Overcoming Vulnerabilities in Public Key Infrastructure and Certificate Authorities

[1]Akash Nair, [2]Chinmay Mahajan, [3]Dr. Shubhalaxmi Joshi,
[#]MCA Dept, MAEER'S MIT School of Management,
Paud Road, Kothrud,
Pune, India

Abstract–For several years now, digital certificates have been implemented as a means of protecting the confidentiality and integrity of data travelling over the internet. However, there have been numerous criticism of certificate based browser encryption by security experts. Several cases of Certificate Authority (CA) and Secure Sockets Layer (SSL) exploits have exposed the vulnerability of CA based authentication.

In March 2011, an Iranian hacker broke into Comodo and forged bogus certificates for Google's email services.

In another similar instance, in August 2011, an unauthorized intrusion into DigiNotar's CA cause several bogus public key certificate requests to be issued which subsequently led to the company getting bankrupt.

In this paper, we propose encrypting web based traffic using dynamically generated keys in order to secure communications between client and server. This research aims at providing an alternative to the conventional CAbased authentication which is prone to several weaknesses as substantiated by the excerpts above.

The encryption model consists of of a set of cryptographic keys which are unique to the corresponding entity. The set of keys are unique and different for each domain (website) being visited.

Using these keys the traffic is encrypted and thus, the data is significantly safeguarded against man-in-the-middle (MITM) type of attacks.

Keywords—certificate authority; public key infrastructure; browser security, man-in-the-middle, SSL, PKI;

I. INTRODUCTION

In a Public Key Infrastructure (PKI), the Certificate Authority assumes the role of authentication done in the form of issuing digital certificates. The issuance of Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates to web site operators are proof that the Certificate Authority(CA) has verified that the web site operator owns the domain name in question.

If the browser is able to tie the certificate to a trusted root CA successfully, it indicates to the user that it is communicating with the true owner of the domain name and not a man-in-the-middle.

However, any root authority can create an illegitimate certificate for purposes of infiltration/hacking. For instance, the Hong Kong post office can create a valid Google certificate and this can be used by them to access your information.

A web browser will give no warning to the user if a web site suddenly presents a different certificate even if it has a different provider. Where certificate providers are under the jurisdiction of governments, those governments may have the freedom to order the provider to generate any certificate, such as for the purposes of law enforcement. Subsidiary wholesale certificate providers also have the freedom to generate any certificate.

In 2011, DigiCertSdn. Bhd., a sub-CA, was revealed to have been issuing certificates with weak keys. Attackers used this vulnerability to impersonate the legitimate owners by making their own code-signing certificates.[1]

Other incidents at larger CA's such asComodo, VeriSign and Trustwavehave exposed the vulnerability of the existing CA authentication system.

A digital certificate issued in an unauthenticated manner can make the problem worse by giving people false confidence in the identity of the sender of a message or the singer of a document.

In order to overcome this weakness presented by Security Certificate authentication, we have proposed a dynamic encryption model to encrypt the traffic being passed between the client and server.

II. OBJECTIVE

To thwart the possibility of an attack due to CA vulnerabilities and to safeguard the user from falling victim to a compromised CA and being fooled into thinking they are using a trusted connection.

We aim to reduce the reliance on CA as much as possible and encrypt data on the fly.

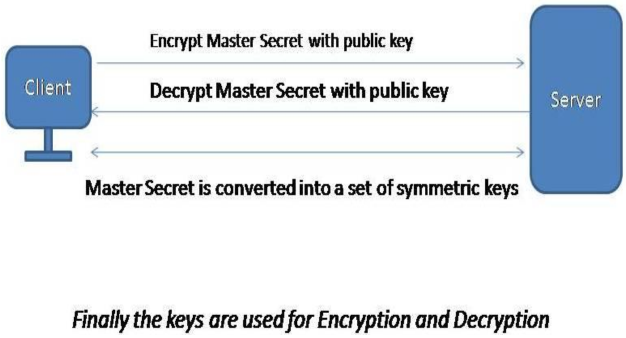

Encrypt Master Secret with public key
Client
Decrypt Master Secret with public key
Server
Master Secret is converted into a set of symmetric keys
Finally the keys are used for Encryption and Decryption

Fig 1.1 Existing encryption model

III. SCOPE

The protocol can majorly be implemented in two ways:
(i) Creating a browser extension/plugin/add-on
(ii) Being built-in to the browser itself

Depending on the type of implementation, the scope can vary from either being browser specific to being a globally implemented protocol.

