



Implementing Vulnerability Assessment Tool as an On-Premises configuration

Christopher Saletta

2019 Laurea



Laurea University of Applied Sciences

Implementing Vulnerability Assessment Tool as an On-Premises configuration

Christopher Saletta
Business Information Technology
Bachelor's Thesis
September 2019

Degree Programme in Business Information Technology
Bachelor's thesis

Christopher Saletta

Implementing Vulnerability Assessment Tool as an On-Premises configuration

Year	2019	Pages	28
------	------	-------	----

The purpose of this thesis project was to find a suitable vulnerability assessment application, that offers comprehensible features that allows the client to scan for vulnerabilities and exploits in their IT infrastructure. The objectives included comparing different options with the requirements set by the client. These requirements included implementing the software on premises instead of a cloud-based SaaS (Software as a Service) model, being easy to use and quick to set up as well as having extensive options for reporting.

The client company is a Finnish software company who offer services for the social and healthcare industry in Finland, as well as a subsidiary company based in Sri Lanka. The services offered by the client reach over 2 million users in Finland, and are used by public and private healthcare centres, military as well other governmental institutions. The client benefits from this project by having an extra layer of security to help safeguard its infrastructure if being implemented fully into the existing environment.

The research part of this project was conducted as a case study. The problem the client had was the lack of in-house tools to search for and find exploits or vulnerabilities in their IT infrastructure, with the only options previously being audits by outside parties. The client had several requirements on what the implemented tool should have. The most important requirement being to have the vulnerability management tool be installed on premises, due to requirements by the clients' customers to minimize use of cloud-based platforms. This in turn meant that a considerable amount of time had to be used to research options that were possible to have on premises.

The results of this project were positive. The possibility of testing Nexpose on a separate environment the client uses for testing purposes for their different products before being pushed into production meant that valuable information was gathered. The software was able to find between 10-30 different vulnerabilities in many assets in the testing environment, and remediation plans as well as executive style reports were able to be generated by the data. This in turn means that the implementation of Nexpose into the clients' IT infrastructure could benefit the client greatly, with having the desired extra layer of security in-house.

Keywords: Cybersecurity, InfoSec, Exploitation,

Table of Contents

1	Introduction	5
1.1	Client Company	5
1.2	Purpose & Goals of this thesis project	5
2	Theoretical Framework and Research Methodology.....	6
2.1	ISACA Vulnerability Assessment	6
2.2	CVE	6
2.3	Research and Development Methodology.....	7
3	Vulnerability assessment Product requirements and comparison	7
3.1	Client Requirements.....	7
3.2	Product listing	8
3.3	F-secure Radar	8
3.4	Nexpose.....	9
3.5	OpenVas	9
3.6	Nessus.....	10
3.7	Product Comparison Results	10
4	Description of Testing Environment.....	11
4.1	Scanning frameworks included in Nexpose	14
5	Scanning results	16
5.1	Remediations	20
5.2	Scanning Results Summarized	21
6	Project results	22
7	Conclusions	22
8	References.....	23
9	Tables	24
10	Figures.....	25
11	Appendices	26

1 Introduction

The world is moving towards deeper levels of digitalisation in almost every area of modern life. Businesses, government entities, schools, military and healthcare institutions are all being digitised. This digitalisation of everything around us has large positive effects. We are able to advance development of new and exciting technologies at an ever-expanding pace, allow companies to grow their global reach easily as well as assist other people in any part of the world due to our connected society. This in turn means that there are also many more ways of exploiting and damaging the services and technologies we use if they are not properly secured against cyber threats, either due to internal or external factors.

Companies have to constantly stay secured and up to date with the everchanging security situations presented to them. This is why there are numerous companies specifically focused on offering security services and tools to companies and government entities, to reduce the possibility of exploitation. While it is very common these days to outsource security services to cloud-based Software as a Service (SaaS) models to make security development as straight forward as possible, sometimes companies might have in-house locally implemented security software because they wish to keep their infrastructure completely localised and secure, without cloud-based services.

1.1 Client Company

The client company for this thesis is a Finnish software company. The client is a leading Ehealth solution provider for the health and social care market in Finland, while also having a subsidiary company based in Sri Lanka. Their services reach over 2 million users in Finland in the public and private sector, their customers ranging from public healthcare centres, private practices to military institutions and universities.

1.2 Purpose & Goals of this thesis project

The purpose of this thesis is split into two goals. The first objective is to find possible options based on the requirements set by the client. The second objective is to test and assess the chosen vulnerability management services on a virtual test platform.

2 Theoretical Framework and Research Methodology

The theoretical frameworks this thesis is being based on are the Vulnerability Assessment guide by ISACA and the CVE database project by Mitre. The research methodology used is case study, in this case the research questions being “what vulnerability assessment tools would fit the company’s needs?”.

2.1 ISACA Vulnerability Assessment

Vulnerability assessment is major part of a working security system. Testing and remediating possible flaws in any infrastructure are often regarded a necessity for mandatory compliance to regulatory bodies as well as good practice in general. Vulnerability assessments can take many different shapes, like network-based scanning, host-based scanning (reviewing different system configurations), application scanning (often included in penetration scanning) and wireless scanning.

ISACA states that “Conducting an assessment does not necessarily improve security on its own; instead it reflects a snapshot of the environment at a particular point in time, and its goal is simply to identify and analyse weaknesses present in a technical environment” (ISACA 2017, 4)

The benefits of vulnerability assessments are very straightforward. The more insight you gain about different types of vulnerabilities, the better you can plan ahead for the future. For example, using network scans you can identify unknown assets connected to a network or find bad configurations that could open backdoors to tampering.

With the multitude of different types of vulnerability assessment software, it is easier to find one that suits the need of the user, albeit not necessarily as quickly as may be desired.

2.2 CVE

CVE means Common Vulnerabilities and Exposures. It is a project launched in the late 1990’s by a non-profit company MITRE, who focus on researching and cataloguing exploits and vulnerabilities in software. The purpose of the CVE database is to standardize the identification of exploits and vulnerabilities. The ID’s allotted to different vulnerabilities allow security technicians to access necessary technical information gathered about the vulnerability in question in one place.

While the CVE project is sponsored by the American Department of Homeland Security, MITRE is in charge of maintaining and updating the database. The site also has the CVE compatibility Program, that is meant to promote the use of CVE identifiers by CVE numbering authorities, or CNA’s

The CVE isn't "just another vulnerability database. It is designed to allow vulnerability databases and other capabilities to be linked together, and to facilitate the comparison of security tools and services" (Armerding 2019). The reason why the CVE database is important to this project is due to the standardisation of identifying vulnerabilities being an important part of security development. If vulnerabilities are found in the clients testing environment, the database is of great use when trying to plan out possible fixes.

