distributed data services are the optimum business policy, the team responsible for application transition must analyze the environment pre-migration to determine the necessary plan of actions and milestones for secure transition. A post-migration analysis is then necessary to validate all associated information assurance controls.

For the case study considered in this paper, the transition team has access to the source code and some of the original documentation. The legacy software suite consists of an internally developed Java application front end, and Apache Tomcat 8 with a SQL Server backend which come together to provide a web application accessible to anyone who has a vetted and approved, password protected account and compatible web browser. The intent of the system is to receive anticipated satellite bandwidth usage data input from a base of users located around the world and produce automated reports documenting satellite throughput. The ultimate goal is to ensure effective use of satellite resources and project when additional satellites will be required to meet increased demand.

Additionally, during the threat analysis process (discussed in further detail in subsequent sections), the transition team responsible for moving the application to the cloud becomes aware that three new cybersecurity controls are required along with the cloud transition: multi-factor authentication, approved encryption algorithms for data storage/transmission, and validation by the application of input fields prior to submitting to the SQL database. For the purposes of this paper, multi-

larry.llewellyn.ii@gmail.com

factor authentication will be a two-factor capability implemented with a Common Access Card (CAC) supported by an existing Public Key Infrastructure (PKI).

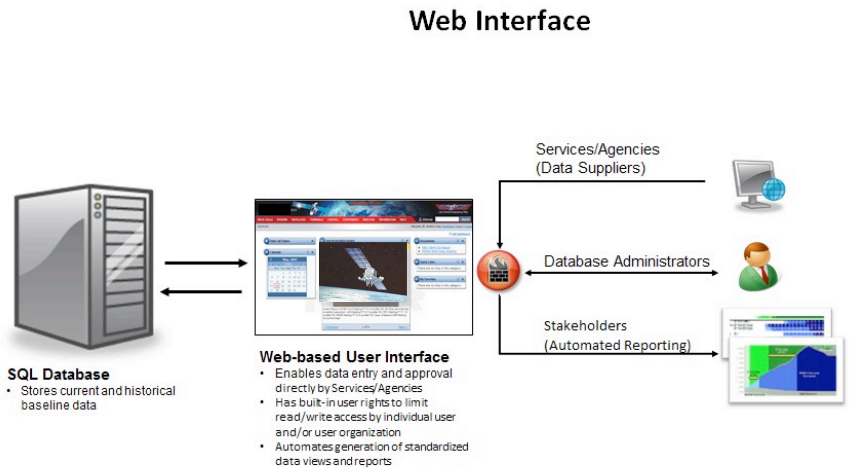

Figure 1

1.2. COMMON SOFTWARE SOURCES AND DEVELOPMENT TRENDS

The term legacy means many things to different people. For the purposes of this paper, legacy means any software currently in use by a given information system, which has been in use for five or more years and was developed using techniques prior to current methodologies. Additionally, here the legacy software source code is available, documentation is limited, and the original developers are no longer available to provide continuity and design consultation.

The Open Organization defines open source software as "software whose source code is available for modification or enhancement by anyone" ("What is open source software? | Opensource.com," n.d.). Source code is the human readable computer instructions written in a particular programming language (in this case C for Apache and Java for the web app), which is then compiled into machine executable files, or object code. In the traditional proprietary, closed-source software environment, it is the object code (or executable) that the consumer purchases. By comparison, object code of a program is very difficult to interpret by a human relative to its source code

larry.llewellyn.ii@gmail.com

counterpart. Open source therefore facilitates modification whereas closed source software does not.

The typical open source software development model is decentralized compared to the model used by developers of proprietary software. More specifically, contributions to code development are less organized with open source software projects. Further, the way the software is patched, bundled, and the roles played by participants may also differ for each approach. There are even differences among the two philosophies. Every FOSS project has a slightly contrasting development model; however, in general the practice includes open sharing, modification (improvement), and redistribution. Many challenges are common in both FOSS and closed source (or proprietary) models. Quality is a concern in both development philosophies as is security.

