An Approach to Implement Cryptographic Protocol Version Downgrade Within a Secure Internal Network: TLS 1.x to SSL

https://doi.org/10.3991/ijim.v13i10.11308

Ganesh Kumar S (✉), Elango Govindaraju
SRM IST, Chennai, India
kumaruday10@yahoo.com

Abstract—The end to end encryption of connections over the internet have evolved from SSL to TLS 1.3 over the years. Attacks have exposed vulnerabilities on each upgraded version of the cryptographic protocols used to secure connections over the internet. Organisations have to keep updating their web-based applications to use the latest cryptographic protocol to ensure users are protected and feel comfortable using their web applications. But the problem is that, web applications are not always standalone systems, there is usually a maze of systems that are integrated to provide services to the end user. The interactions between these systems happens within the controlled internal private network environment of the organisation. While only the front ending web application is visible to the end user. It is not often feasible to upgrade all internal systems to use the latest cryptographic protocol for internal interfaces/integration due to prohibitive cost of redevelopment and upgrades to infra and systems. Here we define an algorithm to setup internal & external firewalls to downgrade to a lower version of the cryptographic protocol (SSL) within the internal network for the integration/interfacing connections of internal systems while mandating the latest cryptographic protocol (TLS 1.x) for end user connections to the web application.

Keywords—SSL; TLS; POODLE; Vulnerabilities; protocol versions upgrade

1 Introduction

The web servers using older versions of SSL and TLS authentication protocols within HTTPS leaves a system vulnerable to attack by hackers, SSL/TLS encrypts a channel between two endpoints (for example, between a web browser and web server) to provide privacy and reliability of data transmitted over the communications channel. Since the release of SSL v3.0, several vulnerabilities have been identified, most recently in late 2014.

The SSL protocol (all versions) cannot be fixed; there are no known methods to remediate vulnerabilities such as POODLE. SSL and early versions of TLS no longer meet the security needs of entities implementing strong cryptography to protect payment data over public or untrusted communications channels. Additionally,

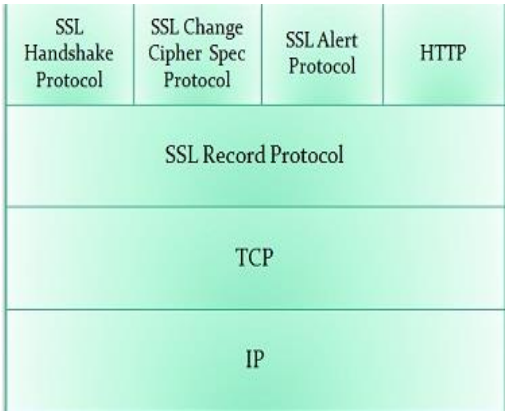
modern web browsers will begin prohibiting SSL connections in the very near future, preventing users of these browsers from accessing web servers that have not migrated to a more modern protocol.

2 The Problem

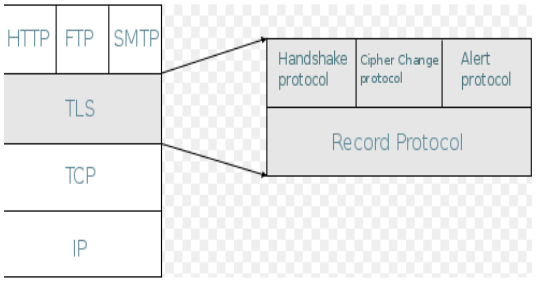
Current State

Protocol	Defined	Deprecated	Comments
SSL 1.0	Never Released	Never Released	Serious flaws in security, so not released
SSL 2.0	1995	2011	First public version, breached by attackers
SSL 3.0	1996	2015	Vulnerable to POODLE attacks
TLS 1.0	1999	2016	Can be downgraded to SSL 3.0 hence considered vulnerable
TLS 1.1	2006	In use	Backward compatible with SSL versions, so not considered safe
TLS 1.2	2008	In use	No backward compatibility
TLS 1.3	Under Draft	In use	Still not completely defined, but released for public use

