Vulnerability Assessment

NDUS – North Dakota University System – Oct. 2014

Document ID: NDUS.docx

December 31, 2014

Submitted to:

NORTH DAKOTA UNIVERSITY SYSTEM
ACCESS. INNOVATION. EXCELLENCE.

Prepared by:

CDI L.R. Kimball
Architecture · Engineering · Communications Technology

TCS TeleCommunication Systems
Enabling Convergent Technologies®

Table of Contents

List of Exhibits

1. Executive Summary

Information technology (IT) security practices are critically important for the North Dakota University System and its institutions to protect large amounts of sensitive and confidential information that are stored on their computer systems, including information for more than 47,000 students and 11,000 faculty and staff . Universities are attractive targets for computer hackers because they traditionally have a strong culture of academic freedom that values open access to information and a free exchange of ideas. By providing numerous computer labs and high-capacity internet access that allows for the exchange of information at high speeds, universities not only accommodate their many users, but also create an attractive target for computer hacking. University IT security problems are occurring more often through weaknesses in network and web-based computer programs and (applications) as well as via social engineering techniques.

On behalf of the North Dakota State Auditor and the North Dakota University System, from October 1 to October 31, 2014, Team Kimball (the team) carried out external and internal vulnerability assessments on the networks associated with the North Dakota University System (NDUS). These networks consisted of the following campuses as well as NDUS networks in the listed locations: Bismarck State College (BSC), Dakota College at Bottineau (DCB), Dickinson State University (DSU), Lake Region State College (LRSC), Mayville State University (MASU), Minot State University (MISU), North Dakota State College of Science (NDSCS), North Dakota State University (NDSU), NDUS Offices (Fargo, Bismarck, Grand Forks), University of North Dakota (UND), Valley City State University (VCSU), Williston State College (WSC)

External assessments were conducted with no privileges in order to mimic anyone surfing the Internet. External network access as well as externally facing web applications were evaluated for each of the campuses. The majority of internal assessments were conducted with the same access and privilege level a student or a member of the faculty would have within the university system. In some cases as part of the internal assessment Team Kimball was provided with additional access to reach internal network components for evaluation. The scans were configured to check for vulnerabilities on any host that was controlled by the campus. Vulnerabilities were identified but no exploitation of the vulnerabilities that were detected was attempted. Access to certain systems was however obtained due to default login credentials being utilized.

Key findings at each of the campuses can be summarized into the following twelve categories. Each of the findings is provided in more detail within this report.

1. Unsupported Operating Systems
2. Missing software patch or required software upgrade
3. Easily guessed or Default Credentials
4. Unsupported Web Server
5. Well Known Exploits
6. Publicly Accessible Web Applications
7. Lack of Firewall / NAT

8. Cross-site Scripting
9. Cleartext submission of Password
10. Session Token in URL
11. SQL Injection
12. Serialized Object in HTTP message

All found vulnerabilities were rated on a scale from Critical to Low based on the common vulnerability scoring system (CVSS). Details associated with each of the findings as well as remediation guidance were provided to NDUS and to each of the campuses. The 12 findings listed above were all considered Critical or High. Appendix A of this document contains a response from NDUS to the findings of the vulnerability assessment.

2. Introduction and Background

Information technology (IT) security practices are critically important for the North Dakota University System and its institutions to protect large amounts of sensitive and confidential information that are stored on their computer systems, including information for more than 47,000 students and 11,000 faculty and staff. Universities are attractive targets for computer hackers because they traditionally have a strong culture of academic freedom that values open access to information and a free exchange of ideas. By providing numerous computer labs and high-capacity internet access that allows for the exchange of information at high speeds, universities not only accommodate their many users, but also create an attractive target for computer hacking. University IT security problems are occurring more often through weaknesses in network and web-based computer programs and (applications) as well as via social engineering techniques

IT security violations have occurred both in North Dakota and other states. For example:

- In February, 2014 an NDUS IT system was inappropriately accessed, which could have exposed the personal information of more than 290,000 current and past students and employees. Officials learned that unauthorized access was initially gained in October 2013. An investigation involving law enforcement and an outside forensics group revealed that the unauthorized party – thought to be based outside of the U.S. – was likely not going after the data, but instead was leveraging the processing power of the server to attack other computers and systems, according to the NDUS website.
- In August, 2014 eight North Dakota State University employees had their paychecks stolen by online scammers. It happened during a payroll cycle when the employees fell victim to what's known as a phishing scam, which involves emails pretending to be from official sources. "The email asked employees to click on a link and verify their information for payroll distribution, and eight employees responded," NDUS said in a statement. "Unfortunately, those employees' paychecks were then re-directed to the scammer's account. They reported the incidents, which were then reported to authorities." North Dakota State University covered about $20,600 in lost wages for employees who fell victim to this email scam according to a report presented to state lawmakers that included previously unreleased details about the so-called "spear phishing" attack.
- In September, 2014 North Dakota State College of Science Information Technology Services department had been alerted to malware activity on a number of NDSCS-owned computers in Wahpeton and Fargo and took immediate steps to ramp up security on its systems. Personal information such as names, Social Security numbers and mailing addresses of more than 15,000 current and former students and employees were contained on some of the affected computers. Those whose information was found were notified of the incident. After the malware was discovered, immediate action was taken to secure NDSCS systems. This included conducting a thorough internal investigation by NDSCS and North Dakota University System Information Technology experts. Law enforcement was contacted, and key systems were sent to a national forensic organization to confirm the analysis.

