School of Computer Science and Information Technology

Security in Computing and IT

Assignment

Semester 1, 2017

Aaron HORLER

s3481341

DECLARATION AND STATEMENT OF AUTHORSHIP

1. I hold a copy of this work which can be produced if the original is lost/damaged.

2. This work is my original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgement is made.

3. No part of this work has been written for me by any other person except where such collaboration has been authorised by the lecturer/teacher concerned.

4. I have not previously submitted this work for this or any other course/unit.

5. I give permission for this work to be reproduced, communicated, compared and archived for the purpose of detecting plagiarism.

6. I give permission for a copy of my/our marked work to be retained by the school for review and comparison, including review by external examiners.

I understand that:

7. Plagiarism is the presentation of the work, idea or creation of another person as though it is my/our own. It is a form of cheating and is a very serious academic offence that may lead to exclusion from the University. Plagiarised material can be drawn from, and presented in, written, graphic and visual form, including electronic data and oral presentations. Plagiarism occurs when the origin of the material used is not appropriately cited.

8. Plagiarism includes the act of assisting or allowing another person to plagiarise or to copy my/our work.

1 Vulnerabilities and Malware

1.1 National Vulnerability Database

A vulnerability in the Google Chrome web browser was reported to the National Vulnerability Database (NVD) on the 17th of February 2017 affecting the browsers displaying of internationalized domain names (IDNs) in its address bar, or Omnibar.

The vulnerability allowed an attacker to spoof an English domain name using Punycode - a means to represent Unicode characters using the limited subset of ASCII characters used for domains on the Internet.

This attack is known as an IDN homograph attack.

This specific vulnerability was assigned CVE-2017-5015, and is documented by the NVD as affected versions of Chrome prior to 56 on Linux, Windows, Mac, and Android. However, on the 14th of April 2017, Xudong Zheng reported a similar vulnerability affecting all versions of Google Chrome released to that date.

Xudong registered the domain xn--80ak6aa92e.com, gave it the standard www subdomain, and requested an HTTPS certificate from certificate authority Let's Encrypt.

Let's Encrypt is a well known provider of free HTTPS certificates, who provides certificates on an automated basis to individuals who can prove domain ownership (by serving content or creating a DNS entry). Let's Encrypt do not manually validate domains, or check them for malicious intent.

In versions of Google Chrome below 58 (58.0.3029.81, specifically) the URL https://www.xn--80ak6aa92e.com rendered as https://www.apple.com in the address bar. Google assigned this specific vulnerability the identifier CVE-2017-5060. Google paid Xudong $2,000 USD for reporting this vulnerability, and Haosheng Wang $2,000 USD for CVE-2017-5015.

i. CVE-2017-5015 received a CVSS v3 Base Score of 6.5 (Medium) from the NVD, and CVE-2017-5060 also received a score of Medium from Google.

ii. CVE-2017-5015 received an Impact Score of 3.6 from the NVD.

iii. Two valid purposes of using the CVSS scoring system are;

1. The system is a standardised means to categorise vulnerabilities and prioritise their patching.
2. The system is an evidence-based empirical means to categorise vulnerabilities independant to media and individual speculation and sensationalism.

iv. CVE-2017-5015 was patched in Google Chrome release 56.0.2924.76, and CVE-2017-5060 was patched in Google Chrome release 58.0.3029.81.

v. This set of vulnerabilities are very specific and were difficult to mitigate even for the most novice of users.

The Australian Signals Directorate (ASD) publishes strategies to mitigate vulnerabilities. From their strategies, the following would have acted to mitigate an IDN homograph attack;

- **Patch applications**
  This would have been the most helpful in mitigating any attacks that resulted from these vulnerabilities. In both cases Google updated its browser in a timely manner, and the updates resolved the issue.

- **Antivirus software with up-to-date signatures**
  This would have been less helpful, but could have potentially mitigated an attack. Antivirus software often prohibits access to malicious domains and IP addresses, and an antivirus software could have potentially discovered a domain being used to conduct an IDN homograph attack before a user visited the site and blocked access to it.

  As with most blacklisting strategies, this solution is limited because it requires the antivirus software become aware of the attack before a user is affected.

