National Public Key Infrastructure: Friend or Foe?

B. Noviansyah, An Independent Security Researcher
@tintinnya

***Abstract*—The needs of secure communication in web-based era is an inevitable aspect in order to support trustworthiness between counterparts. Maintaining trustworthiness could be achieved by providing mathematical aspects to support nonrepudiation, integrity, and confidentiality. Cryptography is one of many ways to make communication channel more secure, by preserving confidentiality and integrity aspect in C-I-A triangle. Public Key Infrastructure ("PKI") provides the ease of cryptography key management, so that one party does not have to trust all of other entities but simply trust the Certification Authority ("CA"). By delegating trust to CA, all of other public keys that are issued by this CA are also trusted by correspondent party. If Government creates their own CA aside from commercial CAs, it could benefit a country in terms of national sovereignty. But it could also be rising up the red flag from citizen towards their privacy. This paper discusses the positioning of National PKI as friend or foe.**

***Index Terms*—Cryptography, Digital signatures, Information security, Public key, National security**

I. Introduction

CRYPTOGRAPHY is widely used to ensure that only parties with the key in possession could read the message. In IT communication, the challenges in creating secure channel between two parties are: (i) to determine the desired cryptography algorithm; (ii) the key with predetermined length, and also (iii) mechanism on how the key is distributed. This paper discusses limited to only two types of cryptography algorithm, i.e. symmetric-key and asymmetric-key along with their advantages and disadvantages.

The main focus will be emphasized on the utilization of asymmetric-key encryption and how it could be used to protect national interests. Protection of this PKI also discussed in technical point of view, but not necessarily technical in-depth. Leaving out the big question whether the implementation will be friendly to citizen, or it is a threat for citizen in daily usage of Internet.

This paper is submitted to be presented in Indonesia IT Security Conference 2016 in Malang, Indonesia at 24—25 September 2016. Using template from IEEE http://www.ieee.org/documents/trans_jour.docx

B. Noviansyah is with Asosiasi Forensik Digital Indonesia (AFDI) in Jakarta Pusat, Indonesia. As independent IT Security Researcher and Java Programmer, his interests and researches include digital forensics, malware analysis, secure code analysis, and IT system vulnerability study. His current research area is in IT for Payment Systems. He shares his works as a guest lecturer in several academic institutions and government agencies in Indonesia. His past works in Public Key Infrastructure include designing, implementing, maintaining, and also providing technical recommendation in several institutions he worked for.

II. Symmetric-key Cryptography

Symmetric-key Cryptography Algorithm is the first type of cryptography that will be discussed. In this type of cryptography, the key that being used to encrypt the message ("plain text") should be similar with the key that is used to decrypt the secret message ("cipher text"). Otherwise, the cipher text will not be able to be decrypted to plain text again.

Fig. 1. Encryption process using key that has been agreed between A and B, denoted with q_{AB}.

As depicted in Fig. 1 above, before A and B exchanges the secret message, both A and B agree to use q_{AB} as the key when A encrypts the plain text, and B decrypts the cipher text. On the other direction, B could have different key with q_{AB} when B wants to send encrypted message with A, namely q_{BA}. To make this explanation simple, assume that q_{AB} is similar with q_{BA}.

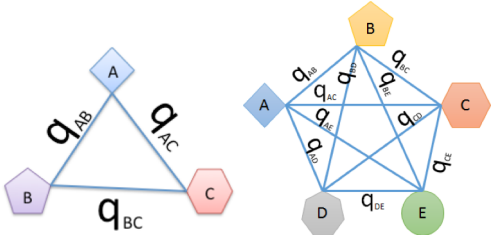


Fig. 2. Asume that q_{AB} is similar with q_{BA}. When 3 parties exchanging secret messages, there are 3 different keys distributed among them. But, when number of parties increased to 5, the number of keys distributed in system increased to 10 different keys.

A. Key Distribution

The number of keys needed to support secure messaging between parties are increasing exponentially with the increasing number of party. Fig. 2 above illustrates the number of keys distributed in system are increasing in geometry progression.

In the system where there are only 3 parties to exchange secret message (denoted in $n = 3$, where n is number of parties), still assuming $q_{AB} = q_{BA}$, each parties will keep 2 keys (that is, $n - 1$ keys). For instance, party A will keep q_{AB} to able to exchange the secret message with B, and also A will keep q_{AC} to exchange secret message with C. Party B will keep q_{AB} and q_{BC}. Party C will keep q_{AC} and q_{BC}. Hence, the key space for this system is only 3 unique keys, that is $\{q_{AB}, q_{AC}, q_{BC}\}$.

This key space will be growing in geometry progression when new parties introduced to system. In Fig 2 above, when 2 new parties are introduced to system, all existing parties namely A, B, and C will also add 2 new keys from D and E. Then each party keep 4 keys, which makes number of keys exchanged in this system increased from 3 to 10, that is $\{q_{AB}, q_{AC}, q_{AD}, q_{AE}, q_{BC}, q_{BD}, q_{BE}, q_{CD}, q_{CE}, q_{DE}\}$. Following the pattern in geometry progression, the formula to calculate number of key in key spaces is:

$$K = \frac{n \times (n-1)}{2} \quad (1)$$

where : K = number of key in key spaces
n = number of party in mutual system, n > 0

B. Challenges

When a new party is introduced to the system, all other existing parties should add new party's key. Otherwise, the system would not be able to communicate securely with new party. This operation is not effective since key registration process to each party should be done directly to related party, without centralized key distribution that play a role as the center of trust. It means, to trust party D, each party should directly perform key exchange mechanism.