- University breaches have been on the rise recently. The University of Maryland and Indiana University both recently announced incidents involving hundreds of thousands of victims, with the Maryland occurrence also being the result of an attack.

IT security is essential to help campuses comply with federal laws and regulations designed to protect sensitive information such as educational records, personally identifiable information, and financial aid records.

On behalf of the North Dakota State Auditor and the North Dakota University System, from October 1 to October 30, 2014, the team carried out external and internal vulnerability assessments on the networks associated with the North Dakota University System (NDUS). These networks consisted of the following campuses as well as NDUS networks: Bismarck State College (BSC), Dakota College at Bottineau (DCB), Dickinson State University (DSU), Lake Region State College (LRSC), Mayville State University (MASU), Minot State University (MISU), North Dakota State College of Science (NDSCS), North Dakota State University (NDSU), NDUS Offices (Fargo, Bismarck, Grand Forks), University of North Dakota (UND), Valley City State University (VCSU), Williston State College (WSC)

External assessments were conducted with no privileges in order to mimic anyone surfing the Internet. External network access as well as externally facing web applications were evaluated for each of the campuses. The majority of internal assessments were conducted with the same access and privilege level a student or a member of the faculty would have within the university system. In some cases as part of the internal assessment Team Kimball was provided with additional access to reach internal network components for evaluation. The scans were configured to check for vulnerabilities on any host that was controlled by the campus. Vulnerabilities were identified but no exploitation of the vulnerabilities that were detected was attempted. Access to certain systems was however obtained due to default login credentials being utilized.

3. Scope

Testing was performed on all networked devices within the ranges specified by each campus. External and Internal ranges were assessed. The scans checked for known vulnerabilities and weaknesses in the network and attached hosts and appliances. All relevant web applications were also audited using Burp Suite Professional vulnerability scanner. All detected vulnerabilities and weaknesses were documented, and guidelines for remediation were provided.

4. Methodology

The team performed an external and internal vulnerability assessment to determine which hosts were visible from outside of the NDUS and each of the institutions networks. The team followed the standard penetration test methodology for the security assessment but stopped at the point where the team would attempt to exploit the target systems. Exhibit 1 shows the methodology. The light blue boxes were completed as part of the assessment.

The team utilized the following tools to assess the network and networked devices:

- NESSUS (commercial version)
- Core Impact Pro
- BURP Suite Professional Vulnerability Scanner
- NMAP Network Scanner

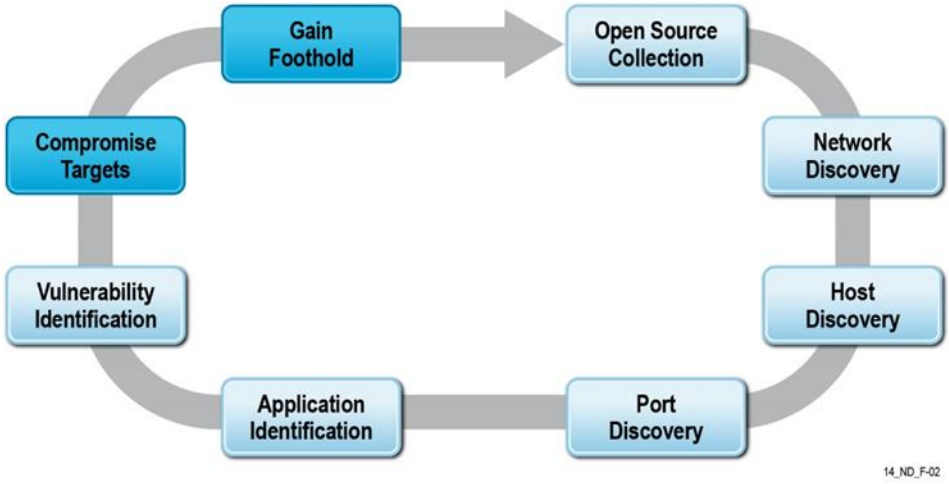

Exhibit 1. Team Kimball's Network Penetration Testing Methodology

The findings associated with each of the assessments: Internal, External, and Web Sites can be found in their respective sections.

5. Network Assessment Findings

The following findings represent a sanitized version of the vulnerability assessments that were presented to each of the NDUS campuses. Detailed assessment results as well as remediation for each of the findings has been provided to the campuses and many of them had resolved a significant portion of the critical items prior to the team leaving the site. This section contains findings that were common to both the external and internal network assessment. For the external assessment, the assessment team was connected to the internet with no special access. The majority of internal assessments were conducted with the same access and privilege level a student or a member of the faculty would have within the university system. In some cases as part of the internal assessment Team Kimball was provided with additional access to reach internal network components for evaluation. Four (4) findings are detailed in this section and these findings are representative of most of the colleges and universities that were assessed.