1.3. COMMON SOFTWARE SECURITY CHALLENGES

All software exists to fulfill a specific task (e.g., write data to disk, read from input devices, start other programs, etc.) that it does only with the permission of the host operating system. In most existing computer programming models, the software is responsible for ensuring that the permitted actions only take place in the manner intended by its developer. History has shown that it is difficult to create software that always functions as intended. Complicating the problem further, most software must process data from external sources. It is this external interaction that exposes most security flaws. This section discusses a few of the most common, dangerous software errors. For a review of the "Top 25 Most Dangerous Software Errors", please see The MITRE Corporation's Common Weakness Enumeration effort ("CWE -2011 CWE/SANS Top 25 Most Dangerous Software Errors," n.d.).

On the topic of security for closed-proprietary versus FOSS, this paper assumes Ross Anderson's answer holds, that "for systems large and complex enough, the attack and defense are helped equally" (Anderson, 2006, p.10). In other words, for software sufficiently large enough, likelihood of exploitation is equal for both FOSS and closed-proprietary software.

larry.llewellyn.ii@gmail.com

The classic buffer overflow problem presents an example of how external interaction can cause an information system to behave unexpectedly. When accepting input from an external source, general programming practice dictates that the programmer ensure data received as input to the data structure is validated prior to load. When the buffer is over-filled, input data has potential to be written into a section of memory containing executable code resulting in unexpected system state upon execution. If the Operating System, application, and CPU programming model are well understood, a malicious actor could force the execution of instructions, which result in privilege escalation or compromise the confidentiality, integrity, or availability of the information system and associated data. Of course, buffer overflow errors are concerns in both FOSS and closed-source software and are the result of poor software coding techniques. The two primary methods of mitigating this type of attack are writing the code with the least privileges required and correct validation of all input.

A new survey from Ponemon Institute found that 84 percent of a respondents stated the most common gateway attack experienced over the past 12 months was SQL injection ("SQL injection has surfaced as the no. 1 attack in 2015," n.d.). Similar to buffer overflow vulnerabilities, SQL injection attacks are the result of trusting non-validated input and granting the application privileges higher than required for successful execution. In this case study, since SQL Server provides the relational database back end, a primary security concern involves the ability of the legacy application to execute sanitization and correct validation of all input.

Broken or antiquated encryption algorithms are another serious software security problem. For example, the Data Encryption Standard or DES was once considered a strong encryption algorithm adopted by the U. S. Government as an official Federal Information Processing Standard. In 1998, a computer built by the Electronic Frontier decrypted a DES-encoded message in 56 hours (Electronic Frontier Foundation, 1998, p. 153); therefore, dependence upon DES for confidentiality of sensitive data is a serious security problem for any software project.

larry.llewellyn.ii@gmail.com

1.4. Legacy software modernization trends

It has been stated that software modernization is more challenging than most software engineers suspect; however, research by the Standish Group International, Inc. shows that only 8 percent of these projects fail (*Modernization clearing a pathway to success*, n.d., p. 4). Additionally, when compared with new application development and commercial-off-the-shelf (COTS) implementation, modernization of legacy code has the highest expected return on investment. It is for these reasons the case study discussed in this paper employs legacy software modernization versus other available options.

