Information and Systems Security Project

Project Title			
Client	WAEC	**Email**	Desmond.chenik@waec.wa.gov.au
Project Team	Ramya Natarajan	**Project Supervisor**	Dr. Denise Gengatharen
Date	Nov 2011		

Western Australia Electoral Commission

Information and System Security Project

Future Internet Security

Final Project Report

Business Analyst:

Ramya Natarajan

Nov 2011

Table of Contents

Introduction:

The report covers the latest threats and vulnerabilities of the Internet and the proper mitigations with examples explained in detail in the appendix. The report also covers e-voting technology and the number of issues faced by the countries, which have adopted them.

Recent developments in information systems have dragged technology towards the cloud; this report lists cloud security threats and suitable recommendations. Initially, the report was planned by developing a comprehensive Risk Template for WAEC. However, WAEC was already developed a risk template and the scope of this project was changed to cover only the latest security issues that could be relevant to WAEC and which we assume are not addressed in WAEC's existing risk template.

Business Values:

- This report helps the WAEC in securing their Information and information technology assets from these 'unconsidered' unexpected threats in the near future.
- It clearly describes the weakness and loopholes in security relating to information technology and information that an organization can face from these new threats. Proper mitigation helps to reduce the damage or outage of systems.
- The report also covers brief information about the cloud environment and future threats. It helps WAEC in getting a clear idea about the risks in cloud services and also the ways to eradicate them.
- This report will also provide WAEC with brief information about e-voting issues in other countries/implementations.

Road to Software Security:

In the past software development, requirements and performance were the main priorities in application design. Recently the priority has changed, not only because of various threats, but also because of potential liability as well as regulatory compliances. Security has become the limiting factor in the broad implementation of web services. As a result, software developers focus on developing high-level security standards and protocols to overcome the issues faced by the organization, but in many cases the simplest application level attacks have been neglected. However it is a common observation from the client side, that web developers hardly take any preventive steps to secure their web applications.

Most of the time web application developers emphasize only on authentication and authorization to secure web applications. This may be a viable approach for designing an intranet application. However, for Internet applications, multiple programming practices need to be followed to prevent future attacks.

Top Cyber Security Attacks:

According to a recent article published in the SOA world magazine (Punjabi, 2011), the common vulnerabilities of web applications were listed as follows:

Bypassing input validation

The above topic is considered as the top most vulnerability in web application.

Web developers often validate user input forms by using JavaScript validations. Once the information is directed to the server side, developers do not validate again, because they assume that JavaScript validations can block all invalid or malicious data.

But here the hackers use the weakness of the developer; they simply save the page to their local hard drive and alter the JavaScript to bypass the validation allowing the server to get compromised and information to be accessed.

Mitigation:

The best way to mitigate the vulnerability is by validating the input twice - first on the client side and then on the server side.

Client-side validation - using JavaScript

Server-side validation - using server side technology like Java, DOT NET, PHP

So, by using the both server and client-side methods WAEC can get the better of the two: *fast response, more secure validation and better user experience*.

SQL Injection attacks:

The most common and dangerous attack is SQL INJECTION ATTACK.

Definition: *It is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution*

SQL statements should be tested for injection vulnerabilities because SQL Server will execute each and every part of the valid queries that it receives so there is a chance of data loss. An experienced and determined attacker can manipulate even the parameterized data to create attacks.

SQL injection results in:

- Loss of confidential data
- Loss of data integrity
- Loss of information
- Compromise of the entire network

The technique behind the SQL injection is a pretty simple one. If the application receives the user data as an input, there is a chance for a malicious user to enter the crafted data in to the input that makes the application interpret it as an SQL query instead of data.

The following script shows a simple SQL injection,

SELECT * FROM Users WHERE Username='$username' AND Password='$password'

This SQL statement is designed to show all the records from the table "Users" for a username and password supplied by a user. Here, the malicious user creates the SQL injection by using the small technique

1' or '1' = '11' or '1' = '1

Now the query results to,

SELECT * FROM Users WHERE Username='1' OR '1' = '1' AND Password='1' OR '1' = '1'

Now the hacker has effectively injected a whole OR condition into the authentication process. Worse, the condition '1' = '1' is always true, so this SQL query bypasses the authentication process and creates the attack. A hacker can potentially delete entire tables, or change the data in the application. Some of the other characters like ‘;’ and ‘- -‘also play a vital part in creating SQL injections.

