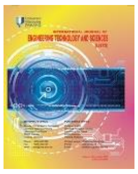
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY AND SCIENCES (IJETS)
ISSN: 2289-697X (Print); ISSN: 2462-1269 (Online) Vol.5 (3) December 2018
© Universiti Malaysia Pahang
DOI: http://dx.doi.org/10.15282/ijets.5.3.2018.1.1010



The Efficient Use of a List of Trusted Certificate Authorities

*[1]*Essam Alnatsheh*
[1]*AMA International University, College of Engineering, Department of Informatics Engineering, Bahrain*
**Email: dr_natsheh@hotmail.com*

Abstract- Nowadays, many Certificate Authorities (CAs) that issue certificates may or may not be trusted because not all CAs are reliable and trustworthy. University laboratories and computers of its people (students, lecturer and staff) are thus susceptible to the risk resulting from this mistrust. This study proposes a university owned notary server, which will be managed by the university, to solve the problem using a Certificate Trust List (CTL). Simply put, when students and others use the Web, the notary server checks the certificate to see whether a conflict exists and verifies the signatures and key references in the certificate. If all the information is correct, the notary server sends a response of approval to the client to accept the certificate. Our system enhanced the security in a university by trusting only genuine CAs. Our proposed server is better than regular notary servers because it uses existing infrastructure and online connections, and it does not introduce any overheads or special configurations to the client's Web browser. Compared with a well known notary server runs over the existing infrastructure, our proposed notary server is 10.8 seconds faster in terms of dealing with untrusted CA and 2.3 seconds faster in terms of dealing with mismatched address of the Web sites.

Index Terms- Man-in-the-middle attack; Certificate Authorities; Notary Servers; SSL protocol; PKI

I. INTRODUCTION

The Secure Socket Layer (SSL) protocol works as a key role in the Internet's security. It provides encrypted channels and secure authentication through its certificate infrastructure. An overview of how Certificate Authorities (CAs) sign SSL server certificates introduced by Wendlandt et al. [1]. This protocol suffers from the Man-in-the-Middle (MitM) attack that uses a malicious certificate. Although there are a number of alternative public key architectures (e.g., EFF's Sovereign Keys Project [17], Certificate Transparency [18]), there is no real solution in sight [2], [19].

In this study, we used notary servers as an alternative approach. Figure 1 shows an overview of the client-notary method: (1) The client sends an approval request to the notary, (2) the notary connects to the client over a public channel, (3) the client sends its certificate, and (4) the notary sends an approval response to the client.

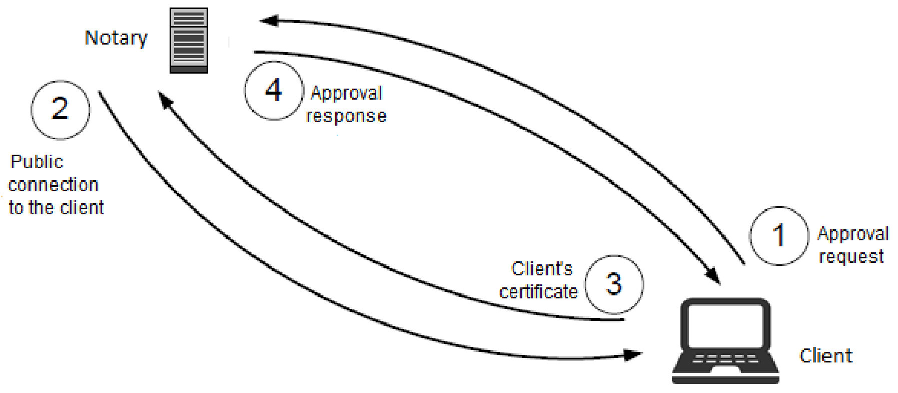


Figure 1: The client-notaries method overview

Our study follows this approach by offering a notary service for SSL clients of university users. We implemented our study at the College of Computer Science and Information Technology (CCSIT) at King Faisal University (KFU) in Saudi Arabia. The motivation for our project is to help CCSIT validate CAs and prevent the Web browsers from accepting SSL users' certificates from untrusted CAs. To our knowledge, no computer of a CCSIT student or staff has a technique to eliminate downloading certificates from unknown or fake CAs. We therefore aim to build a notary server that will give CCSIT users an ability to know which organizations deserve their trust.

II. RELATED WORK

Several proposals have incorporated the idea of observing the server certificate to improve the Web PKI trust model. Wendlandt et al. [3] first introduced this idea in Perspectives in 2008, when they defined notaries as publicly available semi-trusted servers deployed at various locations on the network. The main idea is that, after a client obtains a certificate in the usual way, its Web browser compares the received certificates with a certificate obtained from a notary server. Any differences between the certificates might indicate a certificate substitution.

DoubleCheck [4], proposed in 2009 by Alicherry and Keromytis, trust the CA by retrieving the certificate from a remote server using two alternate paths. The client application then compares the certificate it received on the two paths. If they are different, the connection is aborted as there is a possibility of a MitM attack. It is clear that this method will generate an overhead to the connection time because of needs to multiple path connections. A solution to overcome this overhead was proposed by making the notary receives the server's certificate and observe them [15]. However, this solution will increase the overhead on the notary servers.

In 2010, the Electronic Frontier Foundation (EFF) launched SSL Observatory as a project to investigate the certificates used to secure all of the sites encrypted with https on the Web [5]. The EFF downloaded all the SSL certificates worldwide and built a dataset that is accessible to everyone can access. The certificate observations in the database thus come from a vast range of locations around the Internet. However, their datasets need to be revised and reorganized. Another drawback is that the SSL Observatory is a centralized service that users have to trust, which adds complexity.