The SSL Stack



The TLS Stack [14]

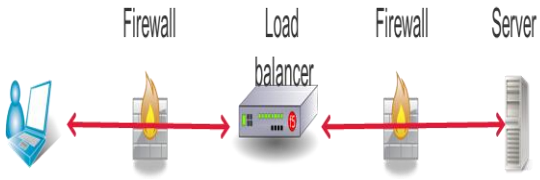


Define the Approach: Define an algorithm to setup internal & external firewalls to downgrade to a lower version of the cryptographic protocol (SSL) within the internal network for the integration/interfacing connections of internal systems while mandating the latest cryptographic protocol (TLS 1.x) for end user connections to the web application. The algorithm converts TLS 1.x to SSL 3.0 for internal connections while keeping external connections at TLS 1.2. This algorithm will define the process to setup the internal and external firewalls to make this downgrade of protocols possible.

3 Approach

The F5 load balancers which are used within the estate have an existing capability to step in and break and renegotiate handshakes in communication flows from the internet to H3. In current operation the F5 works in pass-through mode:

In SLL Offloading mode the handshake is between the browser and the F5 and then invisibly to the User the F5 then handshakes with the target server.



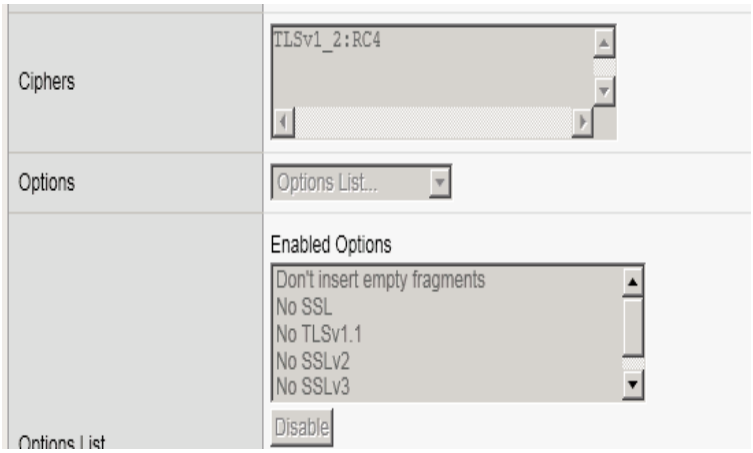
4 Detailed Requirement

The requirement is

- Enforcement of TLS 1.2
- Disabling weak ciphers and algorithms

This can be achieved by change the setup of the virtual servers that are defined on the internet facing (WWW) F5.

Each of the environments Cert's will need to use a profile that has the following options set)



All of the environments that have internet facing services will need changed. Each section below details the changes that will need to be made to each environment along with a comment on the level of testing that will be required.

5 Users Tracking

There is a need to check on Production Clients Ciper's before the change is made in Production. This is to enable the customer to determine how many of their clients may be affected by this change. App Dynamics can be used to place a small piece of code inside the Web System. That reports back from the client's browser information about connection speed etc. This is achieved by using a different URL to report back that information.

The setup for these URL's should have the F5 SSL off-loading in place. Which means that we can add an irule to the virtual IP address, which would allow us to track the client ip address and what cipher is being used.

With this information Customer would be able to tell which of their clients will be affected by this overall change.

