Security Tools Lab 1 – Project

Submitted by

tkokhing (16*92)**

An individual project submitted to the Singapore University of Technology and Design

22 Aug 2022

PDF-XChange Product
Click to buy NOW!
www.tracker-software.com

SSL GRADER

Name: ~~ tkokhing ~~ ID: 1**6*92

Table of Contents

Contents

List of Tables i
List of Figure ii
INTRODUCTION 1
BACKGROUND 1
PRINCIPAL CONSIDERATIONS 1
PART ONE – DESIGN CONSIDERATIONS AND COMPLEXITY 1
UNDERSTANDING THE OF DATA 1
DEFINING THE GRADING SYSTEM 3
COUNTRY AND INDUSTRY FOR STUDY 3
PART TWO – METHODOLOGY 4
METHODS 4
DATA PREPARATION 4
GRADING METHODOLOGY 4
RESULTS 8
VISUALIZATION 8
RESULTS AND EVALUATION 9
DISCUSSION AND RECOMMENDATIONS 11
GOOD SPREAD OF DATASET 11
STUDY OF INVESTMENT IN CYBER WELLNESS 11
GRADING WEBSITE AS A SERVICE 11
CONCLUSION 11

List of Tables

Table 1: Characteristics for Industries Selection. 3
Table 2: Websites Required Data Cleaning. 4
Table 3: Methodology for Points Deduction 4
Table 4: Grades Classification. 5
Table 4: Deduction Points for Outdated Protocols. 5
Table 5: Additional Vulnerabilities Checks for Grading Weightage. 7
Table 6: Grading by Country and Industry. 8
Table 7: Overall Grading by Industry. 8

List of Figures

Figure 1: Deeply Nested Variables in SSL Labs's APIs.....2
Figure 2: Visualization of Common Characteristics for Grade B, C and D.....8
Figure 3: Visualization of Grade A.....9

SSL GRADER

Name: ~~ tkokhing ~~ ID: 1**6*92

INTRODUCTION

BACKGROUND

1. The project is to define a SSL grading system. To accomplish this, the project explored into using APIs from SSL Labs and defined its own grading methodology.

PRINCIPAL CONSIDERATIONS

2. Choice of Industry with Polarity. This should be representative from various industry categories. It must be on **both ends of the polarity so that to have good contrast**.

3. Redefine the Existing Graders. There are existing SSL Graders, notably SSLLabs by Qualys. However, grading scale is blackbox, thus **refinement to allow better differentiate** the gradings.

PART ONE – DESIGN CONSIDERATIONS AND COMPLEXITY

UNDERSTANDING THE OF DATA

4. Capture The Output. A sub-class WriteToFile() was written to capture and record the outputs of the reference program by Narbeh[1]. This sub-class recorded the data fields and outputs in order to study the details of the (i) issued certificates, (ii) website and (iii) the types of vulnerabilities that were provided by current version (v3) of SSL Labs documentation[2]. This, in turn, allow better customization of the SSL Grader. It should be noted that some documentations are inaccurate and therefore, **only well-tested variables were used** for grading. Below showed the key observations.

a) Deeply Nested APIs. As highlighted in Yellow in the snippets, “List” variable resides inside, “suites”, which is inside “details”, which is inside “endpoints”, which is inside “host” in the mixture of Python List and Dict classes (see from bottom of **Figure 1**). This implied that **pulling the correct APIs required deeper investigation** on the exact class type.

```
main_request = json.loads(urlopen(api_url + 'analyze?host={}'.format(host)).read().decode('utf-8'))

endpoint_data = json.loads(urlopen(api_url + 'getEndpointData?host={}&s={}'.format(host,main_request['endpoints'][0]['ipAddress'])).read().decode('utf-8'))

print('\n\nmain_request +++++++++++++++ keys: \n',main_request.keys())
```

[1] https://github.com/narbehaj/ssl-checker
[2] https://github.com/ssllabs/ssllabs-scan/blob/master/ssllabs-api-docs-v3.md

```
main_request +++++++++++++++ keys:
dict_keys(['host', 'port', 'protocol', 'isPublic', 'status', 'startTime', 'testTime', 'engineVersion', 'criteriaVersion', 'endpoints'])

print('\n\nendpoint_data.details.keys() +++++++++++++++ \n',endpoint_data['details'].keys()) # # # www.yahoo.com
endpoint_data.details.keys() +++++++++++++++
dict_keys(['hostStartTime', 'certChains', 'protocols', 'serverSignature', 'prefixDelegation', 'nonPrefixDelegation', 'compressionMethods', 'supportsNpn',
'supportsAlpn', 'sniRequired', 'httpStatusCode', 'rc4WithModern', 'protocolIntolerance', 'miscIntolerance', 'sims', 'openSslCcs', 'openSSLLuckyMinus20',
'ticketbleed', 'bleichenbacher', 'poodleTls', 'freak', 'hasSct', 'logjam', 'hstsPolicy', 'hpkpPolicy', 'hpkpRoPolicy', 'staticPkpPolicy', 'httpTransactions',
'zeroRTTEnabled', 'zombiePoodle', 'goldenDoodle', 'zeroLengthPaddingOracle', 'sleepingPoodle'])

---- <Even though the APIs are pulled from same site, yahoo.com, on the same day, it returns different output as shown in above and below> ----

print('\n\nendpoint_data.details.keys() +++++++++++++++ \n',endpoint_data['details'].keys()) # # # www.yahoo.com
endpoint_data.details.keys() +++++++++++++++
dict_keys(['hostStartTime', 'certChains', 'protocols', 'suites', 'namedGroups', 'serverSignature', 'prefixDelegation', 'nonPrefixDelegation', 'vulnBeast',
'renegSupport', 'sessionResumption', 'compressionMethods', 'supportsNpn', 'npnProtocols', 'supportsAlpn', 'alpnProtocols', 'sessionTickets', 'ocspStapling',
'staplingRevocationStatus', 'sniRequired', 'httpStatusCode', 'supportsRc4', 'rc4WithModern', 'rc4Only', 'forwardSecrecy', 'supportsAead', 'protocolIntolerance',
'miscIntolerance', 'sims', 'heartbleed', 'heartbeat', 'openSslCcs', 'openSSLLuckyMinus20', 'ticketbleed', 'bleichenbacher', 'poodle', 'poodleTls',
'fallbackScsv', 'freak', 'hasSct', 'ecdhParameterReuse', 'logjam', 'chaCha20Preference', 'hstsPolicy', 'hstsPreloads', 'hpkpPolicy', 'hpkpRoPolicy',
'staticPkpPolicy', 'httpTransactions', 'drownHosts', 'drownErrors', 'drownVulnerable', 'implementsTLS13MandatoryCS', 'zeroRTTEnabled', 'zombiePoodle',
'goldenDoodle', 'supportsCBC', 'zeroLengthPaddingOracle', 'sleepingPoodle'])


print('\n\nendpoint_data.details.keys() +++++++++++++++ \n',endpoint_data['details'].keys()) # # # www.cnn.com
endpoint_data.details.keys() +++++++++++++++
 dict_keys(['hostStartTime', 'certChains', 'protocols', 'suites', 'namedGroups', 'prefixDelegation', 'nonPrefixDelegation', 'vulnBeast', 'renegSupport',
'sessionResumption', 'compressionMethods', 'supportsNpn', 'supportsAlpn', 'alpnProtocols', 'sessionTickets', 'ocspStapling', 'staplingRevocationStatus',
'sniRequired', 'httpStatusCode', 'supportsRc4', 'rc4WithModern', 'rc4Only', 'forwardSecrecy', 'supportsAead', 'protocolIntolerance', 'miscIntolerance', 'sims',
'heartbleed', 'heartbeat', 'openSslCcs', 'openSSLLuckyMinus20', 'ticketbleed', 'bleichenbacher', 'poodle', 'poodleTls', 'fallbackScsv', 'freak', 'hasSct',
'ecdhParameterReuse', 'logjam', 'chaCha20Preference', 'hstsPolicy', 'hstsPreloads', 'hpkpPolicy', 'hpkpRoPolicy', 'staticPkpPolicy', 'httpTransactions',
'drownHosts', 'drownErrors', 'drownVulnerable', 'implementsTLS13MandatoryCS', 'zeroRTTEnabled', 'zombiePoodle', 'goldenDoodle', 'supportsCBC',
'zeroLengthPaddingOracle', 'sleepingPoodle'])

print('\n\nendpoint_data.details.suites[0] +++++++++++++++ \n',endpoint_data['details']['suites'][0].keys())
endpoint_data.details.suites[0] +++++++++++++++
 dict_keys(['protocol', 'list', 'preference'])
```

Figure 1: Deeply Nested Variables in SSL Labs's APIs.

b) Differences in Websites Configuration. As shown in **Figure 1**, not all websites (e.g. cnn and yahoo) contain the same set of variables, thus making comparison between websites complicated. Therefore, the **program was heavily modified** (refer to **Annex A**:

i. to incorporate "try" and "exception" methods when calling the APIs so that to prevent any unavailable variables to halt the execution of the program, and,

ii. intentionally added key variable that is to be studied as placeholders to those sites that do not have them. This is to equalize all websites for ease for comparison using ML software.

c) Inaccurate Documentation. After many rounds of testing, some featured variables in version 3 were not present. E.g., "certs[]" (a list of Cert object) was not found in host (as shown in **Figure 1**), or nested under other variables, after testing on more than 100 websites. Some vulnerabilities such as "ticketbleed", "bleichenbacher", and "zombiePoodle" were said to return the status of "0 - unknown" but they were returned with error **signified that the variables were not present**.

