955 boul. Michèle-Bohec, Suite 244
Blainville (Quebec) J7C 5J6
Canada

TECHNICAL SECURITY AUDIT REPORT
External Servers and Web Applications

October 2015

Presented to:

John Smith
Chief Information Officer
ABC inc.

Presented by:

Above Security Inc.
955 Michèle-Bohec Blvd., Suite 244
Blainville (Quebec) J6C 5J6
Canada

[DEMO REPORT]

Table of Contents

1. EXECUTIVE SUMMARY

1.1. Context

Following the recent Request for Proposal from ABC Inc., it was agreed that Above Security would perform a Technical Security Audit of ABC Inc.'s external IT infrastructure.

The project consists of vulnerability assessment and penetration test of five (5) external IP addresses and two (2) Web applications.

Your expectations as we understand them are:

- To evaluate the permeability of your company's internal network infrastructure and Web applications from computer attacks;
- To increase, according to these risks, prevention measures against computer attacks.

1.2. Disclaimer

As defined by the ISO 27001 Auditing Standards, the following limitations should be considered while procuring for an audit:

- The results of an audit are not representative of the total security posture of a company but rather as the security posture relative to the scope and budget of the audit.
- The auditor will endeavor, in line with the scope and budget of its mandate, to obtain a sufficient level of confidence in the results. To this end, the auditor will use appropriate tools and methods.
- The audit is representative of a particular situation at a particular time.
- While the vulnerabilities can be corrected, it is important to look into the original source of the problem and put in place corrective measures to prevent further occurrence.

Above Security will validate the security of the tested assets as completely as possible within the time and budget included in the mandate. It will always be possible to explore deeper into the performed attacks scenarios and to carry out additional tests. We limit our intervention to a level that is generally accepted in our industry and is consistent with the scope of the mandate.

1.3. Objectives

The objective was to validate, as completely as possible within the scope of budget and time constraints, the target's overall security such as agreed within this mandate's framework. The proposed budget allowed simulating the role of a determined attacker having at his disposal all the required resources to carry out such an attack.

The provided test results should be considered as a starting point to improve:

- Information security;
- Physical access controls;
- Methodologies;
- Processes;
- Training plans;
- Security and management policies.

1.4. Scope

The security audit was performed in two (2) phases:

Phase 1: Penetration test of external servers

Within this phase, a vulnerability assessment was carried out of five (5) IP addresses using the same techniques as an attacker located outside of your network.

The assessment includes the following steps:

- Recognition phase;
- Port scanning and systems identification;
- Services drilling/search;
- Vulnerability identification and validation;
- Vulnerability exploitation and evaluation of associated risks.

The intrusion test verified if these servers were be able to resist hostile attacks without revealing too much information on your environment and if identified vulnerabilities can lead to further intrusion and exploitation. The tests were carried out from Above Security's facilities using open source and proprietary tools.

The tests covered the following IP addresses:

- X.X.X.242
- X.X.X.241
- X.X.X.141 (mail.ABC Inc.com)
- X.X.X.254
- X.X.X.142

Phase 2: Penetration test of two (2) external Web applications

The tests were carried out from Above Security's premises using open and proprietary tools and targeted the Web applications located at:

- www.<client_web1>.com
- www.<client_web2>.com

1.5. Methodology

Intrusion tests were carried out employing the same techniques as an attacker located outside your infrastructure. The intrusion tests verified if the applications and systems will resist hostile attacks without revealing too much information on your environment and if identified vulnerabilities can lead to further intrusion and exploitation.

For detailed activities included in our penetration test methodology, please refer to the appendix.

1.5.1. Risk Evaluation

Above Security evaluates the risks according to the following formula:

Risk = Probability of Occurrence × Impact

Impact:

An impact estimate is specified according to the extent of damage an attacker could cause when exploiting an identified vulnerability. This evaluation is done considering the importance of the security weaknesses and the severity of such vulnerabilities. The severity of vulnerabilities is subjectively evaluated on a scale of 0 to 10:

- A vulnerability with a severity of 10 leads to complete compromising and control of the target;
- A severity of zero means that the exploitation does not lead to harmful results.

Probability:

The vulnerabilities' probability of occurrence is evaluated on a scale of 0 to 10. This evaluation is conducted considering:

- The security and authentication mechanisms in place;
- The necessary resources (knowledge, time, budget) for performing an attack;
- The availability of automated attack tools.

A probability of occurrence of "10" means that the vulnerability is easy to exploit even with an automated tool. A probability of "1" means that the vulnerability is unlikely to be exploited.

1.6. Security Posture

Overall, considering the vulnerabilities identified and the risks associated, ABC Inc.'s security posture of the evaluated components is as follows:

SECURITY POSTURE OF ABC INC.'s EXTERNAL SERVERS		
	A	Security meets or exceeds the industry standards and requires little or no improvements.
	B	Security meets most industry standards but a few improvements are required to reach optimum levels.
▶	C	Security meets certain industry standards. A fair amount of improvements are required to reach optimum levels.
	D	Security meets a few industry standards. Several significant improvements are required to reach optimum levels.
	E	Security does not meet the industry standards.

Table 1: Security Posture of ABC Inc.'s External Servers

SECURITY POSTURE PER EXTERNAL SERVERS	
APPLICATION	**POSTURE**
X.X.X.242	C
X.X.X.241	B
X.X.X.141	C
X.X.X.254	B
X.X.X.142	A

Table 2: Security Posture Per External Servers

SECURITY POSTURE OF ABC INC.'s WEB APPLCIATIONS		
	A	Security meets or exceeds the industry standards and requires little or no improvements.
	B	Security meets most industry standards but a few improvements are required to reach optimum levels.
▶	C	Security meets certain industry standards. A fair amount of improvements are required to reach optimum levels.
	D	Security meets a few industry standards. Several significant improvements are required to reach optimum levels.
	E	Security does not meet the industry standards.

Table 3: Security Posture of ABC Inc.'s Web Applications

SECURITY POSTURE PER WEB APPLICATION	
APPLICATION	**POSTURE**
http://www.<client_web1>.com	C
http:// www.<client_web2>.com	C

Table 4: Security Posture Per Web Application

1.7. Recommendations

Above Security, as global IT security service provider, makes the following recommendation for ABC Inc.'s consideration:

Short-term

- Upgrade all the software versions mentioned in this report;
- Apply input data validation filters to prevent cross-site scripting;
- Secure communications and increase the level of encrypted connections between clients and servers;
- Configure servers properly;
- Ensure sensitive information is never returned;
- Use cookies in secure ways as outlined;
- Protected Sensitive Files;
- Disable SSL2 protocol support;
- Apply the necessary patches

Mid-term

- Make follow-up tests after the corrective implementation.

Long-term

- Continue periodic testing.

2. TECHNICAL SECURITY AUDIT

2.1. Phase Description

The security evaluation was performed in two (2) phases:

Penetration test of five (5) external servers

Within this phase, an intrusion test of five (5) external servers was carried out using the same techniques as an attacker located outside your infrastructure. This phase includes the following steps:

- Recognition phase;
- Port scanning and systems identification;
- Services drilling/search;
- Vulnerability identification and validation;
- Vulnerability exploitation and evaluation of associated risks.