2.3 Research and Development Methodology

The research and development of this thesis is done using the case study method. The client has a problem that they need fixed, in this case the absence of in-house vulnerability management tools. The answer to this problem requires the research on what options could work in the confines of the clients' requirements as well as any requirements set by any agreements the client has with other organizations or authorities.

Case study method was chosen due to it being useful in doing in-depth analysis, as well as multi-faceted exploration of issues. Qualitative data is needed, due to the need to analyse different vulnerability assessment tools. In depth knowledge of how these vulnerability assessment tools work and what problems or limitations they have are an important part in finding a suitable tool for the client and as such, qualitative data needs to be found through a case study. Qualitative data is "Qualitative data in statistics is also known as categorical data. Data that can be arranged categorically based on the attributes and properties of a thing or a phenomenon." (Surendran. A. n.d)

The development of this thesis will follow a linear but flexible path. The research questions of "what vulnerability assessment tools would fit the company's needs?" and "are vulnerabilities found during testing" need to be clarified as well as properly researched. Once enough research has been conducted, choices made and data gathered, further steps toward sharing the conclusions can be made. As a baseline, a flexible 6-step linear by Robert K. Yin in the book "Case Study Research: Design and methods" is used to help guide development.

3 Vulnerability assessment Product requirements and comparison

Currently there are numerous different tools available to monitor and scan IT infrastructure and finding one is not simple. The client also has a set of requirements for the application they would wish to have

3.1 Client Requirements

The client company had a set of requirements by which the vulnerability assessment tool was to be chosen based off. The first requirement was for the tool to be available on premises,

instead of a cloud-based SaaS (Software as a Service) model, due to agreements under KATA-KRI.

The second requirement was to keep the budget under €5000 for a possible one-year licence, as a short-term test.

The third requirement was ease of access and good user experience design. The thought behind this was to make sure to find a tool that was accessible enough to not cost undue work-hours being bogged down with something that is too complicated for a small test. What was desired was something that was simple to set up but would still give valuable information of possible security issues.

The fourth requirement was easily configurable reporting. The client wanted to be able to quickly make different types of reports, for different personnel (executive level reports as well as reports more geared towards the development teams).

3.2 Product listing

Over a period of 4 weeks, research was done to list possible options, which was then followed with the arduous task of weighing the different tools against the different requirements by the client. The closest possible options are Openvas, Nexpose, F-Secure Radar and Nessus. Table 1 presents considerable options. OpenVas is an open-source tool whereas Nexpose, F-Secure Radar and Nessus are commercially available software with licence fees.

Table 1 Product list

OPEN SOURCE SOFTWARE	COMMERCIAL SOFTWARE
OpenVas	Nexpose
	F-Secure Radar
	Nessus

3.3 F-secure Radar

A demo was set up with F-secure to see how well Radar would work, though the demo was based off their cloud-based platform with limited functionality. It was not possible to use the demo environment to test the client company's infrastructure, only for pre-set assets that were in the testing environment. The demo was arranged for 14 days so enough data could be reviewed, and decisions made.

While the tool itself was very well made and intuitive to use, with very deeply rooted scanning procedures and solid reporting, Radar had to be dropped due to numerous factors. One big issue with moving forward with Radar was that while it was possible to get it installed on premise with a licence deal for €4000-€5000 per year for 130 assets. One asset being one unique IP address, the installation would have to be done by technicians from F-secure and a separate cost for the customer service meant that the costs of running Radar for one year would be over €20 000 (Kauhanen 2019. Pers. Com.). It was recommended to rather take their cloud-based SaaS model of Radar and save money, but as stated before this was not an option due to prior agreements with company clients. F-Secure Radar is a powerful and well-made tool, that is perfect for vulnerability management, but for this project it was out of reach due to budgetary constraints.

3.4 Nexpose

The other option on the list was Nexpose by American security company Rapid7. While Rapid7's main vulnerability management software is their cloud-based Insight platform, they also offer a scaled down version that is available on premises called Nexpose. Nexpose was referred as being easy to use, with deep scanning capabilities and highly modifiable reporting structures that can produce reports ranging from surface level reports for executive level situations, to more deeper levels of information for the purpose of security development.

After contacting Rapid7 about a possible demo or trial as well as pricing info, they informed that they offer 1 a month trial licence for free, with the opportunity to implement it wherever it was deemed necessary. This enabled deeper testing opportunities and was a positive surprise. Another benefit of Nexpose was its pricing structure.

A one-year licence was price was quoted to the author by L. Truelsen (2019. Pers. com.) at €3400-€3600 ranging from 130 to 150 assets including a year of customer service. The next step was to arrange an environment for the implementation of a Nexpose testing build that could be used to possibly find exploitable vulnerabilities. The decision was made to try out the software on a separate testing environment used by the clients' development team when developing new services or fixes to older services before being pushed into production.

To be sure that the assessment data is of any use, any possible vulnerabilities will be checked against the CVE database for confirmation.

3.5 OpenVas

OpenVas was not taken into larger consideration due to the complicated nature of its design. While it seems to be a powerful tool, the set-up of OpenVas to test was deemed too time consuming by the client that it was quickly ruled out in favour of the other available choices.

3.6 Nessus

On paper, Nessus seems like a perfect choice due to it being a widely used, powerful tool with a well-designed user experience. Nessus was one of the top choices for actual testing, but once Tenable, the developers of the tool, were contacted about a possible trial and pricing, they quoted a yearly price of €7000. This was deemed too high of a price, so Nessus was no longer taken into consideration

3.7 Product Comparison Results

Based on the client requirements, research on possible options and two short preliminary tests with Radar and Nexpose the following table can summarize the results.

Table 2 Product Comparison

Requirements	Nexpose	F-Secure Radar	Nessus	OpenVas
Ease of use (UX)	✓	✓	✓	✗
Price (<€5000)	✓	✗	✗	✓
Possibility of on-premises installation	✓	✓	✓	✓
Configurable reporting possibilities	✓	✓	✓	✓

Of the four most viable options, Nexpose was a clear choice. As mentioned before, the price for a 1-year licence was around €3400-€3600, with the other closest one being OpenVas, as it is an open-source software, it does not cost anything. The reason OpenVas was not chosen was that it is notorious for being very user unfriendly and difficult to set up. F-secure Radar and Nessus were both too expensive, at €25 000 for the former and over €7000 for the latter.