DEFINING THE GRADING SYSTEM

5. "Black Box" with Minimum Learning Value. Common criteria for websites grading, namely (i) Protocol support, encryption strength of keys, and Cipher support, is considered as a "black box". Led by the Qualys, the SSLLAB's grading system did not publish how the sub-components, such as how many points to award or deduct if sites still allow vulnerability prone protocol and settings like SSL1, 2 or renegotiation, attributed to the overall grading. Therefore, this **easy way of simply using the SSL Lab grading system is not selected** because it gave minimum learning value.

6. Define Own Sets of Criteria. Although the modified program called the same APIs, the variables were individually read so that to see inside the "black box" and determine how it could be used to perform calculations forming the grading systems. The details of the testing were repetitive hence not be elaborated here.

COUNTRY AND INDUSTRY FOR STUDY

7. The selected industries were chosen as they each have their unique characteristics (**bold** in **Table 1**). Within each industry, the polarities, positive and negative, were provided to guide the selection of websites[3] so that to provide contrast for deeper analysis.

S/N	Industry	Characteristic	Polarity (+ve)	Polarity (–ve)
1	Academia – University	University is the **cream of the crop in technology**. It will be interesting to find the otherwise and weaknesses in their websites.	Top rankings by independent studies.	Low rankings from the same independent studies.
2	Healthcare	The healthcare industry is **vital to any country** with **information on individuals**, thus there should be much emphasis to maintain the industry websites.	Newsweek reports rankings on hospitals.	Hospitals in rural districts, bad records with high incidents.
3	Arts – Museums	This is **not a high revenue earning industry but age-old**, thus may have fall behind the technology trend leaving their websites more prone to vulnerabilities.	Reputation, high visitorship and size.	Old museums, rural districts and smaller in scale.
4	Banking	Banks are of **high value and transactions**, thus should system in placed to upkeep against vulnerabilities.	Forbes rankings on banks with big earnings	Forbes worst banks, banks in rural districts, small-scale, privately owned banks.

Table 1: Characteristics for Industries Selection.

8. To satisfy all the industries, careful search was done and concluded **one country representing Asia, Europe and Americas** are India, Netherlands, and USA respectively.

[3] The searches were done through google *keywords* search, e.g., "*small-scale privately owned banks* in *India*".

PART TWO – METHODOLOGY

METHODS

DATA PREPARATION

9. Data Cleaning. As defined in **PART ONE – DESIGN CONSIDERATIONS AND COMPLEXITY**, **120 websites data were collected**. This was based 10 websites each from the 4 industries in the 3 countries. Within the datasets, there were few websites that return nothing even though they could be surfed. Since they were already picked, their websites were examined and found to be using HTTP. Their data (2 examples shown in **Table 2**) were retained and cleaned up to represent their status.

S/N	Website	Failed Code and Message
1	sec.secd.ac.in	Failed: [('SSL routines', 'ssl3_read_bytes', 'tlsv1 alert internal error')]
2	www.zakirhusaindelhicollege.ac.in	Failed: [WinError 10061] No connection could be made because the target machine actively refused it

Table 2: Websites Required Data Cleaning.

10. Data Interpretation. Like all datasets, they had to be represented in numerical form so that data analytics could be applied easier. E.g, from the 120 websites, three certificate algorithms, namely ecdsa-with-SHA256, sha256WithRSAEncryption, and sha384WithRSAEncryption, were recorded. They were changed to 1, 2 and 3 respectively. Using EXCEL, the same process was repeated in other fields and -1 was used to represented "No Data". A **new column "Country" was added** to represent India, Netherlands, and the USA as 1, 2 and 3 respectively.

GRADING METHODOLOGY

11. All websites shall **initially be awarded with full 500 points**, each 100 to each of the five criteria. The grading methodology shall **weight on deduction of points** based on known vulnerabilities that were still active in the sites. The older the vulnerabilities, more points (e.g., number of years or months) shall be deducted with illustrations as shown in **Table 3**. A FACTOR of 10 was introduced to create a spread and cater for vulnerabilities that are less than a year old. The grading classification is shown in **Table 4**.

End of Life	Age (Years)	Deduction Points
Mar 2018	4 (Aug 2022 – Mar 2018)	40 (4 X FACTOR)
Nov 2018	3 (Aug 2022 – Nov 2018)	30 (3 X FACTOR)
Mar 2021	1 (Aug 2022 – Mar 2021)	10 (1 X FACTOR)
Nov 2021	>1 (Aug 2022 – Nov 2021)	5 (0.5 X FACTOR)

Table 3: Methodology for Points Deduction.

Grades Classification	Minimum to Achieve
A	400
B	300
C	200
D	100

Table 4: Grades Classification.

12. The **five criteria** that made up of the grading system as shown:

a) Certificate Expiry. Expired certificate put personal information at risk from man-in-the-middle attacks and individual being susceptible to fraud and identity theft[4]. This is manageable and requires little technicality on the website owner. Therefore, a **deduction of 10 points per day** shall be implemented.

b) Outdated Protocols. The current protocols are TLS 1.2 and 1.3 with earlier versions reported with vulnerabilities. As such, the program was added with checks to purposefully test for older protocols[5] which websites should not be using anymore. The protocols were categorized with the **deduction points depending on the age** End of Life (EOF[6]) as shown in **Table 5**.

Protocol	Age (EOF)	Deduction Points
SSL 1.0 and 2.0[7]	5 (Jun 2018)	50
SSL 3.0	4 (Nov 2018)	40
TLS 1.0 and 1.1[8]	5 (Jun 2018)	50

Table 5: Deduction Points for Outdated Protocols.

c) Site Negotiation. As reported in Nov 2009[9], websites should not allow negotiation because SSL/TLS renegotiation would subject the server unavailable with a Denial of Service (DOS) attack or could execute a Man-in-the-Middle injection attack into the HTTPS sessions when a client initiates a renegotiation process. **Deduction points would be 120** (12 years X FACTOR).

d) Using of Known Vulnerabilities. After analysing the codes, six vulnerabilities (**Table 6**) were identified to better gauge whether the websites took remedy actions after such vulnerabilities were being exposed. The age of these vulnerabilities, based on when they were discovered, provided an indication of the company interest in making their website strong.

[4] https://www.globalsign.com/en-sg/ssl-information-center/dangers-expired-ssl-certificates
[5] https://blog.gigamon.com/2019/02/11/dont-tls-1-3-naked-the-risks-and-benefits-of-traffic-decryption-decoded-for-you/
[6] https://endoflife.software/protocols/encryption
[7] SSL 1.0 and 2.0 were group together because SSL 1.0 was never publicly released because of serious security flaws in the protocol. Version 2.0, after being released in February 1995. Reference: https://en.wikipedia.org/wiki/Transport_Layer_Security#SSL_1.0,_2.0,_and_3.0
[8] TLS 1.0 and 1.1 are group together because they were deprecated on the same day, March 25, 2021, by the Internet Engineering Task Force (IETF).
[9] https://owasp.org/www-pdf-archive/OWASP_-_TLS_Renegotiation_Vulnerability.pdf