The intrusion test verified if these servers resist hostile attacks without revealing too much information on your environment and if identified vulnerabilities can lead to further intrusion and exploitation.

Phase 2: Penetration test of two (2) external Web applications

Within this phase, an intrusion test of two (2) Web Applications was carried out employing the same techniques as an attacker located outside your Web infrastructure. This phase includes the following steps:

- Identification and recognition phase;
- Application structure scanning and identification of used technologies;
- Verification and validation of the existence of sensitive files or directories;
- Identification and validation of vulnerabilities related to used products versions, input validation, error code management and authentication;
- Vulnerability exploitation and evaluation of associated risks.

The intrusion test verified if these applications will resist hostile attacks without revealing too much information on your environment and if identified vulnerabilities can lead to further intrusion and exploitation.

2.2. Detailed Methodology

2.1.1. Network Intrusion Tests

In order to evaluate the servers' capacity to prevent attacks against the network, we proceeded with the following steps:

Reconnaissance

- WHOIS and DNS interrogations;
- Network reconnaissance using tools such as Traceroute;
- Search of Web sites and search engines for user names.

Enumeration

- Full Nmap scan of all ten assets to map open ports and services;
- Exploration of discovered services using standard clients;
- Scanning (Saint, Nessus, Nexpose, ...) of all assets for known vulnerabilities;
- Scan for vulnerable IPSec VPNs.

Vulnerability Analysis and Exploitation

- Automated exploitation of discovered vulnerabilities using Core Impact, SAINTexploit, Metasploit and other tools;
- Brute-force Remote Desktop, VPN and other remote services.

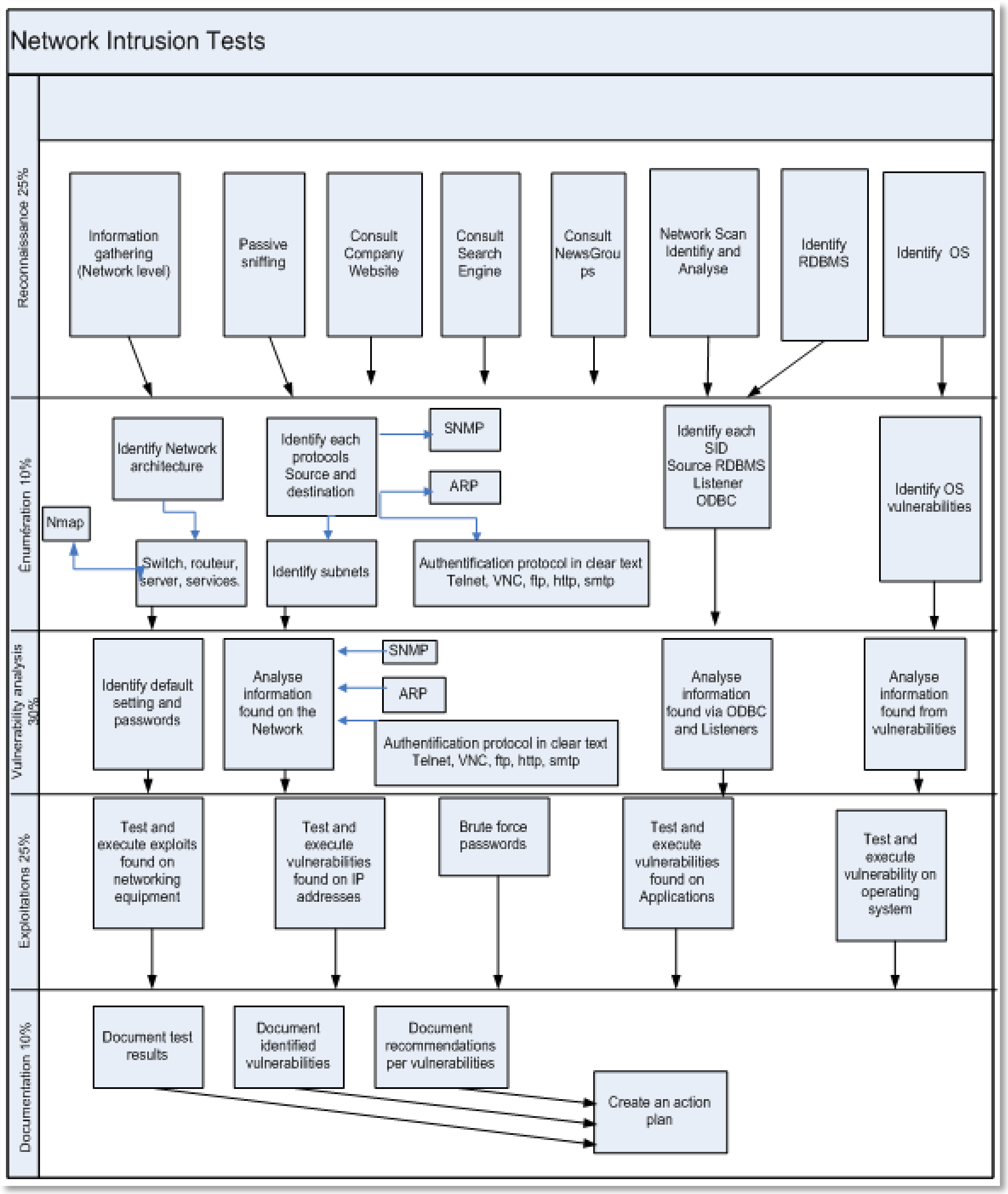

Network Intrusion Tests
Reconnaissance 25%
Enumération 10%
Vulnerability analysis 30%
Exploitations 25%
Documentation 10%
Information gathering (Network level)
Passive sniffing
Consult Company Website
Consult Search Engine
Consult NewsGroups
Network Scan Identifiy and Analyse
Identify RDBMS
Identify OS
Identify Network architecture
Identify each protocols Source and destination
SNMP
ARP
Identify each SID Source RDBMS Listener ODBC
Identify OS vulnerabilities
Nmap
Switch, routeur, server, services.
Identify subnets
Authentification protocol in clear text Telnet, VNC, ftp, http, smtp
Identify default setting and passwords
Analyse information found on the Network
SNMP
ARP
Authentification protocol in clear text Telnet, VNC, ftp, http, smtp
Analyse information found via ODBC and Listeners
Analyse information found from vulnerabilities
Test and execute exploits found on networking equipment
Test and execute vulnerabilities found on IP addresses
Brute force passwords
Test and execute vulnerabilities found on Applications
Test and execute vulnerability on operating system
Document test results
Document identified vulnerabilities
Document recommendations per vulnerabilities
Create an action plan

Figure 1: Network Intrusion Tests Methodology Flow Chart

2.1.2. Web Application Security Audit Process

In the case of Web applications, the methodology we used follows the Open Web Application Security Project (OWASP). The security audit included the following checks using automated and manual tools:

Information Gathering

- Testing Web application fingerprint;
- Application discovery;
- Spidering and googling;
- Analysis of error codes.

Business Logic Testing

Authentication Testing

- Testing for bypassing authentication schema;
- Testing for directory traversal/file included;
- Testing for vulnerable remembered password and password reset;
- Testing for logout and browser cache management testing.