5.1. Finding 1 – Unsupported Operating Systems

The assessment team found unsupported operating systems at 10 of the 12 locations assessed. The operating systems ran the gambit from Windows, UNIX, MacOS, and Linux systems. The proliferation of unsupported and end-of-life products is an issue for many organizations and increases the effort required to minimize risk. As applications and operating systems reach their end-of-life (EOL), vendors stop offering support. Therefore, security and stability decrease, allowing attackers to exploit found vulnerabilities that will never receive a patch or security update. Patches, updates and security fixes will no longer be available, so identifying systems running EOL operating systems and applications is an important part of assessing and minimizing organizational risk

End of Support for XP SP2 (x86) and Windows 2000

Windows 2000™ is no longer supported by Microsoft. Its mainstream and extended support ended on June 30, 2005, and July 13, 2010, respectively. The Service Pack support for the 32-bit edition of Windows XP™ SP2 was retired on July 13, 2010 and the 64-bit edition of XP SP2 was retired on April 8, 2014.

Consumers, business users, and software developers using Windows 2000™ and Windows XP™ SP2 (x86) will no longer receive updates for security fixes and non-security hotfixes.

The Risks in Using Unsupported Operating Systems

There are risks in using Windows 2000™ and XP SP2 (x86) because consumers will no longer receive product support, bug fixes, and patch releases. Any known and unknown vulnerabilities affecting the unsupported operating systems create a risk of exploitation or data breaches from attackers on the vulnerable OS.

Other risks from using Windows XP™ SP2 (x86) and Windows 2000™ occur whenever malware creators release malicious codes targeting unsupported and unpatched operating systems. Over time, the software developers and security software vendors offering protection for an unsupported OS will also stop providing detection signatures and product support. With that in mind, any malware targeting old OS puts an organization at risk of data loss or a security breach. The worst scenario is when critical and sensitive data is stolen by malware attackers.

In Secunia's Half Year Report 2010, Apple OS is the top vendor with the most vulnerabilities. Oracle comes in second place, while Microsoft holds the third position. Other vendors in the list are HP, Adobe Systems, IBM, VMWare, Cisco, Google, and Mozilla. The report also states, "On average, 10 vendors are responsible for 38% of the vulnerabilities per year." This is a cause for concern because consumers have supported platforms or applications installed on the computers that depend on product support, security fixes, and non-security hotfixes to help protect the computer and data. The use of operating systems that are no longer supported will almost always provide attackers with the ability to exploit the targeted systems.[1]

Recommendations for Finding 1 - Unsupported Systems

1. Where possible move from unsupported versions of operating systems to supported versions.
2. For systems where this is not possible or where the cost is too high, consider defense in depth strategies to mitigate risk to these systems:
    a. Shutdown ports and applications not required
    b. Limit access to the machine
    c. Segregate the machine where possible
3. The following operating systems should not be allowed within the NDUS network. If a system is required, a waiver should be provided and defense in depth strategy outlined for protection of the machine and the attached networks.[2]

    - Macintosh System Software (any version)
    - Mac OS 9 and below
    - Microsoft Windows XP Professional and below
    - All 16-bit Windows releases (Windows 98, Window 95 and Microsoft DOS)
    - Solaris 9 / SunOS 5.9 and below
    - AIX 6.1 and below
    - Debian 5.0.x and below
    - FreeBSD 6.x and below
    - Red Hat Enterprise Linux 2.1 and below
    - SUSE Linux Enterprise 11 and below
    - Ubuntu 13.10 and below (LTS version 12.04 is still supported)

5.2. Finding 2 – Missing Software Patch or Required Upgrade

It is imperative that software patches and software upgrades are applied in a timely manner, particularly those that are linked to application security. At the same time, it is important that the IT teams have sufficient time to evaluate the patches and upgrades to determine if their specific

[1] PDF Format of Secunia Half Year Report 2010: http://secunia.com/gfx/pdf/Secunia_Half_Year_Report_2010.pdf
[2] http://blogs.microsoft.com/on-the-issues/2013/10/29/new-cybersecurity-report-details-risk-of-running-unsupported-software/

mix of applications will potentially have issues with the upgrade. This also allows the IT staff to ensure that their customer base does not have an adverse reaction to the patch.

Patch Management is an important IT security practice designed to proactively prevent the exploitation of vulnerabilities on system devices. The expected result is to reduce the time and money spent dealing with vulnerabilities and their exploitation. Taking a proactive approach to patch management can reduce or eliminate the potential for exploitation and involve considerably less time and effort than responding after a vulnerability has been exploited.

This item was the largest finding in the assessment and contributed to the largest reported vulnerability. It is imperative that each of the campuses have a program in place to identify required patches and then to implement them in a timely manner. They must also have a system in place to prioritize those that may not be critical but are well known vulnerabilities that need to be addressed.