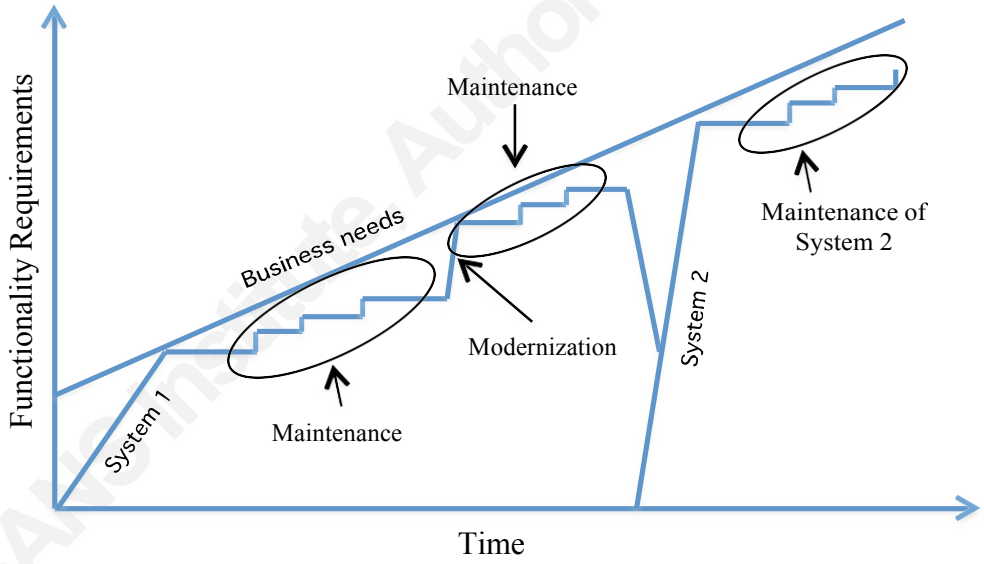

Figure 2

Figure 2 illustrates the information system life cycle (Seacord, Plakosh, & Lewis, 2003, p. 6). As business functionality requirements increase, the information system functionality must also trend upwards or become irrelevant. Less involved maintenance activities (i.e., small, incremental changes such as implementing a new font type) have the ability to improve functionality; however, modernization (i.e.,

larry.llewellyn.ii@gmail.com

extensive modifications such as taking advantage of parallel processing) is required to address functionality requirements. Here the application falls into the System 1 scenario at the modernization slope. As the team works to increase functionality in the application, business functionality requirements grow over time. This cycle continues for the current system until modernization efforts are unable to improve the system and a new system is developed and implemented from the ground up (System 2 in Figure 2).

Complexity is the single greatest challenge to software modernization. This is due to the size and lack of code clarity in most legacy systems. Additionally, modernization efforts must maintain backwards functionality (i.e., continue to satisfy the original set of business requirements while building in additional mission enabling capabilities). Frequent testing and stakeholder engagement will help to mitigate concerns associated with these risks.

2. Transitioning into the Cloud

2.1. Security challenges of the cloud environment

For the purposes of this paper, cloud Infrastructure as a Service (IaaS) is defined using the National Institute of Standards and Technology (NIST) definition found in Special Publication 800-145 dated September 2011. **Figure 3** illustrates IaaS functionality and the traditional private/on-premise datacenter.

To date, only a scant few cloud data breaches have been noted. Jay Heiser, VP of research at technology consultant Gartner Inc., recently stated "Cloud providers put more emphasis on security than other entities. If they didn't, they'd fall over" ("Cloud computing data breaches currently few | Business Insurance," n.d.). Similarly, as cloud increases in popularity and usage, so too will the motivation and innovation of perpetrators who wish to gain from exploiting cloud vulnerabilities.