To ensure communication between party A and party B in a secure way, party C should not have q_{AB} that is used only by A and B. Otherwise, C could decrypt the message; A and B fail to have a secure communication. In this case, A and B should revoke q_{AB} in their system and create new key that is denoted as $q_{A'B'}$.

Revocation is another cumbersome process, there is no centralized way to remove key q_{AB} both in A's system and B's system. If A removed q_{AB} but B still keeps q_{AB}, C would still be able to decrypt the message; even A could not be able to decrypt the message. In this case, C might reply the message using the same key q_{AB} and encrypt the message back to B. B might be deceived that the message was coming from A. This unfortunate event shows that failure to revoke the key on the both sides, the non-repudiation aspect is broken. Even though the confidentiality aspect is still taking place.

III. Asymmetric-key Cryptography

In asymmetric key cryptography, before joining the system, each entity should have their own key pair first; that is private key and public key. These keys are mathematically correlated so they can be proved by certain mathematical calculation that two keys are pair.

A. Encrypt and Decrypt

Let A is the entity in asymmetric key system, q_A is A's private key, and p_A is A's public key. All other entities in system may have and keep p_A because public key is for public. While A should put all of efforts to protect that only A could access and use q_A.

When other party, say B, wants to send encrypted message to A, B should use p_A to encrypt the message. Now, even after B encrypt the message with p_A B could not decrypt the message. It could be done by only applying cryptography decrypt function using q_A. These characteristics give advantages in key management. Regardless of whom the counterparty is, A should only focus on protection of q_A.

In this scenario, when B could encrypt the message with p_A and send the message to A, A would not be able to reply the message to B in encrypted form. Because, in order to reply the message from B, A should have p_B first, then A could send encrypted message to B using B's public key. These processes are depicted in Fig. 3 below.

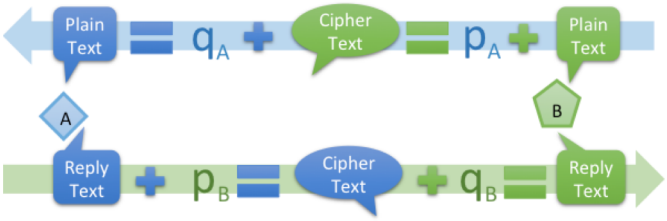


Fig. 3. Encryption process message from B using A's public key p_A and A decrypts the cipher text using A's private key q_A. In order to reply the message in encrypted form, A uses B's public key p_B and B decrypts the cipher text using that B's private key q_B. These processes are simplified, so that the main process is easier to understand.

B. Message Signing

Scenario in Fig. 3, public key p_A is available for all parties. Hence, there is no guarantee whether B is the real sender of the message to A. Using property of private key and public key, a certain cryptography method using private key could be used as a message signing mechanism. Since private key only owned and kept by B, then it could be inferred that B is the sender of the message. This is the property of nonrepudiation in messaging.

To apply this property, B creates message as plain text first. Instead of directly encrypt the plain text using p_A, B creates an additional message from plain text using a mathematical calculation in cryptography with q_B and yields a secured message digest ("hash"). This hash serves as handwritten signature in real life. Along with this hash, plain text then encrypted using p_A and send to A. The last encryption process serves as the digital envelope that wraps the plain text and the hash part.

To read the message, A decrypts the digital envelope using q_A. Thus, party A will have two parts of message: (i) the plain

text as original message, and (ii) hash part as digital signature of B. Unlike real-life signature, this hash is able to detect whether the message in plain text has been tampered or not. This is the property of integrity assurance in messaging.

The integrity of the message itself can be verified by comparing the digital signature part in the message with p_B. With mathematical operation, A calculates digital signature from original message. Also, A decrypts the digital signature part in the message using p_B. The result will be compared with digital signature part in the message. If it is matched, then the message is not tampered. A also confident that the signature is calculated using p_B, that could be inferred the message was sent by the person who has q_B in possession. These processes are depicted in Fig. 4 below.

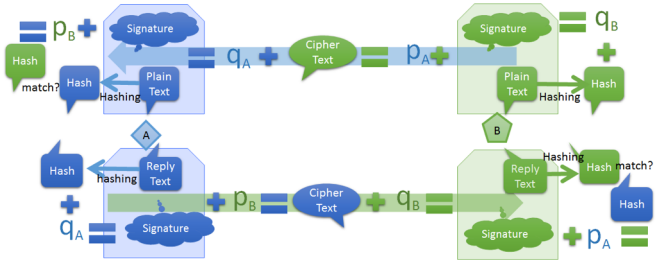


Fig. 4. Asymmetric-key Cryptography satisfies the CIA triangle information system model, especially in Confidentiality (by performing encryption), Integrity (by performing digital signature, which also serves as a non-repudiation control). The illustration above is a simplified mechanism, not necessarily a good model of message exchange. In a secure implementation, exchanging message using asymmetric key cryptography will be added with another encryption using one-time-symmetrical-key algorithm that serves as session key. This will prevent compromising the whole conversation if one session is compromised.

