Pentest

Date 03/25/2016

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Customer Information | | | | | |
| Company Name: |  | | | | |
| City: |  | State: |  | Zip Code: |  |
| URL: |  |  |  |  |  |

|  |  |
| --- | --- |
| Customer Contact Information | |
| Contact Name: |  |
| Title: |  |
| Telephone: |  |
| E-mail: |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Consultant Information | | | | | |
| Company Name: | Serpico Template Report, LLC | | | | |
| Contact Name: | hadhemi | | | | |
| Title: | intern | | | | |
| Telephone: |  | | | | |
| E-mail: |  | | | | |
| Business Address: | 123 Paper St | | | | |
| City | TestCity | State: | MA | Zip Code: | 11111 |
| URL: | <http://www.serpicoreport.com> | | | | |

# Executive Summary

Serpico Template Company (STC) was contracted to perform a penetration test for . This report discusses the results from the assessment. Really, if you are reading this you should update the template to match your executive summary. The symbols throughout this report are used to display the data. Please see the README to understand how they work.

Overall, STC was able to achieve the goals of the assessment and exfiltrate the targeted data. There were a number of critical findings during the assessment including the following:

|  |  |
| --- | --- |
| Finding Name | DREAD Score |
| Cross Site Scripting (XSS) | 46 |
| Direct Object References | 44 |
| Weak SA Password on MSSQL Server | 50 |
| Excessive Ingress Rule Set | 50 |
| Insecure Java RMI Endpoint | 50 |

MOAR STUFF

# Attack Narrative

The following discusses the details of the assessment.

# Findings

## Findings Table

The following were the results from the assessment:

|  |  |  |
| --- | --- | --- |
| Finding Name | DREAD Score | Remediation Effort |
| **High Risk Findings (36-50)** |  |  |
| Cross Site Scripting (XSS) | 46 | Quick |
| Direct Object References | 44 | Quick |
| Weak SA Password on MSSQL Server | 50 | LOW |
| Excessive Ingress Rule Set | 50 | LOW |
| Insecure Java RMI Endpoint | 50 | LOW |
|  |  |  |
| **Moderate Risk Findings (20-35)** |  |  |
| Internal IP Address Disclosure | 29 | LOW |
|  |  |  |
| **Low Risk Findings (1-19)** |  |  |
| HTML form without CSRF protection | 1 | Quick |
| Clickjacking: X-Frame-Options header missing | 1 | Quick |
|  |  |  |

|  |  |
| --- | --- |
| DREAD Score |  |
| Damage Potential | 8 |
| Reproducibility | 10 |
| Exploitability | 8 |
| Affected Users | 10 |
| Discoverability | 10 |
| Total | 46 |

### Summary

The OWASP guide [1] gives the following description for Cross-Site Scripting:

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted web sites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user’s browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

### Proof

### Remediation

The following is recommended to remediate XSS vulnerabilities:

* Never trust user input
* Never insert untrusted data except in allowed locations
* HTML escape before inserting untrusted data into HTML element content
* Use whitelists in place for Black lists for input filtering

|  |  |
| --- | --- |
| DREAD Score |  |
| Damage Potential | 8 |
| Reproducibility | 10 |
| Exploitability | 8 |
| Affected Users | 10 |
| Discoverability | 8 |
| Total | 44 |

### Summary

The OWASP guide [1] gives the following description for Insecure Direct Object Reference:

Applications frequently use the actual name or key of an object when generating web pages. Applications do not always verify the user is authorized for the target object. This results in an insecure direct object reference flaw. Testers can easily manipulate parameter values to detect such flaws and code analysis quickly shows whether authorization is properly verified.

### Proof

### Remediation

Use per user or session indirect object references. This prevents attackers from directly targeting unauthorized resources. For example, instead of using the resource’s database key, a drop down list of six resources authorized for the current user could use the numbers 1 to 6 to indicate which value the user selected. The application has to map the per-user indirect reference back to the actual database key on the server.

Check access. Each use of a direct object reference from an untrusted source must include an access control check to ensure the user is authorized for the requested object.

|  |  |
| --- | --- |
| DREAD Score |  |
| Damage Potential | 5 |
| Reproducibility | 10 |
| Exploitability | 3 |
| Affected Users | 3 |
| Discoverability | 8 |
| Total | 29 |

### Summary

While reviewing 's web server, web servers were discovered to disclose the system's internal IP address via the Content-Location header. The disclosure of the systems internal IP address gives an adversary an indication of how the internal network my be addressed.

### Proof

### Remediation

It is recommended that reconfigure their web servers to use the systems fully qualified domain name (FQDN).

|  |  |
| --- | --- |
| DREAD Score |  |
| Damage Potential | 10 |
| Reproducibility | 10 |
| Exploitability | 10 |
| Affected Users | 10 |
| Discoverability | 10 |
| Total | 50 |

### Summary

Microsoft SQL server comes with a built in System Administrator (SA) account. By default the SA account has full privileges. During the assessment the SA account was found to have a default password of SA or blank. An adversary can use this account to gain administrator level access to the database and can lead to a potential comprise of the system.

### Proof

### Remediation

The default SA account should be disabled. It is recommended to use Windows Authentication. If this is not possible due to business reasons, the SA account should be configured with a strong password. The following guide lines can be used for creating a strong password:

* Use alphanumeric, special characters and spaces
* Use a password that is at least 32 characters long
* Change the password frequently
* Do not reuse previous passwords

|  |  |
| --- | --- |
| DREAD Score |  |
| Damage Potential | 10 |
| Reproducibility | 10 |
| Exploitability | 10 |
| Affected Users | 10 |
| Discoverability | 10 |
| Total | 50 |

### Summary

While reviewing the configuration of externally facing network devices, the consultant noted that the inbound (or ingress) rule set allowed for an excessive number of applications or ports. This is a critical risk to the organization.

### Proof

### Remediation

It is recommended that <<COMPANY>> adopt a DENY unless required policy for inbound traffic.

|  |  |
| --- | --- |
| DREAD Score |  |
| Damage Potential | 10 |
| Reproducibility | 10 |
| Exploitability | 10 |
| Affected Users | 10 |
| Discoverability | 10 |
| Total | 50 |

### Summary

The following server endpoints use an insecure Java RMI endpoint allowing for unauthenticated remote code execution.

Quoting the exploit discussion from [1], the default configuration of the RMI Registry and RMI Activation services, which allow loading classes from any remote (HTTP) URL. As it invokes a method in the RMI Distributed Garbage Collector which is available via every RMI endpoint, it can be used against both rmiregistry and rmid, and against most other (custom) RMI endpoints as well. Note that it does not work against Java Management Extension (JMX) ports since those do not support remote class loading, unless another RMI endpoint is active in the same Java process.

RMI method calls do not support or require any sort of authentication.

### Proof

### Remediation