larry.llewellyn.ii@gmail.com

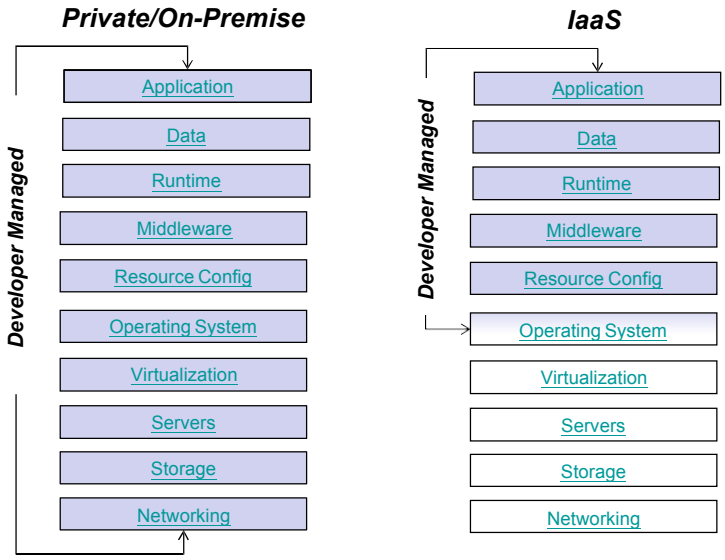

Figure 3

In general, with few exceptions (e.g., satellite command and control, real-time communications, etc.) the "golden nuggets" or highest valued components of most information systems are the sensitive data repositories at the center of the larger information system. In the US, there is no single, comprehensive federal law regulating the collection and use of data. There are federal laws that regulate individual industries (e.g., the health industry's Health Insurance Portability and Accountability Act of 1996 (HIPPA), the financial industry's Gramm-Leach-Bliley Act (GLBA), the Federal Trade Commission Act, etc.) and state laws, which overlap and sometimes conflict with federal laws. Additionally, there are many guidelines developed by industry groups that are part of self-regulatory standards and frameworks that are considered best practices.

The key to successfully navigating these somewhat ambiguous waters lies in a risk based approach to information assurance. The approach starts with system owners fully understanding the mission, information types, network associated with the system, and the applicable laws. Next, the potential impact for each information type processed, stored or transmitted by the system should be assessed from a confidentiality, integrity, and availability perspective. Finally, an evaluation of the

larry.llewellyn.ii@gmail.com

threat sources, threat events, likelihood, and impact feeds into a risk assessment which guides where dollars should be spent to maximize return on information assurance investments.

Every information system has vulnerabilities. Information systems hosted by cloud providers are no different. A primary concern associated with cloud adoption for most candidate organizations is data security. More specifically, storing critical business data into services accessible over the public Internet means that crooks have a target of opportunity with potentially fewer technical challenges. Some additional key concerns which should be evaluated when considering cloud include shared access to cloud resources, virtual exploits, and information system availability.

One key way cloud providers are able to control costs and gain economies of scale is through shared access or multitenancy. Hardware multitenancy means multiple unrelated customers potentially have access to the same CPU, storage, and memory. The risk is that one customer could unintentionally or maliciously gain access to another customer's sensitive data through a multitenancy related flaw. The degree of multitenancy is dependent upon the cloud provider (e.g., from hardware multitenancy to database and application multitenancy) and should be discussed and understood prior to procuring services. Researchers are just beginning to understand many multitenancy security concerns; however, there are many who believe the challenges of multitenancy will be significant as the cloud model matures. Information assurance professionals should pay close attention to this burgeoning field of research.

All cloud providers rely heavily upon virtualization. Virtualization allows cloud providers to maximize resources by abstracting away the underlying hardware and creating multiple Operating System instances on one physical piece of hardware. Virtualization software comes with its own vulnerabilities. According to TechTarget's SearchSecurity.com E-Guide "Top Virtualization Security Risks and How to Prevent Them", the following are the most common attack avenues in the virtual environment: trojaned virtual machines, improperly configured virtual firewalls/networking, improperly configured hypervisor, and data leakage through offline images. The primary defense against vulnerabilities associated with

larry.llewellyn.ii@gmail.com

virtualization is to patch virtualization software as patches become available. Since virtualization software configuration and maintenance is the responsibility of the IaaS provider, details of configuration and maintenance procedures for the virtualization software must be provided by the cloud vendor. Additionally, in most cases, virtualization software updates should be seamless to the customer; however, there are scenarios where virtual machine instances must be rebooted. These events are generally well publicized and in plenty of time ahead of execution to minimize impact to the customer.

Availability is a term often used by service providers to quantify the amount of time an information system is accessible during a given year. For example, 90% or "one nine" availability means the information system has the potential to be down 36.5 days a year. 99% or "two nines" means the information system has the potential to be down 3.65 days per year and so on. Amazon's "Simple Storage Service" or S3 is advertised to provide 99.99% or 52.5 minutes of down time per year; whereas, the DISA milCloud claims 99.999% or 5.25 minutes of down time annually. Both providers' Service Level Agreements back these values. Given that this availability is outside of the customer's control, it's easy to understand why it's such a significant concern.