As long as the injected code is syntactically correct, tampering cannot be identified. The only way is to validate all users’ input carefully and review the code properly.

Mitigation:

Data sanitization and validation are the only way to prevent SQL injection attacks.

Data sanitization usually involves ensuring that the submitted SQL data does not have any dangerous characters like " ' " and ‘- -‘etc.

Validation should be carried out in two ways:

- By blacklisting dangerous or unwanted characters and also by white listing some characters by designing to allow them in certain circumstance. These procedures need to be done by the program developer and consumes lots of time and work
- Though validation is done on the client side, hackers can modify or get around this, so it is fundamental to validate all data on the server side as well.

But sanitization and validation are far from the whole story. Here are ten ways you can help prevent or mitigate SQL injection attack (Rubens, 2010):

Never trust the code: Assume that all user-submitted data is malicious. So, validate and sanitize each and every query.

Avoid dynamic SQL statements: Instead use only prepared statements, parameterized queries or stored procedures.

Try to Update and patch: Hackers can exploit the vulnerabilities in applications and databases by using SQL injection. So it is better to apply patches and updates as soon as practical.

Firewall: Consider a web application firewall (WAF) – either software or appliance based – to help filter out malicious data. A WAF can be particularly useful to provide some security protection against a particular new vulnerability before a patch is available.

Reduce attack surface: Remove the unwanted database functionality that the organization does not need to prevent hackers taking advantage of it.

Don't divulge more information than you need to: These days hackers are learning a great deal of information about database architecture from the error messages it generates, so we have to ensure that we display only minimal information. Use the "RemoteOnly" customErrors mode to display verbose error messages on the local machine, ensuring that an external hacker gets nothing more than the fact that his actions resulted in an unhandled error.

Unprotected Resources

Another threat, which was recently added to the list, is targeting unprotected resources. Nowadays, hackers simply guess the URLs of unprotected resources. For example, by reading the code comments or by assumption they can create cyber attacks.

The unauthorized intruder uses a number of techniques to gather information on applications and also on the networks. For example,

- ➢ Network services that are running but not being utilized.
- ➢ Default user accounts with no passwords specified.
- ➢ Active Guest accounts.

Mitigation*:* The Web content should be protected by authentication.

If it is the case of a Java web application program try to make sure that all the unprotected and sensitive codes are under WEB-INF Directory. A similar solution exists for PHP and other server-side technologies as well. Detailed information about the WEB-INF directory is elaborated in the Appendix.

Reverse Engineering:

Though programmers and developers develop a high-level security environment to control threats and vulnerabilities everyday new attacks are being identified. Another targeted attack is Reverse engineering. It is nothing but the modification of the program that can be done according to the reverse engineer’s wishes.

With the help of some software available in the market, the attackers will decompile the entire source code and gain sensitive information.

Mitigation**:**

The client-side code should not contain any business logic and business logic validation. The code should be encrypted before sending to the client.

Attackers use specific keywords to access the particular code in order to modify the data. However, due to obfuscation, the code gets renamed and makes the attacker's job much more tedious to gain access. There are also other methods and techniques to mitigate reverse engineering for example by using native methods, digital watermarks, etc. (for more detailed information refer to the Appendix).

Weak Authentication:

Most of the time, attackers gain access to a secure website by using common terms like 'admin,' 'test,' etc. Program developers often use these names and passwords for testing purposes and forget to remove them from the production systems.

***Mitigation*:**

Developers should not be given access to production databases for testing purposes. All testing must happen in UAT (User Acceptance Testing) and they should use real user names and passwords (see Appendix).

Cross-Site Scripting (XSS)

When we open two websites in two different browser tabs, we never expect one website on a given tab to steal our passwords from another tab.
However, this is possible, if we are using an old version of the browser or if we are using an infected browser

***Mitigation*:**

Encourage users to upgrade to the latest version of the browsers. Also technologies that use secure sandboxing such as Java Applets and Adobe Flex and many others should be used for creating rich client applications.

Other Web Related Attacks

Zero day vulnerability attacks:

According to the definition given in Wikipedia "*A zero-day (or zero-hour or day zero) attack or threat is a computer threat that tries to exploit* application vulnerabilities *that are unknown to others or the software developer*"

Usually, the client side programmer or customer detects the security issue in the software program and will notify this to the software company. In time, the software company will fix the code and distribute the patch or software update. Even if potential attackers hear about the vulnerability, it may take them some time to exploit it; meanwhile the fix will hopefully become available first. But this is not the case in the zero-day.