PDF-XChange Product Click to buy NOW! www.tracker-software.com

S/n	Vulnerability	Description	Age of Vulnerability[10]	Deduction Points
1	SSL- Change Cipher Spec (SSL-CCS) Injection[11]	This vulnerability requires that both a server and a user's client be vulnerable and enables an attacker to modify traffic from the server and client and subsequently decrypt the entire communication between the server and client.	8 (5 Jun 2014)	80
2	Heartbleed[12]	Heartbleed was a security bug in the OpenSSL cryptography library, which is a widely used implementation of the Transport Layer Security (TLS) protocol. It resulted from improper input validation (due to a missing bounds check) in the implementation of the TLS heartbeat extension. The vulnerability was classified as a buffer over-read, a situation where more data can be read than should be allowed.	8 (1 Apr 2014)	80
3	Padding Oracle On Downgraded Legacy Encryption (Poodle)[13]	This is a security vulnerability which takes advantage of the fallback to SSL 3.0. If attackers successfully exploit this vulnerability, on average, they only need to make 256 SSL 3.0 requests to reveal one byte of encrypted messages. To mitigate this attack, one approach is to completely disable SSL 3.0 on the client side and the server side. However, some old clients and servers do not support TLS 1.0 and above. Thus, it was encouraged that the browser and server to implement TLS_FALLBACK_SCSV, which will make downgrade attacks impossible.	7 (14 Oct 2014)	70
4	Factoring RSA Export Keys (FREAK)[14]	FREAK ("Factoring RSA Export Keys") is a security exploit of a cryptographic weakness in the SSL/TLS protocols introduced decades earlier for compliance with U.S. cryptography export regulations. These involved limiting exportable software to use only public key pairs with RSA moduli of 512 bits or less (so-called RSA_EXPORT keys).	7 (6 Jan 2015)	70
5	logjam	This is a security vulnerability in systems that use Diffie–Hellman key exchange using 512- to 1024-bit keys. In essence, the threat downgrades the Transport Layer Security (TLS) connection and exploits a weakness caused by using the same prime numbers in the encryption to execute a Man-in-the-Middle attack.[15]	7 (20 May 2015)	70
6	Decrypting RSA with Obsolete and Weakened eNcryption (Drown)	DROWN is a serious vulnerability that affects HTTPS and other services that rely on SSL and TLS, some of the essential cryptographic protocols for Internet security. It allows an attacker to decrypt modern TLS connections between up-to-date clients and servers by sending	6 (1 Mar 2016)	60

[10] Dates are in accordance with CVE website.
[11] https://cloudsecurityalliance.org/blog/2014/06/16/openssl-ccs-injection-vulnerability-countdown/
[12] https://en.wikipedia.org/wiki/Heartbleed
[13] https://en.wikipedia.org/wiki/POODLE
[14] https://en.wikipedia.org/wiki/FREAK
[15] https://crashtest-security.com/logjam-tls/

		probes to a server that supports SSLv2 and uses the same private key[16].		

Table 6: Additional Vulnerabilities Checks for Grading Weightage.

e) OCSP Stapling. The Online Certificate Status Protocol[17] (OCSP) checks the validity status of a certificate in real-time unlike Certificate Revocation Lists (CRL) that is cached until it is expires. In addition, OCSP Stapling includes its response within the initial SSL handshake, and shield the client from being seen by the CA for the websites it is visiting. This privacy feature **if not included would have 10 points deducted**.

[16] https://drownattack.com/
[17] https://knowledge.digicert.com/quovadis/ssl-certificates/ssl-general-topics/what-is-ocsp-stapling.html

RESULTS

13. The summarized results of the five criteria as shown in **Table 7** and **Table 8**. Refer to **Annex B** for full details.

Country / Industry	India				Netherlands				USA			
	A	B	C	D	A	B	C	D	A	B	C	D
Academia – University	7	1	0	2	10	0	0	0	7	3	0	0
Healthcare	7	3	0	0	9	1	0	0	9	0	1	0
Arts – Museums	7	2	0	1	8	2	0	0	9	1	0	0
Banking	10	0	0	0	10	0	0	0	9	1	0	0
Country Total	31	6	0	3	37	3	0	0	34	5	1	0

Table 7: Grading by Country and Industry.

Industry / Grades	A	B	C	D
Academia – University	24	4	0	2
Healthcare	25	4	1	0
Arts – Museums	24	5	0	1
Banking	29	1	0	0
Industry Total	102	14	1	3

Table 8: Overall Grading by Industry.

VISUALIZATION

14. Parallel Coordinates. With many websites graded as A, a quick analysis (refer to **Annex C**) using parallel coordinates revealed that the 18 (14 + 1 + 3) websites that did not get a A grade and exhibited a few common characteristics, such as (i) with 14 out of 18 websites had TLS 1.0 & 1.1 enabled when they should have disabled, and (ii) 13 out of 18 websites operating with Key Strength of 3072 when more secure optional Key Strength 4096, 7680 and 15360 were available especially for those in Banking and Healthcare industries.

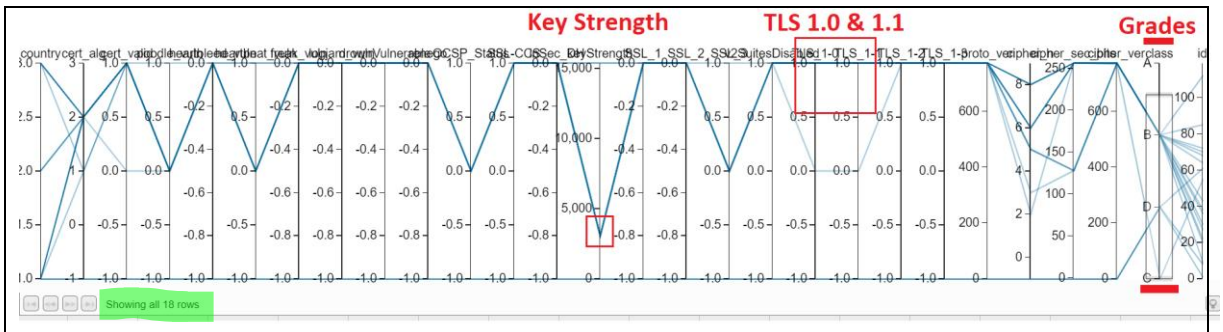

Figure 2: Visualization of Common Characteristics for Grade B, C and D.

15. Grade A Characteristics. Grade A has its healthy characteristics. They are, such as (i) with 92 out of 102 websites disabled TLS 1.0 & 1.1, and (ii) massive adoption of higher Key Strength 4096, 7680 and 15360.

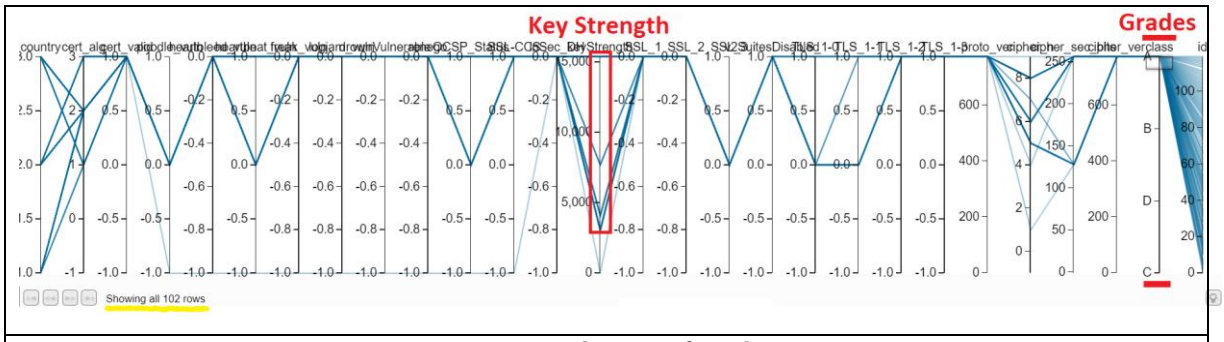

Figure 3: Visualization of Grade A.

RESULTS AND EVALUATION

16. Comparison by Country.

a) India. India had the lowest number of Grade A (31) for Banking, Healthcare and Arts industries. However, they were equally strong in the Banking comparing against Netherlands and USA. **India was the only country found to have websites found still utilizing HTTP** (2 in Academia and 1 in Arts). India also had the greatest number of grade B indicating that they are catching up with the websites cybersecurity health status.

b) Netherlands. **Netherlands scored the best** by having the most number of grade A's and least grade B's. Their grade B in health care was due to website still enabled with TLS 1.0 and 1.1, while 2 websites in Arts industry did not utilize OCSP stapling.

c) USA. For the 3 graded B websites, it was revealed that one of the universities used **self-signed certificate which was expired** on 31 Jul 22. It was also the only website in the dataset that chose not to go through a CA. The other 2 were found with OCSP stapling not enabled and still allowing vulnerable protocols TLS 1.0 and 1.1. For the graded C website (www.thechristhospital.com), it was found to be **susceptible to Poodle attack, and still have not disabled SSL v3, TLS 1.0 and TLS 1.1**. During the ping to the website, the certificate was signed with SSL v3 protocol. This was also the only website in the dataset that failed the 6 chosen vulnerabilities in para 12.d) above.

17. Comparison by Industry.

a) Banking. **Banking was graded as the strongest** among the four industries. This was a healthy sign because even with websites in negative polarity, i.e., Forbes worst banks, banks in rural districts, small-scale, and privately owned banks, did not return a bad state. This indicated that banks across the **countries have put in place**

substantial system and effort to ensure that their websites were well guarded and up to date.

b) Healthcare. Healthcare is a critical industry as it holds many individuals record. As shown in **Table 7**, it was **possible to legislate and tighten the grip** on the industry as exhibited in Netherlands whom out-performed the other two countries.

c) Academia. This was a surprising finding that weaknesses such as (i) expired self-signed certificate in USA and (ii) 2 India universities websites were still running on HTTP. The Academia was chosen because it should be at the frontal of technology and perform better than others. Comparing between India and USA, they were relatively on the same plight with only 7 graded A sites.

d) Arts. In this industry, USA performed better than the other two countries.