- **User education**
  User education would have been highly limited in this scenario. This is because, in the example of CVE-2017-5060, the Punycode URL rendered as https://www.apple.com exactly.

More advanced users may have noticed that the certificate was issued to www.xn--80ak6aa92e.com, however.

References

https://nvd.nist.gov/vuln/detail/CVE-2017-5015
https://www.xudongz.com/blog/2017/idn-phishing/
https://letsencrypt.org/getting-started/
https://www.asd.gov.au/infosec/top-mitigations/mitigations-2017-table.htm
https://chromereleases.googleblog.com/2017/04/stable-channel-update-for-desktop.html
https://chromereleases.googleblog.com/2017/01/stable-channel-update-for-desktop.html

1.2 Antivirus company evaluation

On May 12th 2017, a ransomware threat that was referred to as *WannaCry*, was observed in Europe.

It affected several large-scale organisations and institutions on the continent, including the British National Health Service (NHS).

This section will analyse the detection of *WannaCry* by several antivirus companies, in attempt to display the ability of these companies to quickly respond to an emerging threat.

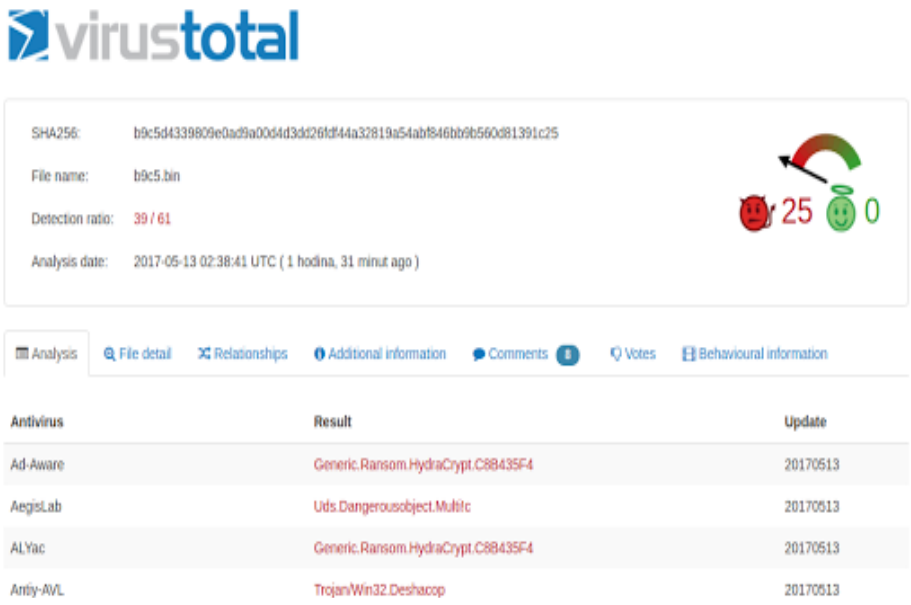
This analysis was performed three times from 10:37pm on the 12th of May 2017 Greenwich Mean Time (GMT). All detection samples are logged in GMT.

Company	Name for *WannaCry*	Detection 12-05-2017 10:37pm	Detection 12-05-2017 11:13pm	Detection 13-05-2017 4:02am
Ad-Aware	Generic.Ransom.HydraCrypt.C8B435F4			
Avast	Win32:WanaCry-A [Trj]			
AVG	Generic_r.SSZ			
Avira	TR/FileCoder.724645			
AVware				
BitDefender	Trojan.Ransom.WannaCryptor.D			
ClamAV				
Comodo	UnclassifiedMalware			
Endgame				
ESET-NOD32	Win32/Filecoder.WannaCryptor.D			
F-Secure	Generic.Ransom.HydraCrypt.C8B435F4			
Kaspersky	Trojan-Ransom.Win32.Wanna.c			
Malwarebytes	Ransom.WanaCrypt0r			
McAfee	Artemis!7BF2B57F2A20			
Microsoft	Ransom:Win32/WannaCrypt			
NANO-Antivirus				
Panda	Trj/RansomCrypt.K			
Sophos	Troj/Wanna-D			
SUPER AntiSpyware				

Symantec	ML.Attribute.HighConfidence			
TrendMicro	RANSOM_WCRY.I			
VIPRE				
Webroot	W32.Ransom.Wannacry			
ZoneAlarm	Trojan-Ransom.Win32.Wanna.c			
Zoner				

Using Google Trends, the absolute first international searches for *WannaCry* via Google started on 12-05-2017 10:00am GMT.

i. To conduct this analysis, I used VirusTotal to scan a binary file belonging to *WannaCry*.