Here, the hacker may be the first to discover the vulnerability. Since the vulnerability is not known in advance, there is no way to guard against the exploit before it happens. Thus once the working exploit of the vulnerability has been released into the wild, particular users with the affected software will be compromised continuously until a software patch is made available or the user takes some form of mitigation.

According to the SANS report released in 2009, *"File Format Vulnerabilities continue to be the first choice for attackers to conduct zero-day and targeted attacks"*. Most of the attacks continue to target Adobe PDF, Flash Player and Microsoft Office Suite (PowerPoint, Excel and Word) software. Specifically, '*vulnerabilities are often found in 3rd party add-ons to these popular and widespread software suites, making the patching process more complex and increasing their potential value to attackers'*.

Mitigation:

- The best approach to protect against zero-day exploits is to follow good security policies in the first place.
- All anti-virus software must be installed and maintained up to date.
- Organizations need to aim to have systems secured 99% from the known vulnerabilities.
- One of the best measures to protect against unknown threats is to employ both hardware and software firewalls in the networks.
- Enable heuristic scanning (a technology which is used to attempt to block viruses or worms that are not yet known about) in anti-virus software.
- Block unnecessary traffic and also block unnecessary access to system resources and services with a software firewall or by using the anti-virus software to detect irregular data.

Insufficient Transport Layer Protection

Insufficient transport layer protection allows communication to be exposed to untrusted third parties, enabling an attack vector to compromise a web application and/or steal sensitive information. Typically, websites use Secure Sockets Layer / Transport Layer Security (SSL/TLS) to provide encryption at the transport layer. However, unless the website is configured to use SSL/TLS properly, the website may be vulnerable to traffic interception and modification.

Lack of Transport Layer Encryption

When the transport layer is not encrypted, all communication between the website and client is sent in clear-text which leaves it open to interception, injection and redirection also known as a man-in-the-middle/MITM attack. An attacker will redirect the communication in such a way that the website and client are no longer communicating with each other, but instead are unknowingly communicating with the attacker in the context of the other trusted party.

Weak Ciphers

Weak ciphers are vulnerable to attack because of the relative ease of breaking them. Today, all modern browsers and websites use much stronger encryption, but some websites are still configured to support outdated weak ciphers. Because of this, an attacker may be able to force the client to downgrade to a weaker cipher when connecting to the website, allowing the attacker to break the weak encryption.

Mitigation

The server should be configured to only accept strong ciphers and not provide service to any client that requests using a weaker cipher.

Broken Authentication and Session Management

When developers are programming web application based solutions they rarely focus on how the user’s session is managed. Failing to keep this in mind can lead developers to introduce session management vulnerabilities in their applications (Broken Authentication and Session Management, 2010).

Session management vulnerabilities occur when developers fail to protect their users' sensitive information such as user names, passwords, and session tokens. Broken authentication vulnerabilities occur when developers fail to use authentication methods that have been adequately tested and rely on their own, often flawed, method for authenticating users.

These vulnerabilities are very hard for developers to identify on their own due to the far-reaching aspect of the code that handles session and authentication. To prevent these types of vulnerabilities from occurring in applications, developers should first ensure that SSL is used for all authenticated parts of the application.

In addition, verify that all credentials are stored in a hashed form.

As with all prevention methods, preventing these vulnerabilities takes careful planning from the design stage of the application. The following should all be considered:

- Only use the native session management mechanism.
- Use a single authentication mechanism. Do not write your own authentication mechanism.
- Do not allow the login process to happen from an unencrypted page.
- Once a user authenticates, issue them a new session cookie and invalidate the previous session cookie. This will prevent session hijacking attacks from occurring.
- Verify that every page of the application has a logout link that is easily identified by the user.
- Have adequate timeouts for inactive sessions. Shorter is better.
- Verify the user knows their old password before changing their password.
- Do not send credentials (including the user name) over insecure channels, such as email.
- Do not expose session identifiers, such as the session token, in the URL.

Source Code Disclosure

This attack allows a malicious user to obtain the source code of a server-side application. Attackers use source code disclosure attacks to try to obtain the source code of server-side applications.

The basic role of Web servers is to serve files as requested by clients. Files can be static, such as image and HTML files, or dynamic, such as ASP, JSP and PHP files. When the browser requests a dynamic file, the Web server first executes the file and then returns the result to the browser. Hence, dynamic files are actually code executed on the Web server.