18. With the above findings, it could be deduced that **different countries seem to invest in the industry that is important to them**, i.e., Banking which is high revenue, regardless of the financial muscle of the institutions. When it comes to other industries such as less money churning, the emphasis seems to be uneven as seen contrastingly in India, Netherlands, and the USA. Notwithstanding this, the **emphasis is also largely depending on the leadership** of the government or institution. As seen in Netherlands, it has a good standing for all the four industries unlike India and the USA.

DISCUSSION AND RECOMMENDATIONS

GOOD SPREAD OF DATASET

19. The design consideration on selecting websites based on polarities (+ve and –ve) guided a good spread of data. The dataset did not only capture big earnings banks or reputable universities, it also collected from low revenue earning, i.e., Arts, industry that might have fall behind the technology trend leaving their websites more prone to vulnerabilities.

20. Should the study grow in depth and width, it is **proposed that more industries, business sector or countries could be enlisted**. With that, deeper thinking shall require to guide the collection of websites to ensure polarities at both ends.

STUDY OF INVESTMENT IN CYBER WELLNESS

21. The current study lacks in examining the investment, if any, the entities put in. For example, can we expect the Academia industry to invest the equivalent amount as the Banking? Must museum set up its own cybersecurity department like the Banking?

22. Therefore, if provided with a longer duration, it is **recommended that the study to include examining the investment into cybersecurity** for the selected entities. A correlation study can be accomplished to measure the effectiveness of the investment.

GRADING WEBSITE AS A SERVICE

23. Grading Website as a Service (GWaaS). As explained in para (4.c) c)above regarding inaccurate data provided by SSL Labs, the same service when privatized could become more competitive and produce better results. As such, **turning the SSL grader into GWaaS** could help entities as a form of early-warning signal to guard the websites.

CONCLUSION

24. Connectivity is the future and web / app surfing is the link bridge. Cyber risk will continue to evolve such that protocols that are working well today may not be so for tomorrow. This project has shown that there are websites that were left behind, knowingly, or unknowingly like a US university having an expired self-signed certificate. The project also revealed that the leadership of the government or institution is important to steer the implementation of cyber defence and hygiene.

SSL GRADER

Python Code

```
# NAME: tkokhing
# STUDENT ID: 1**6*92
# # This program took refernce from https://github.com/narbehaj/ssl-checker
# # heavily modification as stated in the report.

#!/usr/bin/env python3
import socket
import sys
import json
import time as tm

from argparse import ArgumentParser, SUPPRESS
from datetime import datetime
from ssl import PROTOCOL_TLSv1
from time import sleep
from csv import DictWriter


try:
    from OpenSSL import SSL, crypto
    from json2html import *
except ImportError:
    print('Please install required modules: pip install -r requirements.txt')
    sys.exit(1)


class Clr:
    """Text colors."""

    RST = '\033[39m'
    RED = '\033[31m'
    GREEN = '\033[32m'
    YELLOW = '\033[33m'


class SSLChecker:

    # for summing the overall run
    total_failed = 0
    total_valid = 0
    total_warning = 0
    total_expired = 0


    def __init__ (self):
```

SSL GRADER

```
        self.myFileName = 'Write_SSL_output.txt'
        self.saveFile = open(self.myFileName, 'w')
        self.saveFile.write('-----------Recording for SSL Grader-----------\n')

    def WriteToFile(self,myMessage,fileMode):
        i = 0
        self.saveFile = open(self.myFileName, fileMode)
        self.saveFile.write(myMessage + '\n')

    def CloseFile(self):
        self.saveFile.close()

    def testBit(self, int_type, offset):
        mask = 1 << offset
        return(int_type & mask)

    def get_cert(self, host, port, user_args):

        cert_context = {}
        """Connection to the host."""
        if user_args.socks:
            import socks
            if user_args.verbose:
                print('{}Socks proxy enabled{}\n'.format(Clr.YELLOW, Clr.RST))

            socks_host, socks_port = self.filter_hostname(user_args.socks)
            socks.setdefaultproxy(socks.PROXY_TYPE_SOCKS5, socks_host, int(socks_port), True)
            socket.socket = socks.socksocket

        if user_args.verbose:
            print('{}Connecting to socket{}\n'.format(Clr.YELLOW, Clr.RST))
            self.WriteToFile('Connecting to socket','a')


        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        # sock = socket.socket()

        osobj = SSL.Context(PROTOCOL_TLSv1)
        sock.connect((host, int(port)))
        oscon = SSL.Connection(osobj, sock)
        oscon.set_tlsext_host_name(host.encode())
        oscon.set_connect_state()
        oscon.do_handshake()

        # print('Connection state AFTER handshake', oscon.get_state_string().decode())

        # Retrieve the SSL or TLS protocol version of the current connection.
```

SSL GRADER

```
        cert_context['proto_ver_n'] = oscon.get_protocol_version_name()
        print('Protocol Version Name --- >', oscon.get_protocol_version_name())

        cert_context['proto_ver'] = oscon.get_protocol_version()
        print('Protocol Version  --- >', oscon.get_protocol_version())

        # Obtain the name of the currently used cipher.
        cert_context['cipher_n'] = oscon.get_cipher_name()
        print('Name of the currently used cipher  --- >', oscon.get_cipher_name())


        # cert_context['cipher_list'] = oscon.get_cipher_list() # # # this is wrong
        # Retrieve the list of ciphers used by the Connection object, Returns A list of native cipher
strings.
        # print('List of ciphers used by the Connection object --- >', oscon.get_cipher_list())

        # Obtain the number of secret bits of the currently used cipher.
        # Returns The number of secret bits of the currently used cipher or
        # None if no connection has been established.
        # Return type intor NoneType
        cert_context['cipher_sec_bits'] = oscon.get_cipher_bits()
        print('Number of secret bits of the currently used cipher  --- >', oscon.get_cipher_bits())

        # Obtain the protocol version of the currently used cipher,
        # Returns The protocol name of the currently used cipher or
        # None if no connection has been established.
        cert_context['cipher_ver'] = oscon.get_cipher_version()
        print('Protocol version of the currently used cipher  --- >', oscon.get_cipher_version())

        oscrypto = crypto.PKey()
        # print('\t\toscrypto ---> ', oscrypto)
        # print('\t\toscrypto.bits ---> ', oscrypto.bits)
        # print('\t\toscrypto.bits() ---> ', oscrypto.bits())

        cert_bits = oscrypto.bits()
        cert_context['cert_bits'] = cert_bits
        print('Cert Public Key bit ---> ', cert_bits)


        # # # no clue how to do this
        # print('OCSP Request --->', oscon.request_ocsp())

        cert = oscon.get_peer_certificate()

        cert_Pkey = cert.get_pubkey()
        print('Cert Public Key ---> ', cert_Pkey)
        cert_context['cert_Pkey'] = cert_Pkey
```

```
        sock.close()

        # print('Connection state AFTER sock closed', oscon.get_state_string().decode())

        if user_args.verbose:
            print('{}Closing socket{}\n'.format(Clr.YELLOW, Clr.RST))

        return cert, cert_context

    def border_msg(self, message):
        """Print the message in the box."""
        row = len(message)
        h = ''.join(['+'] + ['-' * row] + ['+'])
        result = h + '\n' "|" + message + "|"'\n' + h
        print(result)

    def analyze_ssl(self, host, context, user_args):
        """Analyze the security of the SSL certificate."""
        try:
            from urllib.request import urlopen
        except ImportError:
            from urllib.request  import urlopen

        api_url = 'https://api.ssllabs.com/api/v3/'
        while True:
            if user_args.verbose:
                print('{}Requesting analyze to {}{}\n'.format(Clr.YELLOW, api_url, Clr.RST))

            main_request = json.loads(urlopen(api_url +
'analyze?host={}'.format(host)).read().decode('utf-8'))
            if main_request['status'] in ('DNS', 'IN_PROGRESS'):
                if user_args.verbose:
                    print('{}Analyze waiting for reports to be finished (1
secs){}\n'.format(Clr.YELLOW, Clr.RST))

                sleep(5)
                continue
            elif main_request['status'] == 'READY':
                if user_args.verbose:
                    print('{}Analyze is ready{}\n'.format(Clr.YELLOW, Clr.RST))

                break

        endpoint_data = json.loads(urlopen(api_url + 'getEndpointData?host={}&s={}'.format(
            host, main_request['endpoints'][0]['ipAddress'])).read().decode('utf-8'))

        if user_args.verbose:
```