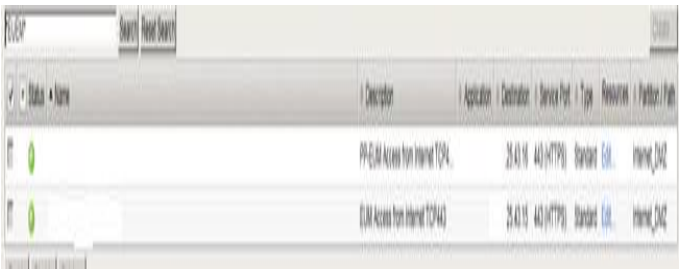
Below is the irule that would need to be added to EUEM virtual addresses.

```
when CLIENTSSL_CLIENTHELLO {

log local0. "IP [IP::client_addr]
[SSL::cipher name] [SSL::cipher version]"

}
```

This will need to be applied to the following Virtual Address on the WWW F5



6 Tests

A few tests have been performed to prove the theory and have been successful. Four test sets are used to prove the version downgrade solution

- Test 1 – This is to test the theory using only one version downgrade (TLS 1.2 to SSL 3.0)
- Test 2 – This is to test the theory using a combination of two versions downgrades (TLS 1.2 to SSL 3.0/Lower)
- Test 3 – This is to test the theory using a combination of three version downgrades (TLS 1.2 to TLS 1.1/1.0/SSL 3.0)
- Test 4 – This is to test the theory using all combinations for the version downgrade (TLS 1.2 to Any Lower TLS and SSL)

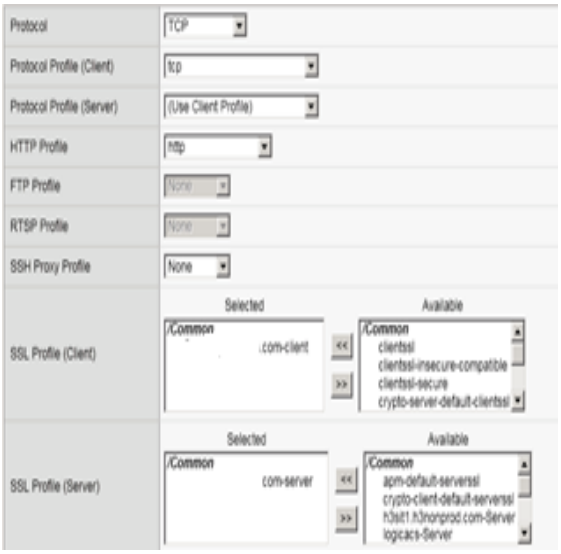
Test 1: F5 Services that will need to be modified. All the above virtual services will need the following change:

The modification should be done in the configuration of the F5 GUI using the front end. Making these config changes helps us in analyzing the first set of results. The F5 inbuilt services pick up the incoming TLS connections (of any version) and offload the secure connection at its level. Once the F5 offloads the connection, then it re-establishes the connection in a secure way using the internal SSL or any other downgraded protocols as required by the applications internal to the network. The requirements of the internal applications are configured in the F5 using the above screen.

Test 2

VI Services	IP address	Port
Incoming internet ip address2	xxx.25.43.30	443 (HTTPS)
Internal address1	xxx.25.43.34	443 (HTTPS)
Internal address2	xxx.25.43.35	443 (HTTPS)
External address1	xxx.25.43.32	443 (HTTPS)
External address2	xxx.25.43.31	443 (HTTPS)

