REN-ISAC SECURITY ALERT
Muzzling the POODLE (While Cleaning Up Other Related Vulnerabilities, Too)

October 22, 2014

To: IT Executives and Security Staff

EXECUTIVE SUMMARY

There have been many recent security advisories involving SSL/TLS encryption-related vulnerabilities, including most notably BEAST[1], the highly publicized Heartbleed[2] bug, the BERserk vulnerability,[3] and most recently, POODLE.[4,5,6]

Many sites have taken specific actions to address one or more of these high profile vulnerabilities, and we commend you for doing so -- your efforts help to protect your systems, the information they store and process, and your users.

However, if you *ONLY* take steps to address those particular high profile issues, your systems that rely on SSL/TLS are likely *still* insecure due to other equally serious (but less well-publicized) SSL/TLS-related issues.

This alert will help you to assess the status of your servers and to understand the steps you should be undertaking to fix the POODLE vulnerability in particular. Also, it will suggest what you should be doing in general to improve the quality of the SSL/TLS cryptography you're depending on.

Specifically we recommend that all sites should (1) identify their servers that use SSL/TLS, (2) assess the status of each of those servers, (3) update server cryptographic libraries, and (4) harden server crypto configurations.

TECHNICAL DETAILS

Step 1. Identify ALL Servers at Your Site That Use SSL/TLS

You can't check and fix SSL/TLS on servers that you don't know exist. Thus, a first step should be to ensure that you have an inventory of all campus servers using SSL/TLS. Here are some ways you may be able to identify these servers:

-- ASK: Ask campus system administrators to self-identify any servers or appliances that may be using SSL/TLS (note that while SSL/TLS is most commonly used to secure web servers, it may also be used to protect SMTP and POP/IMAP, and for other types of network traffic, too). Be sure to think about systems hosted off-site/in the "cloud" as well as systems connected via your own local network.

-- CHECK: If you centrally manage your certificates (rather than delegating certificate procurement to individual sysadmins or departments), check your certificate authority's management console to see a list of systems that have obtained certificates.[7]

---

[1] https://en.wikipedia.org/wiki/Transport_Layer_Security#BEAST_attack
[2] http://heartbleed.com/
[3] http://www.intelsecurity.com/advanced-threat-research/
[4] http://googleonlinesecurity.blogspot.nl/2014/10/this-poodle-bites-exploiting-ssl-30.html
[5] https://www.openssl.org/~bodo/ssl-poodle.pdf
[6] https://tools.ietf.org/html/draft-ietf-tls-downgrade-scsv-00
[7] Note that if you use wildcard certificates (e.g., certificates that cover all systems in a domain, or certificates that cover all systems within a subdomain), it may be hard to identify all systems using such a certificate.

-- SCAN: Use an active network scanning tool to probe for systems doing SSL/TLS. Although servers may offer SSL/TLS secured services on any port, scanning should initially focus on port 443/TCP, the normal secure web port. Note at this point we're not scanning for particular vulnerabilities, we're just trying to understand the population of servers that may need attention. Sites with centrally managed certificates may also be able to use a certificate discovery tool if one is provided as part of the certificate authority's management console or as a standalone tool. Other tools include, but are not limited to:

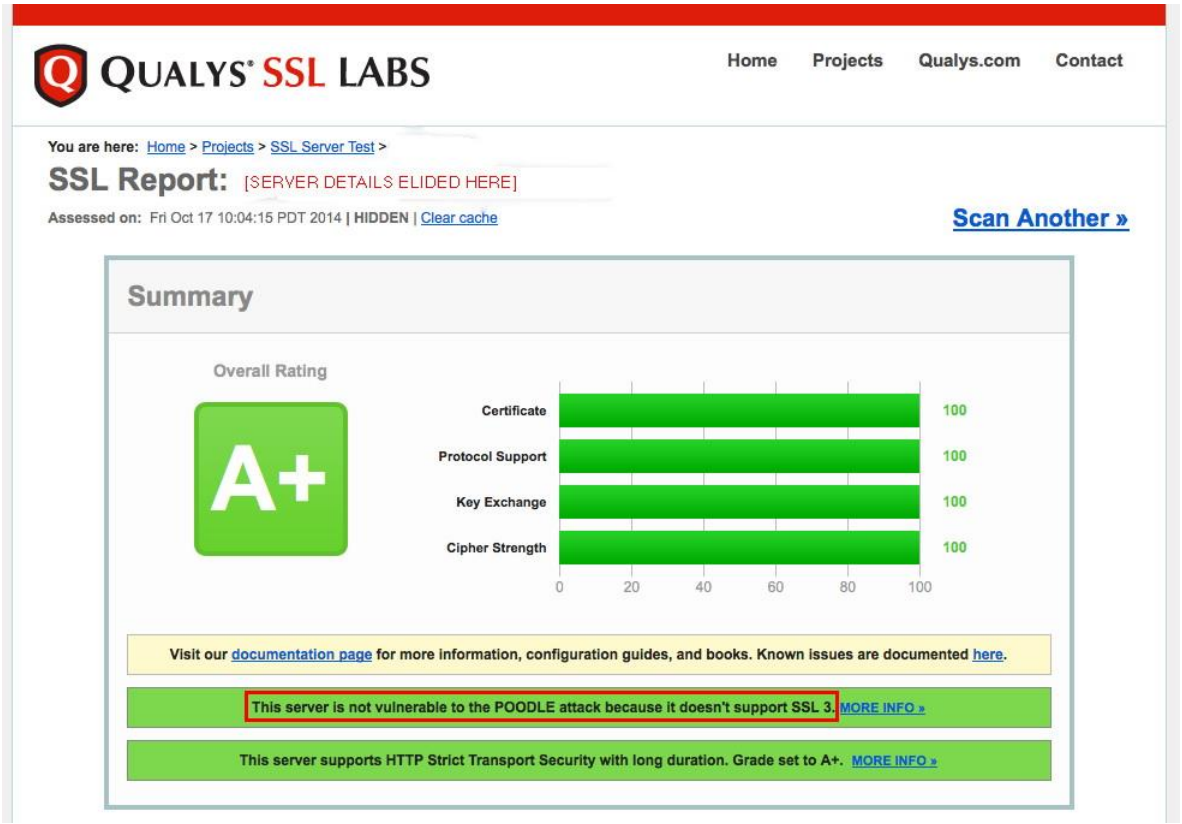
    Nmap: http://nmap.org/nsedoc/scripts/ssl-enum-ciphers.html
    Nessus: http://www.tenable.com/pvs-plugins/8548
    POODLE Prober: https://github.com/jeffmurphy/poodle-prober

-- SNIFF: Your intrusion detection system (Snort,[8] Bro,[9] etc.), Netflow traffic, or other passive methods may also help you to identify campus servers that are using SSL/TLS.

Step 2. Review the SSL/TLS Status of Each of Those Servers

Once you've identified all servers using SSL/TLS at your site, you should assess the status of those SSL/TLS implementations. This can be a complex process to undertake manually. Fortunately, however, you can use automated testing tools to do these tests for you. One highly-regarded tool is the Qualys SSL Labs SSL tester: https://www.ssllabs.com/ssltest/.

Note that you have the option to suppress public display of your server's results by ticking the box under the Domain Name box before hitting submit. After the scan completes, you will receive a summary grade, as well as a detailed report. If your server is not vulnerable to POODLE, that will be explicitly stated in the summary:

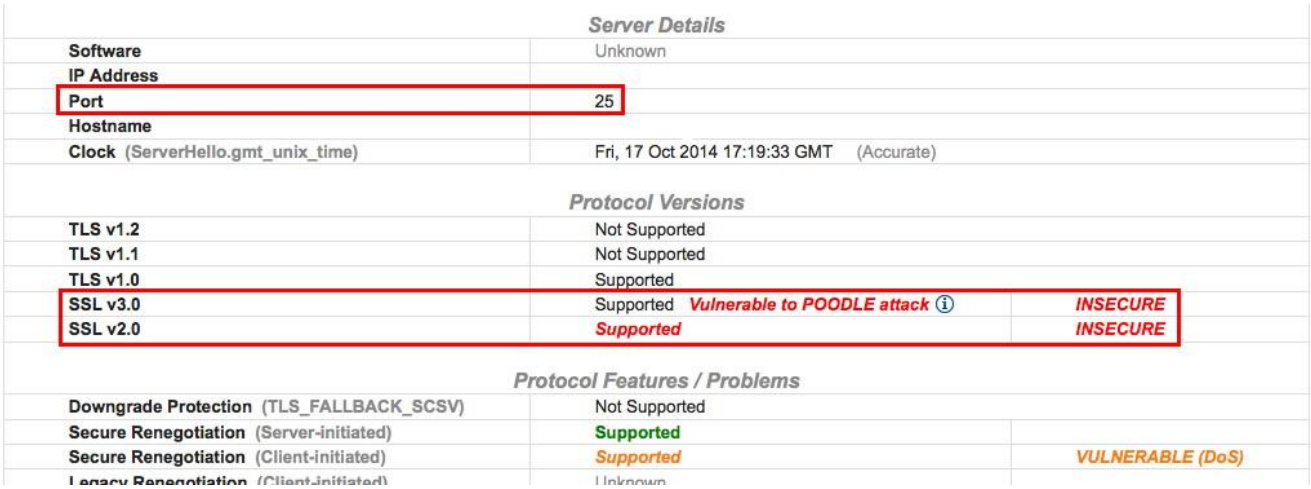


[8] https://www.snort.org/
[9] https://www.bro.org/

After this summary screen, you will also receive detailed information about all other aspects of your server's SSL/TLS configuration. **If your server is vulnerable to POODLE, or if you don't receive as strong an assessment as you might like, you should consider updating your server's SSL/TLS crypto libraries and/or tweaking its configuration.**

Q: "My server doesn't run on port 443. I can't figure out how to tell the Qualys tester to check some other port!"

A: If you have a server doing SSL/TLS on some other port, you may need to use a different tool. For example, you may want to try the Comodo tester, instead (see https://sslanalyzer.comodoca.com/). That tester will let you specify a hostname followed by a colon and a port number (e.g., serverfoo.example.edu:25). Partial output from that tester follows:



Server Details		
Software	Unknown	
IP Address		
Port	25	
Hostname		
Clock (ServerHello.gmt_unix_time)	Fri, 17 Oct 2014 17:19:33 GMT (Accurate)	
Protocol Versions		
TLS v1.2	Not Supported	
TLS v1.1	Not Supported	
TLS v1.0	Supported	
SSL v3.0	Supported *Vulnerable to POODLE attack*	*INSECURE*
SSL v2.0	*Supported*	*INSECURE*
Protocol Features / Problems		
Downgrade Protection (TLS_FALLBACK_SCSV)	Not Supported	
Secure Renegotiation (Server-initiated)	Supported	
Secure Renegotiation (Client-initiated)	*Supported*	*VULNERABLE (DoS)*
Legacy Renegotiation (Client-initiated)	Unknown	

This example output shows a mail transfer agent (MTA) that is vulnerable to the POODLE attack, and which even supports SSL v2.0 (which is as bad as SSL v3.0, as targeted by POODLE).

IMPORTANT NOTE: Be sure to check the status of all SSL/TLS servers/services you identified in Step 1. https://www.[yourschooldomain].edu is not the only secure server at your school!

IMPORTANT NOTE: When updating your servers to fix POODLE, you should also fix any other security vulnerabilities identified during the testing process. POODLE will often not be the only SSL/TLS problem present!

STEP 3. Update Your Servers' Cryptographic Libraries

Cryptographic libraries are continually being patched and enhanced. For example, **OpenSSL, the most broadly used cryptographic library, was just patched on October 14th, 2014,** to address four vulnerabilities.[10] Of those vulnerabilities, POODLE was one, but it was only a medium severity vulnerability. One of the other three was a HIGH severity vulnerability, even if you didn't hear much if anything about it in the press. It is critical that your cryptographic libraries are patched up-to-date!