In 2011, Moxie's Convergence deployed a number of notaries, in the hope that users would deploy many more, and a Firefox extension that the user could configure with their choice of notaries [6] [7]. However, Moxie's version seems to no longer have working public notaries. While a live fork remains, it only contains private notaries, and it does not support Convergence's technique (called "bounce notaries"), which prevents a user's request for certificate validation revealing their browsing history to the notaries. In 2012, Holz et al. [8] introduced the Crossbear system which uses Convergence as a source of independent observations from other vantage points on the Internet. It queries Convergence notaries and stores certificate information. However, this system add a huge overhead to the Internet because of its need to add many databases (Certificate database, Observation database, and Hunting Task database).

In early 2012, Amann et al. [9] presented an ICSI (International Computer Science Institute) notary server that provides clients with a third-party perspective on certificates they should expect to receive from a server. They collect certificates from live Internet traffic at seven different sites, providing users with a near real-time view of certificates in actual use by a large client population. At the end of 2012, they offered a large-scale study of SSL traffic [10]. They reassessed previous findings with a broader data set, examined a range of further aspects, and contributed to improving end-user security by making the collected Web certificates available to the public in the form of an online notary service. In 2013, they shared their results with browser vendors to reevaluate their current warning mechanisms and conserve user attention [11]. In 2014, they used their notary server to trust CAs that their mobile device and its applications trust [12]. However, they have not released their raw data because they claim that they need to account for sensitivity concerns at their data providers; that drawback is likely the most valuable part for the broader community because it provides clients with a measure of notary server reputation.

Taking applications for using notary servers in a new direction, some researchers used notary server concept in providing users with the ability to model trust relationships that reflect their social relationships. Micheloni et al. [13] presented Laribus, a social based network, which allows users to trust their friends, and not unknown organizations, with certificate validation. Huang et al. [14] used notary server for detecting SSL MitM attacks against a Facebook's users.

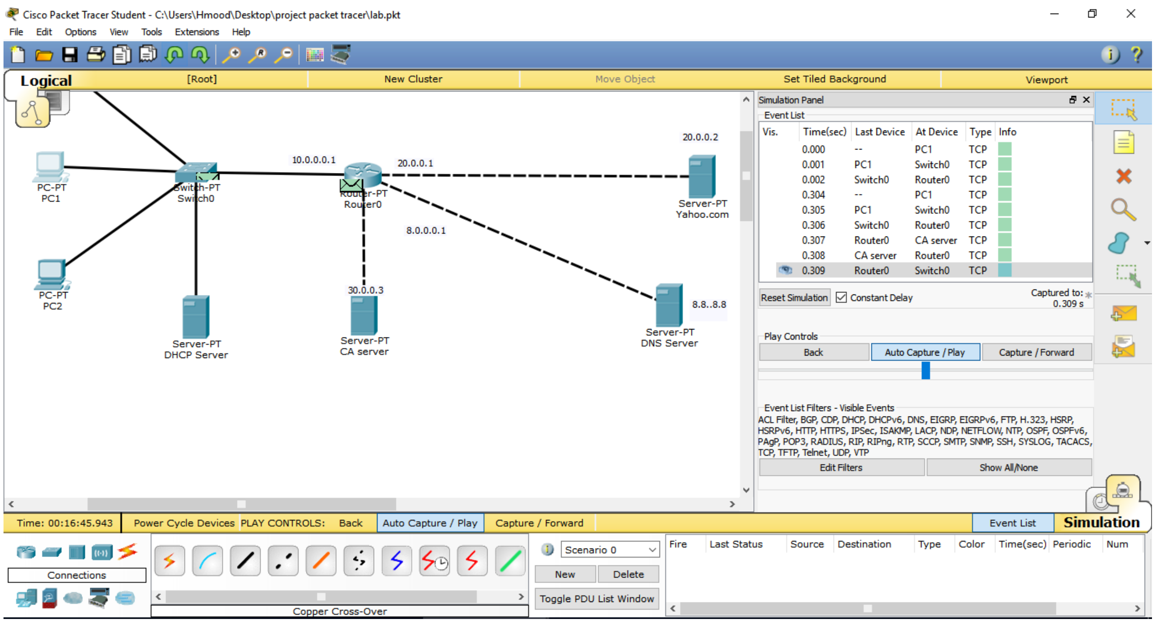
Our CCSIT notary server takes a different approach on how the Web browser determines whether an SSL certificate is valid. Instead of requiring browser users to trust an anointed group of CAs, our CCSIT notary server gives the user the ability to pick a group they trust (e.g., Google, CCSIT CA, KFU CA, Saudi National root CA) and to trust no others.

III. PROPOSED SYSTEM

In the following section, we simulate our proposed network to analyze its requirements. Section 3.2 shows the real implementation of the network, and Section 3.3 focuses on the details of implementing the notary server.

3.1 Simulating CCSIT network

We used Packet Tracer simulation tool to design and analyze our proposed network for CCSIT.



Vis.	Time(sec)	Last Device	At Device	Type	Info
	0.000	--	PC1	TCP	🟩
	0.001	PC1	Switch0	TCP	🟩
	0.002	Switch0	Router0	TCP	🟩
	0.304	--	PC1	TCP	🟩
	0.305	PC1	Switch0	TCP	🟩
	0.306	Switch0	Router0	TCP	🟩
	0.307	Router0	CA server	TCP	🟩
	0.308	CA server	Router0	TCP	🟩
👁	0.309	Router0	Switch0	TCP	🟦

Event List Filters - Visible Events
ACL Filter, BGP, CDP, DHCP, DHCPv6, DNS, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, IPSec, ISAKMP, LACP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, RADIUS, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

Figure 2: Simulation of CCSIT network

All devices in the network were connected as shown in Figure 2. We did the required configuration for Dynamic Host Configuration Protocol (DHCP) scope, Domain Name System (DNS), and Default Gateway, and we created an A record for a website at the DNS server. This record is used in the forward lookup process during the translation from domain name to IP address. As shown, we used www.yahoo.com as our test Web domain name that will host the website.