An even bigger concern is the possibility of critical business data loss due to either an internal cloud issue or a malicious attack. Amazon's S3 claims 99.999999999% of durability ("Amazon S3 Frequently Asked Questions," n.d.). According to Amazon, this translates into an average expected loss of one file every 10,000,000 years. This durability is accomplished by Amazon's process of storing objects across multiple devices across multiple facilities. The service is designed to sustain concurrent datacenter failures by quickly detecting and repairing any lost redundancy. This availability and data durability is likely much better than most private/on-premise data centers; however, the fact that the vendor has primary control over these critical metrics raises concerns.

larry.llewellyn.ii@gmail.com

2.2. Best practices for operating in the cloud

Though there are many challenges of cloud, numerous benefits also exist. Amazon Web Services (AWS) provides many of the same security features traditional IT systems provide. System administration and monitoring keeps operating systems updated with the latest patches. Backup of critical data is a standard service offering among most cloud providers; however, the customer is generally responsible for deploying the necessary tools required by these services. Intrusion detection tools are available to monitor traffic. Some unique security features of cloud are:

- systems are remotely administered across the Internet (no physical access)
- systems are virtual (making procurement and reconstitution a relatively fast process)
- redundancy is often built-in (geographical diversity allowing data to be stored across multiple geographical locations)

2.2.1. Data Access and Encryption in the Cloud

As mentioned previously, unauthorized access to sensitive data is a primary concern in the cloud. One method of ensuring data in the cloud is secure is to encrypt prior to sending to cloud data storage. Using this philosophy, even if thieves were able to gain access to sensitive data, the information would be rendered useless since decryption would not be possible by the wrong doers.

Generally speaking, for encryption algorithms, the security strength is a measure of the difficulty in discovering the key. With a brute force attack (i.e., trying every possible key until the correct one is found), key length could mean the difference between decrypting an encoded message in 10 seconds or 10,000 years. The NIST Special Publication 800-131A recommends using encryption algorithms with key lengths of 112-bits or current FIPS 140-2 compliant encryption algorithms. This level of

larry.llewellyn.ii@gmail.com

encryption should provide the strength necessary to defend against current computing capabilities.

In addition to data encryption, a customer can secure data in the cloud by implementing fine-grained access control. Fine-grained access control is the ability to control who can access what discrete data items and attributes in the database tables and indexes along with the actions that can be performed on them. Using fine-grained access control features provided by AWS, the application can hide information in the database table in a horizontal fashion by matching user attributes with primary key values.

2.2.2. Two-Factor Authentication in the Cloud

Two-factor authentication is defined as something you have (e.g., a token or a Common Access Card (CAC)) and something you know (e.g., a personal identification number (PIN) or a password). As of the writing of this paper, AWS just released its multi-factor offering to its customers ("Multi-Factor Authentication," n.d.). Amazon also offers multiple multi-factor authentication solutions (e.g., tokens, one-time-password via Short Messaging Service, etc). For the purposes of this paper, the two-factor authentication solution will use a CAC and a six digit PIN. More specifically, here two-factor implementation involves the Department of Defense (DoD) PKI solution. Two-factor authentication is neither a panacea nor invulnerable; the successful 2011 cyber-attack on RSA demonstrated this fact. Additionally, before resources are invested implementing two-factor, risk analysis should be performed to insure this authentication solution is warranted. For example, if the true mission risk is the insider threat, two-factor authentication by itself will do little to mitigate this particular risk.

Given that username and password is the most common authentication mechanism used to access cloud resources and the most trivial to hack, implementation of a multi-factor authentication solution increases the difficulty involved in gaining unauthorized access. Moreover, since the DoD PKI solution is a service provided by the milCloud, modifying the application

larry.llewellyn.ii@gmail.com

to use the available PKI application programming interface is relatively straightforward and inexpensive requiring no purchase of additional hardware or software. Challenges do exist with migrating current users from username/password authentication to PKI. These challenges are primarily logistical not technical (i.e., communicating the migration with the users and ensuring users have the required CAC hardware, application, and middleware).

2.2.3. Data Input Sanitization and Validation

Sanitization involves running externally submitted input through a data parser that disallows all characters except those that are specifically relevant. As previously discussed, validation attempts to ensure that received data is in the form that is expected. These functions must be tested and verified prior to operational cloud deployment.