Session Management Testing

- Testing for session management schema;
- Testing for cookie and session token manipulation;
- Testing for exposed session variables.

Data Validation Testing

- Testing for cross site scripting;
- Testing for SQL injection;
- Testing for command injection;
- Testing for code injection;
- Testing for buffer overflow;
- Testing for denial of service.

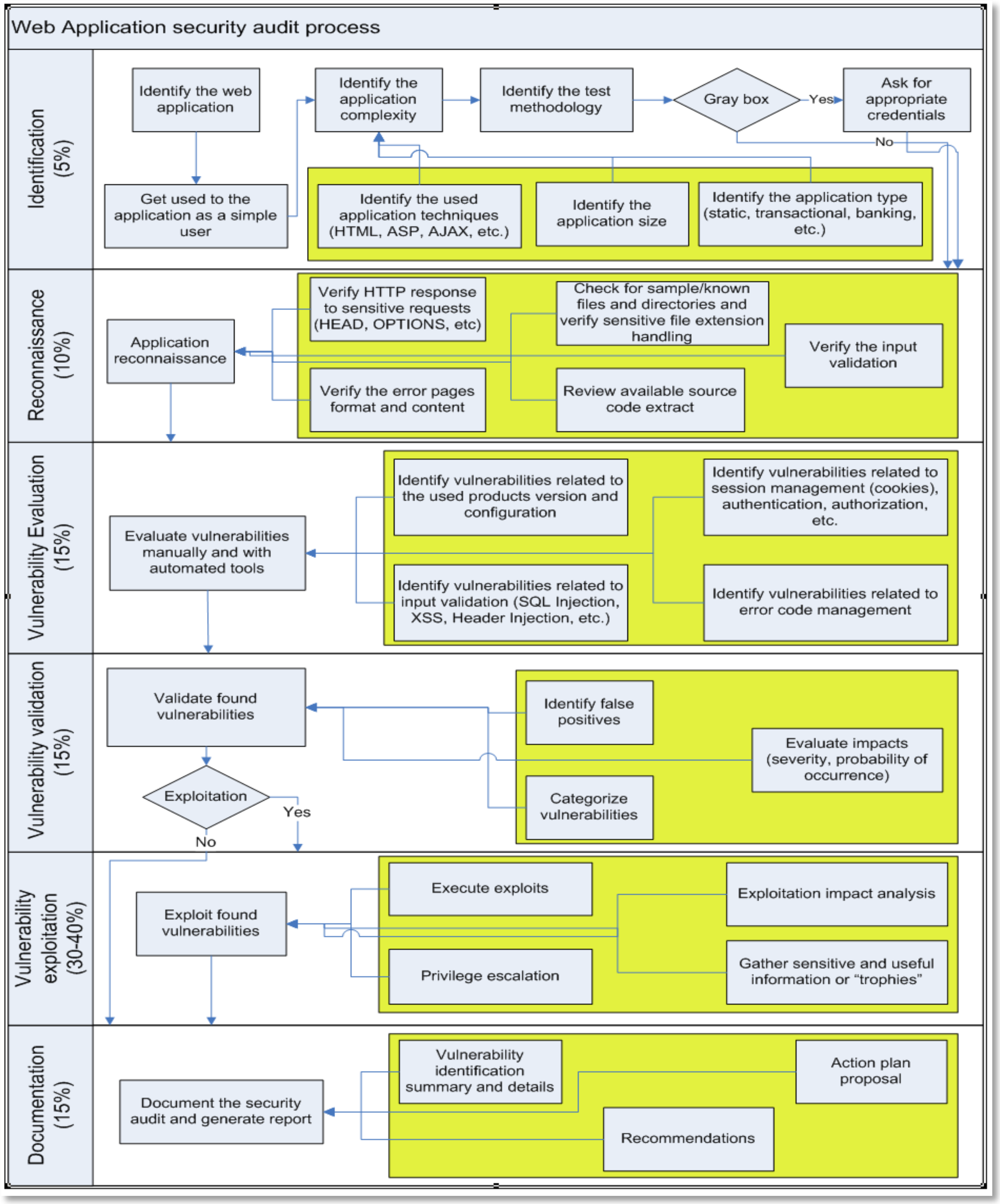

Web Application security audit process
Identification (5%)
Identify the web application
Get used to the application as a simple user
Identify the application complexity
Identify the test methodology
Gray box
Yes
No
Ask for appropriate credentials
Identify the used application techniques (HTML, ASP, AJAX, etc.)
Identify the application size
Identify the application type (static, transactional, banking, etc.)
Reconnaissance (10%)
Application reconnaissance
Verify HTTP response to sensitive requests (HEAD, OPTIONS, etc)
Check for sample/known files and directories and verify sensitive file extension handling
Verify the input validation
Verify the error pages format and content
Review available source code extract
Vulnerability Evaluation (15%)
Evaluate vulnerabilities manually and with automated tools
Identify vulnerabilities related to the used products version and configuration
Identify vulnerabilities related to session management (cookies), authentication, authorization, etc.
Identify vulnerabilities related to input validation (SQL Injection, XSS, Header Injection, etc.)
Identify vulnerabilities related to error code management
Vulnerability validation (15%)
Validate found vulnerabilities
Identify false positives
Evaluate impacts (severity, probability of occurrence)
Categorize vulnerabilities
Exploitation
Yes
No
Vulnerability exploitation (30-40%)
Exploit found vulnerabilities
Execute exploits
Exploitation impact analysis
Privilege escalation
Gather sensitive and useful information or "trophies"
Documentation (15%)
Document the security audit and generate report
Vulnerability identification summary and details
Action plan proposal
Recommendations

Figure 2: Web Application Intrusion Tests Methodology Flow Chart

2.3. Risk Evaluation

Above Security quantifies the risks according to the following formula:

Risk = Probability of Occurrence × Impact

2.3.1. Impact

An impact estimate is specified according to the extent of damage an attacker could cause when exploiting an identified vulnerability. This evaluation is done considering the importance of the security weaknesses and the severity of such vulnerabilities. The severity of vulnerabilities is subjectively evaluated on a scale of 0 to 10:

- A vulnerability with a severity of "10" leads to complete compromising and control of the target;
- A severity of "0" means that the exploitation does not lead to harmful results.

2.3.2. Probability of Occurrence

The vulnerabilities probability of occurrence is evaluated on a scale of 0 to 10. This evaluation is done considering:

- The perimeter and network security and authentication mechanisms;
- The perimeter and Web site security and authentication mechanisms;
- The attacker's knowledge and competence level;
- The necessary attack resources;
- The availability of automated attack tools.

A probability of occurrence of "10" means that the vulnerability is easy to exploit even with an automated tool or by a virus. A probability of "1" means that the vulnerability is unlikely to be exploited.

2.3.3. Risk Rating Methodology

Our risk rating methodology is based on the OWASP Risk Rating Methodology.

OVERALL RISK SECURITY LEVELS

Rate	Level
< 30	LOW
30 – 59	MEDIUM
60 – 100	HIGH

To complete the threat assessment, ABC Inc. can evaluate the effective value of its assets, estimate the cost as the total economic impact of a successful attack, and then calculate a revised risk using the following formula:

Risk = Probability of Occurrence × Impact × Cost to ABC Inc.