C. Key Distribution

In Asymmetric-key Cryptography, numbers of keys exchanged in system are equals to number of party. Regardless number of party, if everyone in the system want to send encrypted message to A, all party will receive same public key p_A. While A's private key will not be distributed into system. As in Fig. 5 below.

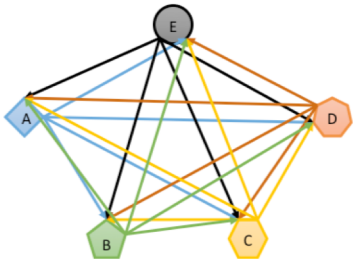


Fig. 5. Asymmetric-key Cryptography will only distribute public key. Regardless the number of party in the system, number of keys is increased linearly and equivalent with number of the parties themselves. This reduces the complexity of number of distributed keys in the system

IV. Public-Key Distribution Mechanism

Distribution Key model as Fig. 5 above is not necessarily guarantee that public key in system is valid. In simple system, this is still acceptable. Entity A should ask first with person that represented by public key A, what is its keyID so that it could be matched with public key when other party received it. There are many variances of key distribution mechanism. This paper only discusses two mechanisms: (i) PGP and (ii) PKI.

A. Pretty Good Privacy (PGP)

One of popular public key distribution mechanism is using PGP Key Server. Refer to [2], Philip Zimmermann invented this mechanism in 1991. With this key server mechanism, all public keys that generated by each entity can be published in centralized key server. Prior this mechanism exists; each entity published their public key on their own server. It became hard to be verified by counterparty. In order to found the correct public key, counterparty should also know to which server did this key exists.

A verification process of public keys is matter. Every party in the system could create their own identity. For example, entity G could generate their own key pair but contain information and identity as if entity G is entity A. If G could manage the server where A put its public key, G could replace A's public key with its public key. Hence, when other entity found A's public key, it is actually the public key of G and only G has private key of purportedly A's public key. In this case, G could impersonate A in digital world.

Moreover, revocation public key in PGP should be done with tricks from [3], it was suggested with helps of trusted friends by signing a new key to announce that previous key with specific keyID was already invalid.

B. Public Key Infrastructure (PKI)

Public Key Infrastructure (PKI) is another mechanism to distribute public key. PKI is widely used by communication using SSL, such as HTTPS, FTPS, VPN over SSL.

Similar with PGP, Entity A should generate the key pair first that is q_A and p_A. The private key q_A remains in A's protection effort. While public key, in order to be issued by CA (Certification Authority), A should wrapped the public key in certain format called PKCS#10 CSR (Public Key Cryptography Standards #10 Certificate Signing Request). This CSR file format is stored in PEM (Privacy Enhanced Mail), that is a Base64-encoded string from DER (Distinguished Encoding Rules) binary file format. Binary DER file format complies with x509 v3 standard that contain of information stored in ASN1 (Abstract Syntax Notation 1) structure. Types of information stored in this CSR file includes: (i) subject's identity such as country name, city, etc., (ii) public key p_A of subject that is generated in key pair process, (iii) additional info such as Certificate Key Usage for SSL.

The CSR file should be submitted to be signed by trusted CA. When there is no technical error, CA will extract several information and creates a Digital Certificate in x509 v3 standard as well with information in CSR along with other information. Inside this Digital Certificate, there is information about chain of trust tells which CA issued the

certificate. If this CA is a SubCA, then there is also information that RootCA issued the SubCA. Prior to use the digital certificate from CA, Entity A should install the CA Certificate first as an agreement to trust the CA. Fig. 6 below tells this process in simplified way.

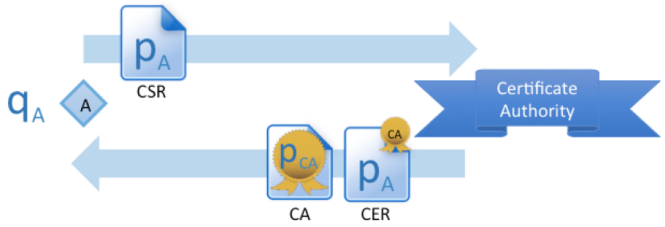


Fig. 6. Entity A creates CSR that contain A's public key. In short, CA will sign the request and create digital certificate based on information inside the CSR. Entity A should trust CA prior to use Digital Certificate by installing the CA Certificate.

When A trusts a CA, A also trusts all of certificates that are issued to B, C, D, E by the same CA. Entity A and will download the certificates from CA and extract the public key inside the certificate to encrypt the message to respective entity B, C, D and E. This is the advantage in PKI in terms of key distribution management, as simple as shown in Fig. 7 below.