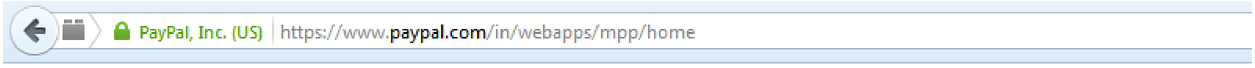
IV. STRENTHENING THE AUTHENTICATION SYSTEM

All web browsers come with an extensive built-in list of trusted root certificates, many of which are controlled by organizations that may be unfamiliar to the user. Each of these organizations is free to issue any certificate for any web site and have the guarantee that web browsers that include its root certificates will accept it as genuine. In this instance, end users must rely on the developer of the browser software to manage its built-in list of certificates and on the certificate providers to behave correctly and to inform the browser developer of problematic certificates. While uncommon, there have been incidents, in which fraudulent certificates have been issued: in some cases, the browsers have detected the fraud; in others, some time passed before browser developers removed these certificates from their software.[6]

The list of built-in certificates is also not limited to those provided by the browser developer: users (and to a degree, applications) are free to extend the list for special purposes such as for company intranets. This means that if someone gains access to a machine and can install a new root certificate in the browser, that browser will recognize websites that use the inserted certificate as legitimate.[6]

Browser extensions like Certificate Patrol, http://patrol.psyced.org, are designed to alert users when certificates change or seem suspiciously inconsistent. Such extensions have enjoyed limited

adoptionrates because they require savvy users who understand the nature of digital certificates. More recent proposals, such as the Internet-Draft “Public Key Pinning Extension for HTTP”, appear poised for greater adoption. These approaches take a Trust on First Use (ToFU) approach and simply terminate connections if the keys are inconsistent with those that were indicated in the first connection[7]


PayPal, Inc. (US) https://www.paypal.com/in/webapps/mpp/home

The padlock displayed in the location bar is an indication of a secure connection. The same padlock will be displayed even if bogus certificates are issued to the browser.

As the CA system was originally designed and is currently implemented, all root CAs are equally trusted by the browsers. That is, each of the 264 root CAs trusted by Microsoft, the 166 root CAs trusted by Apple, and the 144 root CAs trusted by Firefox are capable of issuing certificates for any website, in any country or top level domain [7].

For example, even though Bank of America obtained its current SSL certificate from VeriSign, there is no technical reason why another CA, such as GoDaddy, cannot issue another certificate for the same site to someone else. Should a malicious third party somehow obtain a certificate for Bank of America’s site and then trick a user into visiting their fake web server (for example, by using Domain Name Server (DNS) or Address Resolution Protocol (ARP) spoofing), there is no practical, easy way for the user to determine that something bad has happened, as the browser interface will signal that a valid SSL session has been established.[8]

Implementation Details

The encryption model we have proposed consists of the usage of discrete encryption keys in order provide a strong encryption model. It primarily involves the domain name (URL) of the sites visited in the hashing routines. This makes the keys generated site-specific and therefore substantially increases the robustness of the model. It makes use of the following fundamental entities:

Description of keys and IDs to be used:

(a) GUID: Globally Unique Identifier.

A thirty-two characters string that can be generated using a browser native language like JavaScript.

(b) UID: User Identifier

It is the site specific user ID and is different for every other site. Since it is not in human readable form, a “nick” can be associated with it.

(c) GEN: GENerator

It is specific to each site and acts as the individual password for every site visited.

(d) KEY: Encryption/Decryption key

It acts as the session key, since it is hashed with the timestamp.

Generation of UID, GEN and KEY

(i) Hash the GUIDwith the domain name.

HMAC (GUID, domain) ->UID;

(ii) Hash the master password and salt with the domain:

HMAC (Password+Salt, domain) ->GEN;

(iii) Hash the GEN with a nonce(random number) and a timestamp:

HMAC (GEN, nonce+timestamp) ->KEY;