At the time this advisory was written, OpenSSL users should be running 1.0.1j, although further upgrades over time are certain to take place. See https://www.openssl.org/source/ for the most recently available version.

[10] https://www.openssl.org/news/secadv_20141015.txt

If openssl is in your path, you can check the version that's currently installed with the command:

```
$ openssl version
OpenSSL 1.0.1j 15 Oct 2014
```

Important Note: After updating your cryptographic libraries, be sure to recompile and reinstall any applications that may be statically linked with the old cryptographic libraries.

Important Note: Ensure that any application you recompile with updated crypto libraries is, itself, fully up-to-date. For example, at the time this alert was written, the latest version of Apache was 2.4.10,[11] and the latest mainline version of nginx was 1.7.6.[12] After recompiling and reinstalling those applications, explicitly confirm that you're actually running the new version you've just built.[13]

Q. "I use an enterprise-grade commercially-supported Linux distribution that stresses stability, and it tends to lag when it comes to updates for OpenSSL and other applications that can leverage newer cryptographic and security features. Our local policy does not allow us to install ad-hoc packages except as distributed by the commercial Linux distribution vendor. What should I do?"

A. This is a difficult situation. If you are using a commercially-supported distribution, we encourage you to contact your vendor's support staff to see what they'd advise, and so you can share your perspective on the importance of having current cryptographic libraries and security features even in distributions that strive to avoid unnecessary changes. Most vendors, even those that stress stability over everything else, understand the importance of incorporating critical security-related updates. Also note that some vendors will at times backport specific security fixes and enhancements into older versions of their distributions. Be sure to check the vendor's security notification announcements to check if specific fixes have been backported.

Q. "We run Microsoft Windows Server -- what about our cryptographic libraries?"

A. You're using Schannel. Ensure you've applied all available service packs and patches, including the patches linked from "SHA512 is disabled in Windows when you use TLS 1.2," http://support.microsoft.com/kb/2973337 See also "Cipher Suites in Schannel," http://msdn.microsoft.com/en-us/library/windows/desktop/aa374757%28v=vs.85%29.aspx

Step 4. Harden Your Server's Configuration

You're now ready to tweak your server's configuration to ensure it's doing crypto the way it should. For example, to eliminate the POODLE vulnerability, you'll want to disable use of SSLv3. While you're making that change, however, you should also be reviewing the totality of your server's SSL/TLS configuration.

If you are not a cryptography buff, you may wonder what settings you should be configuring -- there are a somewhat daunting set of options, and there are legitimate differences of opinion even among experts about how

[11] http://httpd.apache.org/download.cgi
[12] http://nginx.org/
[13] Some operating systems may have atypical conventions for the appropriate location of applications and other configuration files. If you build cryptographic libraries or applications from scratch AND you fail to adjust the default installation locations, you run the risk of installing updated cryptographic libraries or applications ALONGSIDE existing out-of-date applications. That is, rather than ending up with one current installation, you can easily end up with TWO parallel installations, one legacy installation and one updated installation. This can be very confusing. It is thus critical that you pay attention to the correct location for all libraries, header files, and binaries during configuration, build, and installation, and that you verify the version of the products you're running after you've concluded your upgrades.

you should configure your systems, with the largest differences turning on the importance of compatibility (particularly for users with legacy systems) vs. strong security.

Examples of resources you may want to consult when it comes to configuring your server's crypto include:

Mozilla Security/Server Side TLS
https://wiki.mozilla.org/Security/Server_Side_TLS

Applied Crypto Hardening
https://bettercrypto.org/static/applied-crypto-hardening.pdf

Cryptographic Best Practices in the Post-Snowden Era
http://www.educause.edu/sites/default/files/library/presentations/SEC14/SESS32/crypto-bcp.pdf

Qualys SSL/TLS Deployment Best Practices
https://www.ssllabs.com/downloads/SSL_TLS_Deployment_Best_Practices_1.3.pdf

Summary recommendations AS OF THIS TIME would include:

-- Ensure your server's configuration DOES offers TLS 1.2. Ensure your server does NOT offer SSLv2, NOR SSLv3 (disabling SSLv3 will fix the POODLE vulnerability).[14]

-- While you're tweaking your server, check to confirm that you **ARE using a SHA2 server certificate,** and NOT using a SHA1 certificate. SHA1 certificates are relatively weak, and will begin to generate warnings in popular browsers (such as Chrome) as early as November 2014 in some cases. If you are NOT using a SHA2 cert, request a SHA2 certificate from your certificate authority.