The navigation flow diagram shown in Figure 3 shows the certificate request process (between the client browser and the certificate authority), the certificate acceptance process (between the client browser and the notary server), and the certificate approval process (between the notary server and the certificate authority).

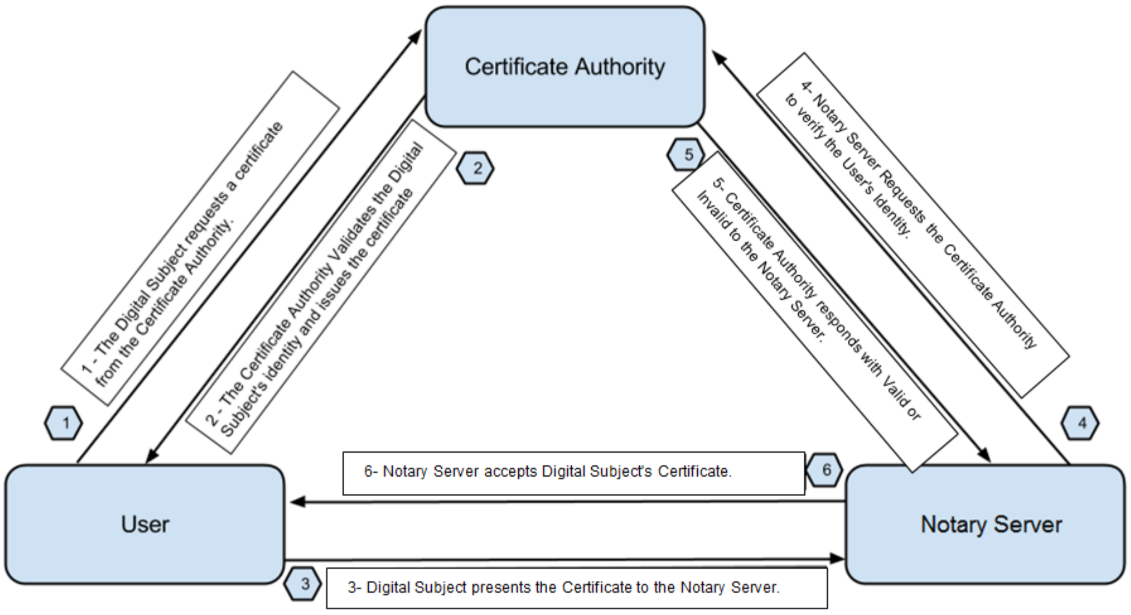

Certificate Authority
User
Notary Server
1 - The Digital Subject requests a certificate from the Certificate Authority.
2 - The Certificate Authority Validates the Digital Subject's identity and issues the certificate
3- Digital Subject presents the Certificate to the Notary Server.
4- Notary Server Requests the Certificate Authority to verify the User's Identity.
5- Certificate Authority responds with Valid or Invalid to the Notary Server.
6- Notary Server accepts Digital Subject's Certificate.

Figure 3: Navigation flow diagram

3.2 Implementing CCSIT Network

In this phase, we conducted a real implementation of our system, as shown in Figure 4. First, we installed and configured Microsoft Windows Server 2012 on three computers, which we named Server1, Server2, and Server3, and we installed Microsoft Windows 7 operating system on one client computer.

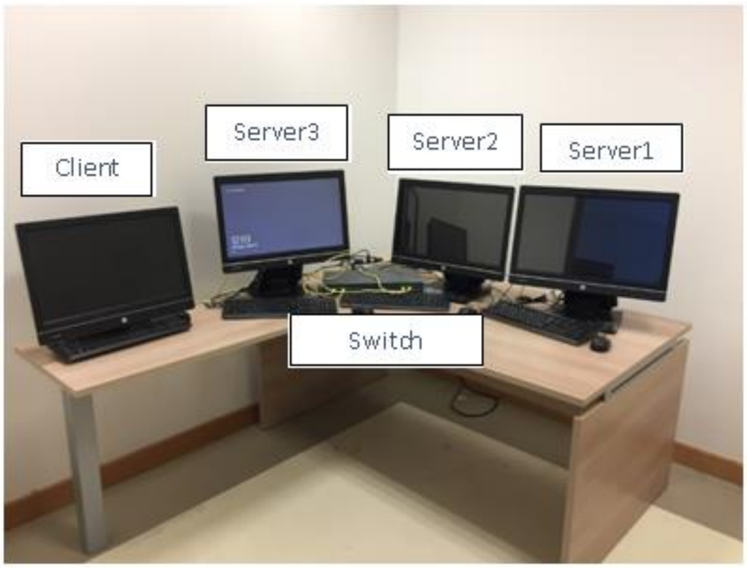
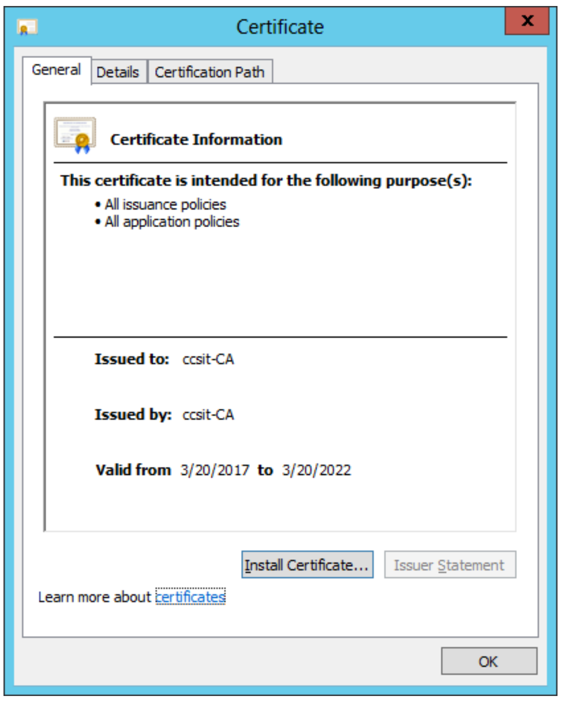


Figure 4: Real implementation of the proposed system

We installed a DNS service on Server2 and created a primary zone, which we called ccsit.edu. We also installed an Active Directory Domain Service, created the ccsit.edu domain, and added all computers to that domain.