4 Description of Testing Environment

The final testing build was constructed in a virtual testing platform the company uses to test out new services, updates, fixes etc. before pushing them into production. The testing platform is run off VMware Vcloud Director, with the different testing environments segmented off into different branches. The Nexpose testing environment was installed into the “General Testing Cloud” branch. The virtual machine used in the testing process used Centos 7 Linux Distribution by Red Hat, with 2 virtual cores and 6 gigabytes of RAM. The figure below shows mock-up of the virtual environment. The other virtual machines running in the same cloud environment are running on different operating systems from Centos7 to Windows Server 2008 R2 to Debian Linux depending on what they are used for. The diagram below gives a simplified view how the Vcloud environment is set up.

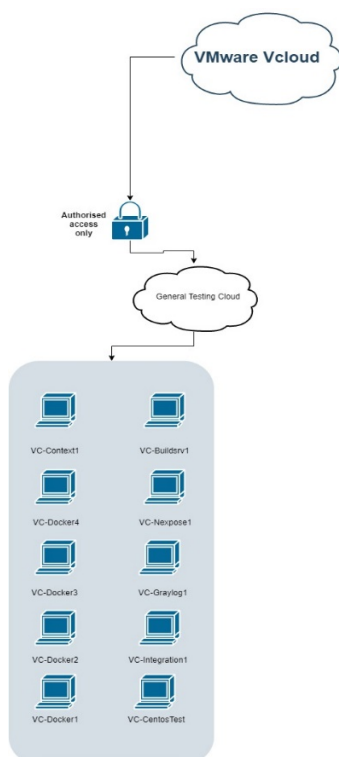


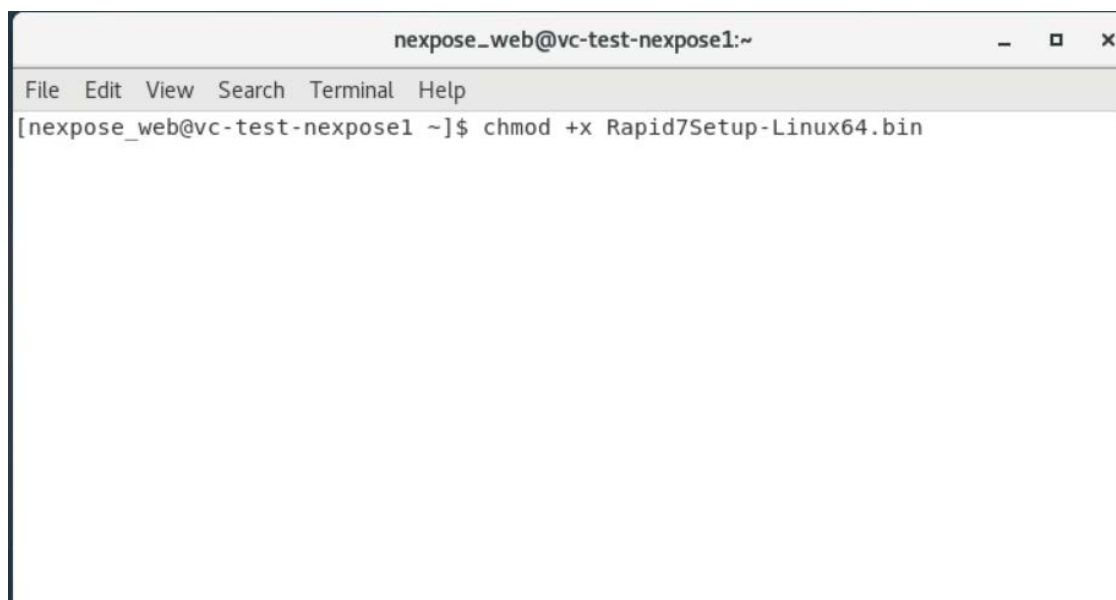
Figure 1 Vcloud basic environment

As the figure below shows (Figure 2), the installation of VC-Nexpose1 is done on Centos7. The next step was to make the Centos 7 operating system be more user friendly. This meant the installation of a separate desktop environment for the Linux distribution, in this case GNOME by the GNOME Project. Separate web browsers were also installed, including Mozilla Firefox as well as Google Chrome, to see how well the Nexpose dashboard works with different browsers.

Virtual Machine name:	<input type="text" value="vc-test-nexpose1"/> *
	A label for this VM that appears in VCD lists.
Computer name:	<input type="text" value="vc-test-nexpose1"/> *
	The computer name / host name set in the guest OS of this VM that identifies it on a network. This field is restricted to 15 characters for Windows. For non-Windows systems it can be 63 characters long and contain dots.
Description:	<input type="text" value="Nexpose Testipalvelin"/>
Operating System Family:	Linux
Operating System:	Red Hat Enterprise Linux 7 (64-bit)

Figure 2 VC-Nexpose1

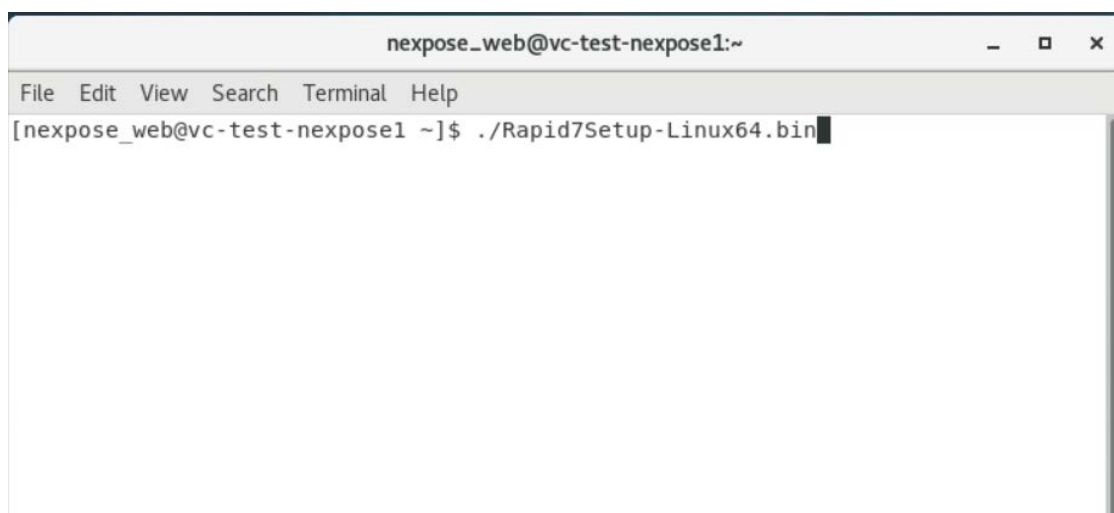
The installation of Nexpose was very straightforward, with installation started via the Centos 7 terminal. Using the command "Chmod +x Rapid7Setup-Linux.bin", as shown in figure 3, allows the installer file to be turned into an executable file.



```
nexpose_web@vc-test-nexpose1:~  
File Edit View Search Terminal Help  
[nexpose_web@vc-test-nexpose1 ~]$ chmod +x Rapid7Setup-Linux64.bin
```

Figure 3 Nexpose Linux install step 1

This is then proceeded with the command “./Rapid7Setup-Linux64.bin” in figure 4, that runs the installer.