The above screenshot displays the filename and SHA256 hash of the binary file, along with a subset of the results. VirusTotal is can be used as a *cross-referencing* site.

First-party sites that publish malware information include McAfee Labs, MalwareBytes Labs, Kaspersky, and Symantec. These sites, however, may not indiscriminately publish all detected threats.

ii. Antivirus websites vary in descriptiveness of their malware reports. Some sites, like MalwareBytes labs, publish articles explaining major incidents in simple terminology. Other websites, like McAfee labs, do not publish all threats and delay the publishing of threats they do publish.

iii. My analysis of *WannaCry* detection shows that many major antivirus companies had detected the threat several hours after its genesis, although some less-major companies had not.

By 12-05-2017 10:37pm GMT, 15 out of the 25 antivirus companies I analysed had detected *WannaCry*. This set included several well-known companies, like Kaspersky and Symantec, and discluded Comodo and Sophos.

By 13-05-2017 4:02am GMT, almost five hours later, that number had increased to 18 out of 25. Antivirus companies that had not detected *WannaCry* by this time include SUPER Anti-Spyware and Zoner - both arguably lesser-known.

iv. For this section, I used recently updated lists of threats from Symantec, ESET, Microsoft (for Windows Defender), and Avira. I used VirusTotal as a cross-referencing site.

I used *WannaCry* because all four companies had documented it, and had used similar names.

	Symantec	**Microsoft**	**ESET**	**Avira**
Detection	12-05-2017	16-05-2017	16-05-2017	16-05-2017
Order	1	2	2	2

The table above finds that Symantec claims to have detected the virus on the 12th of May 2017, and all others on the 16th of May 2017.

My earlier analysis of the time of *WannaCry* detection proves this information to be false, and seriously questionable. According to VirusTotal, all four companies had detected *WannaCry* by the 13th of May 2017 (GMT).

My statement, based on these findings, is not to trust first-party information from antivirus companies. Documentation is largely outdated and inconsistent. Instead, rely on independent sources like VirusTotal to analyse actual malicious files against the detection of antivirus software.

1.3 Recent vulnerability analysis

Threat 1: *WannaCry/WanaCrypt0r 2.0*

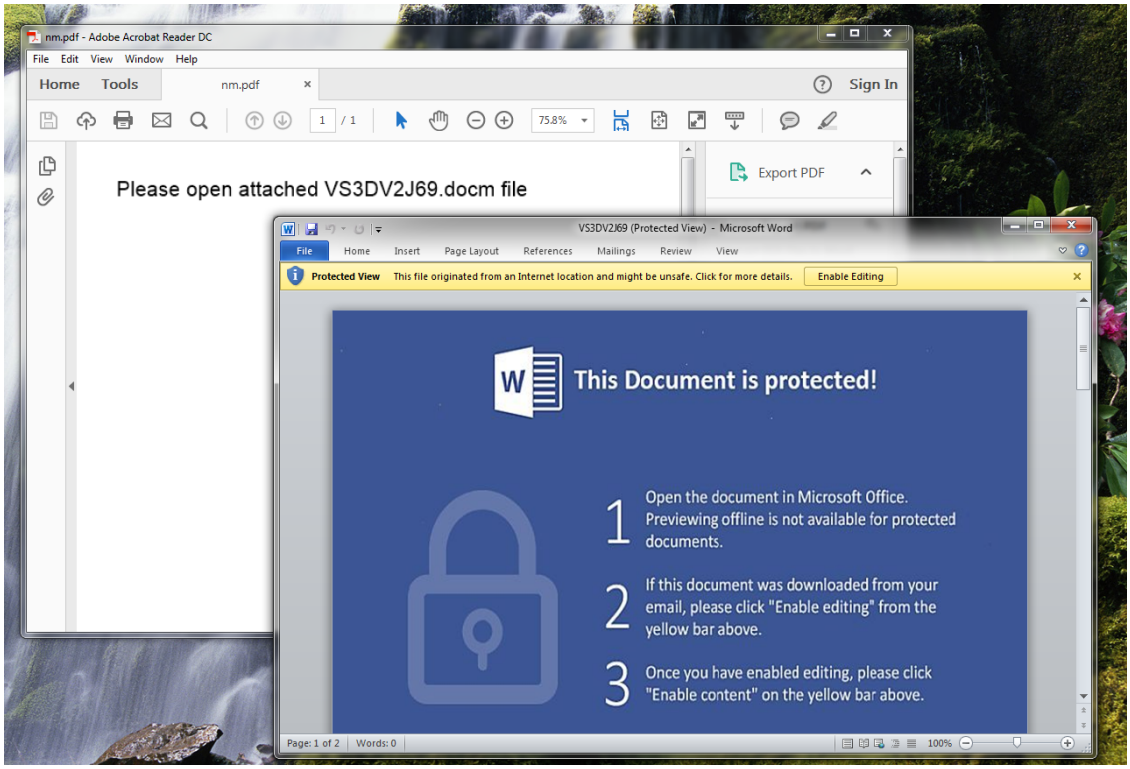
i. The attack strategy of *WannaCry* is via the MS17-010 vulnerability in Microsoft Windows SMB servers. This vulnerability was patched on March 14 2017. The vulnerability was published after a National Security Agency (NSA) leak.