2.4. Summary of Vulnerabilities

Here is a brief listing of the vulnerabilities found during each phase of the security audit. Detailed explanations can be found in the indicated sections:

2.4.1. External Servers

VULNERABILITY	DETAILS	RISK (/100)
Multiple Vulnerabilities in Lighttpd version	Section 3.1	49
Vulnerable Mail Server version	Section 3.2	42
SMTP Relaying (OPEN Relay)	Section 3.3	36
Unsecure Logon Page	Section 3.4	35
Deprecated SSL Version	Section 3.5	24
Vulnerable Version of OpenSSH	Section 3.6	24
Possible Vulnerabilities in Microsoft Exchange	Section 3.7	24
Telnet sessions enabled on Internet-facing network device	Section 3.8	21
FTP Service Allows Clear-Text Authentication	Section 3.9	21
Weak Encryption Ciphers Supported	Section 3.10	18
SSH v1 supported	Section 3.11	14

Table 5: List of ABC Inc.'s external server vulnerabilities

2.4.2. External Web Applications

VULNERABILITY	DETAILS	RISK (/100)
Cross-Site Scripting (XSS) Vulnerability	Section 4.1	72
Directory Listing - Unprotected Sensitive Files	Section 4.2	72
Cross-Frame Scripting Vulnerability	Section 4.3	56
No Brute Force Protection on Login Page	Section 4.4	54
Login credentials sent over unencrypted connection	Section 4.5	48
Session Fixation Vulnerability	Section 4.6	28
Runtime Error Messages	Section 4.7	27

Table 6: List of Web Applications Vulnerabilities

The vulnerabilities are also categorized following the OWASP Top Ten Vulnerabilities in Web applications. The right column of this table shows what categories passed or failed our audit:

✓	**Pass**
!	**Area of concern**
✗	**Fail**

VULNERABILITY	DETAILS	RISK (/100)
Cross-Site Scripting (XSS)	XSS flaws occur whenever an application takes user supplied data and sends it to a Web browser without first validating or encoding that content. XSS allows attackers to execute script in the victim's browser which can hijack user sessions, deface Web sites, possibly introduce worms, etc.	✗
Injection Flaws	Injection flaws, particularly SQL injection, are common in Web applications. Injection occurs when user-supplied data is sent to an interpreter as part of a command or query. The attacker's hostile data tricks the interpreter into executing unintended commands or changing data.	✓
Malicious File Execution	Code vulnerable to remote file inclusion (RFI) allows attackers to include hostile code and data, resulting in devastating attacks, such as total server compromise. Malicious file execution attacks affect PHP, XML and any framework which accepts filenames or files from users.	✓
Insecure Direct Object Reference	A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, database record, or key, as a URL or form parameter. Attackers can manipulate those references to access other objects without authorization.	✓
Cross-Site Request Forgery (CSRF)	A CSRF attack forces a logged-on victim's browser to send a pre-authenticated request to a vulnerable Web application, which then forces the victim's browser to perform a hostile action to the benefit of the attacker. CSRF can be as powerful as the Web application that it attacks.	✓
Information Leakage and Improper Error	Applications can unintentionally leak information about their configuration, internal workings, or violate	✗

Handling	privacy through a variety of application problems. Attackers use this weakness to steal sensitive data or conduct more serious attacks.	
Broken Authentication and Session Management	Account credentials and session tokens are often not properly protected. Attackers compromise passwords, keys, or authentication tokens to assume other users' identities.	✓
Insecure Cryptographic Storage	Web applications rarely use cryptographic functions properly to protect data and credentials. Attackers use weakly protected data to conduct identity theft and other crimes, such as credit card fraud.	✓
Insecure Communications	Applications frequently fail to encrypt network traffic when it is necessary to protect sensitive communications.	✗
Failure to Restrict URL Access	Frequently, an application only protects sensitive functionality by preventing the display of links or URLs to unauthorized users. Attackers can use this weakness to access and perform unauthorized operations by accessing those URLs directly.	✗

Table 7: Web Vulnerabilities Categories

3. Vulnerability Details – External Servers

3.1. Multiple Vulnerabilities in Lighttpd version

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
49	7	7	MEDIUM

Impact

The Lighttpd webserver is prone to multiple vulnerabilities. Successfully exploiting these vulnerabilities allows a remote attacker to cause a denial of service or to obtain sensitive information.

Details

According to the system scan, one of your servers is running Lighttpd version 1.1.19 which is prone to an important number of security issues.

We've listed below the different CVE affecting this Lighttpd version:

- CVE-2008-1531
- CVE-2008-4298
- CVE-2008-4359
- CVE-2008-4360
- CVE-2010-0295

Vulnerable Assets

- X.X.X.141 (mail.ABC Inc.com)
- X.X.X.242

Solution

Upgrade Lighttpd to version 1.4.26 or higher.

3.2. Vulnerable Mail Server version

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
42	6	7	MEDIUM

Impact

The SMTP component in Microsoft 2003 is prone to multiple vulnerabilities. A remote attacker could crash the mail service or gain user-level privileges to the service, including the ability to use the server as a mail relay.

Details

According to the mail server version 6.0.3790.3959, it seems that few patches are missing on this server (MS04-035) and (MS10-024).

We've listed below the different CVE affecting this Mail Server version:

- CVE-2010-0024
- CVE-2010-0025
- CVE-2010-1689
- CVE-2010-1690

Vulnerable Assets

- X.X.X.141 (mail.ABC Inc.com)
- X.X.X.242

Solution

Apply the patches referenced in (MS04-035) and (MS10-024).

3.3. SMTP relaying; Open Relay

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
36	6	6	MEDIUM

Impact

Unauthorised user or a spammer can send spam emails targeting internal users (Open Relay).

Details

Without authentication, using Telnet we were able to connect to ABC Inc. SMTP server titan.ABC Inc.-cnc.com and send an email from an internal user called A to another internal user called B. User A did not know that we used his email address to send the email. The way the server is configured, it will relay only to the addresses of the ABC INC. domain. User B confirmed the reception of the email as shown in the figure below.

Figure 3: Confirmation of the email sent from user A to user B (figure removed)

Vulnerable Assets

- X.X.X.242
- X.X.X.141

Solution

Configure the SMTP server to not be used as (OPEN Relay).

3.4. Unsecure Logon Page

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
36	4	9	MEDIUM

Impact

Any section of a website containing confidential information or giving access to privileged functionality as remote administration should be protected by SSL or another form of encryption. A logon page allows unauthorized users or malicious attackers, under certain conditions, to intercept the authentication data such as user names and passwords and gain access.

Details

The access page to XXXX WAN AGGREGATOR contains a form of authentication. This form uses HTTP Basic authentication. This mechanism does not encrypt the data. The user names and passwords are transmitted in clear and can be recovered by a striker positioned to sniff the network.

Figure 4: Unsecure HTTP Access (figure removed)

Vulnerable Assets4

- X.X.X.242
- X.X.X.141
- X.X.X.254

Solutions

- It is recommended to use secure authentication over SSL (HTTPS) for any administration interfaces;
- Restrict this URL to only IP addresses who are allowed to connect to this web page, either internally or externally.