A terminal window titled "nexpose_web@vc-test-nexpose1:~" with a menu bar (File, Edit, View, Search, Terminal, Help). The command prompt shows "[nexpose_web@vc-test-nexpose1 ~]\$./Rapid7Setup-Linux64.bin" with a cursor at the end of the command.

```
nexpose_web@vc-test-nexpose1:~  
File Edit View Search Terminal Help  
[nexpose_web@vc-test-nexpose1 ~]$ ./Rapid7Setup-Linux64.bin
```

Figure 4 Command to run the installer

The installer itself is simple and straightforward, with basic requirements informed in the beginning. To meet the basic installation requirements as shown in figure 5, some configuration with Linux is needed, for example turning off SELinux.

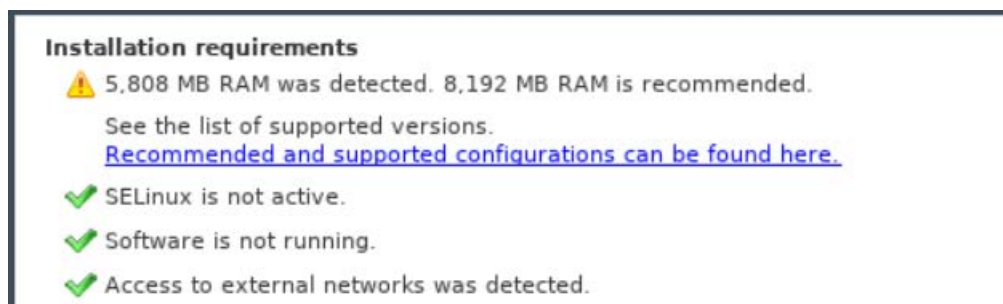


Figure 5 Installation requirements

The rest of the installation process is automated and once it is finished, the next step is to access the Nexpose Security Console. This is done by using a web browser to go to the machines' localhost address with the port 3780 and signing in with the default user and password of "NXadmin".

Once the Nexpose Security Console was accessible, the preparations for vulnerability scanning could begin. First step was to find a proper scanning template that would yield useful data about vulnerabilities

4.1 Scanning frameworks included in Nexpose

After researching articles and the numerous instructional posts on the Rapid7 website like the quick start guide as well as going through the different available templates that are pre-installed in Nexpose, ranging from simple asset discovery scans to Denial of Service scans up to Full Audit scans.

Scan Templates				
Name ^	Asset Discovery	Service Discovery	Checks	
<input type="radio"/> Denial of service	ICMP, TCP, UDP	Default TCP, Default ...	Custom	
<input type="radio"/> Discovery Scan	ICMP, TCP, UDP	Custom TCP, Custo...	Disabled	
<input type="radio"/> Discovery Scan - Aggressive	ICMP, TCP, UDP	Custom TCP, Custo...	Disabled	
<input checked="" type="radio"/> Exhaustive	ICMP, TCP, UDP	Full TCP, Default UDP	Safe Only	
<input type="radio"/> Full audit	ICMP, TCP, UDP	Default TCP, Default ...	Custom	
<input type="radio"/> Full audit enhanced logging without Web Spider	ICMP, TCP, UDP	Default TCP, Default ...	Custom	
<input type="radio"/> Full audit without Web Spider	ICMP, TCP, UDP	Default TCP, Default ...	Custom	
<input type="radio"/> HIPAA compliance	ICMP, TCP, UDP	Default TCP, Default ...	Safe Only	
<input type="radio"/> Internet DMZ audit	Disabled	Default TCP	Custom	
<input type="radio"/> Linux RPMs	ICMP, TCP, UDP	Custom TCP	Custom	

Figure 6 List of scanning templates

In the end, the scanning template that was used for initial testing was the Exhaustive scan, which is described as a thorough network scan of all systems and services uses only safe checks, including patch/hotfix inspections, policy compliance assessments, and application-layer auditing. An intensive scan that could take several hours, or even days, to complete, depending on the number of target assets. Scans run with this template are thorough, but slow.

The Exhaustive scan was deemed perfect due to it using safe checks to allow the scans to be run during work hours without it affecting any of the other virtual machines running in the IP scanning range as shown in figure 7 below.

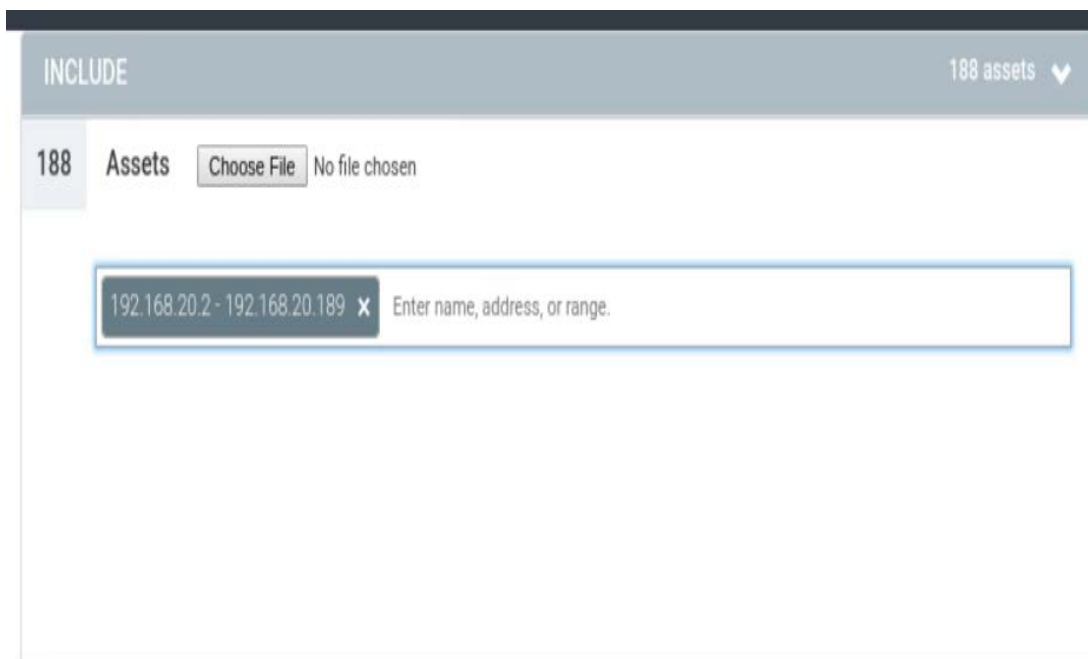














Figure 7 Example of the IP range used in the scanning. **NOTE: THIS IP RANGE IS NOT ACTUALLY USED IN THE TESTING ENVIRONMENT.** Due to security reasons the real IP range cannot be shown, even though it is not available outside the company's local network.