ii. *WannaCry* is ransomware. The target or aim of the malware is therefore monetary gain. *WannaCry* demands $300 USD in Bitcoin to decrypt a user's files.

iii. Being ransomware, *WannaCry* does not aim to permanently hide itself. Instead, it seeks only to hide itself during the initial encryption phase. After that, it makes itself known with a very clear pop-up dialog.

Threat 2: *Jaff*

i. The attack strategy of *Jaff* is via a PDF that opens a DOCM file. The DOCM file contains a Macro with the payload. *Jaff* attempts to bypass Microsoft Office security mechanisms by instructing the user to enable both editing and macros with a message claiming that the document is ‘protected’.



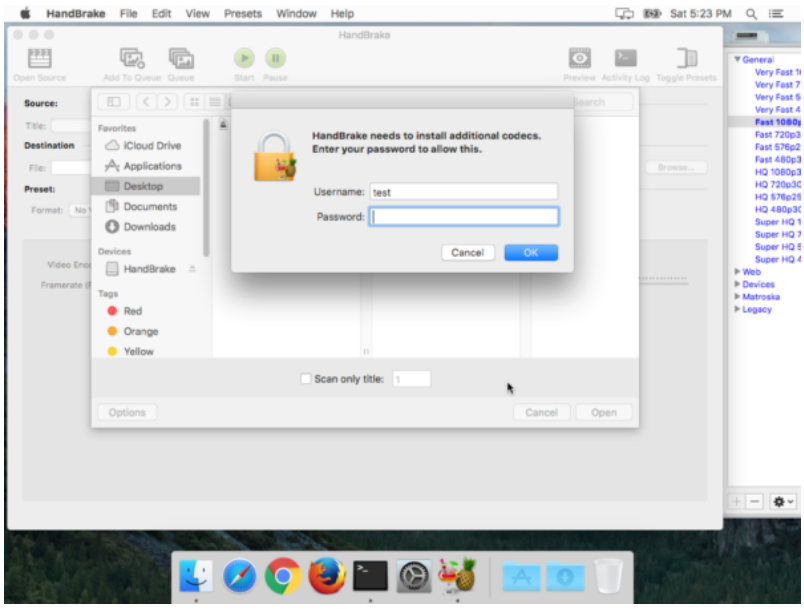
ii. *Jaff*, like *WannaCry*, is ransomware. The target or aim of the malware is therefore monetary gain. *Jaff* demands 2 Bitcoin to decrypt a user’s files. This, at the time of writing this report, is approximately $4645 AUD.

iii. Being ransomware, *Jaff* does not aim to permanently hide itself. Instead, it seeks only to hide itself during the initial encryption phase. After that, it makes itself known with a pop-up dialog.

Threat 3: *Proton in HandBrake*

i. The attack strategy of *Proton*, in this specific case, is complicated and not entirely clear. HandBrake, a popular macOS media-manipulation application, had their servers hacked to replace version 1.0.7 of the application with a malicious copy.