We installed an IIS service on Server3 to host the website of CCSIT. In IIS, we bound the CCSIT website with its name, physical path, http protocol, IP address, and port address. To test the new web site, we opened the website (http://www.ccsit.edu) page using the client's Web browser.
We also installed the CCSIT CA on Server3 and configured it with the name ccsit-CA. Because it has access to Active Directory, it is configured as an enterprise CA. Additionally, because it is the first CA in our domain, it is configured to be the root CA. Besides that, other configurations including the cryptographic provider, key length, and hash algorithm for signing certificates issued by the CA have been configured. Figure 5(a) shows the certificate issued for the CCSIT root CA.



Certificate Information

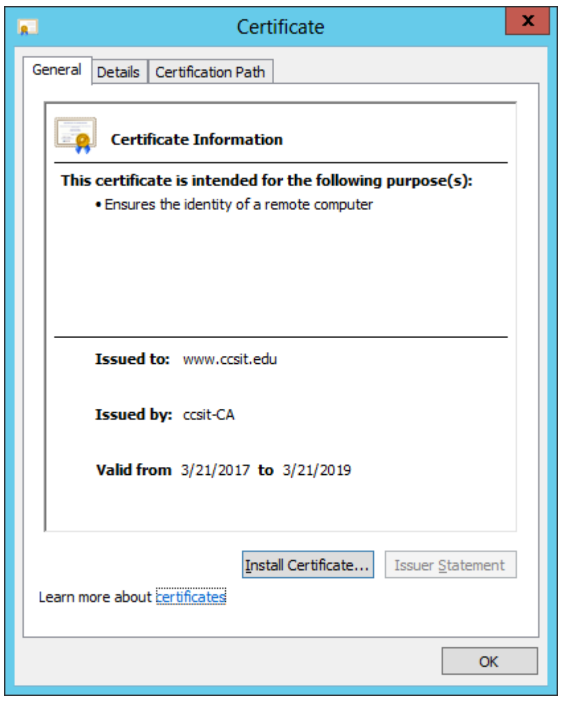
This certificate is intended for the following purpose(s):
- All issuance policies
- All application policies

Issued to: ccsit-CA

Issued by: ccsit-CA

Valid from 3/20/2017 to 3/20/2022

(a) The certificate issued for the root CA



Certificate Information

This certificate is intended for the following purpose(s):
- Ensures the identity of a remote computer

Issued to: www.ccsit.edu

Issued by: ccsit-CA

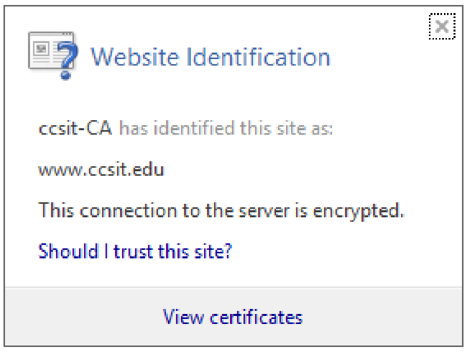
Valid from 3/21/2017 to 3/21/2019

(b) CCSIT website certificate

Figure 5: The issued certificates

To apply the SSL technology, we used https to access the college website, https://www.ccsit.edu, by making it use ccsit-CA certificates to prove the identity of its Web address and to make a secure channel through which to transfer information between the client and the server. For SSL to work properly, each website hosted on our Web server needs a separate certificate. Figure 5(b) shows the certificate that our ccsit-CA issued to our college website, and then the certificate bound with the website using the Site Binding page on the IIS server.

As our root CA certificate would not be recognized or trusted by the client browsers, we imported the website certificate to the browser's authority's store to get the browsers to trust our root CA. Then, we initiated SSL connections using the URL, https://www.ccsit.edu/. The https protocol used our SSL certificate and opened our college website. The Web browser showed a message clarifying that the CA had identified our CCSIT website (Figure 6).


Website Identification

ccsit-CA has identified this site as:

www.ccsit.edu

This connection to the server is encrypted.

Should I trust this site?