3.5. Deprecated SSL Version

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
24	3	8	LOW

Impact

Servicing 'secure' transactions via encrypted communications, support version 2 of the SSL protocol. This version of SSL is deprecated and offers weak security; it should be disabled and replaced by SSLv3/TLSv1. Indeed, a skilled attacker could decrypt users' communications or force their browser to use weak encryption keys.

Details

Some assets accept connections made with SSL version 2. Several weaknesses have been exposed in SSL v2 and this version is therefore not considered secure any longer.

Vulnerable Assets

- X.X.X.141 (mail.ABC Inc.com)
- X.X.X.242

Solutions

- Disable the support of all deprecated versions of SSL (versions before SSLv3/TLSv1);
- With Microsoft IIS it is necessary to edit the Windows registry. The steps involved are detailed in the Microsoft Support page reference below;
- With Apache web servers disable deprecated versions of SSL using the following configuration directive in httpd.conf or ssl.conf: SSLProtocol -ALL +SSLv3 +TLSv1.

References

- Microsoft Support: How to disable PCT 1.0, SSL 2.0, SSL 3.0, or TLS 1.0 in Internet Information Services http://support.microsoft.com/kb/187498
- Bruce Schneier: Analysis of the SSL Protocol http://www.schneier.com/paper-ssl.pdf
- Apache documentation: Apache Module mod_ssl – SSLProcol http://httpd.apache.org/docs/2.0/mod/mod_ssl.html#sslprotocol

3.6. Vulnerable Version of OpenSSH

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
24	3	8	LOW

Impact

OpenSSH is prone to information disclosure vulnerability. An attacker can exploit this vulnerability to insert arbitrary commands into a session, or to gain remote root access to the OpenSSH server.

Details

One of the tested assets is using OpenSSH 5.1 which is known to be vulnerable. We encourage ABC Inc. to apply the corrective actions presented in the solution section.

Vulnerable Asset

- X.X.X.254

Solution

Upgrade to OpenSSH version 5.8 or higher.

Reference

- New release OpenSSH 5.8 http://www.openssh.com/txt/release-5.8

3.7. Possible Vulnerabilities in Microsoft Exchange

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
24	3	8	LOW

Impact

Multiple vulnerabilities were reported in Microsoft Exchange server. If exploited successfully, a remote attacker could crash the mail service or execute arbitrary code.

Details

Some of the tested assets are using Microsoft Exchange server with possibly some vulnerabilities that were addressed in Microsoft Security Bulletin MS06-003 and MS05-021.

We encourage ABC Inc. to apply the corrective actions presented in the solution section.

- CVE: CVE-2006-0002
- CVE: CVE-2005-0560

Vulnerable Asset

- X.X.X.242

Solution

Apply the patches referenced in (MS06-003) and (MS05-021).

955 boul. Michèle-Bohec, Suite 244
Blainville (Quebec) J7C 5J6
Canada

3.8. Telnet sessions enabled on Internet-facing network device

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
21	3	7	LOW

Impact

The network device allows users to connect remotely via the Telnet protocol. This protocol is unsecure as it transmits all data, including user credentials, in clear-text format. Telnet is deprecated, subject to brute-force attacks and should not be used in production environments as any well-position attacker could steal the transmitted credentials.

Moreover, the Telnet access is available from the Internet which suggests that no ACL has been defined to restrict access to a pre-defined set of valid IP addresses.

Details

As is shown on *Figure 4*, Telnet is enabled on the Cisco device:

Figure 5: Telnet service enabled (figure removed)

Vulnerable Assets

- X.X.X.242
- X.X.X.241

Solutions

- Disable Telnet and replace it with SSH which is much more secure and encrypts all communications.
- Define an ACL to only accept connection from a pre-approved set of IP addresses.
- Apply the best practice of stopping all unnecessary services.

3.9. FTP Service Allows Clear-Text Authentication

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
21	3	7	LOW

Impact

The server allows users to connect remotely via the File Transfer Protocol (FTP). This protocol is insecure as it transmits all data, including user credentials, in clear-text format. FTP is deprecated, subject to brute-force attacks and should not be used in production environments as any well-positioned attacker could steal the transmitted credentials.

Details

The remote FTP does not encrypt its data and control connections. The user name and password are transmitted in clear text and may be intercepted by a network sniffer, or a man-in-the-middle attack.

Vulnerable Asset

- X.X.X.242

Solutions

- Disable FTP and switch to SFTP (part of the SSH suite) or FTPS (FTP over SSL/TLS) and configure the server such as data and control connections must be encrypted;
- Define an ACL to only accept connections from a pre-approved set of IP addresses;
- Apply the best practice of stopping all unnecessary services.

3.10. Weak Encryption Ciphers Supported

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
18	3	6	LOW

Impact

These SSL-protected servers support communications using low and medium-grade encryption. These mechanisms do not ensure the confidentiality of transmitted data anymore and should therefore be disabled.

Details

The remote FTP does not encrypt its data and control connections. The user name and password are transmitted in clear text and may be intercepted by a network sniffer, or a man-in-the-middle attack.

The servers allow users to connect using 40-bit export-grade encryption keys (low encryption) or keys smaller than 112 bits (medium encryption). Moreover, some of the encryption algorithms are outdated (e.g. RC2, RC4 or DES).

Indeed, these mechanisms are not considered secure as numerous it has been demonstrated that their encryption could be broken. Stronger encryption is preferred, starting with 128-bit keys.

Vulnerable Assets

- X.X.X.242
- X.X.X.141

Solutions

- Disable support for all ciphers implementing encryption lower than 128-bit, as well as the DES and RC2 algorithms;
- For Microsoft IIS web servers, see Microsoft Knowledgebase article “245030” for instructions on disabling weak ciphers: http://support.microsoft.com/kb/245030/
- With Apache web servers, disable support for weak encryption ciphers using the following directive or similar in httpd.conf or ssl.conf: SSLCipherSuite ALL:!aNULL:!eNULL:!LOW:!EXP:RC4+RSA:+HIGH:+MEDIUM

3.11. SSHv1 supported

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
12	2	6	LOW

Impact

The SSH services support connections made using versions 1.99 of the protocol. Versions 1.x of the SSH protocol are not considered safe and their use can lead to session corruption and total compromise of the asset.

Details

The use of SSH 1.x is discouraged as cryptographic issues have been discovered. Attackers can gain victim user privileges.

- CVE-2001-0361
- CVE-2001-1473

Vulnerable Asset

- X.X.X.242

Solution

- Disable compatibility with version 1.x of the protocol.

Reference

http://www.openssh.org/

4. Vulnerability Details – Web Applications

4.1. Cross-Site Scripting (XSS) Vulnerability

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
18	3	6	LOW

Impact

If successful, Cross-Site Scripting vulnerabilities can be exploited to manipulate or steal cookies, create requests that can be mistaken for those of a valid user, compromise confidential information, or execute malicious code on end user systems.