This malicious copy contained the *Proton* malware. The *Proton*-containing HandBrake will request administrative credentials when launched. This is unlike previous genuine versions.



ii. It is believed that *Proton* attempts to decrypt the contents of a user's keychain, and upload any credentials to a remote server.

iii. In this scenario, *Proton* masquerades as a genuine application. This is, therefore, an advanced form of a Trojan - in that it isn't obvious that the application is malicious after installation.

References

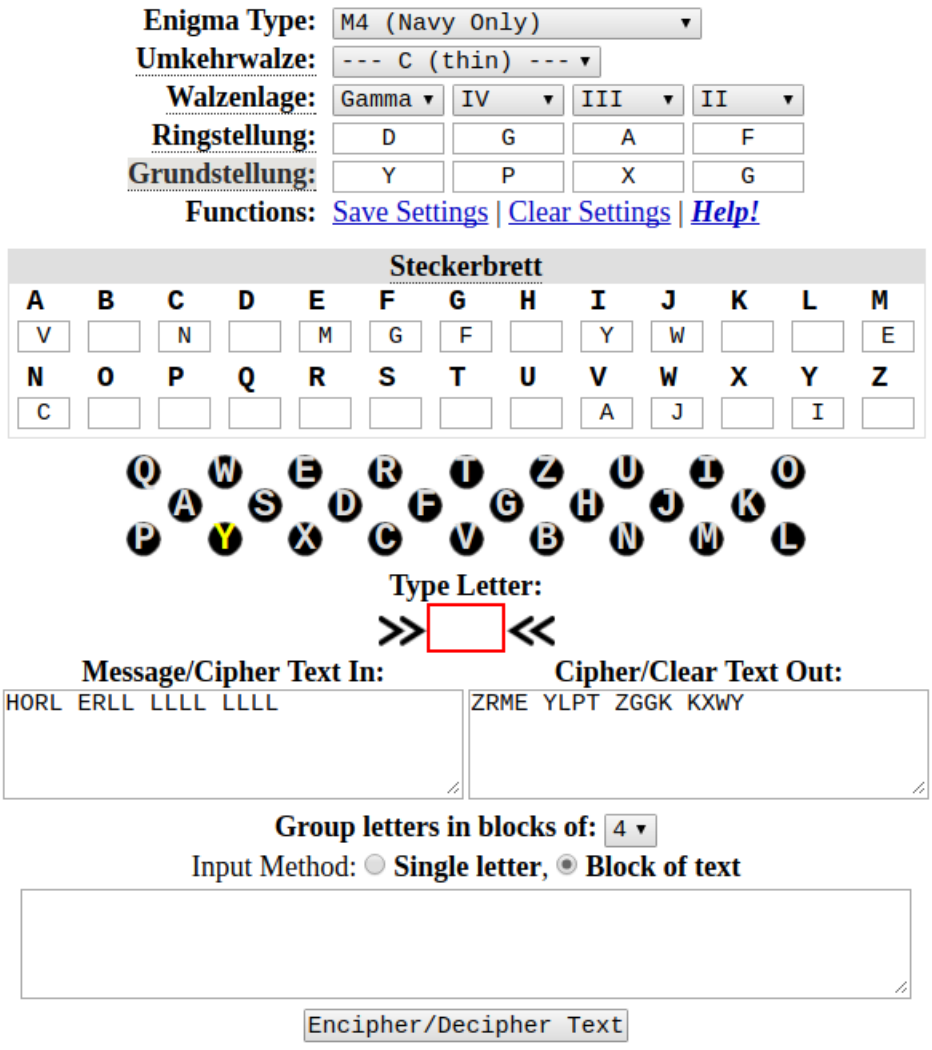
References for Task 1.2 and Task 1.3 are compounded.