The IP range used, covered 188 unique assets. Each having its own hostname, operating system, users and applications.

5 Scanning results

Over the course of a month, 22 test scans and reports showed the basic information about the virtual machines running in the test environment segment, including numerous vulnerabilities, exploits and outdated software that were found on the assets in the network. Figure 8 shows the different information on operating systems, number of vulnerabilities and scanning dates.

Site	Operating System			Vulnerabilities	Risk ▼	Assessed	Last Scan	Delete
Test1	Microsoft Windows Server 2012 R2 Standard Edition	0	6	245	46,065	Yes	Thu Aug 8 2019	
Test1	Microsoft Windows Server 2012 R2 Standard Edition	0	7	229	42,494	Yes	Thu Aug 8 2019	
Test1	Microsoft Windows Server 2008 R2, Standard Edition SP1	0	6	61	25,638	Yes	Thu Aug 8 2019	
Test1	Microsoft Windows	0	9	78	24,660	Yes	Thu Aug 8 2019	
Test1	Microsoft Windows Server 2012 R2 Standard Edition	0	3	33	16,944	Yes	Thu Aug 8 2019	
Test1	Microsoft Windows Server 2012 R2 Standard Edition	0	1	21	8,980	Yes	Wed Jul 17 2019	
Test1	Debian Linux 8.0	0	0	11	5,501	Yes	Wed Jul 17 2019	
Test1	Microsoft Windows	0	1	12	4,009	Yes	Thu Aug 8 2019	
Test1	Linux 3.11	0	0	6	3,258	Yes	Tue Jul 16 2019	
Test1	CentOS Linux	0	0	0	0	No	Tue Jul 16 2019	

Rows per page: 10 of 4

Figure 8. 10 of the 188 assets scanned, with basic info. **Note: The names of the assets have been redacted due to security reasons.**

In addition to the basic data about the assets in the network segment, there were detailed information about what types of vulnerabilities, what categories they fall under and links to references. Figure 9 shows the different vulnerabilities by different risk categories.

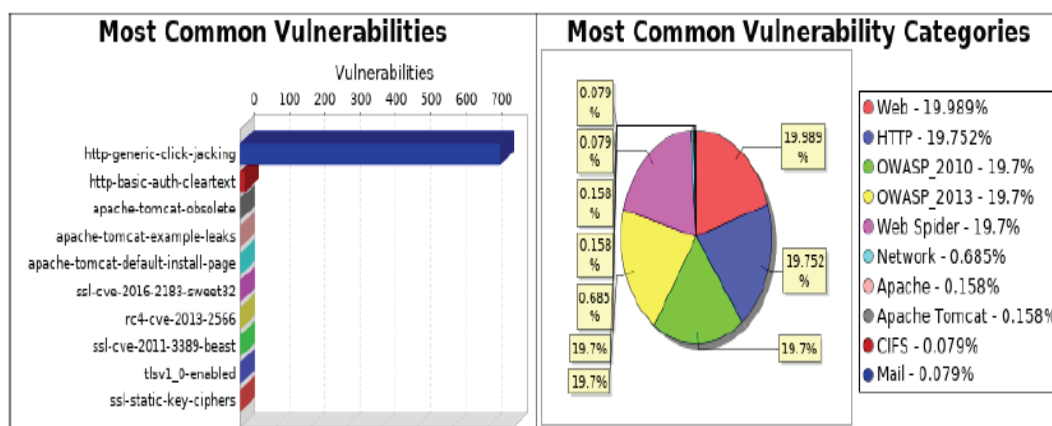
If a scan was running with authenticated access to the assets, there would be also information about what users have accessed the assets, what has been installed, system logs, possible malware and other user data that is stored on the assets. These scans were not set up to run with authenticated access to assets as it was deemed unnecessary for the testing of Nexpose, though if the software is to be implemented into the company's production infrastructure it might be more pertinent to have authenticated access to an asset that would need to be thoroughly scanned.

Risk Index	Risk Factors
16,880	<ul style="list-style-type: none"> •This device is in the Test1 site with normal importance. •5 critical vulnerabilities were discovered. •19 severe vulnerabilities were discovered. •9 moderate vulnerabilities were discovered. •10 DCE RPC services were discovered. •4 HTTP services were discovered. •One DCE Endpoint Resolution service was discovered. •One RDP service was discovered. •One CIFS service was discovered. •One HTTPS service was discovered. •One SMTP service was discovered.

Figure 9 Different types of vulnerabilities found in an asset. **Note: Asset names have been redacted due to security reasons**

The next picture (figure 10) shows in-depth information on the highest-risk vulnerabilities in the aforementioned asset. Many of them are basic vulnerabilities that should be patched out if they were in public use, but as these assets are used for testing purposes for the company's developers and are completely separate from the company's own production infrastructure, some of these vulnerabilities are known but are not needed to be fixed due to the assets being used for testing application fixes etc. These vulnerabilities are relevant according to the CVE database. For example, the vulnerability "RC4-cve-2013-2566" is a known vulnerability in the RC4 algorithm, where when used in the TLS and SSL protocols, has many single-byte biases which means that possible attackers can remotely conduct plaintext-recovery attacks. "They can exploit this by using statistical analysis of ciphertext in a large number of sessions that use the same plaintext". (MITRE 2013.)

The important factor here is that the Nexpose scans successfully found relevant vulnerabilities and provided the instructions and information on how to fix them. This was one of the major goals of this project.



There were 735 occurrences of the http-generic-click-jacking vulnerability, making it the most common vulnerability. There were 759 vulnerability instances in the Web category, making it the most common vulnerability category.