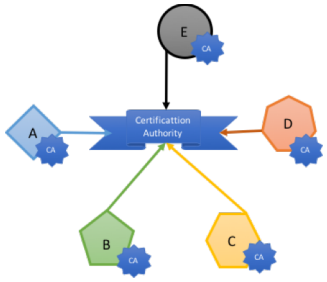


Fig. 7. By trusting CA, all of certificate issued and signed by this CA will automatically trusted by all entities in the system. Hence, each entity only build a single trust to CA only, as opposed to trust all other public key of entities in PGP system.

When, say entity C lost C's private key, entity C should tell CA immediately so CA can revoke Certificate that contain C's public key. So, when entity A wants to send encrypted message to C using revoked certificate, A will check to CA's CRL (Certificate Revocation List), or using OCSP (Online Certificate Status Protocol). If the certificate already revoked, then A will denied the connection to C using C's public key in C's certificate.

V. Plan for National PKI

According to [1], Indonesian Government via Kemkominfo (Ministry of Communication and Informatics) and Republic of Korea via KOICA (Korea International Cooperation Agency) agreed to launch project for National Certification Authority for e-Government in Indonesia in February 2015. There is no other details document to support this project and what will this National CA for. Based on [1] materials, Indonesia Government will create National Root CA only, and all other CA will be handled by each ministry and even third party Issuing CA. Fig. 8 below shows the governance of CAs.

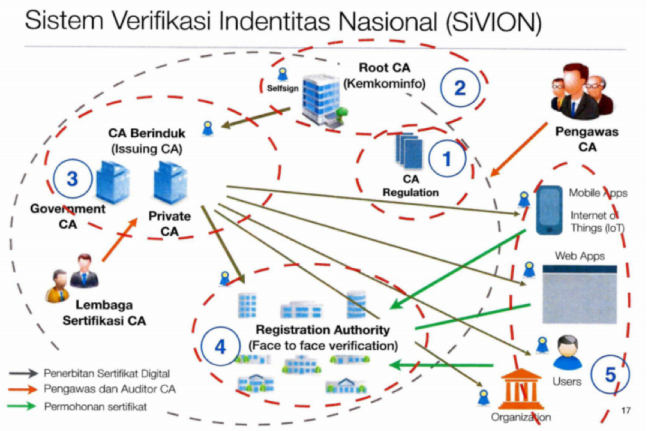


Fig. 8. National ID Verification System (SiViON) shows that Ministry of Communication and Informatics will create National Root CA [4]

This project is available to support ITE Law on article 11 about Digital Signature. By implementing National Root CA, Indonesian Government will create a certain mechanism to support digital signature to be applied on electronic transaction or electronic document using Citizen's private key. No further information what will the private key be.

It is planned that the usage of PKI in Nation-wide include [4]:

1. SSL certificate for official website,
2. Secure Email, Internet Banking,
3. e-Taxation, e-Custom
4. e-Commerce, Cyber Trading, e-Banking
5. IoT, FIDO

VI. Friend or Foe?

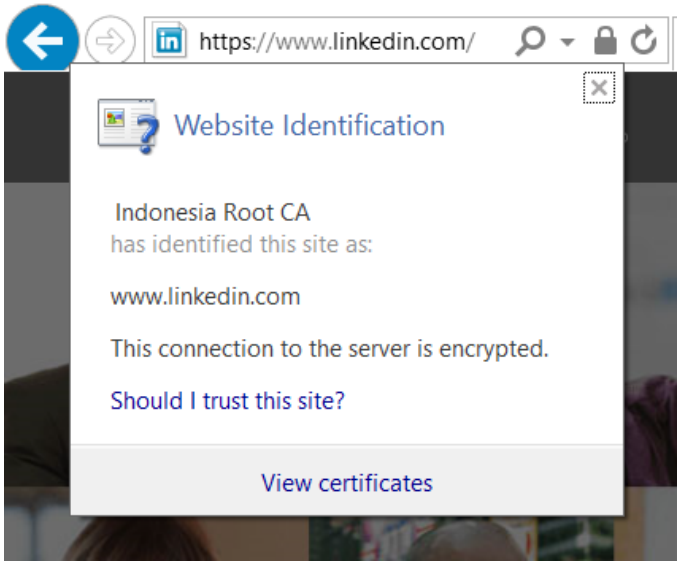
When Indonesia has their own RootCA, there is another homework for government to embedded the RootCA certificate as Trusted CA in several Operating Systems such as Microsoft Windows with Microsoft Trusted Root Certificate [6], Apple macOS/iOS with Apple Root Certificate Program [7], Google Chromium Projects with Root Certificate Policy [8], Mozilla Firefox Certificate Store with Mozilla CA Certificate Policy [9]. Otherwise, there will be error messages in Client-side saying that the connection is not secure since the certificate is not trusted.