https://www.theguardian.com/society/2017/may/12/hospitals-across-england-hit-by-large-scale-cyber-attack
https://virustotal.com/cs/file/b9c5d4339809e0ad9a00d4d3dd26fdf44a32819a54abf846bb9b560d81391c25/analysis
https://web.archive.org/web/*/https://virustotal.com/cs/file/b9c5d4339809e0ad9a00d4d3dd26fdf44a32819a54abf846bb9b560d81391c25/analysis/
https://www.symantec.com/security_response/landing/threats.jsp
https://www.microsoft.com/en-us/security/portal/threat/threats.aspx
http://www.virusradar.com/en/threat_encyclopaedia
https://www.avira.com/en/support-virus-lab
https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
https://www.youtube.com/watch?v=Zy4G30kSPnY
https://www.malwaretech.com/2017/05/how-to-accidentally-stop-a-global-cyber-attacks.html
https://blog.malwarebytes.com/threat-analysis/2017/05/the-worm-that-spreads-wanacrypt0r/
https://blog.malwarebytes.com/cybercrime/2017/05/new-jaff-ransomware-via-necurs-asks-for-2-btc/
https://blog.malwarebytes.com/threat-analysis/mac-threat-analysis/2017/05/handbrake-hacked-to-drop-new-variant-of-proton-malware/

2 Symmetric and asymmetric ciphers

The Enigma machine was a piece of encryption hardware used by the Germans to protect commercial, diplomatic and military communication before and during World War Two.

The task in this section is to encrypt a string consisting of my family name plus ten ‘L’ characters using Enigma with specified settings. This, known as the *plain text* is, **HORLERLLLLLLLLL**.



Plain text: **HORL ERLL LLLL LLLL**
Cipher text: **ZRME YLPT ZGGK KXWY**
Final ground setting (Grundstellung): **YPXG**

3 Defence Mechanisms

In this section, it is required that I calculate the result of my RMIT student number (*3481341*) modulo three, and answer questions depending on its result.

3481341 mod 3 = 0

As above, the result of my student number modulo three is zero. I will therefore, as per the assignment specification, answer question 3.0 on defense mechanisms.

Question 3.0

This question addresses the effectiveness of various *defense mechanisms* on *threats*. *Defense mechanisms* are listed in the first row, and *threats* are listed in the first column.

I answered this questions using coloured-coded responses, followed by an explanation. My definition of these responses is below.

- Very effective - Completely effective by any realistic interpretation.
- Effective - Effective by realistic interpretation, but not without limit.
- Partially effective - Only effective from certain elements of a threat, the threat in certain scenarios, or the threat's ramifications.
- Potentially effective - Effective or partially effective depending on interpretation of either the threat or the defense mechanism.
- Not effective - Not effective by any realistic interpretation.

	Firewalls embedded in the application	TLS/SSL	Two-factor authentication (2FA)	Signature-based intrusion detection
Trojans	Partially effective. A firewall can block connections from certain known malicious hosts that may serve malware, including trojans.	Not effective. Transport security cannot guarantee connections are void of malware.	Not effective. 2FA relates to authentication. 2FA can, however, protect a user from the ramifications of a keylogger (which could potentially be delivered as a trojan) logging their primary password.	Effective. The files associated with a common trojan can be analysed by an anti-malware program. The limitation is that signature-based intrusion detection requires that an anti-malware solution is updated with the required signature before the user is attacked.
Social engineering	Partially effective. Firewalls cannot protect against the human social element of social engineering, but they can block fraudulent emails based either on content or origin.	Not effective. Trusted HTTPS certificates can be issued to anyone.	Potentially effective. Some forms of social engineering involve stealing passwords. Stolen passwords are less useful when 2FA is used. 2FA, however, does not protect against all forms of social engineering. An example of this is calling a call center and requesting 2FA be disabled, or subverted, for an emergency.	Not effective. Social engineering, by its nature, involves people. People cannot be analysed in this way.
Spoofing	Potentially effective. An advanced firewall could verify a host beyond the level that a client browser, or other application, would. This is limited, regardless. The firewall itself could be victim to a spoofing attack.	Partially effective. Servers can identify themselves using their HTTPS certificate. Servers can also use response headers like HTTP Public Key Pinning to force a client to only accept certain certificates. HTTPS certificates, however, can be sought by anyone from numerous authorities. Authorities have been know to erroneously issue certificates to parties without provable ownership.	Potentially effective. 2FA does not protect against spoofing attacks, but it can protect against its ramifications. And example is an attacker spoofing a web server to intercept a user's password. Certain forms of 2FA (the types that change over time) would render this useless in the long term.	Not effective. Spoofing usually refers to spoofing of a service or host. This cannot be signatured in a means that realistically suits this question.