Recommendations for Finding 2 – Missing Patches or Upgrades

1. Ensure that all campuses are running Nessus or equivalent tools for vulnerability assessment. This will allow them to determine what patches are required and be in a better position to provide prioritization associated with patching.
2. All campuses must apply all applicable hardware, software, and applications patches in a reasonable timeframe based on the severity of the issue. NDUS and the campuses should define the severity of the issue based on their current policies and procedures and risk associated with the software.

   Typical Patch Timelines:

   - Critical – 1 week
   - High – 45 days
   - Medium/Low – up to 1 year
3. Ensure a patch management program is in place that is tracking systems that are affected and timeline to resolution.
4. NDUS and campuses should evaluate commercially available patch management products to expedite patching and updates. Some commercially available products include:

   Commercially Available Patch Management Products:

   - ManageEngine Desktop Central (Win/Mac/3rd)
   - Symantec Altiris (Win/Mac/Linux)
   - Dell Kace (Win/Mac/Linux/3rd)
   - GFI LanGuard (Win/Mac/Linux/3rd)

5.3. Finding 3 – Easily Guessed or Default Credentials

Passwords are instrumental in the protection of data, systems, and networks. For example, passwords are used to authenticate users of operating systems and applications such as email, labor reporting, and remote access. In addition, passwords are often used in less visible ways;

for example, a biometric device may generate a password based on a fingerprint scan, and that password is then used for authentication.

Organizations should be aware of the drawbacks of using password-based authentication. There are many types of threats against passwords, and most of these threats can only be partially mitigated. Also, users are burdened with memorizing and managing an ever-increasing number of passwords. However, although the existing mechanisms for enterprise password management can somewhat alleviate this burden, they each have significant usability disadvantages and can also cause more serious security incidents because they permit access to many systems through a single authenticator. Therefore, organizations should make long-term plans for replacing or supplementing password-based authentication with stronger forms of authentication for resources with higher security needs.

During our assessment, every campus was found to have systems with easily guessed or default credentials. These systems are goldmines for hackers as this provides easy access to the campuses internal network and may allow a hacker to move around (pivot) and gain additional footholds within the organization at will. In addition, if they are able to gain access to administrator level accounts the attackers will have full access to the system and any files or network access associated with the account.

Recommendations for Finding 3 – Easily Guessed or Default Credentials

1. Create a password policy that specifies NDUS password management related requirements
2. Protect passwords from attacks that capture passwords (use HTTPS for web password submission or use multifactor authentication)
3. Configure password mechanisms to reduce the likelihood of successful password guessing and cracking
4. Determine requirements for password expiration based on balancing security and usability
5. Ensure systems are not deployed with default or out of the box user/password settings

5.4. Finding 4 – Unsupported Web Server

During the Assessment it was determined that five of the twelve campuses assessed had unsupported web servers operating in their networks. These represent a security issue based on the risk associated with discovered vulnerabilities that cannot be patched or remedied by the web server supplier. Since these applications are directly connectable via the internet, it is easy for an attacker to find these targets and exploit them.

Recommendations for Finding #4: Unsupported Web Server

1. Evaluate the need for the web server. If it is no longer being used shut it down.
2. Upgrade the server to a supported release.
3. If the server is no longer supported, look for a web server that is supported and will meet the requirements associated with your applications.

6. Internal Network Assessment

In addition to the above common findings, the following findings were unique to the internal network assessment:

6.1. Finding 5 – Systems with well-known exploits

While performing the internal assessment, nine out of the twelve campuses assessed contained systems with well-known exploits. Systems were found with Heartbleed, Shellshock, and IPMI Cipher Suite Zero exploits. Each of these exploits has been in the news and patches have been available for some time. In addition, Nessus and other vulnerability assessment tools have supported identification of each of these vulnerabilities. These are called out because each of these vulnerabilities is known to be exploitable. It is important to note that these vulnerabilities were only found internal to the campuses that were assessed, however, if an attacker is able to breach the outer perimeter and then finds additional easily accessible machines internal to the network, it is then very easy to setup multiple points of presence which will make removal from the network much more difficult to detect and accomplish.

6.1.1. Heartbleed

The Heartbleed bug tricks a server into spilling out extra information from its memory. A server's memory often includes sensitive personal information, such as your passwords, credit card numbers, and other data you wouldn't want anyone else to see or have access.

This information is usually encrypted, which means it's translated to an indecipherable code when it's transferred between servers, but Heartbleed can decode this encryption and store the codes used to protect your data. Heartbleed takes advantage of a vulnerability in OpenSSL, a popular encryption standard used to power a giant chunk of the Web.

Heartbleed attacks a vulnerability in OpenSSL called Heartbeat, which is a means of calling out to a server to make sure the connection is secure. The Heartbeat message usually contains arbitrary data and length field denoting how many bytes of data are in the message. The server would then transmit that exact message back to the original sender to prove that the connection is secure. The Heartbleed bug involves an issue with the server reading the length field incorrectly, which in turn tricks your server into transmitting more data than it should without realizing it.

6.1.2. Shellshock

Shellshock, also known as Bashdoor, is a family of security bugs in the widely used Unix Bash shell, the first of which was disclosed on 24 September 2014. Many Internet-facing services, such as some web server deployments, use Bash to process certain requests, allowing an attacker to cause vulnerable versions of Bash to execute arbitrary commands. This can allow an attacker to gain unauthorized access to a computer system.