Using a source code disclosure attack, an attacker can retrieve the source code of server-side scripts, such as ASP, PHP and JSP. Obtaining the source code of server-side scripts grants the attacker deeper knowledge of the logic behind the Web application, how the application handles requests and their parameters, the structure of the database, vulnerabilities in the code and source code comments. Having the source code, and possibly a duplicate application to test on, helps the attacker to prepare an attack on the application.

An attacker can cause source code disclosure using one of the following techniques:

- Using known source disclosure vulnerabilities
- Exploiting a vulnerability in the application which may allow source disclosure
- Exploiting detailed errors which may sometime include source code
- Using other types of known vulnerabilities, which may be useful for source disclosure.

To prevent disclosure of sensitive application data:

- Use strong encryption to secure the data.
- Authorize each caller prior to performing data access so that users are only able to see their own data

Insecure Cryptographic Storage

Passwords, credit card information and other sensitive data should be encrypted before storing into the database. If the attacker somehow manages to get the data in an organization's database, it is still not completely compromised as it is already encrypted and the attacker will require a long time to decrypt it. This of course only can happen if the data is properly encrypted with strong encryption algorithms and securely store the encryption key.

In certain applications, the developer encrypts the data but leaves the key together with the data. Running brute force algorithm can easily decrypt the encrypted data in order to crack password of the encryption key. The encryption key should always be separated from the application server. It is also recommended to use a hardware key container (HSM – Hardware Security Module) in applications to bring the security level of application to a higher level.

Aurora:

It enters the computer system without our knowledge while installing third party software. It is one of the abundant spyware found on the Internet today, which is really very hard to remove. It is also called "a better internet" also referred to as "browser hijacker". It alters our settings, and pushes advertising to the hijacked pages and search engines etc.

Many files are always in default locations, some files can easily be moved to different locations or change names like many spyware, adware, or popup programs do. If spyware or adware is suspected, check to see if there are files of similar names stored elsewhere. It always better to check task list and also to see what is currently running on the computer. Task manager shows the number of tasks running on the computer. DLL files are not shown in the task list, as DLL files are part of other processes. Similar program names may be running if spyware or adware is on the computer. Always check the location of tasks or processes if you are really concerned.

This is the survey taken by the OWASP shows the top ten application vulnerabilities for the year 2010.

Vulnerability / Risk	2010 rank	2007 rank	2004 rank
Injection	1	2	6
Cross-site scripting (XSS)	2	1	4
Broken authentication and session management	3	7	3
Insecure direct object references	4	4	2
Cross-site request forgery (CSRF)	5	5	
Security misconfiguration	6		10
Insecure cryptographic storage	7	8	8
Failure to restrict URL access	8	10	2
Insufficient transport layer protection	9	9	10
Invalidated redirects and forwards	10		
Malicious file execution		3	
Information leakage and improper error handling		6	7
Invalidated input			1
Buffer overflows			5
Denial of service			9

So far the report covers the Latest Internet threats and suitable recommendations. Now it explains about e-voting technology pros and cons.

E-voting

Broadband connection is playing a vital role in the 21st century. Australia's National Broadband Network is now in progress, which helps to provide fast broadband connection. Most large organizations are now planning to move operation and functionalities online to make them more user-friendly and also to reduce cost. As technology becomes an essential part of our daily life, automated voting systems become an appropriate vehicle to attain a solid and legitimate democratic model. WAEC also has the same idea and has plans to improve the country's election process by moving towards electronic voting. This report helps WAEC to know more about the e-voting pros and cons and also the e-voting issues faced by other countries.

Advantage

Convenience: Due to use of well-designed software and user-friendly system, voters feel free to use this equipment in the minimal time.

Mobility: Voters can cast their votes at home, or any place in which they can get access to the Internet.

Voter participation: Due to the convenience and mobility of the system, people get motivated and interested. This would increase the participation of voters.

Tally Speed: The system is used to calculate the results in a short period.

Less Cost: Though in the beginning, the expense for building the e-voting system would be high, eventually the total expense would be reduced much lower than the paper ballot system.

Disadvantage

Inequality problem: Some people might not have a computer and a broadband connection so they might lose their privilege in voting.

Vulnerable to Security: one of the major weaknesses in the e-voting system is security. So far, there are many kinds of attacks and issues faced by other countries, which are really very hard to prevent.