Replay attack	Not effective. A firewall that is embedded in an application cannot prevent a third-party from interfering with a request while it is in transit.	Very effective. Transport security will prevent a third-party from listening to network requests - thus making replaying those requests impossible.	Not effective. 2FA relates to authentication It cannot prevent replay attacks, although it can prevent some ramifications of replay attacks on authentication requests.	Not effective. Replay attacks simply cannot be prevented using this defense mechanism. This is due to the nature of replay attacks, and signature-based intrusion detection.
Person in the middle attacks	Not effective. A firewall that is embedded in an application cannot prevent a third-party from interfering with a request or response while it is in transit.	Very effective. Transport security makes it practically impossible for a third-party to intercept the connection without the victim accepting another root certificate, or downgrading the security of the connection.	Potentially effective. Depending on the nature of the attack. 2FA cannot prevent PITM (MITM) attacks, but can protect against some of its ramifications (eg. password theft).	Not effective. Person in the middle attacks simply cannot be prevented using this defense mechanism. This is due to the nature of replay attacks, and signature-based intrusion detection.
Denial of service attacks	Not effective. A firewall that is embedded in an application will not prevent requests from reaching a server.	Not effective. Transport security cannot prevent unwanted requests to a server. HTTP is identical to HTTPS in this case, in that the request is received in both cases.	Not effective. 2FA relates to authentication. DoS attacks don't care about authentication.	Potentially effective. Depending on interpretation. Network filters can implement 'signature-based' detection to detect a suspicious client connecting to a web server.
Cross-site scripting	Not effective. A firewall that is embedded in an application cannot prevent scripts from running in any scenario, beyond scripts hosted on malicious hosts. This would not prevent cross-site scripting.	Not effective. Transport security cannot prevent the client from injecting scripts into a page, willingly or unwillingly.	Not effective. 2FA relates to authentication. Cross-site scripting does not require user authentication to work.	Partially effective. Browsers are capable of doing this to some degree when the server has correctly set the X-XSS-Protection response header.
SQL injection attack	Not effective. A firewall that is embedded in an application cannot prevent SQL injection beyond blocking malicious requests. A firewall cannot judge SQL queries.	Not effective. Transport security will not filter unwanted and malicious requests	Not effective. 2FA relates to authentication. SQL injection mainly targets databases open to the greater Internet.	Not effective. As per my answer for DoS attacks, network filters can analyse the 'signature' of requests for suspicious intent. This does not prevent SQL injection in all cases, so it's not

				effective regardless.
Drive-by download attack	Not effective. A firewall that is embedded in an application could block a download if the content is hosted on a remote and untrusted server. This is, however, very limited and not realistic.	Potentially effective. Transport security cannot guarantee connections are void of malware. TLS/SSL can prevent content from being injected into pages - thus preventing a drive-by download attack via injection during transport.	Not effective. 2FA relates to authentication.	Partially effective. Signature-based intrusion detection cannot prevent an unwanted download, but it can prevent the download of a known malicious file. The limitation is that signature-based intrusion detection requires that an anti-malware solution is updated with the required signature before the user is attacked.
Session hijacking	Not effective. A firewall that is embedded in an application cannot prevent a third-party from interfering with a request or response while it is in transit.	Very effective. Transport security will encrypt a cookie while in transport. This can be boosted by setting the secure flag on cookies.	Not effective. A session can be hijacked regardless of the means of initial authentication.	Not effective. Session hijacking simply cannot be prevented using this defense mechanism. This is due to the nature of replay attacks, and signature-based intrusion detection.

Assumptions

- "*Firewalls embedded in the application*" refers to firewalls running in the Application layer of the TCP/IP model.
- "*TLS/SSL*" refers only to modern versions of TLS.

References

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Public-Key-Pins
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection
https://blog.mozilla.org/security/2016/10/24/distrusting-new-wosign-and-startcom-certificates/
https://arstechnica.com/security/2017/03/google-takes-symantec-to-the-woodshed-for-mis-issuing-30000-https-certs/

Submitted by Aaron Holer on May 18, 2017.