-- When you request that SHA2 certificate, consider generating a CSR with a **4096 bit RSA key**. Although a 2048 bit RSA key is the current industry standard/most commonly employed, most sites can do a stronger 4096 bit RSA key. Note that leveraging a 4096 bit key requires additional system resources for each connection. For sites with a moderate amount of traffic this should not be an issue. If you are hosting a high traffic site, and system resources are already near capacity, you should consult with system engineers to determine if the extra resources required by a 4096 bit key will exceed capacity.

-- Prioritize cipher suites that use **AES-256 or AES-128** as a symmetric cipher. Do NOT use weak ciphers with less than 128 bit keys, including so-called "export grade" ciphers. Do NOT use RC4. Ensure that server cipher preferences are honored.

-- Ensure you have prioritized cipher suites that use ephemeral key exchange. Ephemeral key exchange delivers **"forward secrecy,"** and protects any traffic that may have been vacuumed up and archived by an adversary, in the event your private keys are ever compromised. Having strong DHPARAMs is a key part of this. However note that some web servers do NOT allow you to specify the use of strong DHPARAMs.

-- Consider configuring your site to employ **http strict transport security.**[15]

-- If you are running a crypto library that supports **elliptic curve crypto** (ECC), consider evaluating it. ECC has the potential to offer a highly desirable combination of increased strength and enhanced performance.

---

[14] If you're using Microsoft IIS, you may find this free tool helpful:
https://www.nartac.com/Products/IISCrypto/Default.aspx

[15] http://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security

Step 5. Retest Your Server(s)

After you've completed all tweaks and have restarted your newly reconfigured servers, retest them with the same tools mentioned in step 2, just to double check that everything went the way you wanted it to go. Hopefully you'll now have secure systems/services earning top marks!

Step 6. Don't Forget About Web Browsers on Laptops/Workstations/Tablets/etc.

Now that your servers are in better shape, don't forget to ALSO check the browsers on laptops, workstations, tablets, smart phones, etc. Keeping them up-to-date is essential to your security.

While the process of checking browsers to make sure they're up-to-date will vary from browser to browser, as an example, in Firefox, go to the Firefox menu item, and then go to "About Firefox" to automatically check for updates. At the time this was written, you should be running Firefox 33.0, or if you use one of the Firefox Extended Support Releases (ESR), please check that you have the latest version installed. You may wish to refer to "POODLE Disabling SSLv3 Support in Browsers", https://zmap.io/sslv3/browsers.html, for information concerning browser status.

And while you're working to ensure that your browsers are up to date, also check your systems for any updates that may be pending for *other* parts of your system.

Feedback on this document?

We welcome your feedback on this document. Please send your comments or suggestions to soc@ren-isac.net

Credits

We wish to thank the REN-ISAC Technical Advisory Group[16] for assistance in developing this Alert and give particular thanks to Joe St Sauver, Farsight Security (effective November 1st, 2014).

References

Copy of this Alert is available at: http://www.ren-isac.net/alerts/REN-ISAC_Alert_POODLE_and_Crypto_20141022.pdf

About REN-ISAC: The REN-ISAC mission is to aid and promote cybersecurity operational protection and response within the research and higher education (R&E) communities. The mission is conducted through private information sharing within a community of trusted representatives at member organizations, and as a computer security incident response team (CSIRT) supporting the R&E community at-large. REN-ISAC serves as R&E's trusted partner in commercial, governmental and private information sharing relationships, in the formal U.S. ISAC community, and for served networks. http://www.ren-isac.net

[16] http://www.ren-isac.net/about/advisory.html#technical