All the above virtual services will need the following change: -

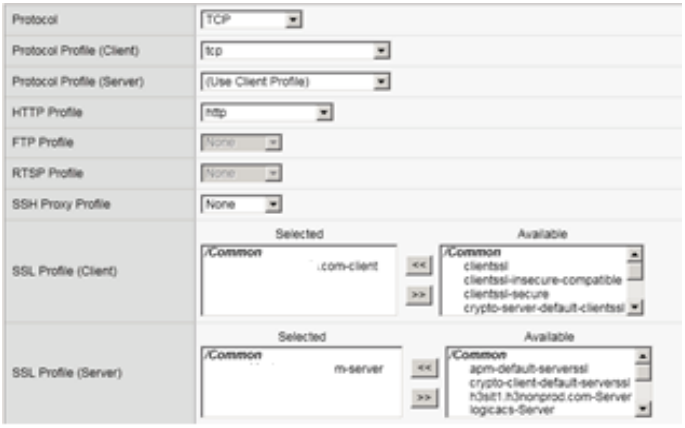


The modification should be done in the configuration of the F5 GUI using the front end. Making these config changes helps us in analyzing the second set of results. The F5 inbuilt services pick up the incoming TLS connections (of any version) and offload the secure connection at its level. Once the F5 offloads the connection, then it re-establishes the connection in a secure way using the internal SSL or any other downgraded protocols as required by the applications internal to the network. The requirements of the internal applications are configured in the F5 using the above screen.

Test 3

VI Services	IP address	Port
Incoming internet ip address2	xxx.25.43.24	443 (HTTPS)
Internal address1	xxx.25.43.20	443 (HTTPS)
Internal address2	xxx.25.43.22	443 (HTTPS)
External address1	xxx.25.43.25	443 (HTTPS)
External address2	xxx.25.43.23	443 (HTTPS)
Incoming internet ip address2	xxx.25.43.21	443 (HTTPS)

All the above virtual services will need the following change

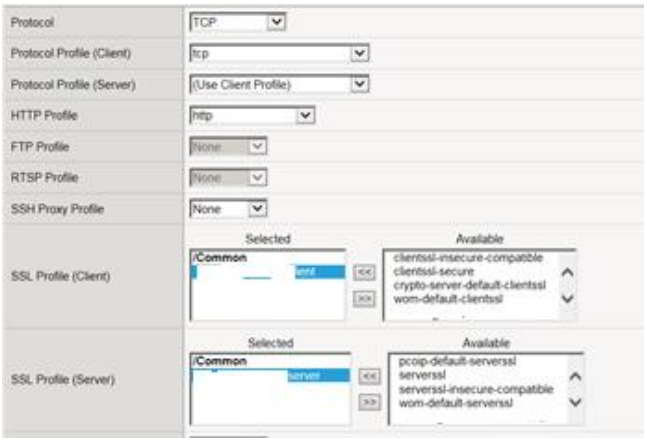


The modification should be done in the configuration of the F5 GUI using the front end. Making these config changes helps us in analyzing the third set of results. The F5 inbuilt services pick up the incoming TLS connections (of any version) and offload the secure connection at its level. Once the F5 offloads the connection, then it re-establishes the connection in a secure way using the internal SSL or any other downgraded protocols as required by the applications internal to the network. The requirements of the internal applications are configured in the F5 using the above screen.

Test 4

VI Services	IP address	Port
Incoming internet ip address2	xxx.25.43.2	443 (HTTPS)
Internal address1	xxx.25.43.4	443 (HTTPS)
Internal address2	xxx.25.43.5	443 (HTTPS)
External address1	xxx.25.43.3	443 (HTTPS)
External address2	xxx.25.43.1	443 (HTTPS)

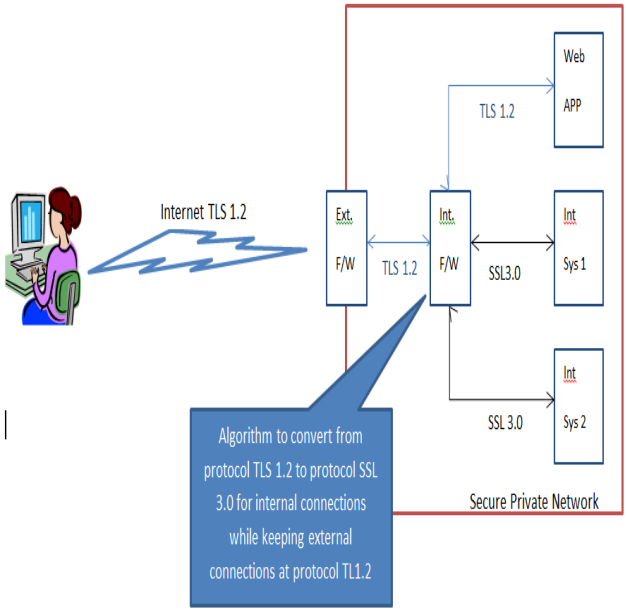
All the above virtual services will need the following change:-