Though the application performs some data input sanitization and validation, additional rigor must be implemented to insure recently discovered threats are addressed and associated risks are mitigated. Moving forward, the application will implement a whitelist of acceptable input that strictly conform to predetermined conditions, and where possible prepared or parameterized statements will be used. For validation, all potentially relevant input will be considered which coincide with business rules. Additionally, the application will be run using the lowest privileges that are required to accomplish the necessary tasks.

2.2.4. Managing Cloud Adoption Risk

As previously discussed, transitioning legacy code from an on premise datacenter to the cloud has inherent risks. Due to code complexity and unknown system interdependencies, hidden technical challenges, which are not discovered until code is deployed into the cloud, are often a reality. To buy down this risk, cloud providers offer a “sandbox” which is analogous to the operational cloud except for the fact that non-authorized, external entities are unable to access the sandbox without explicit authentication credentials

larry.llewellyn.ii@gmail.com

and required software (e.g., specific Virtual Private Network client software and unique soft-certs). This resource is where developers can learn, design the environment, and install/configure all necessary software so that functional and cybersecurity testing can be planned and executed. During this testing, technical challenges can be exposed and resource estimates developed. In summary, cost benefit analysis can be performed with very little investment to determine if cloud adoption of this particular application is feasible.

2.2.5. Cloud Service Level Agreement

A key piece of the overall security posture for cloud is the agreement, or Service Level Agreement (SLA), between the customer and the cloud provider. This agreement documents the services provided by the cloud vendor to include roles and responsibilities in support of information assurance and the logical and physical points of demarcation. The SLA should also document the Incident Response Plan used to report security incidents and outages and establish reporting channels and associated points of contact to help facilitate an appropriate and prompt response. It is also vitally important that this document be reviewed at least annually to determine if any modifications or amendments are necessary to reflect the customer's requirements and the cloud provider's services.

2.3. Assessing and Addressing Security challenges

The application in this scenario was developed for the benefit of the satellite communications community, as an integrated tool to synchronize the acquisition and deployment of satellite assets. It reduces the risk of systems being fielded in an uncoordinated manner by correlating schedules, budgets, satellite terminal fielding plans, and delivery plans. The system processes, stores, and disseminates the following information types: budget, strategic planning, system maintenance, information security, and system/network monitoring. All information types are sensitive but unclassified (SBU) and have been assessed by the stakeholders as low impact should confidentiality, integrity, or availability be compromised; however,

larry.llewellyn.ii@gmail.com

when aggregated with other SBU information (e.g., enterprise architecture, system capabilities, etc.) this data could be elevated to medium impact should confidentiality of strategic planning information be compromised.

Based on the system's mission, associated information types, known and potential vulnerabilities, threat research, and analysis, it has been determined that the primary adversaries of system data are nation-state sponsored bad actors external to the United States. Out of an abundance of caution, based on data aggregation risk and the likely threat, it has been determined that Federal Information Processing Standard (FIPS) Publication 140-2 compliant encryption must be used for data storage and transmission. Additionally, to eliminate vulnerabilities incurred with username/password authentication, two-factor authentication shall be used to secure access to the application

2.4. "Cloudification" and the Security Implications

Along with the ability to transfer the cost of datacenter hardware technology purchase, installation, and sustainment; the cloud also brings the ability to quickly add or remove IT capacity to meet increasing or decreasing demand. The catch to taking advantage of cloud IT elasticity is that applications must be built to scale vertically or horizontally as resources become saturated. Horizontal scaling involves adding more instances of the virtual machine to the infrastructure. Vertical scaling involves standing up instances with more RAM and CPU capacity. This capability brings with it a significant weapon against distributed denial of service (DDoS) attacks.

2.4.1. Thwarting Distributed Denial of Service Attacks with Elasticity

The key to being prepared for Distributed Denial of Service (DDoS) attacks is to develop a response strategy. Most cloud providers offer multiple services that combine to significantly improve resiliency against this type of assault. The goal of DDoS attacks is to exhaust resources resulting in rendering the information system unusable to legitimate users. DDoS attacks can be categorized as application, protocol, or bandwidth exhausting. Identifying an attack is accomplished by first understanding the

larry.llewellyn.ii@gmail.com

systems baseline traffic metrics. Once this is established, alarms can be created to alert on potential attacks or other anomalies.