Figure 10 Screen capture of the common vulnerabilities from a report produced by the Nexpose Security Console

Figures 11 and 12 show the highest risk vulnerabilities found in 1 asset that is used by the User Experience team. For instance the vulnerability 4.2 “HTTP Basic Authentication Enabled” means that “HTTP Basic authentication (BA) implementation is the simplest technique for enforcing access controls to web resources because it doesn’t require cookies, session identifiers, or login pages; rather, HTTP Basic authentication uses standard fields in the HTTP header, obviating the need for handshakes.” (Chandel 2018)

What this means is that there is no proof of confidentiality when trying to access web resources as stated by Chandel (2018) “The BA mechanism provides no confidentiality protection for the transmitted credentials. They are merely encoded with Base64 in transit, but not encrypted or hashed in any way. HTTPS is, therefore, typically preferred used in conjunction with Basic Authentication. This asset is normally used to test out changes made to the UX design of different services offered by the client company. These vulnerabilities are scheduled to be fixed as soon as possible. “

As mentioned before, all the vulnerabilities found in these scans are not in the production infrastructure of the client company and thus do not present any threat to any services the company offers.

4. Highest Risk Vulnerability Details

4.1. Click Jacking ([http-generic-click-jacking](#))

Category	OWASP_2010, OWASP_2013, HTTP, Web, Web Spider
CVSSv2 score	4.3 (AV:N/AC:M/Au:N/C:N/I:P/A:N)
Risk Score	168,210
References	URL: https://www.owasp.org/index.php/Clickjacking

4.2. HTTP Basic Authentication Enabled ([http-basic-auth-cleartext](#))

Category	OWASP_2010, OWASP_2013, HTTP, Web, Web Spider
CVSSv2 score	6.5 (AV:N/AC:L/Au:S/C:P/I:P/A:P)
Risk Score	9,780
References	OWASP-2010: A9 , OWASP-2013: A6 , URL: http://tools.ietf.org/html/rfc2617

4.3. Apache Tomcat: Obsolete version ([apache-tomcat-obsolete](#))

Category	Obsolete Software, Apache, Web, Apache Tomcat
CVSSv2 score	10.0 (AV:N/AC:L/Au:N/C:C/I:C/A:C)
Risk Score	1,706
References	URL: https://tomcat.apache.org/tomcat-55-eol.html , URL: https://tomcat.apache.org/whichversion.html

4.4. Apache Tomcat Example Scripts Information Leakage ([apache-tomcat-example-leaks](#))

Category	XSS, Apache, Web, Apache Tomcat
CVSSv2 score	7.8 (AV:N/AC:L/Au:N/C:C/I:N/A:N)
Risk Score	1,473

4.5. Apache Tomcat default installation/welcome page installed ([apache-tomcat-default-install-page](#))

Category	Apache, Web, Apache Tomcat
CVSSv2 score	5.0 (AV:N/AC:L/Au:N/C:P/I:N/A:N)
Risk Score	1,176
References	OSVDB: 2117

Figure 11 Highest Risk Vulnerabilities

In the end, there were 696 distinct vulnerabilities in the 188 assets that were discovered in the segment. Some were very basic configuration related vulnerabilities, some more serious and high risk. None of these vulnerabilities were present in the production infrastructure of the company during its last audit by an outside auditing security team.

VULNERABILITIES 

EXCLUDE			RECALL			RESUBMIT					
<input type="checkbox"/>	Title			CVSS	CVSSv3	Risk	Published On	Modified On	Severity 		
<input type="checkbox"/>	Apache Tomcat: Important: Remote Code Execution on Windows (CVE-2019-0232)			9.3	8.1	387	Fri Apr 12 2019	Fri Jul 26 2019	Critical		
<input type="checkbox"/>	Apache Tomcat: Moderate: Denial of Service (CVE-2016-3092)			7.8	7.5	294	Thu Jun 23 2016	Tue Jun 18 2019	Critical		
<input type="checkbox"/>	Apache Tomcat: Low: CORS filter has insecure defaults (CVE-2018-8014)			7.5	9.8	586	Wed May 16 2018	Thu Aug 08 2019	Critical		
<input type="checkbox"/>	Apache Tomcat: Important: Remote Code Execution (CVE-2016-8735)			7.5	9.8	629	Wed Nov 23 2016	Tue Jun 18 2019	Critical		
<input type="checkbox"/>	Apache Tomcat Example Scripts Information Leakage			7.8		737	Mon Nov 01 2004	Tue Jun 18 2019	Critical		
<input type="checkbox"/>	Invalid CIFS Logins Permitted			7.5		747	Tue Jan 25 2005	Fri Jul 11 2014	Critical		
<input type="checkbox"/>	HTTP Basic Authentication Enabled			6.5		753	Wed Jan 01 1997	Thu Jun 20 2013	Severe		
<input type="checkbox"/>	SMB signing disabled			7.3		838	Mon Nov 01 2004	Wed Feb 21 2018	Severe		
<input type="checkbox"/>	Apache Tomcat: Important: Remote Code Execution (CVE-2017-12615)			6.8	8.1	435	Tue Sep 19 2017	Tue Jun 18 2019	Severe		
<input type="checkbox"/>	Apache Tomcat: Moderate: Security Manager bypass (CVE-2016-0763)			6.5	6.3	466	Tue Feb 23 2016	Tue Jun 18 2019	Severe		

Figure 12 Even more information on vulnerabilities and different risk scores

5.1 Remediations

The reports that were produced through the reporting module of Nexpose Security Console gave useful remediation info. The reports included “remediation plans”, as shown in figure 13, that gave time estimates, difficulty estimates as well as a structure on which to start the

vulnerability fixes. These reports were sent out to the technical team lead who helped plan out the possible fixes to the assets running with any high-risk vulnerabilities.

6.1. Remediation Plan for [REDACTED]

6.1.1. For Apache Tomcat

These vulnerabilities can be resolved by performing the following 17 steps. The total estimated time to perform all of these steps is 859 hours 25 minutes.

Use HTTP X-Frame-Options

Estimated time: 734 hours

Send the HTTP response headers with X-Frame-Options that instruct the browser to restrict framing where it is not allowed.

This will address 734 instances of the following issue: Click Jacking ([http-generic-click-jacking](#)).

Use Digest Authentication

Estimated time: 52 hours

Replace Basic Authentication with the alternative Digest Authentication scheme. By modern cryptographic standards Digest Authentication is weak. But for a large range of purposes it is valuable as a replacement for Basic Authentication. It remedies some, but not all, weaknesses of Basic Authentication. See RFC 2617, section 4, [Security Considerations](#) for more information.

This will address 13 instances of the following issue: HTTP Basic Authentication Enabled ([http-basic-auth-cleartext](#)).

Use Basic Authentication over TLS/SSL (HTTPS)

Estimated time: 52 hours

Enable HTTPS on the Web server. The TLS/SSL protocol will protect cleartext Basic Authentication credentials.