Denial of Service attack: A denial of service is an attack that prevents legitimate users from using their resources.

E-voting- Worldwide:

There are many critics regarding e-voting systems. Countries like India, Brazil, Belgium and the Philippines are using electronic-voting systems for all kinds of elections namely state and the local. Other countries such as Estonia, Indonesia, Kazakhstan, Nepal, Norway, Pakistan, Russia and the United States are at various stages of piloting or partially using electronic voting and also Internet voting.

On the other hand some countries are moving in the opposite direction. In 2008 the Netherlands, after several decades of increasing use of electronic voting machines, decertified all of its machines and moved back to manual paper based balloting. Germany likewise recently banned electronic voting machines that were already in use, and in Ireland €52 million worth of electronic voting machines were bought but used only for a small pilot project. Furthermore, the use of electronic voting and counting technologies in the United States is deeply controversial and generates fierce debate between advocates and opponents of these technologies.

In general, almost all e-voting systems in the world are antiquated. E voting does exist in various forms throughout the world. While we can cast our ballot for almost everything online from Pop Idol votes to customer surveys, political voting still uses pen and paper for the elections. A handful of countries have experimented with remote, internet-based voting for expatriates and armed forces, but the only country to ever hold a fully digital election - where anyone in the country could vote via the Internet -is Estonia, which started in 2007. In 2011, 15% of Estonia’s population voted via the Internet.

Estonia’s e-voting success story lies in the country’s explicit and effective risk management and in addressing all expected risks by enhancing the capacities of the procurer, carrying out in-depth risk analyses, and endeavoring to generate trust through consistent dialogue and openness. E-voting also involves many institutional risks, namely: political and legal risks. Political risks entail the possibility that e-voting could bring in new voters who otherwise would not participate and that it could worsen the position of those parties whose supporters prefer traditional voting at the polling station. Societal risks include the lack of public acceptance: i.e., since e-voting results cannot be verified by the people themselves, they need to have absolute faith in the accuracy, honesty, and security of the whole electoral apparatus (people, software, hardware).

Why is Internet voting so slowly to take off? Security is one aspect after 100 years of refining the paper-and-pen approach, it’s understandable that some states would be apprehensive about switching to that terrifying, publicly-owned entity that is the internet but as Estonia has shown, with ID cards that also act as public key encryption smart cards, it’s possible to create a secure internet voting platform.

Basic Requirements for Internet voting:

Authorization and authentication: Only authorized persons (legal voters) can vote. Authorization can be done by a trusted authority to people who can show their identity and prove their eligibility. Authentication is the process of validating each voter at the time of casting the vote.

Flexibility: Voters should be able to use different devices such as desktops, laptops, palmtops, mobile phones, and other newly manufactured devices, and different networks such as Ethernet, dial-up connection, and wireless networks. There should be special devices and software assisting disabled voters to cast their vote.

Count ability: Only valid votes are counted.

Anonymity: There should be no link between a particular vote and the person who cast the vote.

Introducing Blind signatures in voting system

It seems a contradiction that the vote should have no information about the voter, while at the same time there should be a way of checking that the vote comes from a legal voter. With the blind signature techniques invented by Chaum (Blind Signatures), anonymity of electronic votes becomes possible. Blind signatures allow an electronic document to be signed without revealing its contents. One way of achieving anonymity is to request a blind signature on a token from the authorization authority. Given the valid identity information of the legal voter, the authority cannot refuse to provide a blind signature. The signed token can then be extracted from the blind signature and sent to the vote counting authority. This authority accepts the vote since the authorization authority has signed it. Since the vote does not have any information about the voter, and the authority signing the vote cannot link the vote with a particular request, the vote can be provided anonymously.

Other Security related papers:

In terms of security for e-voting, some of the more recent suggestions in the academic literature include voting schemes with multiple parties (Abdul Based and Mjølsnes, 2011), partitions of the ballot to provide implicit security (Parakh & Kak, 2010), double-blind identity based encryption (Gray & Sheedy, 2011), a formal, symbolic definition of election verifiability for electronic voting protocols (Kremer, Ryan, & Smyth, 2010). Purdue et al. (2010) developed a threat tree for risk assessment of Internet Voting security. Pan, Hou, & Ansari (2010) developed a cryptographic based security algorithm developed to ensure voters and candidates' confidentiality in E-voting systems

However, some authors also highlight issues with existing e-voting systems e.g., Balzarotti et al. (2010) performed security testing of the electronic voting systems in California and Ohio and found major vulnerabilities in all the systems identified. Esteghari and Desmedt (2010) demonstrated a hack on Helios 2.0 to demonstrate the impact of software vulnerabilities on such cryptographic voting schemes.