A. SSL Interceptor

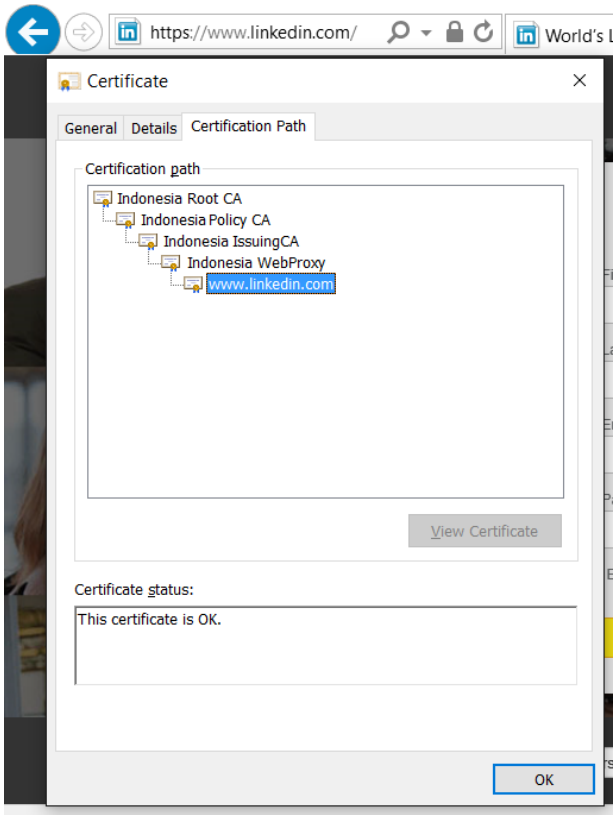
After National Root CA certificate has been installed into Citizens' computer, Government could do pretty much whatever they want to. Along with ID-SIRTII, Government could increase national network visibility by inspecting encrypted traffic in SSL tunnel. This task is getting easier if Government enforce the implementation of transparent web proxy on every NAP (Network Access Point). The idea to implementing SSL inspection via transparent proxy nation-wide is still theoretical. But using this transparent proxy, Government could enforce the client to use National Root CA by applying HSTS (HTTP Strict Transport Security) [10].

Based on author's experiences, dissecting SSL traffic and saw what was lurking on SSL traffic is possible. Author set up

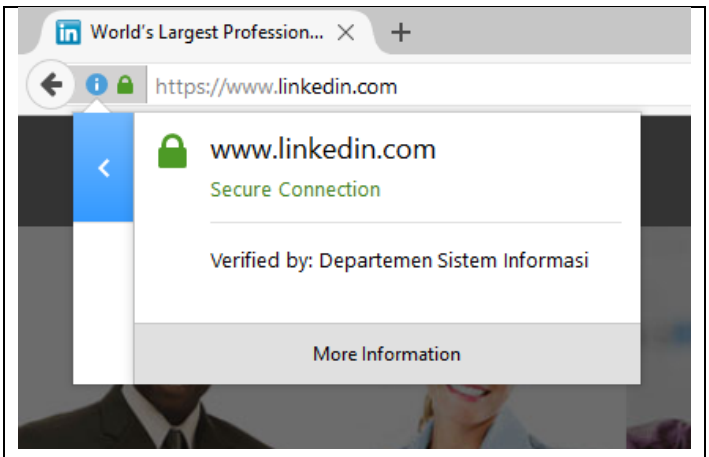
a Root CA, with two Sub CA, and implemented transparent proxy. Root CA was successfully installed in web browser and OS level. When client accessed a website that is secured by HTTP over SSL, user did not see anything suspicious. Several screenshots are embedded as proof of concept what author did on previous works.



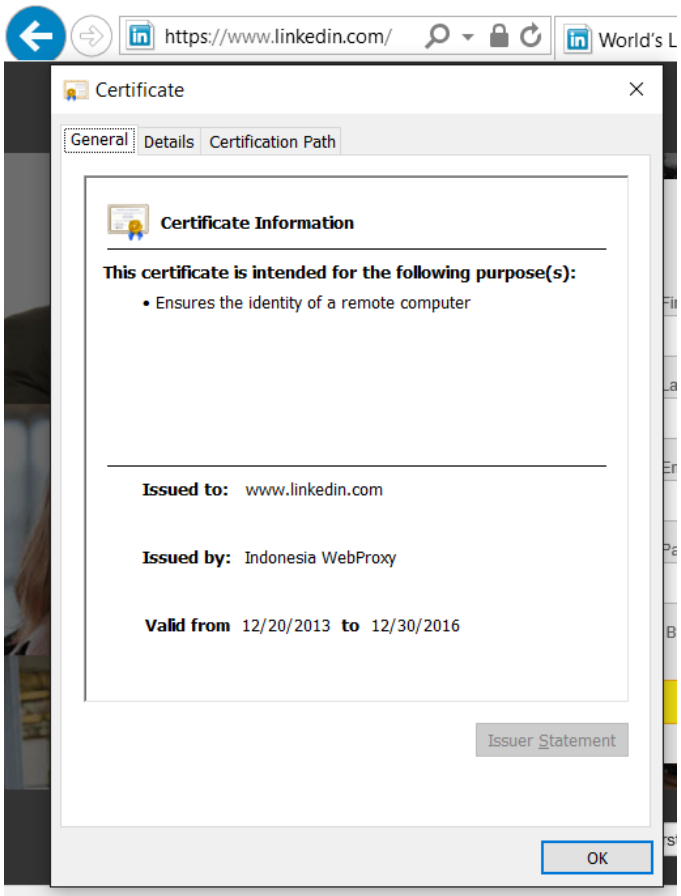
In original SSL certificate, site www.linkedin.com has DigiCert Global Root CA as Root CA, not Indonesia Root CA