This will address 13 instances of the following issue: HTTP Basic Authentication Enabled ([http-basic-auth-cleartext](#)).

Figure 13 Remediation plan with time estimates. Note: asset hostname has been redacted due to security reasons.

5.2 Scanning Results Summarized

Table 3 shows that in total there were over 696 distinct vulnerabilities found in the 188 assets (or IP's) scanned, with 300 of them being categorized as "critical". These scans were conducted using the Exhaustive scanning framework included in Nexpose.

Table 3 Scanning results

Scanning results	Amounts
Number of distinct vulnerabilities found	696
Vulnerabilities in total	2000+
Number of assets in the testing environment	188

6 Project results

The project brought in positive results. The research conducted to find different options of vulnerability management software was time consuming but in the end two good choices were found.

F-Secure and Nessus are powerful tools that are easy to use, powerful and can be recommended for larger companies with a large IT infrastructure but they were both too expensive to be considered possible by the client. Openvas, while being an open-source tool, without licence fees, was way too difficult to set up and use.

For this project Nexpose is the clear choice. It was quick to set up and start working and easy to learn with nice sets of instructions, the price was within the allotted budget, it was available on-premises and had good reporting possibilities, so it ticked all the client requirements.

The scanning engines embedded in Nexpose are powerful but allow normal work to continue while scanning is underway, and find vulnerabilities ranging from basic configuration issues to larger patchable exploits that could cause major negative consequences if unfixed in a real production environment or local IT infrastructure.

During the evaluation of Nexpose in the separate testing environment the company uses, many vulnerabilities were found in almost all of the scanned assets. Out of 188 assets, 38 had numerous medium-to-high risk vulnerabilities that can be fixed easily with basic configuration of ports, patches to applications as well as operating system updates. While these do not pose any threat to the services the company produces, or to the IT infrastructure of the company, the client is urged to take into consideration the fixes found in the remediation plan reports generated from the Nexpose scans.

7 Conclusions

Finally, it is recommended that the company pursue the implementation of Nexpose into their IT infrastructure. Its licence price is well in the confines of the allotted budget, with the possibility of having it as a on premises service to stay under the agreements the company has with clients, as well as being easy to use and with the ability to quickly generate readable, concise reports or more in-depth plans for fixing found exploits.

Integrating Nexpose into the client's production infrastructure would greatly boost the level of security and give the client a heads-up on possible remediation opportunities over the coming years.

8 References

Electronic sources

Armerding, T. (2019). What is CVE, its definition and purpose?

<https://www.csoonline.com/article/3204884/what-is-cve-its-definition-and-purpose.html> Accessed 25.5.2019

MITRE. 2017. Become a CAN

https://cve.mitre.org/blog/index.html#August022017_Become_a_CNA Accessed 31.5.2019

J. Kauhanen, Key Account Manager. F-secure. Email to author. 15 July 2019. Personal Communication.

L. Truelsen, Channel Manager. Rapid7. Telephone conversation with the author. 10 August 2019. Personal Communication

Red Hat. n.d. Hello World!

<https://developers.redhat.com/products/rhel/hello-world#fndtn-windows> Accessed 26.6.2019

Rapid 7. n.d. Installing Nexpose

<https://nexpose.help.rapid7.com/docs/install> Accessed 10.7.2019

Rapid 7. n.d. Security Console Quick start guide

<https://nexpose.help.rapid7.com/docs> Accessed 10.7.2019

Rapid7. n.d. Scan template appendix

<https://nexpose.help.rapid7.com/docs/scan-templates#section-exhaustive> Accessed 10.7.2019

Chandel, Raj. (2018). Multiple Ways To Exploiting HTTP Authentication)

<https://www.hackingarticles.in/multiple-ways-to-exploiting-http-authentication/>

9 Tables

Table 1 Product list

OPEN SOURCE SOFTWARE	COMMERCIAL SOFTWARE
OpenVas	Nexpose
Metasploit Framework	F-Secure Radar
Retina CS	Nessus
Burp Suite Free Edition	Qualys

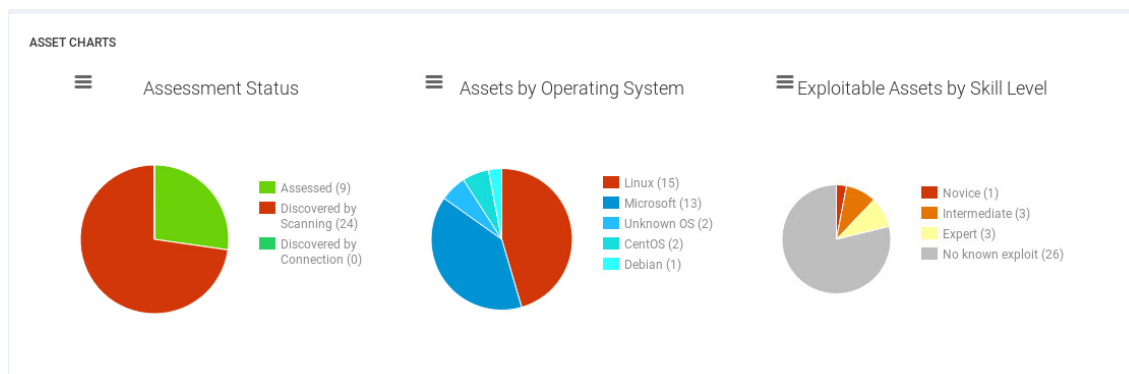
Table 2 Product Comparison

Requirements	Nexpose	F-Secure Radar	Nessus	OpenVas
Ease of use (UX)	✓	✓	✓	✗
Price (<€5000)	✓	✗	✗	✓
Possibility of on-premises installation	✓	✓	✓	✓
Configurable reporting possibilities	✓	✓	✓	✓

10 Figures

Figure 1 CNA world map, with participating countries helping to catalogue CNA's. MITRE, 2017, Become a CNA, accessed 31.5.2019, available: https://cve.mitre.org/blog/index.html#September112018_A_Look_at_the_CVE_and_CVSS_Relationship	Error! Bookmark not defined.
Figure 2 Nexpose virtual machine information	Error! Bookmark not defined.
Figure 3 VC-test-nexpose 1 running GNOME.....	Error! Bookmark not defined.
Figure 4 Nexpose Linux install step 1	12
Figure 5 Command to run the installer	13
Figure 6 Installation requirements.....	13
Figure 7 List of scanning templates	14
Figure 8 Example of the IP range used in the scanning. NOTE: THIS IP RANGE IS NOT ACTUALLY USED IN THE TESTING ENVIRONMENT. Due to security reasons the real IP range cannot be shown, even though it is not available outside the company's local network.	15
Figure 9 10 of the 188 assets scanned, with basic info. Note: The names of the assets have been redacted due to security reasons.	16
Figure 10 Different types of vulnerabilities found in an asset. Note: Asset names have been redacted due to security reasons	17
Figure 11 Screen capture of the common vulnerabilities from a report produced by the Nexpose Security Console	18
Figure 12 Highest Risk Vulnerabilities.....	19
Figure 13 Even more information on vulnerabilities and different risk scores.....	20
Figure 14 Remediation plan with time estimates. Note: asset hostname has been redacted due to security reasons.	21