When an attack is believed to be underway, additional distributed server instances may be scaled up both horizontally and vertically to absorb the additional traffic. Note that most cloud providers will obviously charge for the addition of new server instances; however, the DDoS can be dispersed over a wider area forcing the attackers to expend additional resources. An example of this is the AWS Availability Zones (three in the Asian Pacific, two in the European Union, one in South America, and three in the US). With this approach, it is possible to prevent loss of service to legitimate customers. Though this is a key cloud security feature that should be considered in the overall security design, threat analysis has shown the system is not a likely target of DDoS attacks; however, this option could be exercised in the future should the need arise.

2.5. Operational Transition (securely, and seamlessly moving users and data)

Having gained familiarity with cloud resources, installed/configured the application, implemented source code security improvements, and resolved all functionality issues, the next step is to transition operations from the on premise datacenter to the cloud. This part of the project requires careful planning and execution.

2.5.1. Migrating Data to the Cloud Securely

As discussed in the "Assessing and Addressing Security Challenges" section of this paper, moving sensitive data to the cloud securely requires understanding threats to confidentiality, integrity, and availability of the data. Adherence to the applicable data-related laws and regulations is also of the utmost importance. To work through these issues, one must first understand the content and context of the data to be moved. This requires listing all information types associated with the system and understanding the confidentiality, integrity, and availability (CIA) implications of each

larry.llewellyn.ii@gmail.com

based on mission use cases, as previously discussed. Understanding who has access to what data and the CIA associations allows decision makers to balance risk with information assurance control investments for the movement of data to the cloud and for the overall operational system.

Protecting data as it moves into the cloud is a straightforward matter of implementing a data-in-motion encryption algorithm such as Secure Sockets Layer (SSL) or a Virtual Private Network (VPN) with Internet Protocol Security (IPSec). For the system in this scenario, since the milCloud requires a Cisco VPN client to access provisioned images, the Cisco client with FIPS 140-2 compliant encryption will be employed.

2.5.2. Validating Functionality

With the data securely in the cloud and the requisite software installed, the process of initial operational testing can begin. For the system in this scenario, the team has procured a sandbox with a server and several additional virtual machines which function as the operational user clients during initial testing. This configuration allows for functional testing without opening the application up to be accessible by the world prior to information assurance testing. The client virtual machines will execute all possible standard use cases to insure the system meets the intent, correctly responds to various inputs, and achieves the results operational stakeholders require. Upon completion of functional testing, the system's virtual machine image can be saved for future security testing.

2.5.3. Validating Security Controls

To insure the system meets essential information assurance requirements, the system will be scanned and assessed using a tool which analyzes the registry, services, and file structure pre and post install. This way, a comparison can be made of the baseline operating system image before and after installation revealing any resulting registry vulnerabilities, added/modified or deleted services, added dynamically linked libraries, and the location of the installation. The WireShark network traffic packet

larry.llewellyn.ii@gmail.com

capture tool will be up and running prior to and after installation. This will allow for comparison of network traffic showing any new network connections or listening ports. Additionally, execution of WireShark during operational testing will allow validation of implemented encryption algorithms as well as PKI functionality.

A SQL Injection scanner will be used to verify effective sanitization and validation of all inputs. One such common tool is Hewlett Packard's WebInspect. By creating a macro with valid logon credentials, WebInspect can access the web application under test and attempt to develop an initial analysis and fingerprint of the technologies used. Once verification of the initial analysis is complete, WebInspect is ready to run through a targeted set of the SQL Injector tool data injection and manipulation checks. Another common commercially available SQL inject scanner is Acunetix Ltd.'s Web Vulnerability Scanner. The Acunetix scanner functions similar to WebInspect; however, Acunetix provides additional details for each vulnerability found (e.g., dynamics of the test performed to discover the flaw, including the http headers exchanged during the test). Once SQL injection vulnerabilities are discovered, they should be verified, remedied, and rescanned to insure discovered flaws are resolved.