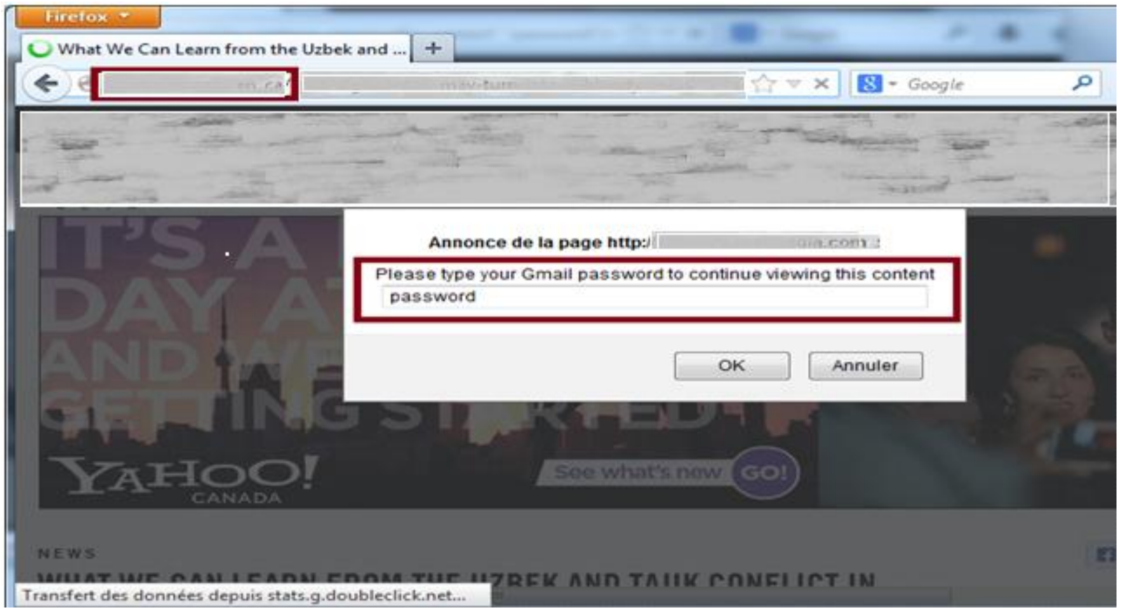
Details

XSS vulnerability has been identified in the Web applications. It is possible for an attacker to inject malicious script in the URL, which can be used to perform successful phishing attacks, steal session's cookie, etc...

This happens when user input from a web client is immediately included via server-side scripts in a dynamically generated web page. via some social engineering, an attacker can trick a victim, such as through a malicious link or "rigged" form, to submit information, which will be altered to include attack code and then sent to the legitimate server. The injected code is then reflected back to the user's browser, which executes it because it came from a trusted server.

For instance, a malicious user may e-mail his victim with the following link:

http://<client_web1>.com/Login.aspx?lang=EN&ReturnUrl=%22%3E%3CsCrIpT%3Eprompt%28%22Please%20type%20your%20username%20and%20password%20to%20continue%20viewing%20this%20content%20(username/password)%22,%22username/password%22%29%3C/sCrIpT%3E


Firefox
What We Can Learn from the Uzbek and ...
Google
Annonce de la page http://
OK
Annuler
YAHOO! CANADA
See what's new GO!
NEWS
WHAT WE CAN LEARN FROM THE UZBEK AND TAJIK CONFLICT IN
Transfert des données depuis stats.g.doubleclick.net...

Please type your Gmail password to continue viewing this content

password

Figure 6: Cross-site Scripting Vulnerability exploited

The result of open the malicious URL in a browser is show in figure 3. The web page asks the victim to type his credentials to continue browsing. As soon as the users enters his password and hits OK button, it will be sent to a server belonging to the attacker, even though the message seems to be coming from ABC's servers.

Vulnerable Link(s)

The attack works by appending the following string to any of the URLs listed below or to the vulnerable parameters in the URL:

%22%3E%3CsCrIpT%3Eprompt%28%22Please%20type%20your%20username%20and%20password%20to%20continue%20viewing%20this%20content%20(username/password)%22,%22username/password%22%29%3C/sCrIpT%3E

- https://www.<client_web1>.com/Login.aspx
    - Vulnerable parameter(s) : ReturnUrl, Source, d, t

Solution

Cross-Site Scripting attacks can be avoided by carefully validating all input, and properly encoding all output. When validating user input, verify that it matches the strictest definition of valid input possible. For example, if a certain parameter is supposed to be a string, attempt to convert it to a string data type in your programming language.

Developpers should use the following methods when using ASP.NET:

- **ASP.NET: int.TryParse(Request.QueryString["q"], out val);**

The same applies to date and time values, or anything that can be converted to a stricter type before being used. When accepting other types of text input, make sure the value matches either a list of acceptable values (white-listing), or a strict regular expression. If at any point the value appears invalid, do not accept it. Also, do not attempt to return the value to the user in an error message.

Most server side scripting languages provide built in methods to convert the value of the input variable into correct, non-interpretable HTML. These should be used to sanitize all input before it is displayed to the client.

- **ASP.NET: Server.HTMLEncode (strHTML String)**

When reflecting values into JavaScript or another format, make sure to use a type of encoding that is appropriate. Encoding data for HTML is not sufficient when it is reflected inside of a script or style sheet. For example, when reflecting data in a JavaScript string, make sure to encode all non-alphanumeric characters using hex (\xHH) encoding.

If you have JavaScript on your page that accesses unsafe information (like location.href) and writes it to the page (either with document.write, or by modifying a DOM element), make sure you encode data for HTML before writing it to the page. JavaScript does not have a built-in function to do this, but many frameworks do. If you are lacking an available function, something like the following will handle most cases:

```
s= s.replace(/&/g,'&amp;').replace(/"/i,'&quot;').replace(/</i,'&lt;').replace(/>/i,'&gt;').replace(/'/i,'&apos;')
```

Ensure that you are always using the right approach at the right time. Validating user input should be done as soon as it is received. Encoding data for display should be done immediately before displaying it.

References

http://download.hpsmartupdate.com/asclabs/cross-site_scripting.pdf

http://www.owasp.org/documentation/topten/a4.html

http://support.microsoft.com/default.aspx?scid=kb;EN-US;q252985

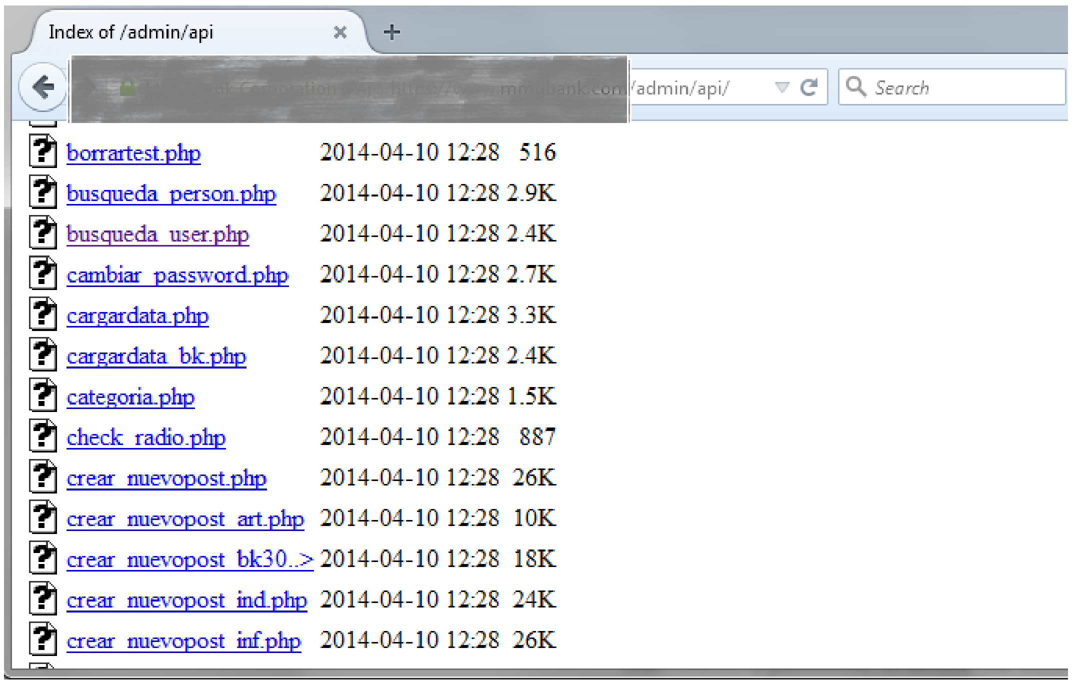
http://httpd.apache.org/info/css-security/apache_specific.html