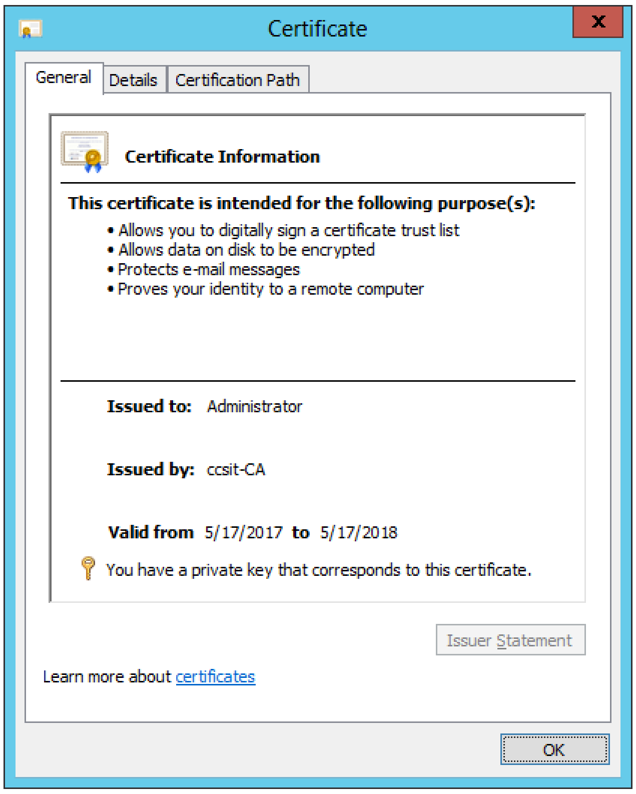
View certificates

Figure 6: A message clarifying that the CA identified the CCSIT website

3.3 Implementing CCSIT Notary Server

In this section, we created the CCSIT notary server using a Certificate Trust List (CTL) for the CCSIT's domain. A CTL is a predefined list of items signed by a trusted entity. All items in the list are authenticated and approved by a trusted signing entity. After creating the CTL, we prevented the CCSIT's domain users from using any certificate not existing on our CTL by using a domain group policy.

To create the CTL, we first needed to configure our domain with a list of CAs the domain trusted. We also needed an Administrator certificate or an explicit Trust Signing Certificate. Therefore, on Server2, which contains the active directory, we successfully requested and installed an Administrator Certificate issued by CA ccsit-CA (on Server3), which would allow the administrator to digitally sign a CTL (Figure 7).



Certificate Information

This certificate is intended for the following purpose(s):

- Allows you to digitally sign a certificate trust list
- Allows data on disk to be encrypted
- Protects e-mail messages
- Proves your identity to a remote computer

Issued to: Administrator

Issued by: ccsit-CA

Valid from 5/17/2017 to 5/17/2018

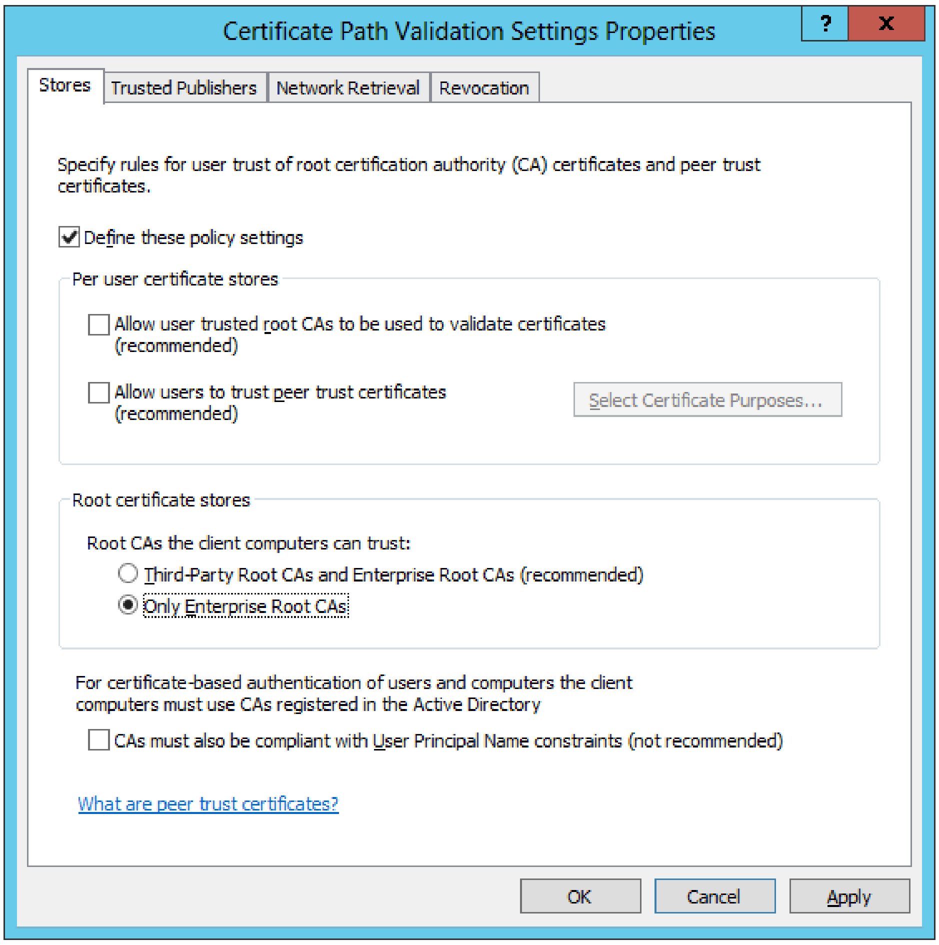
You have a private key that corresponds to this certificate.

Figure 7: An administrator certificate that allows the administrator to create a CTL

Using a domain group policy, we initiated a new CTL. In CTL wizard, we typed the name of the CTL, and we selected its purposes to Any Purpose. Then, we added certificates we trusted from the store to the CTL.

Certificate path validation in Microsoft Windows Server allowed us to manage the settings for certificate path discovery and validation for all users in our CCSIT domain, as shown in Figure 8. We used the domain group policy to easily configure and manage these certificate validation settings. Using these settings, we deployed CA certificates and blocked certificates that were not trusted. Hence, we prevented users in the CCSIT domain from configuring their own sets of trusted root certificates and deciding which root certificates within CCSIT could be trusted. The Stores tab was used to accomplish this using the following options [16]:

- Allow user-trusted root CAs to be used to validate certificates. Clearing this check box prevents users from deciding which root CA certificates to use to validate certificates.
- Allow users to trust peer trust certificates. Clearing this check box prevents users from deciding which peer certificates to trust. This option can help prevent users from trusting certificates from a source that is not secure.
- Only Enterprise Root CAs. By restricting the trust to only enterprise root CAs, we effectively restrict the trust to certificates issued by an internal enterprise CA, which obtains authentication information from and publishes certificates to the Active Directory Domain Services (AD DS).