2.5.4. Operational Testing and Final Transition

With the data securely stored in the cloud, all initial functional testing complete, and information assurance controls validated, full functional testing from user clients outside of the sandbox can be accomplished. In this portion of the project, public access to the application will be tested for the first time. During operational testing, the focus will need to be on the performance of the application in the new environment when under various loads. Additionally, the capacity of supporting services (e.g., cloud bandwidth, disk input/output, PKI, etc.) to perform as required will also be established.

larry.llewellyn.ii@gmail.com

2.5.5. Risk Acceptance

Risk acceptance is the final stage in the cloud transition process. Having assessed known threats to system data, implemented mitigations against those threats, and tested the performance and security of the application in the cloud, a clear picture of likelihood and consequence for known risks can be achieved. With this risk picture, the leaders responsible can be confident in their decision to support the final product and have empirical data to back their position.

3. Conclusion

This paper investigates existing methods, and proposes a practical approach, for mitigating risks associated with transition of legacy and free and open source software to the cloud. This approach includes the following milestones:

Item #	Milestone	Description
1	Mission System Identification	Understand the system's purpose, hardware, software, configuration, applicable laws, and use cases.
2	System Information Assessment	List and assess the potential impact for each information type should confidentiality, integrity, or availability be compromised by known threats.
3	Assess Target Environment	Analyze vulnerabilities and security features associated with candidate cloud vendor.
4	Determine Information Assurance Controls	Based on the system information assessment and target environment analysis, perform cost benefit study of relevant information assurance controls.
5	Implement Necessary Security Controls	For the system in this scenario, this step involved employing two-factor authentication, FIPS 140-2 compliant encryption algorithms, and data input

larry.llewellyn.ii@gmail.com

		sanitization and validation.
6	Data Migration	Data was transferred from the on premise data center to the cloud using the Cisco AnyConnect VPN client with FIPS 140-2 compliant encryption.
7	Validation	To ensure the system meets the functional intent and achieves the required security goals, the application was first tested in the cloud sandbox environment with applicable mission threads. Then a method of test was developed and executed for the relevant security controls.
8	Risk Acceptance	Item # 7 provides the system owner with empirical data supporting the acceptable information assurance risk posture of the application.

larry.llewellyn.ii@gmail.com

References

References

Amazon S3 Frequently Asked Questions. (n.d.). Retrieved from https://aws.amazon.com/s3/faqs/

Anderson, R. (2006). *Security in open versus closed systems – The dance of Boltzmann, Coase, and Moore*. Retrieved from Cambridge University website: https://www.cl.cam.ac.uk/~rja14/Papers/toulouse.pdf

Cloud computing data breaches currently few | Business Insurance. (n.d.). Retrieved from http://www.businessinsurance.com/article/99999999/NEWS070101/39999

CWE -2011 CWE/SANS Top 25 Most Dangerous Software Errors. (n.d.). Retrieved from http://cwe.mitre.org/top25/

Electronic Frontier Foundation. (1998). *Cracking DES: Secrets of encryption research, wiretap politics & chip design*. San Francisco, CA: Author.

Modernization clearing a pathway to success. (n.d.). Retrieved from The Standish Group website: https://www.standishgroup.com/sample_research_files

Multi-Factor Authentication. (n.d.). Retrieved from https://aws.amazon.com/iam/details/mfa/

Seacord, R. C., Plakosh, D., & Lewis, G. A. (2003). *Modernizing legacy systems: Software technologies, engineering processes, and business practices*.

SQL injection has surfaced as the no. 1 attack in 2015. (n.d.). Retrieved from http://net-security.org/secworld.php?id=19220

Top virtualization security risks and how to prevent them. (n.d.). Retrieved from http://searchcloudsecurity.techtarget.com/tip/Top-virtualization-security-risks-and-how-to-prevent-them

larry.llewellyn.ii@gmail.com

What is open source software? | Opensource.com. (n.d.). Retrieved from https://opensource.com/resources/what-open-source

larry.llewellyn.ii@gmail.com