SSL GRADER

```
            print('{}Analyze report message: {}{}\n'.format(Clr.YELLOW,
endpoint_data['statusMessage'], Clr.RST))

        # if the certificate is invalid
        if endpoint_data['statusMessage'] == 'Certificate not valid for domain name':
            return context

        try:
            context[host]['grade'] = main_request['endpoints'][0]['grade']
        except:
            context[host]['grade'] = "NO DATA"

        try:
            context[host]['poodle_vuln'] = endpoint_data['details']['poodle']
        except:
            context[host]['poodle_vuln'] = "NO DATA"

        try:
            context[host]['heartbleed_vuln'] = endpoint_data['details']['heartbleed']
        except:
            context[host]['heartbleed_vuln'] = "NO DATA"

        try:
            context[host]['heartbeat_vuln'] = endpoint_data['details']['heartbeat']
        except:
            context[host]['heartbeat_vuln'] = "NO DATA"

        try:
            context[host]['freak_vuln'] = endpoint_data['details']['freak']
        except:
            context[host]['freak_vuln'] = "NO DATA"

        try:
            context[host]['logjam_vuln'] = endpoint_data['details']['logjam']
        except:
            context[host]['logjam_vuln'] = "NO DATA"

        try:
            context[host]['drownVulnerable'] = endpoint_data['details']['drownVulnerable']
        except:
            context[host]['drownVulnerable'] = "NO DATA"

        try:
            context[host]['renego'] = "N"
            renegvar = int(endpoint_data['details']['renegSupport'])
            renegp = self.testBit(renegvar, 0)
            if renegp != 0:
                context[host]['renego'] = "Y"
```

SSL GRADER

```
except:
    context[host]['renego'] = "NO DATA"

#sgrade
try:
    context[host]['sgrade'] = endpoint_data['gradeTrustIgnored']
except:
    context[host]['sgrade'] = "NO DATA"

try:
    context[host]['OCSP_Status'] = endpoint_data['details']['ocspStapling']
except:
    context[host]['OCSP_Status'] = "NO DATA"

# OpenSSL CCS Injection Vulnerability (CVE-2014-0224) Alert
try:
    # Refer to SSL lab doc
    # 1
    # -1 - test failed
    # 0 - unknown
    # 1 - not vulnerable
    # 2 - possibly vulnerable, but not exploitable
    # 3 - vulnerable and exploitable
    context[host]['SSL-CCS'] = endpoint_data['details']['openSslCcs']

except:
    context[host]['SSL-CCS'] = "NO DATA"
    pass # test failed -1 will take care of this


#insecure dh
try:
    insecureProtocolList = []
    protocolList = []
    kxStrengthList = []
    # not all sites have this but created as placeholder for easy comparison
    context[host]['cipher_list'] = []
    context[host]['cipher_Cnt'] = 0
    context[host]['inSec_DH_list'] = []
    context[host]['inSec_DH']= 0
    context[host]['keyStrength'] = 0

    # suites may have more than a Dict
    for s in range(len(endpoint_data['details']['suites'])):
        # list may have more than a List
        for suite in endpoint_data['details']['suites'][s]['list']:
            # print('each suite name is  ---> ', suite['name'])
```

```
            # capture the no duplicate list of secure protocol
            if suite['name'] not in protocolList:
                protocolList.append(suite['name'])

            # not all fields inside suite will be key() kxStgrength
            try:
                kxStrengthList.append(suite['kxStrength'])
                # print('kxStrength is  ---> ', suite['kxStrength'])
            except Exception as e:
                pass
            # print('kxStrength is  ---> ', type(suite['kxStrength']))
            # print('Looping -------->',s)
            # input()

            try:
                if "DHE" in suite['name']:
                    try:
                        if suite['q']==0: # 0 if the protocol is insecure, null otherwise
                            if suite['name'] not in insecureProtocolList:
                                insecureProtocolList.append(suite['name'])
                            # print("This is possible insec DH " + suite['name'])
                        # elif suite['q']==1: # 1 if the protocol is secure
                        #     print("This is Secure DH " + suite['name'])
                        # else:
                        #     print('q is not 0 or 1 haha')
                    except Exception as e:
                        # enter here when there is no field q
                        pass

            except Exception as e:
                pass

    # return protocol list and count
    if len(protocolList) != 0:
        context[host]['cipher_list'].extend(protocolList)
        context[host]['cipher_Cnt'] = len(protocolList)

    # return insecure protocol list and count
    if len(insecureProtocolList) != 0:
        context[host]['inSec_DH_list'].extend(insecureProtocolList)
        context[host]['inSec_DH'] = len(insecureProtocolList)

    # return key Strength of protocols, it should be the same for all protocol
    # but i cannot be 100% sure
    if len(kxStrengthList) != 0:
        context[host]['keyStrength'] = max(kxStrengthList)
```