Attackers exploited Shellshock within hours of the initial disclosure by creating botnets of compromised computers to perform distributed denial-of-service attacks and vulnerability scanning. Security companies recorded millions of attacks and probes related to the bug in the days following the disclosure.

Shellshock could potentially compromise millions of unpatched servers and other systems.

6.1.3. IPMI Cipher Suite Zero

"The Intelligent Platform Management Interface (IPMI) is a standardized computer system interface used by system administrators for out-of-band management of computer systems and monitoring of their operation. It is a way to manage a computer that may be powered off or otherwise unresponsive by using a network connection to the hardware rather than to an operating system or login shell.

Recommendations for Finding #5: Systems with Well-Known Exploits

1. Patch all systems that are found with these exploits with appropriate vendor-supported patch.

7. External Network Assessment

The following findings are unique to the external network assessment.

7.1. Finding 6: Publicly Accessible Web Applications

A number of publicly accessible web applications (not part of the campus web pages) were found to be externally accessible as part of the university network. Many of these applications were manually evaluated by Team Kimball and were found to be configured with default or easily guessed username and password for the administrator of the application. In addition, many of these systems had a web interface that should be disabled or blocked from external (public) access. Some of the applications were found to store and allow access to personally identifiable information (PII). Team Kimball was also able to send traffic to these applications, which could lead to denial of service and prevent the legitimate use of the applications by students and staff.

Recommendations for Finding #6: Publically Accessible Web Applications

1. Systems with web applications or appliances should be disabled if not required and should not be publicly accessible. If they need to be publicly accessible proper authentication mechanisms should be in place.
2. Ensure systems are not setup with default user or administrator credentials. These should be changed before deployment of the device.

7.2. Finding 7: Firewall / NAT

During the internal assessment it was noted that, four of the twelve campuses evaluated utilize public IP addresses for internal routing within the campus. In some cases, a bridged firewall is in place to provide routing rules and to provide application level filtering. Network Address Translation (NAT) allows a network to be protected by a single host (the router). The router acts as the gateway to the Internet, separating and protecting the hosts downstream from potential threats on the Internet. Due to the configuration of these 4 campuses their networks can be considered external. This means that hosts residing on this network can be accessed by anyone connected to the Internet; an attacker does not have to be on campus to access these hosts. It should also be noted that the usage of a firewall or NAT is not an end all be all solution. These measures should be implemented as part of a multi-faceted security approach.

Recommendation for Finding #7: Firewall/NAT:

1. Install and configure a routing firewall to provide NAT to these networks so they are no longer publicly facing.
2. Implement a multi-faceted security effort.

8. Web Application Assessment Findings

The website for each of the campuses and NDUS was assessed. The assessment team evaluated the web sites using Burp Suite Professional Vulnerability Scanner. This tool audits the web site for any potential attack vectors by issuing a number of requests and processing the results it receives from the server.

8.1. Finding 8: Cross-Site Scripting

During the web application assessment, nine of the twelve campus websites had cross-site scripting related issues.

Reflected cross-site scripting vulnerabilities arise when data is copied from a request and echoed into the application's immediate response in an unsafe way. An attacker can use the vulnerability to construct a request which, if issued by another application user, will cause JavaScript code supplied by the attacker to execute within the user's browser in the context of that user's session with the application.

The attacker-supplied code can perform a wide variety of actions, such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes.

Users can be induced to issue the attacker's crafted request in various ways. For example, the attacker can send a victim a link containing a malicious URL in an email or instant message. They can submit the link to popular web sites that allow content authoring, for example in blog comments. The attacker can create an innocuous looking web site which causes anyone viewing it to make arbitrary cross-domain requests to the vulnerable application (using either the GET or the POST method).

The security impact of cross-site scripting vulnerabilities is dependent upon the nature of the vulnerable application, the kinds of data and functionality which it contains, and the other applications which belong to the same domain and organization. If the application is used only to display non-sensitive public content, with no authentication or access control functionality, then a cross-site scripting flaw may be considered low risk. However, if the same application resides on a domain which can access cookies for other more security-critical applications, then the vulnerability could be used to attack those other applications, and therefore may be considered high risk. Similarly, if the organization which owns the application is a likely target for phishing attacks, then the vulnerability could be leveraged to lend credibility to such attacks, by injecting Trojan functionality into the vulnerable application, and exploiting users' trust in the organization in order to capture credentials for other applications which it owns. In many kinds of applications, such as those providing online banking functionality, cross-site scripting should always be considered high risk.

Recommendations for Finding #8: Cross-site Scripting

1. Input should be validated as strictly as possible on arrival, given the kind of content which it is expected to contain. For example, personal names should consist of alphabetical and a small range of typographical characters, and be relatively short; a year of birth should consist of exactly four numerals; email addresses should match a well-

defined regular expression. Input which fails the validation should be rejected, not sanitized.

2. User input should be HTML-encoded at any point where it is copied into application responses. All HTML metacharacters, including < > " ' and =, should be replaced with the corresponding HTML entities (< > etc).