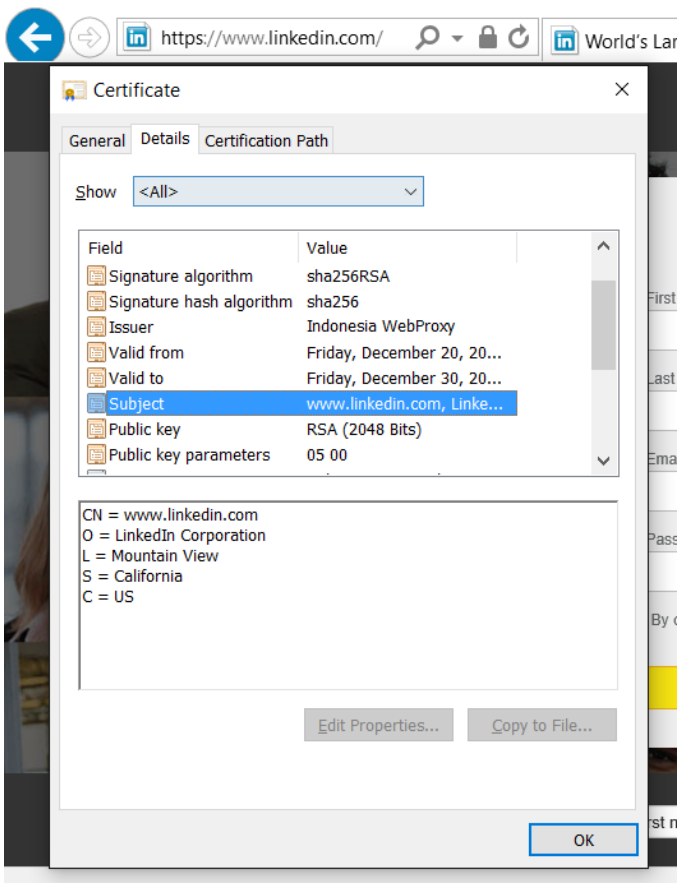
Notice that www.linkedin.com was issued by DigiCert SHA2 Secure Server CA, not by Indonesia WebProxy



Compared to the original, site www.linkedin.com is verified by DigiCert Inc., not Departemen Sistem Informasi



Site www.linkedin.com has SSL certificate that is issued by DigiCert SHA2 Secure Server CA



This transparent proxy copies Subject information from original SSL Certificate and create a new SSL Certificate, but issued by Indonesia WebProxy not by DigiCert

With network visibility as transparent as this, Government gain advantage in controlling the National Internet. Unfortunately, intercepting SSL as above would only success if the client-side is not implementing Certificate Pinning [11]. Most of web browsers are implemented without Certificate Pinning feature, but mobile apps are usually having their server's certificate pinned into the application. When clients initiate secure connection, server performs calls back to perform certificate inspections. When client and server perform certificate inspection, then this transparent proxy to inspect the encrypted channel becomes fail to achieve the goal.

In terms of flow monitoring, no node could see the traffic flow if the communication is using secure layer such SSL. To evaluating network flow, this kind of communication should be decrypted first so that each packet transmitted will be able to be index at least these information: (i) source IP, (ii) source port, (iii) destination IP, (iv) destination port, and (v) protocol. Using these information, Government would be able to detect the anomaly traffic inside encrypted traffic. One of the advantage includes ability to detect tunneling access to illegal content that is prohibited by Indonesian Law.

B. Multi-purpose Digital Certificate

PANDI (Pengelola Nama Domain Internet Indonesia) already began the campaign "Proud using .ID domain" since August 17th, 2014 [12]. The idea is to protect visitor from visiting fraudulent website when using domain .ID, since the registration processes are very rigid to avoid anonymity. In addition to it, Government also could enforce that all digital certificates should be issued by National Root CA. These digital certificates include:

1) SSL Certificate

This type of certificate is for protecting and give additional secure layer for HTTPS, FTPS. Another extra layer of validation to avoid fraudulent website claiming false identity, National Root CA for SSL could also add DV feature (Domain Validation).

2) Application Signing

This certificate is intended to protect applications being tampered in application distribution process. These applications could be application for National ID (KTP) registration process in Municipal Office (Kantor Walikota) or District Office (Kantor Kelurahan).

3) Driver/Library Signing

This certificate is intended to protect applications being tampered in application while in operational routines. Case of Bangladesh Heist, SWIFT Alliance Access has been tampered in library/driver level [17].

VII. Proper Security Segregates Friend and Foe

With that great power in controlling Internet, National Root CA would be very lucrative by adversaries if it could be taken down or hijack. When author using his hat as a regulatory body, inspecting encrypted traffic is inevitable since the trends in cybercrime already moving their arsenal into encrypted channel. Malware and APT that threatening national critical information infrastructure such Stuxnet driver was signed by a valid certificate from Realtek Semiconductor Corps [13]. Hence, this malware was considered as a trusted application and safe to be executed by system.