4.2. Directory Listing – Unprotected Sensitive Files

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
18	3	6	LOW

Impact

Any unauthenticated user located inside your network can access the vulnerable directory containing sensitive information. A file found on the vulnerable server contains Credit Card information with names, addresses, phone number, etc..



```
borrartest.php              2014-04-10 12:28  516
busqueda_person.php         2014-04-10 12:28 2.9K
busqueda_user.php           2014-04-10 12:28 2.4K
cambiar_password.php        2014-04-10 12:28 2.7K
cargardata.php              2014-04-10 12:28 3.3K
cargardata_bk.php           2014-04-10 12:28 2.4K
categoria.php               2014-04-10 12:28 1.5K
check_radio.php             2014-04-10 12:28  887
crear_nuevopost.php         2014-04-10 12:28  26K
crear_nuevopost_art.php     2014-04-10 12:28  10K
crear_nuevopost_bk30..>     2014-04-10 12:28  18K
crear_nuevopost_ind.php     2014-04-10 12:28  24K
crear_nuevopost_inf.php     2014-04-10 12:28  26K
```

Figure 7: Unprotected Directory

Vulnerable Link(s)

- www.<client_web1>.com/downloads/
- www.<client_web2>.com/creditcards/

Solution

Restrict access to the directories containing sensitive information.

4.3. Cross-Frame Scripting Vulnerability

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY

18	3	6	LOW

Impact

A Cross-Frame Scripting weakness could allow an attacker to embed the vulnerable application inside an iframe. Exploitation of this weakness could result in:

1. Hijacking of user events such as keystrokes
2. Theft of sensitive information
3. Execution of privileged functionality through combination with Cross-Site Request Forgery attacks

Details

A Cross-Frame Scripting (XFS) vulnerability can allow an attacker to load the vulnerable application inside an HTML iframe tag on a malicious page. The attacker could use this weakness to devise a Clickjacking attack to conduct phishing, frame sniffing, social engineering or Cross-Site Request Forgery attacks.

The goal of a Clickjacking attack is to deceive the victim user into interacting with UI elements of the attacker's choice on the target web site without her knowledge and in turn executing privileged functionality on the victim's behalf. To achieve this goal, the attacker must exploit the XFS vulnerability to load the attack target inside an iframe tag, hide it using Cascading Style Sheets (CSS) and overlay the phishing content on the malicious page. By placing the UI elements on the phishing page to overlap with those on the page targeted in the attack, the attacker can ensure that the victim is forced to interact with the UI elements on the target page not visible to the victim.

Vulnerable Link(s)

- http://www.<client_web2>.com

Try it

Create a test page containing an HTML iframe tag whose src attribute is set to https://wwws.<client_web2>.com/ . Successful framing of https://wwws.<client_web2>.com/ indicates the application's susceptibility to XFS. All visible pages on the site may be vulnerable to XFS as well and hence should be protected against it with an appropriate fix.

Solution

Browser vendors have introduced and adopted a policy-based mitigation technique using the X-Frame-Options header. Developers can use this header to instruct the browser about appropriate actions to perform if their site is included inside an iframe. Developers must set the X-Frame-Options header to one of the following permitted values:

- DENY

Deny all attempts to frame the page

- SAMEORIGIN

The page can be framed by another page only if it belongs to the same origin as the page being framed

- ALLOW-FROM origin

Developers can specify a list of trusted origins in the origin attribute. Only pages on origin are permitted to load this page inside an iframe

Developers must **strictly avoid** the use of client-side frame busting JavaScript as a protection against XFS.

References

https://developer.mozilla.org/en-US/docs/HTTP/X-Frame-Options

http://tools.ietf.org/html/draft-ietf-websec-x-frame-options-01

https://www.owasp.org/index.php/Clickjacking

https://www.owasp.org/images/0/0e/OWASP_AppSec_Research_2010_Busting_Frame_Busting_by_Rydstedt.pdf

4.4. No Brute Force Protection on Login Page

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
54	6	9	MEDIUM

Impact

As the web application does not include any specific security mechanism to prevent automated and spoofed requests, it is vulnerable to several types of serious attacks, including brute force attacks.

Brute Force attacks are the simplest form of secret-guessing attacks and are usually launched to determine valid password given a user ID (or both): it is conducted by trying all possible combinations. Such attacks are often quite lengthy to complete but they are performed by automated scripts and can be successful, especially if strict password complexity rules are not enforced. In case of success the targeted user account will likely be completely compromised.

Details

ABC's extranet applications do not seem to provide any brute force attack protection mechanism for its authentication page, which means that malicious individuals or automated scripts can launch as many requests as they want – i.e. try as many login/password combinations as they need.

Vulnerable Links

- http://www.<client_web>.com

Solution

- Implement a limit on the number of tries that can be made to authenticate to the application.
- A common procedure is to lock the account once this maximum number of tries (usually ranging from 3 to 6) is reached. Then, users can get their account unlocked either by:
    - Contacting the application's customer service;
    - Entering some additional private information matching the account's records, or a one-time code delivered by e-mail (in addition to a valid password);
    - Resolving a CAPTCHA
    - Simply waiting for the account lock-out to time out by itself.

The last solution can in many cases be considered the most reasonable. Indeed, it does not require any action from the user or the application's maintaining unit. Moreover, setting a low number of maximum login attempts and a moderate lock-out period proves effective against password-guessing attacks, which a lock-out is aimed at preventing. In any case, ABC should select the solution that is best fitted to its business needs.

References

https://www.owasp.org/index.php/Blocking_Brute_Force_Attacks

4.5. Login credentials sent over unencrypted connection

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
48	6	8	MEDIUM

Impact

An attacker that exploits this design vulnerability would be able to use the information to escalate his method of attack, possibly leading to impersonating a legitimate user, obtaining proprietary data, or simply executing actions not intended by the application designers.

Details

The web application sends User authentication credentials through a simple HTTP session. These credentials are transmitted in clear text and may be intercepted by a network sniffer, or a man-in-the-middle attack. It should not be possible to access this information using a non-SSL connection.