Certificate Path Validation Settings Properties

Stores | Trusted Publishers | Network Retrieval | Revocation

Specify rules for user trust of root certification authority (CA) certificates and peer trust certificates.

[✓] Define these policy settings

Per user certificate stores

[ ] Allow user trusted root CAs to be used to validate certificates (recommended)

[ ] Allow users to trust peer trust certificates (recommended)

Select Certificate Purposes...

Root certificate stores

Root CAs the client computers can trust:

○ Third-Party Root CAs and Enterprise Root CAs (recommended)

● Only Enterprise Root CAs

For certificate-based authentication of users and computers the client computers must use CAs registered in the Active Directory

[ ] CAs must also be compliant with User Principal Name constraints (not recommended)

What are peer trust certificates?

OK | Cancel | Apply

Figure 8: Configuring certificate path validation

3.4 Testing CCSIT Notary Server

To test our system, we created a fake website and bound it with a certificate from an untrusted CA. Then, we proved that the client could access that site and received a message indicating an invalid certificate due to an untrusted CA.

First, on the IIS server, we bound the website www.fake.com with its name, physical path, http protocol, IP address, and port address.

Next, we bound the fake website with a certificate from an untrusted CA. To do that, we used a self-signed certificate from our Server3. Figure 9 shows that Site Binding page used a fake SSL certificate. We forced the website to use the https protocol only. At the client Web browser, we connected to https://www.fake.com using a secure SSL session. Figure 10 shows clearly that the browser at the client PC refused to access the website and showed a message saying that this website uses a certificate that is not issued by a trusted CA. To continue our test, we clicked “Continue to this website.” Access to the website was denied because the CA does not exist in CCSIT’s CTL. The Certificate Invalid message appeared in the client’s Web browser, as shown in Figure 11. We clicked “View Certificate” to read its contents. Figure 12 shows that this CA root certificate was not trusted.

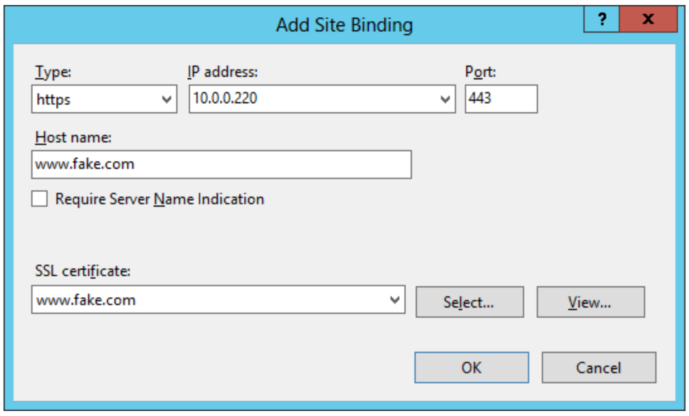
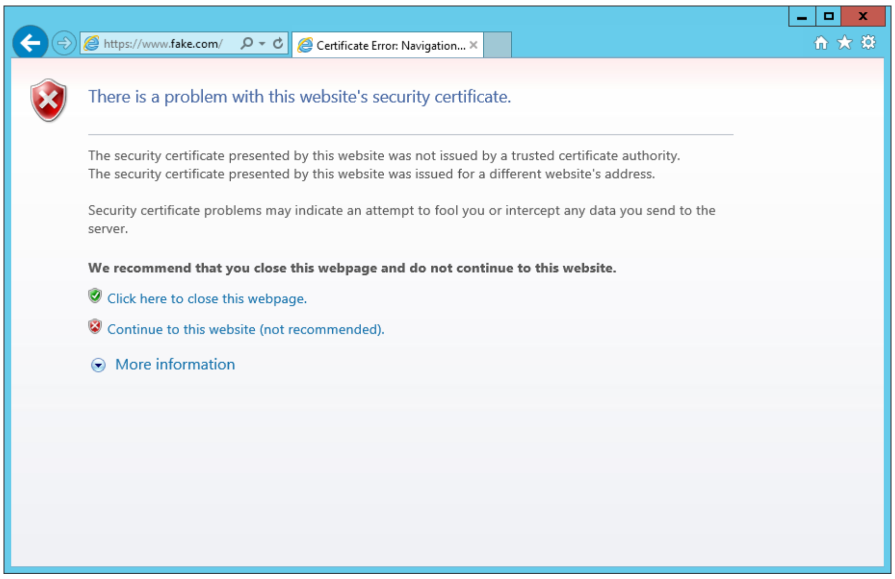


Figure 9: Binding the website with a fake SSL certificate



There is a problem with this website's security certificate.

The security certificate presented by this website was not issued by a trusted certificate authority.
The security certificate presented by this website was issued for a different website's address.

Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server.

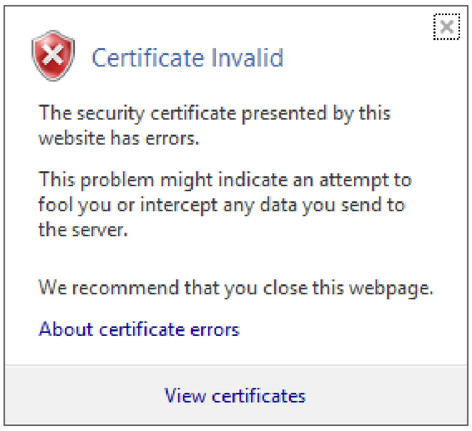
We recommend that you close this webpage and do not continue to this website.

Click here to close this webpage.

Continue to this website (not recommended).

More information

Figure 10: Accessing the website with a non-trusted certificate



Certificate Invalid

The security certificate presented by this website has errors.