But, it also becomes a threat to privacy for Citizen. When author using his hat as a citizen in this country, author definitely does not want his communication being tapped, or inspected because author and any other Internet user deserve my privacy in Internet. As long as there is a guarantee that insider in Government does not misuse the inspected communication, good people should not be worried about this interception. Hence, proper security measurement and better risk management in operating this National Root CA becomes thin red line that segregates Government as Friend of Good Citizen or Foe for Good Citizen.

Managing National Root CA should be subjected to be audited by third party with certain qualification. Government should also gain Good Citizen trust by providing Certificate Policy (CP) and/or Certificate Practice Statement. These are some kind of contract between Government as CA management and Citizen as Internet user. Moreover, the party that operates this Root CA should also put technical security aspect in place.

A. Planning for CA Hierarchy

When designing Root CA, Government or Kemkominfo should also designed on how many level the hierarchy of CA down to personal certificate. It is not a best practice to have Root CA directly issues a personal certificate to user, since compromising the issuing CA that is equals to Root CA will create big effort to revoke Root CA in Web Trust and registering the new Root CA to Web Trust back [5].

It is recommended by Microsoft Technet Team to create a 3-tier hierarchy of CA. Government or Kemkominfo operates Root CA and the task is only to issues Policy CA. In this case, Policy CA could be delegated to another Ministry or another Government Body (Kementerian/Lembaga or K/L) that responsible in each sector, for example Ministry of Transportation as Government body representative that responsible in transportation sector. As Policy CA, Ministry of Transportation could appoint third party as certificate issuer. Then Airlines could request certificate to certificate issuer to have SSL certificate in their website. Customer of this airlines also request certificate to same issuer, so that any transaction performed in airlines' website could be digitally signed to maintain aspect of nonrepudiation. So, both airlines and customer are gaining trust in cyber space.

B. Key Properties

For all hierarchies, to avoid breaking private key easily, keys should have higher bit length as possible. In author's experiences when designing Root CA with RSA 4096-bit key length, the key is more than enough to handle brute force attempt. By math, guessing a 2048-bit that is half lower than 4096-bit key would take 13.75 billion years to complete [14] and of course cracking 4096 bit takes longer than 2048 bit. But unfortunately, legacy operating system that is installed base of many Point-of-Sales (POS) device still using Windows XP POS Edition. Experiences said that Windows XP was unable to import, process or accept CA with keys greater than 2048-bit length. Moreover, embedded system or System-on-Chip (SoC) such as Yubico's YubiKey USB also unable to accept more than 2048 bit [15].

Not only key length is matter, randomization when generating key also matter. Weak 1024-bit key was easier to break using GPU, compared to strong 1024-bit key. Several Capture The Flag problems also introduce the way to crack weak 1024-bit key.

C. Protecting Private Key of National Root CA

Choosing key length with random is not enough. Private key should be stored and kept in a very secure way. These are security measurement on how to protect it.

1) Password Protected

Do not create private key without applying password prior to use the key. Otherwise, anyone who can have access to private key would not be challenged of the possession. There are various file format that support password protection of private key such as Microsoft PFX file format (PKCS#12 P12 or PFX) and Java Keystore (JKS).

2) Registry Hive or File System

After protecting private key with password, now the file should be kept on secure place that need another security layer to access it. In Microsoft Windows, your private key was stored in such way that only administrator with access to registry hives could use the private key. But either P12/PFX or JKS could be kept as simple as a file format in Operating System.

The disadvantage of storing private key in file system is this file is prone to be attacked with brute force attempt. No certain mechanism to protect unauthorized access to file in file system. Once adversaries gain access to file system, this file could be copied and put on a special rig that perform brute force attack.

3) HSM: USB, PCIe, Network

Instead of storing and kept the private key on file system, there is more secure way to do it. HSM (Hardware Security Module) is simply a device like storage of certificate and public key. Prior to access the file that contain private key, user will be challenged by several security measurements. USB HSM will require certain software application and driver before using it. These software and driver blocks the brute force attempt, for example disabling the driver or hardware when 10 failure attempt.

Network-based HSM adds another security layer. Before communication took place, this kind of HSM requires NTL (Network Trust Links). HSM will only accept network communication from encrypted link that has been established before by performing symmetrical-key exchange. Then, after gain access in network layer, HSM will challenge the user by asking PIN or password in order to gain access in HSM's Operating System. This will ensure that only server and correct application could access HSM's OS and HSM's network layer. Then, as an extra layer, HSM also challenge another PIN or password prior accessing HSM's partition. Hence, by layering access from network, OS, and partition, only specific module can access the specific keys. By this layered protection, Module X in Application A that is installed on server with IP 1234 is guaranteed that could not access private key that is stored in same HSM but different partition, intended to be access only by Module Y in Application A on the same server IP 1234.

VIII. END NOTES

PKI uses asymmetric-key cryptography to ensure the symmetric-key exchange proses with specific party. It is not common and inefficient to encrypt message using asymmetric-key since the processing time to decrypt and encrypt it are longer than using symmetric-key. The purpose of Asymmetric-key algorithm is to make interception process of encryption key harder.