11 Appendices
Appendix 1: Asset chart



Appendix 2: Another list of scanned assets

1. Discovered Systems

Node	Operating System	Risk	Aliases
[REDACTED]	Microsoft Windows Server 2012 R2 Standard Edition	44,785	[REDACTED]
[REDACTED]	Microsoft Windows Server 2012 R2 Standard Edition	41,271	[REDACTED]
[REDACTED]	Microsoft Windows Server 2008 R2, Standard Edition SP1	25,352	[REDACTED]
[REDACTED]	Microsoft Windows	24,284	[REDACTED]
[REDACTED]	Microsoft Windows Server 2012 R2 Standard Edition	16,876	[REDACTED]
[REDACTED]	Microsoft Windows Server 2012 R2 Standard Edition	8,931	[REDACTED]
[REDACTED]	Debian Linux 8.0	5,492	[REDACTED]
[REDACTED]	Microsoft Windows	3,970	[REDACTED]
[REDACTED]	Linux 3.11	3,253	[REDACTED]
[REDACTED]	Linux 3.11	0.0	[REDACTED]
[REDACTED]	Linux 3.11	0.0	[REDACTED]
[REDACTED]	Microsoft Windows Server 2012 R2 Standard Edition	0.0	[REDACTED]
[REDACTED]	CentOS Linux	0.0	[REDACTED]
[REDACTED]	Microsoft Windows Server 2012 R2 Standard Edition	0.0	[REDACTED]
[REDACTED]	Linux 3.11	0.0	[REDACTED]
[REDACTED]	Linux 3.11	0.0	[REDACTED]
[REDACTED]	Microsoft Windows Server 2012 R2 Standard Edition	0.0	[REDACTED]
[REDACTED]	Linux 3.11	0.0	[REDACTED]
[REDACTED]	Linux 3.11	0.0	[REDACTED]
[REDACTED]	Linux 3.11	0.0	[REDACTED]

Appendix 3: Second remediation list

REMIEDIATIONS			
BEST SOLUTIONS	APPLICABLE SOLUTIONS	SOLUTIONS BY VULNERABILITY	
			<p>Adjust the share permissions to be more secure</p> <p>Change the default page, or stop and disable the Tomcat server completely</p> <p>Configure SMB signing for Windows</p> <p>Disable HTTP DELETE method</p> <p>Disable HTTP OPTIONS method</p> <p>Disable ICMP timestamp responses</p> <p>Disable SSLv2, SSLv3, and TLS 1.0. The best solution is to only have TLS 1.2 enabled</p> <p>Disable TLS/SSL support for 3DES cipher suite</p> <p>Disable TLS/SSL support for RC4 ciphers</p> <p>Disable TLS/SSL support for static key cipher suites</p> <p>Disable autocomplete for all sensitive fields</p> <p>Disable insecure TLS/SSL protocol support</p> <p>Disable web directory browsing for all directories and subdirectories</p> <p>Fix Apache Tomcat v4.x Example Scripts Information Leakage</p> <p>Generate random Diffie-Hellman parameters</p> <p>Obtain a new certificate from your CA and ensure the server configuration is correct</p> <p>Replace TLS/SSL self-signed certificate</p> <p>Restrict access to NetBIOS</p> <p>Restrict database access</p> <p>Restrict invalid guest logins</p> <p>Stop Using SHA-1</p> <p>Upgrade to the latest version of Apache Tomcat</p> <p>Upgrade to the latest version of Apache Tomcat, Upgrade to the latest version of Apache Tomcat version 8.0</p> <p>Use Basic Authentication over TLS/SSL (HTTPS)</p>

Appendix 4: More remediations from asset scan

Stop Using MD5

Estimated time: 8 hours

Stop using signature algorithms relying on MD5, such as "MD5withRSA", when signing X.509 certificates. Instead, use the SHA-2 family (SHA-224, SHA-256, SHA-384, and SHA-512).

This will address the following issue: MD5-based Signature in TLS/SSL Server X.509 Certificate (tls-server-cert-sig-md5).

Disable SSLv2, SSLv3, and TLS 1.0. The best solution is to only have TLS 1.2 enabled

Estimated time: 1 hour

There is no server-side mitigation available against the BEAST attack. The only option is to disable the affected protocols (SSLv3 and TLS 1.0). The only fully safe configuration is to use Authenticated Encryption with Associated Data (AEAD), e.g. AES-GCM, AES-CCM in TLS 1.2.

This will address the following issue: TLS/SSL Server is enabling the BEAST attack (ssl-cve-2011-3389-beast).

Disable TLS/SSL support for RC4 ciphers

Estimated time: 1 hour

Configure the server to disable support for RC4 ciphers.

For Microsoft IIS web servers, see Microsoft Knowledgebase article [245030](#) for instructions on disabling rc4 ciphers.

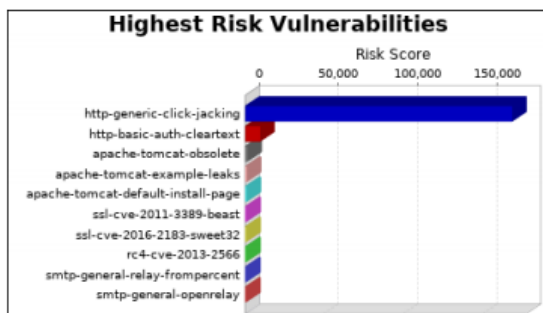
The following recommended configuration provides a higher level of security. This configuration is compatible with Firefox 27, Chrome 22, IE 11, Opera 14 and Safari 7. SSLv2, SSLv3, and TLSv1 protocols are not recommended in this configuration. Instead, use TLSv1.1 and TLSv1.2 protocols.

Refer to your server vendor documentation to apply the recommended cipher configuration:

ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-

Appendix 5: Executive Overview

3. Executive Overview



The http-generic-click-jacking vulnerability poses the highest risk to the organization with a risk score of 168,210. Risk scores are based on the types and numbers of vulnerabilities on affected assets.