Vulnerable Link(s)

- http://www.<client_web2>.com

Solution

Ensure that any area of a web application that possibly contains sensitive information or access to privileged functionality such as remote site administration functionality utilize SSL or another form of encryption to prevent login information from being sniffed or otherwise intercepted.

References

http://www.kb.cert.org/vuls/id/466433

4.6. Session Fixation Vulnerability

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
28	4	7	LOW

Impact

Session Fixation is an attack that permits an attacker to hijack a valid session by stealing user's session ID after a successful login authentication the web application.

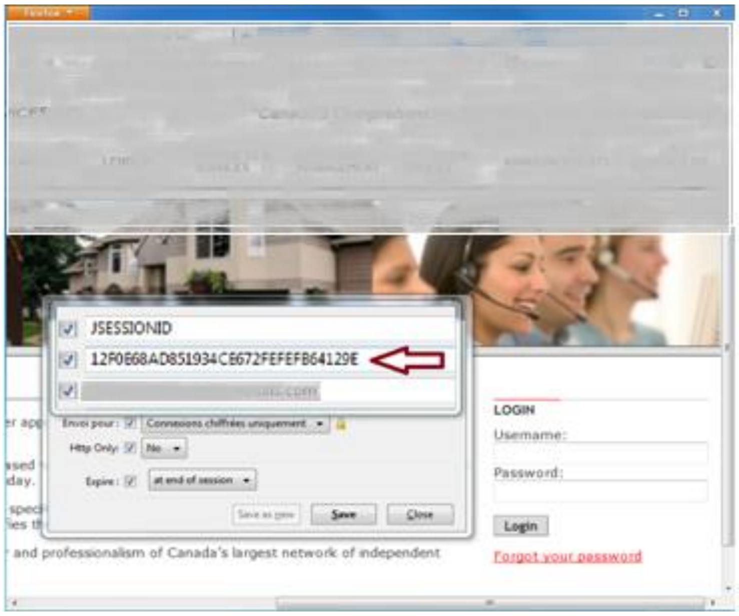
Details

Session Fixation is an attack technique that forces a user session ID to an explicit value. Unfortunately, cookie based session are the easiest to attack. Depending on the functionality of the target web site, a number of techniques can be used to "fix" the session ID value. After a User session ID has been fixed; the attacker will wait for that user to login. Once the user does so, the attacker uses the predefined session ID value to assume the same online identity.

As we can see in the figures bellow, the session cookie (JSESSIONID) of the web application stays the same "before and after" authentication. This weakness can be exploited by an attacker to perform a session fixation attack. The Session Fixation attack is normally a three steps process:

- Session Set-Up: an attacker may select an arbitrary session ID (JSESSIONID) from the web application. He can modify the value of the expiration date of the cookie by extending its value.
- Session fixation: the attacker introduces the JSESSIONID cookie into the victim's browser and fixed the user's session ID.
- Session entrance: the attacker waits until the victim logs into the target web site. When the victim does so, the fixed session ID value will be used and the attacker may take over.

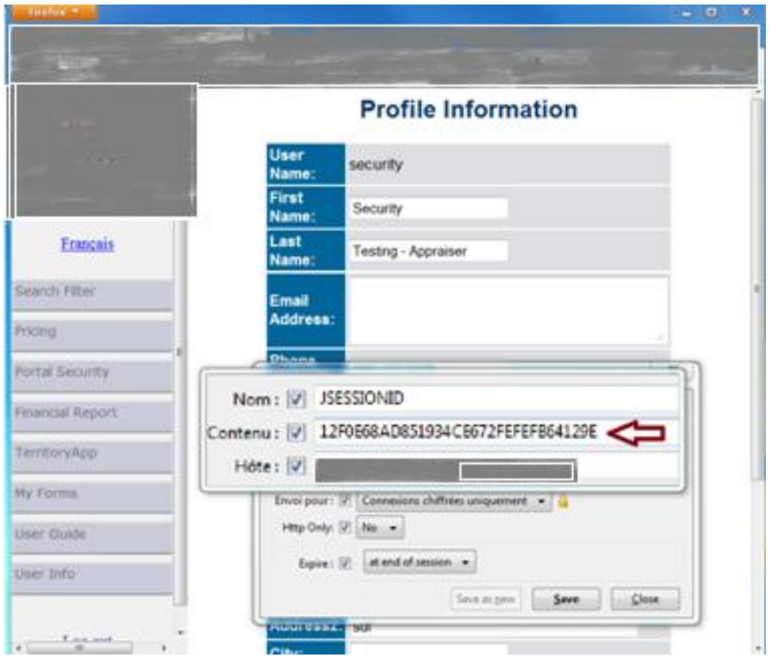
In the manual tests performed, we were able to end an active session for a legitimate user and force the user to re-authenticate



JSESSIONID
12F0E68AD851934CE672FEFEFB64129E

Envoi pour : Connexions chiffrées uniquement
Http Only: No
Expire : at end of session

Figure 8: SessionID before Authentication



Profile Information

User Name:	security
First Name:	Security
Last Name:	Testing - Appraiser
Email Address:	

Nom : JSESSIONID
Contenu : 12F0E68AD851934CE672FEFEFB64129E
Hôte : [redacted]

Envoi pour : Connexions chiffrées uniquement
Http Only: No
Expire : at end of session

Figure 9: SessionID after Authentication

Vulnerable Link(s)

- www.<client_web1>.com

Solution

A web application on a strict system should only issue session IDs of newly generated sessions to users after they have successfully authenticated (as opposed to issuing them along with the login form). This means that an attacker who isn't a legitimate user of the system will not be able to get a valid session ID and will therefore be unable to perform a session fixation attack.

References

https://www.owasp.org/index.php/Session_fixation

4.7. Runtime Error Messages

RISK (/100)	PROBABILITY OF OCCURRENCE	IMPACT (/10)	RISK SEVERITY
27	9	3	LOW

Impact

Information disclosure vulnerabilities reveal sensitive information about a system, network or web application to an attacker. An attacker can use this information to learn more about a system when attempting to gain unauthorized access. Disclosure of sensitive information facilitates a wide range of attacks.

Details

A runtime error message indicates that an unhandled exception was generated in the web application code. Unhandled exceptions are circumstances in which the application has received user input that it did not expect and doesn't know how to deal with.

Vulnerable Link(s)

- http://www.<client_web2>.com/scripts/badfile123.ashx

Solution

From a testing perspective, ensure that the error handling scheme is consistent (Use a uniform error handling scheme) and does not reveal private information about your web application.

References

http://support.microsoft.com/kb/823379

http://msdn.microsoft.com/en-us/library/bhs3h30t%28VS.71%29.aspx

About Above Security

Above Security – A Hitachi Group Company is a Global IT Security Service Provider who builds and delivers customized services for monitoring and protecting the most critical and sensitive IT assets in our clients' infrastructures 24/7. With a relentless focus on risk management, and continuous improvement of our technology and incident response processes, our clients count on us to provide the right solutions for their businesses – quickly, effectively and with expertise beyond the industry standards.

Global Presence

Canada (World Headquarters)
Mexico
Switzerland
United Arab Emirates
United States of America

Security Operations Centers

Montreal, Canada
Winnipeg, Canada
Sierre, Switzerland
Mexico City, Mexico