8.2. Finding 9: Cleartext Password

Passwords submitted over an unencrypted connection are vulnerable to capture by an attacker who is suitably positioned on the network. This includes any malicious party located on the user's own network, within their Internet Service Provider (ISP), within the ISP used by the application, and within the application's hosting infrastructure. Even if switched networks are employed at some of these locations, techniques exist to circumvent this defense and monitor the traffic passing through switches.

The application should use transport-level encryption (SSL or TLS) to protect all sensitive communications passing between the client and the server. Communications that should be protected include the login mechanism and related functionality, and any functions where sensitive data can be accessed or privileged actions can be performed. These areas of the application should employ their own session handling mechanism, and the session tokens used should never be transmitted over unencrypted communications. If HTTP cookies are used for transmitting session tokens, then the secure flag should be set to prevent transmission over clear-text HTTP.

Recommendations for Finding #9: Cleartext Password

1. Replace HTTP web services with HTTPs version in instances where data must be protected.
2. Replace unsecured services, such as telnet and rlogin, with a secured SSH service. If you must operate unsecured command line services, it is recommended that you operate them within a secured tunnel like SSL/TLS or VPN.
3. Training for user awareness.

8.3. Finding 10: Session Token in URL

Sensitive information within URLs may be logged in various locations, including the user's browser, the web server, and any forward or reverse proxy servers between the two endpoints. URLs may also be displayed on-screen, bookmarked or emailed around by users. They may be disclosed to third parties via the Referrer header when any off-site links are followed. Placing session tokens into the URL increases the risk that they will be captured by an attacker.

Recommendations for Finding #10: Session Token in URL

1. The application should use an alternative mechanism for transmitting session tokens, such as HTTP cookies or hidden fields in forms that are submitted using the POST method.

8.4. Finding 11: SQL injection

SQL injection vulnerabilities arise when user-controllable data is incorporated into SQL database queries in an unsafe manner. An attacker can supply crafted input to break out of the data context in which their input appears and interfere with the structure of the surrounding query.

Recommendations for Finding #11: SQL Injection

The most effective way to prevent SQL injection attacks is to use parameterized queries (also known as prepared statements) for all database access. This method uses two steps to incorporate potentially tainted data into SQL queries: first, the application specifies the structure of the query, leaving placeholders for each item of user input; second, the application specifies the contents of each placeholder. Because the structure of the query has already been defined in the first step, it is not possible for malformed data in the second step to interfere with the query structure. The affected campuses should review the documentation for the database and application platforms to determine the appropriate APIs which can be used to perform parameterized queries. It is strongly recommended that the affected campuses parameterize every variable data item that is incorporated into database queries, even if it is not obviously tainted, to prevent oversights occurring and avoid vulnerabilities being introduced by changes elsewhere within the code base of the application.

Organizations should be aware that some commonly employed and recommended mitigations for SQL injection vulnerabilities are not always effective.

One common defense is to double up any single quotation marks appearing within user input before incorporating that input into a SQL query. This defense is designed to prevent malformed data from terminating the string in which it is inserted. However, if the data being incorporated into queries is numeric, then the defense may fail, because numeric data may not be encapsulated within quotes, in which case only a space is required to break out of the data context and interfere with the query. Further, in second-order SQL injection attacks, data that has been safely escaped when initially inserted into the database is subsequently read from the database and then passed back to it again. Quotation marks that have been doubled up initially will return to their original form when the data is reused, allowing the defense to be bypassed.

Another often cited defense is to use stored procedures for database access. While stored procedures can provide security benefits, they are not guaranteed to prevent SQL injection attacks. The same kinds of vulnerabilities that arise within standard dynamic SQL queries can arise if any SQL is dynamically constructed within stored procedures. Further, even if the procedure is sound, SQL injection can arise if the procedure is invoked in an unsafe manner using user-controllable data.

8.5. Finding 12: Serialized Object in HTTP message

Our assessment identified that an application appears to submit a serialized object in a request parameter. This behavior can expose the application in various ways, including:

- Any sensitive data contained within the object can be viewed by the user.
- An attacker may be able to interfere with server-side logic by tampering with the contents of the object and re-serializing it.

- An attacker may be able to cause unauthorized code execution on the server, by controlling the server-side function that is invoked when the object is processed.

Actual exploitation of any code execution vulnerabilities arising from the application's use of serialized objects will typically require the attacker to have access to the source code of the server-side application. This may mitigate the practical impact of this issue in many situations. However, it is still recommended to fix the underlying vulnerability.

Recommendations for Finding #12: Serialized Object in HTTP Message

1. The best way to avoid vulnerabilities that arise from the use of serialized objects is not to pass these in request parameters, or expose them in any other way to the client. Generally, it is possible to transmit application data in plain non-serialized form, and handle them with the same precautions that apply to all client-submitted data. If it is considered unavoidable to place serialized objects into request parameters, then it may be possible to prevent attacks by also placing a server-generated cryptographic signature of the object into the same request, and validating the signature before performing deserialization or other processing on the object.