Cloud computing:

Cloud computing is a new way of delivering computing resources Cloud computing is a general term for anything that involves delivering hosted services over the Internet. These services are broadly divided into three categories:

- Infrastructure-as-a-Service (IaaS),
- Platform-as-a-Service (PaaS) and
- Software-as-a-Service (SaaS).

Infrastructure as a Service is a provision model; here the organization outsources the equipment like storage, hardware, servers and network related components. The service provider owns the equipment and will being charge for housing, running and maintaining it.
Platform as a Service: Here the organization will rent operating systems, hardware, storage and network capacity through Internet.
Software as a Service: This is the most common form of SaaS where, applications will be hosted by the vendor and made available to customers via the Internet.

Amazon, Salesforce.com, Google Apps, VMware and Rackspace are some examples of leading cloud vendors at present. Telstra is one of the major cloud vendors in Australia to offer on-demand ICT services including software, platform, infrastructure and network. AT&T, CA, Cisco, CSC, Dell, HP, IBM, Microsoft, Oracle, and Verizon among others are companies that will continue to vie for market share in cloud computing.

This new frontier involves serious risks that aren’t obvious to most. In this report we will discuss the risks involved in cloud computing and the responses required to deal with these risks.

Cloud Threats and Vulnerabilities:

Abuse & Nefarious Use of Cloud Computing:

Hackers, spammers and other criminals are gaining anonymous access from cloud services to launch various attacks ("Top Threats to Cloud Computing," March 2010).

Some of the attacks included are password and key cracking; DDOS; malicious data hosting; launching dynamic attack points; building rainbow tables; and botnet command/control.

Mitigation

- Stricter initial registration and validation processes.
- Comprehensive introspection of customer network traffic.
- Monitoring public blacklists for one’s own network blocks.

Insecure Interfaces & APIs

Customers interact with cloud services through interfaces or APIs. Providers must ensure that security is integrated into their service models, while users must be aware of security risks in the use, implementation, management, and monitoring of such services. Reliance on a weak set of interfaces can expose a corporation to a wide variety of security issues related to confidentiality, availability, and password integrity etc ("Top Threats to Cloud Computing," March 2010).

Mitigation

- Analyze the security model of cloud provider interfaces.
- Ensure strong authentication and access controls are implemented in concert with encrypted transmission.
- Understand the dependency chain associated with the API.

Shared Technology Issues

IaaS is based on shared infrastructure (e.g. disk partitions, CPU caches, GPUs, etc.), which is frequently not designed to accommodate a multi-tenant architecture ("Top Threats to Cloud Computing," March 2010).

Overlooked flaws have allowed guest operating systems to gain unauthorized levels of control and/or influence on the platform.

For example, a user may gain inappropriate levels of control over the underlying platform, thus impacting other customers on the shared platform.

Mitigation

- Implement security best practices for installation/configuration.
- Monitor environment for unauthorized changes/activity.
- Promote strong authentication and access control for administrative access and operations.
- Enforce service level agreements for patching and vulnerability remediation.

- Conduct vulnerability scanning and configuration audits.

Data Loss or Leakage

The threat of data compromise is increased in the cloud due to a number of underlying risk and challenges. Examples include insufficient authentication, authorization or audit controls, operational failures, and data center reliability ("Top Threats to Cloud Computing," March 2010).

Mitigation:

- Try to implement strong API access controls.
- Encrypt and protect integrity of data in transit.
- Analyse the data protection at both design and run time.
- Implement strong key generation, storage and management, and destruction practices.

Account or Service Hijacking

An attack method such as phishing, fraud, and exploitation of software vulnerabilities paves the way for account hijacking.

With cloud services, if an attacker gains access to credentials, they can eavesdrop on activities, transactions, and manipulate and falsify data.

Mitigation:

- Try to prohibit the sharing of account credentials information between users and services.
- Leverage the strong two-factor authentication techniques wherever as possible.
- Try to employ the proactive monitoring to detect unauthorized activity.
- Make sure and try to understand the cloud provider security policies and SLAs.

Unknown Risk Profile

The use of cloud services means organizations are less involved with hardware and software ownership and maintenance because the cloud providers will handle everything.