SSL GRADER

```
except Exception as e:
    pass

# SSL_versions
try:
    # not all sites have this but created as placeholder for easy comparison
    context[host]['SSL_1']=[]
    context[host]['SSL_2']=[]
    context[host]['SSL_3']=[]
    context[host]['v2SuitesDisabled']=[]
    ssl1, ssl2, ssl3 = "N", "N", "N"
    v2SuitesDisabled= "Y"
    for protocol in endpoint_data['details']['protocols']:
        if "SSL" in protocol['name'] and protocol['version']=="1.0":
            ssl1 = "Y"

        if "SSL" in protocol['name'] and protocol['version']=="2.0":
            ssl2 = "Y"
            if protocol['v2SuitesDisabled']== True:
                v2SuitesDisabled = "Y"
            else:
                v2SuitesDisabled = "N"

        if "SSL" in protocol['name'] and protocol['version']=="3.0":
            ssl3 = "Y"

    context[host]['SSL_1']=ssl1
    context[host]['SSL_2']=ssl2
    context[host]['SSL_3']=ssl3
    context[host]['v2SuitesDisabled']=v2SuitesDisabled


except Exception as e:
    pass

# TLS_versions
try:
    # not all sites have this but created as placeholder for easy comparison
    context[host]['TLS_1-0']=[]
    tls10= "N"
    context[host]['TLS_1-1']=[]
    tls11= "N"
    context[host]['TLS_1-2']=[]
    tls12= "Y"
    context[host]['TLS_1-3']=[]
    tls13= "Y"

    for protocol in endpoint_data['details']['protocols']:
```

```
                if "TLS" in protocol['name'] and protocol['version']=="1.0":
                    tls10 = "Y"
                if "TLS" in protocol['name'] and protocol['version']=="1.1":
                    tls11 = "Y"
                if "TLS" in protocol['name'] and protocol['version']=="1.2":
                    tls12 = "Y"
                if "TLS" in protocol['name'] and protocol['version']=="1.3":
                    tls13 = "Y"

            context[host]['TLS_1-0']=tls10
            context[host]['TLS_1-1']=tls11
            context[host]['TLS_1-2']=tls12
            context[host]['TLS_1-3']=tls13

        except Exception as e:
            pass

        return context

    def get_cert_sans(self, x509cert):
        """
        Get Subject Alt Names from Certificate. Shameless taken from stack overflow:
        https://stackoverflow.com/users/4547691/anatolii-chmykhalo
        """

        # print('\t\t\t\tCert Cryptography', x509cert.to_cryptography())
        # # # Cert Cryptography <Certificate(subject=<Name(C=SG,L=Singapore,O=Oversea-Chinese Banking
Corporation Limited,CN=www.ocbc.com)>, ...)>

        san = ''
        ext_count = x509cert.get_extension_count()
        for i in range(0, ext_count):
            ext = x509cert.get_extension(i)
            if 'subjectAltName' in str(ext.get_short_name()):
                san = ext.__str__()
        # replace commas to not break csv output
        san = san.replace(',', ';')
        return san

    def get_cert_info(self, host, cert):
        """Get all the information about cert and create a JSON file."""
        context = {}
        # print('cert --->',cert)
        # # # # cert ---> <OpenSSL.crypto.X509 object at 0x000001A4C5F0E6D0>
        # # # # cert is not a Dict or Tuple or List of List
        # print('Class cert --->',type(cert))
        # # # # Class cert ---> <class 'OpenSSL.crypto.X509'>
```

```
        cert_subject = cert.get_subject()
        # print('cert_subject --->', cert_subject)
        # # # # cert_subject ---> <X509Name object '/C=US/ST=California/L=Los Gatos/O=Netflix,
Inc./CN=www.netflix.com'>

        # # # keeps getting
        cert_Pkey = cert.get_pubkey()
        print('Cert Public Key ---> ', cert_Pkey)

        context['host'] = host
        context['issued_to'] = cert_subject.CN
        context['issued_o'] = cert_subject.O

        context['issuer_c'] = cert.get_issuer().countryName
        context['issuer_o'] = cert.get_issuer().organizationName

        context['issuer_ou'] = cert.get_issuer().organizationalUnitName
        context['issuer_cn'] = cert.get_issuer().commonName

        context['cert_sn'] = str(cert.get_serial_number())
        context['cert_sha1'] = cert.digest('sha1').decode()
        context['cert_sha256'] = cert.digest('sha256').decode()
        context['cert_alg'] = cert.get_signature_algorithm().decode()
        context['cert_ver'] = cert.get_version()
        context['cert_sans'] = self.get_cert_sans(cert)
        context['cert_exp'] = cert.has_expired()
        # context['cert_valid'] = False if cert.has_expired() else True
        context['cert_valid'] = 0 if cert.has_expired() else 1

        # # #  Valid from
        valid_from = datetime.strptime(cert.get_notBefore().decode('ascii'),
                                       '%Y%m%d%H%M%SZ')
        context['valid_from'] = valid_from.strftime('%Y-%m-%d')

        # # #  Valid till
        valid_till = datetime.strptime(cert.get_notAfter().decode('ascii'),
                                       '%Y%m%d%H%M%SZ')
        context['valid_till'] = valid_till.strftime('%Y-%m-%d')

        # # #  Validity days (total number of days the cert will be valid)
        context['validity_days'] = (valid_till - valid_from).days

        # # #  Valid days left
        context['valid_days_to_expire'] = (datetime.strptime(context['valid_till'],
                                           '%Y-%m-%d') - datetime.now()).days

        if cert.has_expired():
            self.total_expired += 1
```

```
        else:
            self.total_valid += 1

        # If the certificate has less than 30 days validity
        if context['valid_days_to_expire'] <= 30:
            self.total_warning += 1

        # print('get_cert_info() == Class type of <context> ',type(context))
        # # # # get_cert_info() == Class type of <context>  <class 'dict'>
        # # # # <context> is declared as a  <class 'dict'> to fill up the site details gathereds

        # print('get_cert_info() == These are inside <context>  --->', context)
        # # # # get_cert_info() == These are inside <context>  --->
        # # # # {'host': 'www.dbs.com', 'issued_to': 'www.dbs.com',
        # # # # 'issued_o': 'DBS Bank Ltd', 'issuer_c': 'US',
        # # # # 'issuer_o': 'Entrust, Inc.', 'issuer_ou': 'See www.entrust.net/legal-terms',
        # # # #  'issuer_cn': 'Entrust Certification Authority - L1M',
        # # # #  'cert_sn': '138536995130289955891549547836603450318',
        # # # #  'cert_sha1': '62:69:8B:70:63:4A:81:BB:C4:0E:13:99:D7:6F:E0:18:53:7F:7F:F5',
        # # # #  'cert_alg': 'sha256WithRSAEncryption', 'cert_ver': 2,
        # # # #  'cert_sans': 'DNS:www.dbs.com; DNS:cug.www.dbs.com',
        # # # #  'cert_exp': False, 'cert_valid': True, 'valid_from': '2021-09-08',
        # # # #  'valid_till': '2022-10-07', 'validity_days': 394,
        # # # #  'days_left': 47, 'valid_days_to_expire': 47}
        # # # .
        # # .
        # #

        return context

    def show_result(self, user_args):
        """Get the context."""

        print('\n')
        host_counter = 1
        context = {}
        all_context ={}
        start_time = datetime.now()
        hosts = user_args.hosts

        if not user_args.json_true and not user_args.summary_true:
            self.border_msg(' Analyzing {} host(s) '.format(len(hosts)))
            self.WriteToFile(' Analyzing ' +  str(len(hosts)) + ' host(s)','a')


        if not user_args.json_true and user_args.analyze:
            print('{}Warning: -a/--analyze is enabled. It takes more time...{}\n'.format(Clr.YELLOW,
Clr.RST))
```

```
            self.WriteToFile(' Warning: -a/--analyze is enabled. It takes more time...' ,'a')

        for host in hosts:
            if user_args.verbose:
                print('\n{} ({}) Working on host: {}{}\n'.format(Clr.YELLOW, host_counter, host,
Clr.RST))
                self.WriteToFile('\n(' + str(host_counter) +')' + ' Working on host: ' +  host,'a')

            host_counter += 1

            host, port = self.filter_hostname(host)

            # Check duplication
            if host in context.keys():
                continue

            try:
                cert, cert_context = self.get_cert(host, port, user_args)
                context[host] = self.get_cert_info(host, cert)
                context[host]['tcp_port'] = int(port)


                # Analyze the certificate if enabled
                if user_args.analyze:
                    context = self.analyze_ssl(host, context, user_args)

                if not user_args.json_true and not user_args.summary_true:
                    self.print_status(host, context, user_args.analyze)

                all_context[host] = {**context[host], **cert_context}

            except SSL.SysCallError:
                if not user_args.json_true:
                    print('\t{}[-]{} {:<20s} Failed: Misconfigured SSL/TLS\n'.format(Clr.RED, Clr.RST,
host))
                    self.WriteToFile(' [-] ' +  (host) + ' Failed: Misconfigured SSL/TLS','a')
                    self.total_failed += 1
            except Exception as error:
                if not user_args.json_true:
                    print('\t{}[-]{} {:<20s} Failed: {}\n'.format(Clr.RED, Clr.RST, host, error))
                    self.WriteToFile('\t[-] ' +  host + ' Failed: ' + str(error),'a')
                    self.total_failed += 1

            except KeyboardInterrupt:
                print('{}Canceling script...{}\n'.format(Clr.YELLOW, Clr.RST))
                self.WriteToFile('Canceling script... ','a')
                sys.exit(1)
```

```
        if not user_args.json_true:
            self.border_msg(' Successful: {} | Failed: {} | Valid: {} | Warning: {} | Expired: {} |
Duration: {} '.format(
                len(hosts) - self.total_failed, self.total_failed, self.total_valid,
                self.total_warning, self.total_expired, datetime.now() - start_time))

            self.WriteToFile('\nSuccessful: ' + str(len(hosts) - self.total_failed) + ' | Failed: ' +
str(self.total_failed) + ' | Valid: ' + str(self.total_valid) + ' | Warning ' +
str(self.total_warning) + ' | Expired: ' + str(self.total_expired) + ' | Duration: ' +
str(datetime.now() - start_time),'a')

            if user_args.summary_true:
                # Exit the script just
                return

        # CSV export if -c/--csv is specified
        if user_args.csv_enabled:
            self.export_csv(all_context, user_args.csv_enabled, user_args)

        # HTML export if -x/--html is specified
        if user_args.html_true:
            self.export_html(all_context)

        # While using the script as a module
        if __name__ != '__main__':
            return json.dumps(all_context)

        # Enable JSON output if -j/--json argument specified
        if user_args.json_true:
            print(json.dumps(all_context))

        if user_args.json_save_true:
            for host in all_context.keys():
                with open(host + '.json', 'w', encoding='UTF-8') as fp:
                    fp.write(json.dumps(all_context[host]))

        self.CloseFile()

    def print_status(self, host, context, analyze=False):
        """Print all the usefull info about host."""
        print('\t{}[+]{} {}\n\t{}'.format(Clr.GREEN, Clr.RST, host, '-' * (len(host) + 5)))
        print('\t\tIssued domain: {}'.format(context[host]['issued_to']))
        print('\t\tIssued to: {}'.format(context[host]['issued_o']))
        print('\t\tIssued by: {} ({})'.format(context[host]['issuer_o'], context[host]['issuer_c']))
        print('\t\tValid from: {}'.format(context[host]['valid_from']))
        print('\t\tValid to: {} ({} days left)'.format(context[host]['valid_till'],
context[host]['valid_days_to_expire']))
        print('\t\tValidity days: {}'.format(context[host]['validity_days']))
```