9. Conclusion

Team Kimball identified vulnerabilities in the NDUS network and within the NDUS campuses networks. Vulnerabilities were identified in all networks that were evaluated as part of this report. Individual reports were provided to each of the campuses as well as the NDUS network. These reports provided detailed explanations associated with each of the vulnerabilities as well as recommended remediation guidance.

This report identified the common vulnerabilities that were presented in each of the campus reports for the NDUS as a whole. It identifies that there are 12 finding areas that need to be addressed and has provided recommendation guidance for addressing each of the findings. More specific and detailed guidance has been provided to each individual campus as part of their individual assessment.

Summary of recommendations:

- Training
    - Cybersecurity Awareness Training for NDUS personnel.
    - Specialized training for IT staff on best practices in information security and use of vulnerability scanning tools
- Deploy Defense-in-Depth strategies (multifaceted security)
- Patch/update process and automated patching where applicable
- Implement quarterly Social Engineering testing to baseline and track user awareness and response
- Expand vulnerability assessment to include penetration testing in order to validate the results and demonstrate the ramifications of the vulnerabilities.

10. Points of Contact for this Report

Program Manager – Team Kimball

RD Porter
Email: rd.porter@lrkimball.com
Phone: 573-355-6881

Principal Architect - Enterprise Security & Protection

Erik Wallace
Email: ewallace@telecomsys.com
Phone: 410-280-1184

Security Analysts - Enterprise Security & Protection

Ryan Miller
Brent Borton
Kyle Jarrett
Brian Lemons
Phil Stoner

Appendix A: NDUS Response

NDUS Response to ***North Dakota University System Vulnerability Assessment*** dated 31 Dec 2014

Prepared by: Brad Miller, NDUS Director of Information Security

Date: January 12, 2015

Finding 1 – Unsupported Operating Systems

Recommendations for Finding 1 - Unsupported Systems

1. Where possible move from unsupported versions of operating systems to supported versions.
2. For systems where this is not possible or where the cost is too high, consider defense in depth strategies to mitigate risk to these systems:
    a. Shutdown ports and applications not required
    b. Limit access to the machine
    c. Segregate the machine where possible
3. The following operating systems should not be allowed within the NDUS network. If a system is required, a waiver should be provided and defense in depth strategy outlined for protection of the machine and the attached networks.

    - Macintosh System Software (any version)
    - Mac OS 9 and below
    - Microsoft Windows XP Professional and below
    - All 16-bit Windows releases (Windows 98, Window 95 and Microsoft DOS)
    - Solaris 9 / SunOS 5.9 and below
    - AIX 6.1 and below
    - Debian 5.0.x and below
    - FreeBSD 6.x and below
    - Red Hat Enterprise Linux 2.1 and below
    - SUSE Linux Enterprise 11 and below
    - Ubuntu 13.10 and below (LTS version 12.04 is still supported)

    Response: Many of the systems identified in the assessment have already been either removed from the NDUS CTS and campus networks or have been migrated to a supported operating system. Some of the systems identified with unsupported operating systems are embedded in vendor hardware or appliances, and therefore cannot be easily removed or migrated without significant cost. These devices have been, or are in the process of being, secured with additional defense in depth techniques as mentioned in the recommendations until they can be updated or migrated. NDUS CTS and campuses will need to develop policy and procedures regarding methods of identifying systems with unsupported operating systems and actions to take when found.

Finding 2 – Missing Software Patch or Required Upgrade

Recommendations for Finding 2 – Missing Patches or Upgrades

1. Ensure that all campuses are running Nessus or equivalent tools for vulnerability assessment. This will allow them to determine what patches are required and be in a better position to provide
2. All campuses must apply all applicable hardware, software, and applications patches in a reasonable timeframe based on the severity of the issue. NDUS CTS and the campuses should define the severity of the issue based on their current policies and procedures and risk associated with the software.
3. Ensure a patch management program is in place that is tracking systems that are affected and timeline to resolution.
4. NDUS and campuses should evaluate commercially available patch management products to expedite patching and updates.

   Response: NDUS Core Technology Services (CTS) is currently using Nessus Enterprise to identify vulnerabilities on systems within key areas of the CTS network. The CTS Nessus Enterprise vulnerability scanning system has been made available to all NDUS campuses to use for ongoing scanning of their campus networks. Most of the campuses have accounts on the CTS Nessus system or have their own on-campus vulnerability scanning capabilities. Also, CTS and many of the NDUS campuses have and use commercial patch management systems. NDUS CTS and the campuses will need to develop vulnerability management and patching policies, procedures, and capabilities to ensure consistent and expedited patching of systems and applications.

Finding 3 – Easily Guessed or Default Credentials

Recommendations for Finding 3 – Easily Guessed or Default Credentials

1. Create a password policy that specifies NDUS password management related requirements
2. Protect passwords from attacks that capture passwords (use HTTPS for web password submission or use multifactor authentication)
3. Configure password mechanisms to reduce the likelihood of successful password guessing and cracking
4. Determine requirements for password expiration based on balancing security and usability
5. Ensure systems are not deployed with default or out of the box user/password settings

Response: Section 3.4 of *NDUS Procedure 1901.2 Computer and Network Usage* currently outlines password management requirements for NDUS CTS and campuses. These requirements currently address password length, complexity, secrecy, and default accounts. These requirements will be examined as part of an ongoing policy review, and any identified deficiencies or recommended changes will be addressed. All identified web password submissions not using HTTPS have been remediated. CTS is currently piloting a multifactor authentication solution for critical systems and applications.