Though it offers multiple advantages, organizations should aware of the issues such as internal security procedures, security compliance, configuration hardening, patching, auditing and logging may be overlooked.

Mitigation:

- Disclosure of applicable logs and data.
- Partial/full disclosure of infrastructure details (e.g., patch levels, firewalls, etc.).

- Monitoring and alerting on necessary information.

Cloud Computing and Virtualization

Each company is on a unique journey to implement new IT services to support its business objectives. As businesses deploy new virtualization, cloud computing, and endpoint architecture, security must be an integral part of the implementation. Threats to the IT infrastructure can impact business operations severely.

Many companies may believe that they can deploy their current physical security solutions in their consumerized, virtualization and cloud computing environments. However, migrating traditional security approaches to new system infrastructures creates real performance and service issues. In addition, traditional physical security does not address the security risks unique to virtual and cloud computing environments.

Some Cloud Security related papers

Takabi et al. (2010) provide an overview of security and privacy issues in cloud computing, while Grobauer et al. (2010) define indicators based on sound definitions of risk factors and cloud computing (2010). Hay et al. (2011) discuss complex security challenges that are introduced by the trend towards Infrastructure as a Service (IaaS)-based cloud computing. Virvilis et al. (2011) provide a good summary on the available infrastructure and architecture for secure cloud storage. Kong (2010) identifies the possible solutions to protect the confidentiality of virtual machines against untrusted host. Maggi & Zanero (2011) discuss about the future web security and solutions for new-old security issues and measures. Zissis and Lekkas (2011) propose a high-level electronic governance and electronic voting solution, supported by cloud computing architecture and cryptographic technologies.

Appendix:

Definitions

WEB-INF directory: -

The root directory of the application is called document root. Root directory contains a directory named WEB-INF (Vispute, 2011). The information other than the WEB-INF directory is publically available, and can be retrieved by URL from the Internet. WEB-INF directory is a private area of the web application; any files under WEB-INF directory cannot be retrieved directly from the Internet. However the content of the WEB-INF directory is accessible by the classes within the application. To prevent access to the certain resources like JSPs or HTML documents from Internet, we can keep it under WEB-INF directory.

Reverse engineering:

Some other mitigation techniques,

- One of the clear ways to protect JAR files is to simply not allow unauthorized persons to read, modify, or download the files (Moore, 2005).
- Another technique is using native methods. It is nothing but writing in a platform-specific language such as C that can be accessed by a Java class. We can either create native source code, or compile the entire JAR package into the native machine code.
- One more technique is attaching a *digital signature* to the JAR file. This can be used to validate the specific JAR file
- *Digital watermarks* are a type of unique identifier that proves the true developer of the software. It does not protect our code from intrusion, but it will help to prove the appropriate vendor. A tool called jmark helps to create digital watermarks on Java class files.

Data sanitization: The records in testing environments must be sanitized in order to protect important business information.

Heuristic scanning: It is use to detect new, unknown viruses from the system. This method is used to examine the instruction piece-by-piece so that it differentiates the virus from the normal programs.

HSM: A Hardware Security Module is a hardware-based security device that generates, stores and protects cryptographic keys.

Bibliography

Aurora.exe. (n.d.). From IObit: http://www.iobit.com/exedll/aurora-exe.html

Balzarotti, D.; Banks, G.; Cova, M.; Felmetsger, V.; Kemmerer, R.; Robertson, W.; Valeur, F.; Vigna, G.;, "An Experience in Testing the Security of Real-World Electronic Voting Systems," *Software Engineering, IEEE Transactions on* , vol.36, no.4, pp.453-473, July-Aug. 2010
Doi: 10.1109/TSE.2009.53
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=5210119&isnumber=5532334

Based, A., & Mjolsnes. (2011). *Algorithms and Architectures for Parallel Processing.* Springer Berlin / Heidelberg.

Blind Signatures. (n.d.). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Blind_signature

Broken Authentication and session Management. (2010 йил 22-April). From The open Web Application Security Project: https://www.owasp.org/index.php/Broken_Authentication_and_Session_Management

Dimitrios, Z., & Dimitrios, L. (2011) Securing e-Government and e-Voting with an open cloud computing architecture, *Government Information Quarterly*, 28(2), 239-251
URL: http://www.sciencedirect.com/science/article/pii/S0740624X10001383

Estehghari, S., & Desmedt, Y. (2010). Exploiting the client vulnerabilities in Internet Voting System. *Paper presented at the 2010 Electronic Voting Technology Workshop*, University College London, Tokyo, Japan

Gray, D., & Sheedy, C. (2011). *Public Key Infrastructures, Services and Applications* (Vol. 6711). (J. Camenisch, Ed.) Springer Berlin / Heidelberg.