SSL GRADER

```
        print('\t\tCertificate valid: {}'.format(context[host]['cert_valid']))
        print('\t\tCertificate S/N: {}'.format(context[host]['cert_sn']))
        print('\t\tCertificate SHA1 FP: {}'.format(context[host]['cert_sha1']))
        print('\t\tCertificate SHA256 FP: {}'.format(context[host]['cert_sha256']))
        print('\t\tCertificate version: {}'.format(context[host]['cert_ver']))
        print('\t\tCertificate algorithm: {}'.format(context[host]['cert_alg']))


        self.WriteToFile('\t[+] ' +  host + '\n\t' + '-' * (len(host) + 5),'a')
        self.WriteToFile('\t\tIssued domain              : ' + str(context[host]['issued_to']),'a')
        self.WriteToFile('\t\tIssued to                  : ' + str(context[host]['issued_o']),'a')
        self.WriteToFile('\t\tIssued by                  : ' + str(context[host]['issuer_o']) + '(' +
str(context[host]['issuer_c']) + ')','a')
        self.WriteToFile('\t\tValid from                 : ' + str(context[host]['valid_from']),'a')
        self.WriteToFile('\t\tValid to                   : ' + str(context[host]['valid_till']) + ' ('
+ str(context[host]['valid_days_to_expire']) + ' days left)','a')
        self.WriteToFile('\t\tValidity days              : ' +
str(context[host]['validity_days']),'a')
        self.WriteToFile('\t\tCertificate valid          : ' + str(context[host]['cert_valid']),'a')
        self.WriteToFile('\t\tCertificate S/N            : ' + str(context[host]['cert_sn']),'a')
        self.WriteToFile('\t\tCertificate SHA1 FP        : ' + str(context[host]['cert_sha1']),'a')
        self.WriteToFile('\t\tCertificate SHA256 FP      : ' + str(context[host]['cert_sha256']),'a')
        self.WriteToFile('\t\tCertificate version        : ' + str(context[host]['cert_ver']),'a')
        self.WriteToFile('\t\tCertificate algorithm      : ' + str(context[host]['cert_alg']),'a')

        if analyze:
            print('\t\tKey Strength: {}'.format(context[host]['keyStrength'] ))
            print('\t\tCertificate grade: {}'.format(context[host]['grade']))
            print('\t\tCiphers Supported by Server: {}'.format(context[host]['cipher_list']))
            print('\t\tPoodle vulnerability: {}'.format(context[host]['poodle_vuln']))
            print('\t\tHeartbleed vulnerability: {}'.format(context[host]['heartbleed_vuln']))
            print('\t\tHeartbeat vulnerability: {}'.format(context[host]['heartbeat_vuln']))
            print('\t\tFREAK vulnerability: {}'.format(context[host]['freak_vuln']))
            print('\t\tLogjam vulnerability: {}'.format(context[host]['logjam_vuln']))
            print('\t\tDROWN vulnerability: {}'.format(context[host]['drownVulnerable']))

            self.WriteToFile('\t\tKey Strength               : ' +
str(context[host]['keyStrength']),'a')
            self.WriteToFile('\t\tCertificate grade          : ' + str(context[host]['grade']),'a')
            self.WriteToFile('\t\tCiphers Supported by Server: ' + (',
'.join(context[host]['cipher_list'])),'a')
            self.WriteToFile('\t\tPoodle vulnerability       : ' +
str(context[host]['poodle_vuln']),'a')
            self.WriteToFile('\t\tHeartbleed vulnerability   : ' +
str(context[host]['heartbleed_vuln']),'a')
            self.WriteToFile('\t\tHeartbeat vulnerability    : ' +
str(context[host]['heartbeat_vuln']),'a')
```

```
            self.WriteToFile('\t\tFREAK vulnerability        : ' +
str(context[host]['freak_vuln']),'a')
            self.WriteToFile('\t\tLogjam vulnerability       : ' +
str(context[host]['logjam_vuln']),'a')
            self.WriteToFile('\t\tDROWN vulnerability        : ' +
str(context[host]['drownVulnerable']),'a')

        print('\t\tExpired: {}'.format(context[host]['cert_exp']))
        print('\t\tCertificate SAN\'s: ')

        self.WriteToFile('\t\tExpired:  ' + str(context[host]['cert_exp']),'a')
        self.WriteToFile('\t\tCertificate SAN\'s:   ','a')

        for san in context[host]['cert_sans'].split(';'):
            print('\t\t \\_ {}'.format(san.strip()))
            self.WriteToFile('\t\t \\_   ' + str(san.strip()),'a')


        print('\n')


    def export_csv(self, context, filename, user_args):
        """Export all context results to CSV file."""
        # prepend dict keys to write column headers
        if user_args.verbose:
            print('{}Generating CSV export{}\n'.format(Clr.YELLOW, Clr.RST))

        with open(filename, 'w') as csv_file:
            csv_writer = DictWriter(csv_file, list(context.items())[0][1].keys())
            csv_writer.writeheader()
            for host in context.keys():
                csv_writer.writerow(context[host])

    def export_html(self, context):
        """Export JSON to HTML."""
        html = json2html.convert(json=context)
        file_name = datetime.strftime(datetime.now(), '%Y_%m_%d_%H_%M_%S')
        with open('{}.html'.format(file_name), 'w') as html_file:
            html_file.write(html)

        return

    def filter_hostname(self, host):
        """Remove unused characters and split by address and port."""
        host = host.replace('http://', '').replace('https://', '').replace('/', '')
        port = 443
        if ':' in host:
            host, port = host.split(':')
```

```
        print('HOST ----- >',host,'PORT ----- >', port)
        return host, port

    def get_args(self, json_args={}):
        """Set argparse options."""
        parser = ArgumentParser(prog='ssl_checker.py', add_help=False,
                                description="""Collects useful information about given host's SSL
certificates.""")

        if len(json_args) > 0:
            args = parser.parse_args()
            setattr(args, 'json_true', True)
            setattr(args, 'verbose', False)
            setattr(args, 'csv_enabled', False)
            setattr(args, 'html_true', False)
            setattr(args, 'json_save_true', False)
            setattr(args, 'socks', False)
            setattr(args, 'analyze', False)
            setattr(args, 'hosts', json_args['hosts'])
            return args

        group = parser.add_mutually_exclusive_group(required=True)
        group.add_argument('-H', '--host', dest='hosts', nargs='*',
                           required=False, help='Hosts as input separated by space')
        group.add_argument('-f', '--host-file', dest='host_file',
                           required=False, help='Hosts as input from file')
        parser.add_argument('-s', '--socks', dest='socks',
                            default=False, metavar='HOST:PORT',
                            help='Enable SOCKS proxy for connection')
        parser.add_argument('-c', '--csv', dest='csv_enabled',
                            default=False, metavar='FILENAME.CSV',
                            help='Enable CSV file export')
        parser.add_argument('-j', '--json', dest='json_true',
                            action='store_true', default=False,
                            help='Enable JSON in the output')
        parser.add_argument('-S', '--summary', dest='summary_true',
                            action='store_true', default=False,
                            help='Enable summary output only')
        parser.add_argument('-x', '--html', dest='html_true',
                            action='store_true', default=False,
                            help='Enable HTML file export')
        parser.add_argument('-J', '--json-save', dest='json_save_true',
                            action='store_true', default=False,
                            help='Enable JSON export individually per host')
        parser.add_argument('-a', '--analyze', dest='analyze',
                            default=False, action='store_true',
                            help='Enable SSL security analysis on the host')
```

SSL GRADER

```
        parser.add_argument('-v', '--verbose', dest='verbose',
                            default=False, action='store_true',
                            help='Enable verbose to see what is going on')
        parser.add_argument('-h', '--help', default=SUPPRESS,
                            action='help',
                            help='Show this help message and exit')

        args = parser.parse_args()

        # Get hosts from file if provided
        if args.host_file:
            with open(args.host_file) as f:
                args.hosts = f.read().splitlines()

        # Checks hosts list
        if isinstance(args.hosts, list):
            if len(args.hosts) == 0:
                parser.print_help()
                sys.exit(0)
        return args


if __name__ == '__main__':

    SSLCheckerObject = SSLChecker()
    SSLCheckerObject.show_result(SSLCheckerObject.get_args(json_args={}))
```

SSL GRADER

DATA FIELDS REPLACED FOR BETTER INTERPRETATION

country

India 1

Netherlands 2

USA 3

cert_alg (certificate algorithm)

ecdsa-with-SHA256 1

sha256WithRSAEncryption 2

sha384WithRSAEncryption 3

cipher_n

AES128-GCM-SHA256 1

AES256-SHA 2

DHE-RSA-AES128-GCM-SHA256 3

ECDHE-ECDSA-CHACHA20-POLY1305 4

ECDHE-RSA-AES128-GCM-SHA256 5

ECDHE-RSA-AES256-GCM-SHA384 6

TLS_AES_128_GCM_SHA256 7

TLS_AES_256_GCM_SHA384 8

TLS_CHACHA20_POLY1305_SHA256 9

	A	B	C	D
Academia	24	4	0	2
Healthcare	25	4	1	0
Arts	24	5	0	1
Banking	29	1	0	0
	102	14	1	3

	India				Netherlands				USA			
	A	B	C	D	A	B	C	D	A	B	C	D
Academia	7	1	0	2	10	0	0	0	7	3	0	0
Healthcare	7	3	0	0	9	1	0	0	9	0	1	0
Arts	7	2	0	1	8	2	0	0	9	1	0	0
Banking	10	0	0	0	10	0	0	0	9	1	0	0
	31	6	0	3	37	3	0	0	34	5	1	0