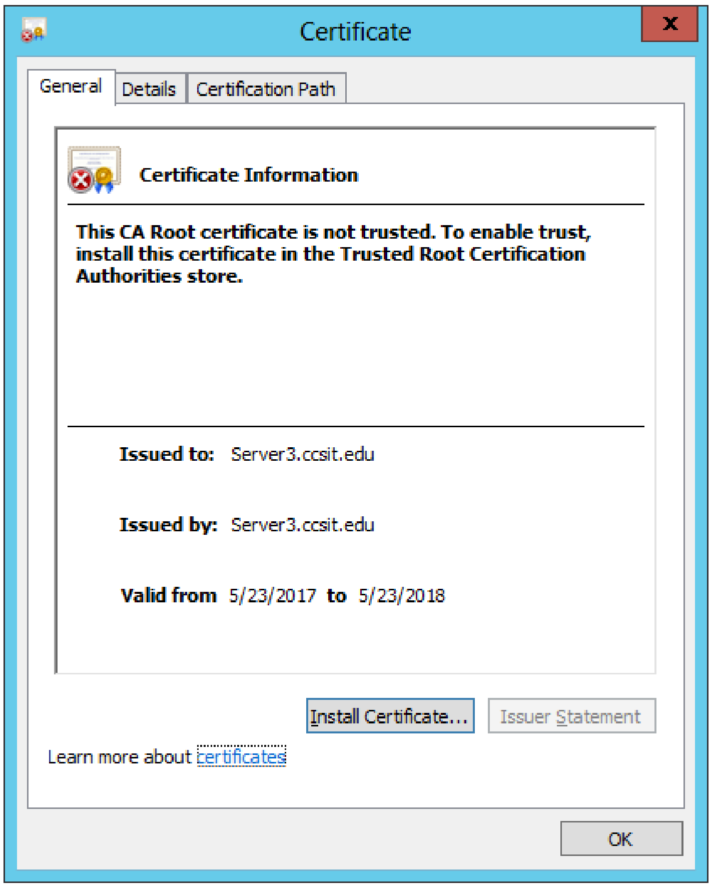
This problem might indicate an attempt to fool you or intercept any data you send to the server.

We recommend that you close this webpage.

About certificate errors

View certificates

Figure 11: Denying access to the fake website



Certificate Information

This CA Root certificate is not trusted. To enable trust, install this certificate in the Trusted Root Certification Authorities store.

Issued to: Server3.ccsit.edu

Issued by: Server3.ccsit.edu

Valid from 5/23/2017 to 5/23/2018

Figure 12: Untrusted CA certificate

To test certificates with mismatched addresses, we bound the https://www.fake.com website with a certificate from a trusted CA, ccsit-CA. We used the certificate of the website, www.ccsit.edu. The connection was accepted and the Web page was displayed. However, we received a message indicating a mismatch between the address of the Web page and the address on the certificate; this proves that the Web browser of the client accepted the certificate because its CA exists in the CTL of the CCSIT Notary Server.

IV. PERFORMANCE ANALYSIS

4.1 Impact of CCSIT notary server on Web browsing

To study the impact of CCSIT notary server on the clients' Web browsing, we measured the time it takes to load the Web pages that have certificates with untrusted CAs and Web pages that have certificates with an address mismatch. We ran each experiment 10 times and compared the average time with DoubleCheck [4]. Figure 13 shows that our proposed notary server is 10.8 seconds faster in terms of dealing with untrusted CA and 2.3 seconds faster in terms of dealing with mismatched address of the Web sites.

When there is a certificate with an untrusted CA or an address mismatch, DoubleCheck fetches the server certificate using the Tor network. This incurs an additional overhead compared to the CCSIT notary server. However, when using the CCSIT notary server or using DoubleCheck with a disabled status and the website has a certificate with a mismatched address, the user is notified of the mismatched address and is allowed to open the Web page. If DoubleCheck is enabled, it will not allow the user to open that Web page. Finally, if there is no error in the certificate, the https sessions will be established, and the Web page will be opened with no more overheads on either notary server.

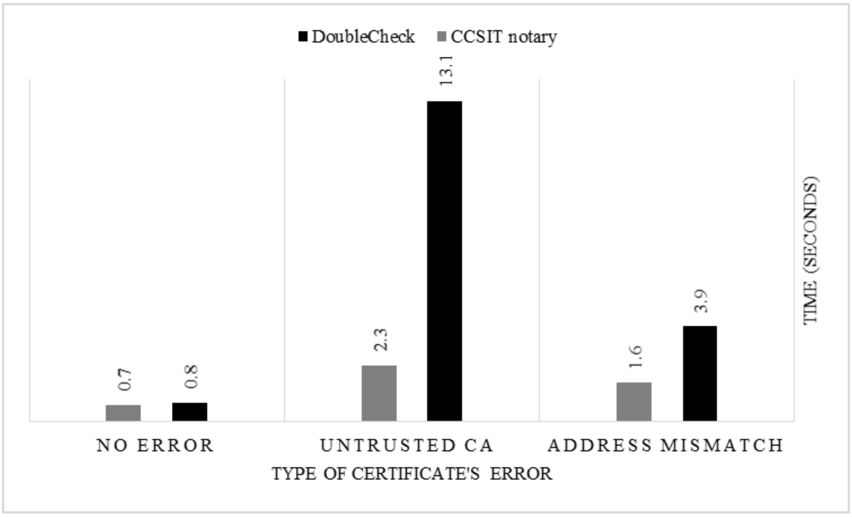


Figure 13: Impact of the CCSIT notary server on web browsing

4.2 Comparing the CCSIT notary server with others

In this section, we compare our CCSIT notary server with Perspectives [3] and DoubleCheck [4]. Our CCSIT notary server performs a subset of the functions that other notary servers support, but with additional advantages. Table 1 summarizes the main differences.