Grobauer, B.; Walloschek, T.; Stocker, E.; , "Understanding Cloud Computing Vulnerabilities," *Security & Privacy, IEEE* , vol.9, no.2, pp.50-57, March-April 2011
doi: 10.1109/MSP.2010.115
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=5487489&isnumber=5739630

Hay, B.; Nance, K.; Bishop, M.; , "Storm Clouds Rising: Security Challenges for IaaS Cloud Computing," *System Sciences (HICSS), 2011 44th Hawaii International Conference on* , vol., no., pp.1-7, 4-7 Jan. 2011
doi: 10.1109/HICSS.2011.386
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=571900

3&isnumber=5718420

Haijun Pan; Hou, E.; Ansari, N.; , "Ensuring voters and candidates' confidentiality in E-voting systems," *Sarnoff Symposium, 2011 34th IEEE* , vol., no., pp.1-6, 3-4 May 2011
Doi: 10.1109/SARNOF.2011.5876452
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=5876452&isnumber=5876433

Jinzhu Kong;, "Protecting the Confidentiality of Virtual Machines Against Untrusted Host," *Intelligence Information Processing and Trusted Computing (IPTC), 2010 International Symposium on* , vol., no., pp.364-368, 28-29 Oct. 2010
Doi: 10.1109/IPTC.2010.11
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=5663256&isnumber=5662254

Maggi, F.; Zanero, S.;, "Is the future Web more insecure? Distractions and solutions of new-old security issues and measures," *Cyber security Summit (WCS), 2011 Second Worldwide* , vol., no., pp.1-9, 1-2 June 2011
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=5978790&isnumber=5978775

Moore, L. (2005). From Reverse-Engineering Java Class Files and Common Mitigation Techniques: http://www.issa.org/Library/Journals/2005/November/Moore%20-%20Reverse-Engineering%20Java%20Class%20Files.pdf

Parakh, A.; Kak, S.;, "Internet Voting Protocol Based on Implicit Data Security," *Computer Communications and Networks, 2008. ICCCN '08. Proceedings of 17th International Conference on*, vol., no., pp.1-4, 3-7 Aug. 2008
Doi: 10.1109/ICCCN.2008.ECP.140
URL: http://0-ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=4674300&isnumber=4674160

Punjabi, M. k. (2011). *Common Web Application Security Vulnerabilities and Mitigation.* From SOA World Magazine: http://soa.sys-con.com/node/2013759

Pardue, H.; Yasinsac, A.; Landry, J.; , "Towards Internet voting security: A threat tree for risk assessment," *Risks and Security of Internet and Systems (CRiSIS), 2010 Fifth International Conference on* , vol., no., pp.1-7, 10-13 Oct. 2010
Doi: 10.1109/CRISIS.2010.5764925
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=5764925&isnumber=5764913

Rubens, P. (2010 йил 23-February). *10 way to prevent or mitigate sql injection.* From Enterprise Networking Planet: http://www.enterprisenetworkingplanet.com/netsecur/article.php/3866756/10-Ways-to-Prevent-or-Mitigate-SQL-Injection-Attacks.htm

Takabi, H.; Joshi, J.B.D.; Ahn, G.;, "Security and Privacy Challenges in Cloud Computing Environments," *Security & Privacy, IEEE*, vol.8, no.6, pp.24-31, Nov.-Dec. 2010

Doi: 10.1109/MSP.2010.186
URL: http://0ieeexplore.ieee.org.library.ecu.edu.au/stamp/stamp.jsp?tp=&arnumber=5655240&isnumber=5655229

Top Threats to Cloud Computing. (2010 йил March). From Cloud Security Alliance: https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf

Transport Layer Protection Cheat Sheet. (2011 йил 16-October). From The Open Web Application Security Project: https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet

Vispute, R. (2011). *WEB-INF directory structure.* From http://www.bestdesigns.co.in/blog/web-inf-directory-structure

Virvilis, N., Dritsas, S., & Gritzalis, D. (2011). *Trust, Privacy and Security in Digital Business.* Springer Berlin / Heidelberg.