Grades	Min Points
A	400
B	300
C	200
D	100

host	coun	cert_alg	cert_valid	poodle_vul	heartbleed	heartbeat_	freak_vuln	logjam_vul	drownVuln	renego	OCSP_Stat	SSL-CCS	inSec_DH	keyStrengt	SSL_1	SSL_2	SSL_3	v2SuitesDi	TLS_1-0	TLS_1-1	TLS_1-2	TLS_1-3	proto_ver	cipher_n	cipher_sec	cipher_ver	GRADES	500 GRADING	TTL DEDUCTION
www.kanyashreeuniversity.ac.in	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
sec.secd.ac.in	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	D	500	0
www.hitm.ac.in	1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
giet.edu.in	1	2	1	0	0	1	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.rjit.ac.in	1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	1	1	1	1	771	5	128	771	A	400	100
www.bitwardha.ac.in	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.igtamsu.ac.in	1	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	A	400	100
www.kbvsasun.ac.in	1	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.zakirhusaindelhicollege.ac.in	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	D	500	0
www.karunya.edu	1	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B	390	110
www.tudelft.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.uu.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.utwente.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.maastrichtuniversity.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.tilburguniversity.edu	2	3	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.hanze.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.tue.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.rsm.nl	2	3	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.nyenrode.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.avans.nl	2	3	1	0	0	0	0	0	0	0	1	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.aufonline.org	3	2	0	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	B	300	200
www.ni-u.edu	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.stmary.edu	3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B	390	110
www.ozarks.edu	3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.athens.edu	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.buc.edu	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.santarosa.edu	3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.gustavus.edu	3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	6	256	771	B	390	110
www.eku.edu	3	2	1	0	0	1	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.stanford.edu	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	500	0
gorakhpur-medicity-hospital.business.site	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B	390	110
www.sumhospital.com	1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	7	128	772	A	500	0
www.amrihospitals.in	1	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.nanavatimaxhospital.org	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	1	1	1	772	8	256	772	A	440	60
delhi.apollohospitals.com	1	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
rubyhall.com	1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.manipalhospitals.com	1	2	1	0	0	1	0	0	0	0	0	1	0	15360	0	0	0	1	1	1	1	1	771	6	256	771	B	390	110
www.wockhardthospitals.com	1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.fmri.in	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771	B	390	110
pgimer.edu.in	1	2	1	0	0	1	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.ziekenhuisrivierenland.nl	2	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.rijnstate.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.spijkenissemc.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.mijnantonius.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	1	1	1	1	771	3	128	771	B	390	110
www.st-anna.nl	2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	500	0
www.asz.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.isala.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.vumc.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.umcg.nl	2	3	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.umcutrecht.nl	2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.johnmuirhealth.com	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	7	128	772	A	490	10
www.elliothospital.org	3	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
nsuh.northwell.edu	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.iuhealth.org	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	1	1	1	771	5	128	771	A	440	60
www.valleyhealth.com	3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	A	400	100
www.thechristhospital.com	3	2	1	1	0	0	0	0	0	0	0	1	0	3072	0	0	1	1	1	1	1	1	771	2	256	768	C	280	220
www.pennstatehealth.org	3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.henryford.com	3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.inova.org	3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.hopkinsmedicine.org	3	2	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	1	0	0	1	1	771	6	256	771	A	500	0
indianmuseumkolkata.org	1	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	A	400	100
www.nationalmuseumindia.gov.in	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	D	500	0
www.dakshinachitra.net	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.chumashmuseum.org	1	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.goamuseum.gov.in	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B	390	110
www.worldtelugu.co	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.salarjungmuseum.in	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
kolkatatourism.travel	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
gandhimmm.org	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B	390	110
www.karnatakatourism.org	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.redlightsecrets.com	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	6	256	771	B	390	110
www.franshalsmuseum.nl	2	2	1	0	0	0	0	0	0	0	1	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.teylersmuseum.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	6	256	771	B	390	110
www.rmo.nl	2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771	A	400	100

[illegible]olenn[illegible]eumdevalk.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.[illegible]ershuis.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.[illegible]uller.nl	2	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	771	6	256	771	A	500	0
www.dordrechtsmuseum.nl	2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
rijksmuseumboerhaave.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.haarlemmermeermuseum.nl	2	2	1	0	0	1	0	0	0	0	0	1	0	4096	0	0	0	1	0	1	1	1	771	5	128	771	A	440	60
www.guggenheim.org	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
vmfa.museum	3	2	1	0	0	1	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.charlestonmuseum.org	3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.thewadsworth.org	3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
americanhistory.si.edu	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.philamuseum.org	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771	B	390	110
www.dmns.org	3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.mfah.org	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.fieldmuseum.org	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.spam.com	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.jkgb.in	1	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	A	400	100
www.iobnet.co.in	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.canarabank.com	1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.pnbindia.in	1	2	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.sbi.co.in	1	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	1	128	771	A	490	10
www.yesbank.in	1	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	500	0
punjabandsindbank.co.in	1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.idbibank.in	1	2	1	0	0	0	0	0	0	0	1	1	0	7680	0	0	0	1	0	0	1	1	771	6	256	771	A	500	0
www.icicibank.com	1	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	500	0
www.finobank.com	1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.caceis.com	2	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.triodos.com	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.nibc.com	2	2	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.devolksbank.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.rabobank.com	2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.banktencate.nl	2	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
www.binck.com	2	2	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
www.anadolubank.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A	490	10
nwbbank.com	2	2	1	0	0	0	0	0	0	0	1	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.achmeabank.nl	2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	9	256	772	A	490	10
www.capitalone.com	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.salemfivedirect.com	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.nbkc.com	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771	B	390	110
www.axosbank.com	3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	4	256	771	A	500	0
www.quontic.com	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
ffin.com	3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	490	10
www.easternbank.com	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
public.websteronline.com	3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A	490	10
investor.firstmidwest.com	3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0
www.huntington.com	3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A	500	0

countr	cert_alg	cert_valid	poodle_vul	heartbleed	heartbeat_	freak_vuln	logjam_vul	drownVuln	renego	OCSP_Stat	SSL-CCS	inSec_DH	keyStrengt	SSL_1	SSL_2	SSL_3	v2SuitesDis	TLS_1-0	TLS_1-1	TLS_1-2	TLS_1-3	proto_ver	cipher_n	cipher_sec	cipher_ver	class
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	D
1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	1	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	771	5	128	771	A
1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	1	1	1	1	771	5	128	771	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
1	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	D
1	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B
2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A
2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
2	3	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A
2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
2	3	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A
2	3	1	0	0	0	0	0	0	0	1	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	0	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	B
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B
3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	6	256	771	B
3	2	1	0	0	1	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	771	6	256	771	A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B
1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	7	128	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	1	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	1	0	0	0	0	0	1	0	15360	0	0	0	1	1	1	1	1	771	6	256	771	B
1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771	B
1	2	1	0	0	1	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	771	6	256	771	A
2	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
2	2	1	0	0	0	0	0	0	0	0	1	0	15360	0	0	0	1	1	1	1	1	771	3	128	771	B
2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771	A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
2	3	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	5	128	771	A
2	3	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
2	3	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	7	128	772	A
3	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	1	1	1	771	5	128	771	A
3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	A
3	2	1	1	0	0	0	0	0	0	0	1	0	3072	0	0	1	1	1	1	1	1	771	2	256	768	C
3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771	A
3	2	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0	0	1	0	0	1	1	771	6	256	771	A
1	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	A
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	D
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
1	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772	B
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772	A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	6	256	771	B
2	2	1	0	0	0	0	0	0	0	1	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772	A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	6	256	771	B
2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771	A

2	2	1	0	0	0	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772 A
2	2	1	0	0	0	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772 A
2	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	771	6	256	771 A
2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
2	2	1	0	0	0	0	0	0	0	0	1	0	4096	0	0	0	1	0	0	1	1	771	5	128	771 A
2	2	1	0	0	1	0	0	0	0	0	1	0	4096	0	0	0	1	0	1	1	1	771	5	128	771 A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	1	0	0	0	0	0	1	0	15360	0	0	0	1	0	0	1	1	771	6	256	771 A
3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	1	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771 A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771 B
3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
1	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	772	8	256	772 A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771 A
1	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771 A
1	2	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	6	256	771 A
1	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	1	128	771 A
1	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771 A
1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772 A
1	2	1	0	0	0	0	0	0	0	1	1	0	7680	0	0	0	1	0	0	1	1	771	6	256	771 A
1	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771 A
1	2	1	0	0	0	0	0	0	0	1	1	0	15360	0	0	0	1	0	0	1	1	772	8	256	772 A
2	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771 A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771 A
2	2	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	6	256	771 A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
2	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
2	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771 A
2	2	1	0	0	0	0	0	0	0	0	1	0	7680	0	0	0	1	0	0	1	1	771	5	128	771 A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	6	256	771 A
2	2	1	0	0	0	0	0	0	0	1	1	0	4096	0	0	0	1	0	0	1	1	772	8	256	772 A
2	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	9	256	772 A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	1	1	1	1	771	5	128	771 B
3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	771	4	256	771 A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	1	0	0	0	0	0	1	0	3072	0	0	0	1	0	0	1	1	771	5	128	771 A
3	1	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A
3	2	1	0	0	0	0	0	0	0	1	1	0	3072	0	0	0	1	0	0	1	1	772	8	256	772 A