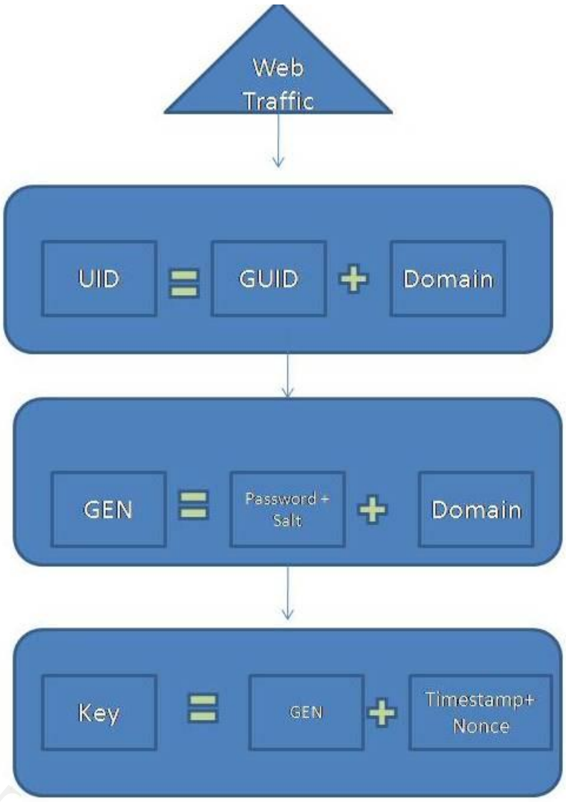


Fig 1.2 Set of entities and their generation process

Experimental Procedure

Encrypting the traffic

On the client side, the traffic is simply encrypted by first deriving the KEY, and then using it in the encryption algorithm.

Decrypting the traffic

Step 1: Client sends the following to the server:

(i) UID
(ii) Timestamp
(iii) The generated nonce
(iv) Encrypted payload

Step 2: The server goes through the following steps

(i) Looks up the UID (to find the password)
(ii) Generate the GEN
(iii) Subsequently generate the KEY to decrypt the traffic.

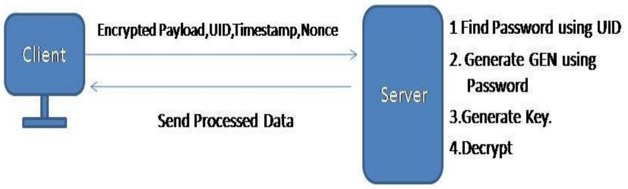

Client
Encrypted Payload,UID,Timestamp,Nonce
Send Processed Data
Server
1 Find Password using UID
2. Generate GEN using Password
3.Generate Key.
4.Decrypt

Fig 1.3 Proposed model decryption process

V. VULNERABILITIES

A vulnerable point is during the establishment of your site specific generator and UID. The only time CA authentication should be used is during user connection initialization. All other requests don't need to go through the PKI. The time window of the initial user connection using PKI would be too small to be practical to attack.

Once the connection has been established and traffic transferred, the SSL strip could have been exploited with a MITM attack, state actors or compromised roots (like DigiNotar).

VI. CONCLUSION

In this paper, we have described an alternative encryption system which can overcome the shortcomings of CA based Public Key Infrastructure. It increases the dynamics of the security algorithm and strengthens the cryptographic keys by making use of a unique string, i.e, the domain name. Thus it is equivalent to having a different cryptographic key for each domain being visited.

The system proposed requires minimal amount of additional resources to support it since it makes use of the existing encryption protocols and machinery. It will help significantly reduce the reliance for security on the CA and strengthen it by making it less vulnerable to attacks.

REFERENCES

[1] J. Nightingale, "Revoking Trust in DigiCertSdn. Bhd Intermediate Certificate Authority," blog, 23 Nov. 2011; https://blog.mozilla.org/security/2011/11/03/ revoking-trust-in-digicert-sdn-bhd-intermediate-certificate-authority/

[2] Wes Kussmaul (2007) "Own Your Privacy" – PKI Press.

[3] Markus Jakobsson, Steven Myers"Phishing and countermeasures understanding the increasing problem of electronic identity theft"

[4] DaviOttenheimer, Matthew Wallace "Securing the Virtual Environment: How to Defend the Enterprise Against Attack"

[5] Carolyn V. King "Online Privacy and Security of Internet Digital Certificates: A Study of the Awareness, Perceptions, and Understanding of Internet Users"

[6] Microsoft Security Advisory (2916652)"Improperly Issued Digital Certificates Could Allow Spoofing"

[7] Public key certificate "Weaknesses" http://en.wikipedia.org/wiki/Public_key_certificate

[8] Steven Roosa, Stephen Schultze, "Trust Darknet: Control and Compromise in the Internet's Certificate Authority Model"

[9] Christopher Soghoian,Sid Stamm"Certified Lies: Detecting and Defeating Government Interception Attacks Against SSL"