Finding 4 – Unsupported Web Server

Recommendations for Finding #4: Unsupported Web Server

1. Evaluate the need for the web server. If it is no longer being used shut it down.
2. Upgrade the server to a supported release.
3. If the server is no longer supported, look for a web server that provides supports and will meet the requirements associated with your applications.

   Response: Many of the unsupported web servers identified in the assessment have already been remediated. Ongoing Nessus vulnerability scans are needed to help identify any further, or future unsupported web servers, as well as other unsupported systems and applications, in order to secure, upgrade, or migrate them to a supported platform.

Finding 5 – Systems with Well-known Exploits

Recommendations for Finding #5: Systems with Well-Known Exploits

1. Patch all systems that are found with these exploits with appropriate vendor supported patch.

   Response: Most of the well-known exploits categorized as critical or high on systems identified in this assessment have already been patched or remediated. NDUS CTS and the campuses will need to develop vulnerability management and patching policies, procedures, and capabilities to ensure consistent, expedited, and ongoing patching of systems and applications.

Finding 6: Publicly Accessible Web Applications

Recommendations for Finding #6: Publically Accessible Web Applications

1. Systems with web applications or appliances should be disabled if not required and should not be publicly accessible. If they need to be publicly accessible proper authentication mechanisms should be in place.
2. Ensure systems are not setup with default user or administrator credentials. These should be changed before deployment of the device.

   Response: Efforts to evaluate and address publicly accessible systems and applications on NDUS and campus networks are currently underway. These publicly accessible systems do pose a significant risk. Policies, procedures, and capabilities will need to be developed by NDUS CTS and campuses to address the security of public-facing systems and web applications and determine whether or not they should be exposed to the external Internet. It will take considerable resources to remediate these issues due to the open network architecture at many of the campuses, the large number of systems currently on these networks, and the diverse population of faculty, staff, and students.

Finding 7: Firewall / NAT

Recommendation for Finding #7: Firewall/NAT:

1. Install and configure a routing firewall to provide NAT to these networks so they are no longer publicly facing.
2. Implement a multi-faceted security effort.

   Response: NDUS CTS and many of the NDUS campus networks do currently use Network Address Translation (NAT) to create 'private' networks, but some of the campuses do not, and in some cases, this results in public facing systems that do not need to be. NDUS CTS and campuses will need to evaluate their environment and identify and implement firewalls, NAT, or other security mechanisms to protect their network and devices.

Finding 8: Cross Site Scripting

Recommendations for Finding #8: Cross Site Scripting

1. Input should be validated as strictly as possible on arrival, given the kind of content which it is expected to contain. For example, personal names should consist of alphabetical and a small range of typographical characters, and be relatively short; a year of birth should consist of exactly four numerals; email addresses should match a well-defined regular expression. Input which fails the validation should be rejected, not sanitized.
2. User input should be HTML-encoded at any point where it is copied into application responses. All HTML metacharacters, including < > " ' and =, should be replaced with the corresponding HTML entities (< > etc).

   Response: See Response under Finding 12.

Finding 9: Cleartext Password

Recommendations for Finding #9: Cleartext Password

1. Replace HTTP web services with HTTPs version in instances where data must be protected.
2. Replace unsecured services, such as telnet and rlogin, with a secured SSH service. If you must operate unsecured command line services, it is recommended that you operate them within a secured tunnel like SSL/TLS or VPN.
3. Training for user awareness.

   Response: See Response under Finding 12.

Finding 10: Session Token in URL

Recommendations for Finding #10: Session Token in URL

1. The application should use an alternative mechanism for transmitting session tokens, such as HTTP cookies or hidden fields in forms that are submitted using the POST method.

   Response: See Response under Finding 12.

Finding 11: SQL Injection

Recommendations for Finding #11: SQL Injection

1. The most effective way to prevent SQL injection attacks is to use parameterized queries (also known as prepared statements) for all database access.

   Response: See Response under Finding 12.

Finding 12: Serialized Object in HTTP Message

Recommendations for Finding #12: Serialized Object in HTTP Message

1. The best way to avoid vulnerabilities that arise from the use of serialized objects is not to pass these in request parameters, or expose them in any other way to the client.

   Response for Findings 8, 9, 10, 11, and 12: NDUS CTS and campuses have already remediated, or are currently working to remediate, web application vulnerabilities identified in this assessment. NDUS CTS and campuses will need to develop an ongoing strategy to establish capabilities for identifying and remediating web application vulnerabilities. In addition, NDUS CTS and campuses will explore development platforms, secure coding standards, and training to facilitate secure application development. NDUS CTS and campuses will also explore exploit prevention technologies such as Web Application Firewalls (WAFs).