The modification should be done in the configuration of the F5 GUI using the front end. Making these config changes helps us in analyzing the fourth set of results. The F5 inbuilt services pick up the incoming TLS connections (of any version) and offload the secure connection at its level. Once the F5 offloads the connection, then it re-establishes the connection in a secure way using the internal SSL or any other downgraded protocols as required by the applications internal to the network. The requirements of the internal applications are configured in the F5 using the above screen.

VI Name	IP Address	Port
Incoming internet ip address1	xxx.25.43.42	443 (HTTPS)
Incoming internet ip address2	xxx.25.43.41	443 (HTTPS)
Internal address1	xxx.25.43.43	443 (HTTPS)
Internal address2	xxx.25.43.44	443 (HTTPS)
External address1	xxx.25.43.40	443 (HTTPS)
External address2	xxx.25.43.39	443 (HTTPS)

7 Solution – Pictorial View



The solution shown in the picture above corresponds to the secure protocol version downgrade as implemented using the solution explained in the previous sections of this document.

8 References

[1] Kyatam 2017 – Heartbleed Attacks Implementation and Vulnerability. https://doi.org/10.1109/LISAT.2017.8001980
[2] Bhargavan 2016 - miTLS: Verifying Protocol Implementations against Real-World Attacks. https://doi.org/10.1109/MSP.2016.123
[3] Eldewahi 2015 - SSL/TLS Attacks: Analysis and Evaluation. https://doi.org/10.1109/ICCNEEE.2015.7381362
[4] Jawi 2015 - Non-intrusive SSL/TLS Proxy
[5] Kim 2015 – A method for Service Identification of SSL/TLS Encrypted Traffic
[6] Larisch 2017 – CRLite: Scalable system for pushing TLS revocations to all browsers. https://doi.org/10.1109/SP.2017.17
[7] Liu 2010 - SSL-DP: A Rootkit of Network Based SSL and TLS Traffic Decryptor
[8] Mortazavi 2014 - A NOVEL SECURE PROTOCOL CALLED FSSL USING FUZZY CONTROLLER FOR WEB SECURITY. https://doi.org/10.1109/SMC.2014.6974076
[9] Koschuch 2017 - Practical Evaluation of TLS Cipher Suites Compatibility
[10] Ranjbar 2016 - An SDN-Based Approach to Enhance the End-to-End Security: SSL/TLS Case Study. https://doi.org/10.1109/NOMS.2016.7502823
[11] Stricottarboton 2016 - Taxonomy of Man-in-the-Middle Attacks on HTTPS. https://doi.org/10.1109/TrustCom.2016.0106
[12] Suga 2012 - Countermeasures and tactics for transitioning against the SSL/TLS renegotiation vulnerability. https://doi.org/10.1109/IMIS.2012.138
[13] https://tools.ietf.org/html/rfc5246
[14] https://commons.wikimedia.org/wiki/File:TLS_protocol_stack.svg

9 Authors

S. Ganesh Kumar is currently working a Associate Professor in SRMIST, Chennai in Computer Science and Engineering department. His areas of research are web services, semantic web, data structures, ontology. He is a member of International Association of Computer and Information Technology (IACSIT). **Email id**: kumaruday10@yahoo.com

Elango Govindaraju is currently a student at SRM Institute of Science and Technology, Chennai, in Computer Science and Engineering department. **Email id**: g.elango@gmail.com.

Article submitted 2019-07-18. Resubmitted 2019-08-23. Final acceptance 2019-08-25. Final version published as submitted by the authors.