The major advantage of the CCSIT notary server is that it runs over the existing infrastructure. Perspective’s clients need to connect to Perspective’s servers, while DoubleCheck clients need to connect to the Tor neywork (as an alternate path to the CAs) to trust the CAs. The CCSIT clients receive a list of trusted CAs when they login to the domain. Unlike Perspectives and DoubleCheck, no proxy or servers connected to the Internet are needed for the CCSIT notary server. The domain group policy will affect the clients at the time they access the domain. Hence, the CCSIT notary server does not need additional tools and is easier to deploy in practice. In addition, the clients Web browser does not need Firefox extensions, unlike with Perspectives and DoubleCheck.

Privacy concerns are possible in the CCSIT notary server because the CTL contains only the specified CA that the clients need to connect to them. A client using Perspectives will not be able to validate the keys for new servers unknown to the notaries. This can be solved easily when using CCSIT notary server by updating the CTL.

Table 1: Comparison of the CCSIT notary server with others

	CCSIT notary server	Perspectives	DoubleCheck

New infrastructure	Not needed	Needed	Needed
Client connection to notary server	Offline	Online	Online
Methodology	Offline retrieval	Offline retrieval	On-demand
Easy of deployment	Yes	No	No
Special configuration on the clients' Web browser	No	Yes	Yes
Privacy concerns (Private trusted CA)	Yes	No	No
New server or key change	Supported	Not supported	Supported

V. CONCLUSIONS

As evidenced by its widespread use, SSL authentication offers a simple and attractive technique for overall security using PKI. Unfortunately, "Trust-on-first-use" leaves users vulnerable to simple MitM attacks, limiting the effectiveness of current trusted CAs. To enhance security, we designed the CCSIT notary server to define trusted CAs using the CTL method and deployed it to the domain using a domain group policy. Our implementation demonstrates that the notary concept is practical; our CCSIT clients found the notary server to be invaluable on several occasions, when logging in to CCSIT and trying to connect to the Internet, or when logging in to a CCSIT and trying to connect to the CCSIT servers. As a result, we believe that the CCSIT notary server is a practical approach to improving the security of CCSIT users communicating with SSL and https.

REFERENCES

[1] D. Wendlandt, D. G. Andersen, and A. Perrig, Perspectives project. Retrieved June 28, 2017, from https://perspectives-project.org/

[2] "What Is A Certificate Authority?". Retrieved April 16, 2018, from https://perspectivessecurity.files.wordpress.com/2011/07/perspectives_usenix08.pdf

[3] D. Wendlandt, D. G. Andersen, and A. Perrig, "Perspectives: Improving SSH-style host authentication with multi-path probing", USENIX Annual Technical Conference, pp. 321–334, 2008.

[4] M. Alicherry and A. D. Keromytis, "Doublecheck: Multi-path Verification Against Man in-the-Middle Attacks", Computers and Communications, IEEE Symposium, pp. 557–563, 2009.

[5] EFF: The EFF SSL Observatory, Retrieved April 16, 2018, from https://www.eff.org/observatory

[6] M. Marlinspike, Convergence, Retrieved April 16, 2018, from http://conergence.io

[7] M. Marlinspike, blog post, Retrieved April 16, 2018, from https://moxie.org/blog/ssl-and-the-future-of-authenticity/

[8] R. Holz, T. Riedmaier, N. Kammenhuber, and G. Carle, "X. 509 Forensics: Detecting and Localising the SSL/TLS Men-in-the-Middle", Computer Security–ESORICS 2012, pp. 217–234, Springer, 2012.

[9] J. Amann, M. Vallentin, S. Hall, and R. Sommer, "Extracting Certificates from Live Traffic: A Near Real-Time SSL Notary Service", International Computer Science Institute ICSI Technical Report, November 2012.

[10] J. Amann, M. Vallentin, S. Hall, and R. Sommer, "Revisiting SSL: A Large Scale Study of the Internet's Most Trusted Protocol" ICSI Technical Report, December 2012.

[11] D. Akhawe, J. Amann, M. Vallentin, and R. Sommer, "Here's My Cert, So Trust Me, Maybe? Understanding TLS Errors on the Web", Proc. of International World Wide Web Conference, May 2013

[12] N. Vallina-Rodriguez, J. Amann, C. Kreibich, N. Weaver and V. Paxson, "A Tangled Mass: The Android Root Certificate Stores", ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT), December 2014.

[13] Micheloni, K. Fuchs, D. Herrmann, and H. Federrath, "Laribus: Privacy-Preserving Detection of Fake SSL Certificates with a Social P2P Notary Network", Eighth International Conference on Availability, Reliability and Security (ARES), pp. 1–10, November 2013.

[14] L.S. Huang, A. Rice, E. Ellingsen, and C. Jackson, "Analyzing Forged SSL Certificates in the Wild", IEEE Symposium on Security and Privacy, pp. 83–97, 2014.

[15] E. Yüce and A. Selçuk, "Server Notaries: A Complementary Approach to the Web PKI Trust Model", IACR Cryptology ePrint Archive 2016: 126, 2016.

[16] Manage Certificate Path Validation, Retrieved April 16, 2018, from https://technet.microsoft.com

[17] The Sovereign Keys Project, Retrieved April 16, 2018, from https://www.eff.org/ar/sovereign-keys

[18] J. Gustafsson, G. Overier, M. Arlitt, and N. Carlsson, "A First Look at the CT Landscape: Certificate Transparency Logs in Practice", Proc. of 18th International Conference on Passive and Active Measurement, pp. 87-99, 2017.

[19] T. Fadai, S. Schrittwieser, P. Kieseberg, and M. Mulazzani, "Trust me, I'm a Root CA! Analyzing SSL Root CAs in Modern Browsers and Operating Systems", Proc. of International Conference on Availability, Reliability and Security (ARES), pp. 174-179, 2015.