With the possibilities of PKI usage, Government could have a better Internet visibility, even in encrypted tunnel. But in order to gain trust from Good Citizen, Government should also put proper security in daily operation of Root CA. Also, announce the result of periodic audit toward the operation.

REFERENCES

[1] Kang Hyeug, "National Certification Authority (CA) and Government Security Emergency Response (SER) System for e-Government in Indonesia," presented at the 4th Public Key Infrastructure Awareness and EJBCA/TOOLKIT Seminar, Jakarta, Indonesia, Feb. 29, 2016.

[2] Philip Zimmermann. (2010, June). Where to get PGP. Phil Zimmermann & Associates LLC. Santa Cruz, California. [Online]. Available: https://philzimmermann.com/EN/findpgp/

[3] Wouter Slegers. (2002). Chapter 7 Revoking a key. Your Creative Solutions. Roermond, Netherlands. [Online]. Available: http://www.pgp.net/pgpnet/pgp-faq/pgp-faq-key-revocation.html

[4] Riki Arif Gunawan, "Digital Signature Roadmap," presented at the 4th Public Key Infrastructure Awareness and EJBCA/TOOLKIT Seminar, Jakarta, Indonesia, Feb. 29, 2016.

[5] Microsoft TechNet. (2012). Securing PKI: Planning a CA Hierarchy. Microsoft. Redmond, Seattle. [Online]. Available: https://technet.microsoft.com/en-us/library/dn786436(v=ws.11).aspx

[6] Microsoft TechNet. (2012). Microsoft Trusted Root Certificate: Program Requirements. Microsoft. Redmond, Seattle. [Online]. Available: https://technet.microsoft.com/en-us/library/cc751157.aspx

[7] Apple. (2016). Apple Root Certificate Program. Apple Inc. Cuppertino, California. [Online]. Available: https://www.apple.com/certificateauthority/ca_program.html

[8] Chromium Projects. (2016). The Chromium Projects Root Certificate Policy. Google. Mountain View, California. [Online]. Available: https://www.chromium.org/Home/chromium-security/root-ca-policy

[9] Mozilla Contributors. (2016). Mozilla CA Certificate Policy. Mozilla Foundation. San Francisco, California. [Online]. Available: https://www.mozilla.org/en-US/about/governance/policies/security-group/certs/policy/

[10] J. Hodges, C. Jackson, A. Barth. (2012 November). HTTP Strict Transport Security (HSTS) IETF RFC6797. PayPal, Carnegie Mellon University, Google, Inc. California. [Online] Available: https://tools.ietf.org/html/rfc6797

[11] Jeffrey Walton. (2013 February 7th,). Securing Wireless Channels in Mobile Space. Presented at OWASP Virginia's Chapter. [Online]. Available: https://www.owasp.org/images/8/8f/Securing-Wireless-Channels-in-the-Mobile-Space.ppt

[12] Kompas Tekno Team. (2014 August). Ini PR Pandi soal apapun.id. Jakarta, Indonesia. [Online]. Available: http://tekno.kompas.com/read/2014/08/17/20150027/Ini.PR.Pandi.soal.Apapun.id.

[13] Nicolas Falliere, Liam O Murchu, and Eric Chien. (2011, February). Symantec Security Response: W32.Stuxnet Dossier. Symantec. Cuppertino, CA. [Online] Available: http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf

[14] DigiCert. (2009 December). The Math Behind Estimations to Break a 2048-bit Certificate. Lehi, Utah. [Online]. Available: https://www.digicert.com/TimeTravel/math.htm

[15] Alessio Di Mauro. (2015 February). The Big Debate, 2048 Vs. 4096, Yubico's Position. Palo Alto, California. [Online]. Available: https://www.yubico.com/2015/02/big-debate-2048-4096-yubicos-stand/

[16] Kerry Scharfglass, Darrin Weng, Joseph White, Christopher Lupo. Breaking weak 1024-bit RSA keys with CUDA. California Polytechnic State University. San Luis Obispo, CA. [Online] Available: http://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1241&context=csse_fac

[17] Sergei Shevchenko. (2016 Apri 25). Bae Systems Threat Research Blog: Two Bytes To $951M. BAE Systems. Sydney, Australia. [Online]. Available: http://baesystemsai.blogspot.co.id/2016/04/two-bytes-to-951m.html

B. Noviansyah was born in Palembang, Indonesia in 1980. He received Bachelor degree ("Sarjana Teknik" or S.T.) in Informatics from Institut Teknologi Bandung in 2002. And he also received Master of Science in Information Security Policy Management (MSISPM) with Cyber Forensics Incident Response Track (CyFIR) from Carnegie Mellon University Heinz College, Pittsburgh, Pennsylvania US in 2014 with Master Thesis title "Increasing Security Awareness in Enterprise Using Automated Feature Extraction with Long n-gram Analysis for File Type Identification".

From 2002 to 2012, he was a 3-Tiered-Java EE programmer, also application security and quality assurance. His works include redesigning Public Key Infrastructure joins with Principals in PKI Industry. When pursuing his Master degree, he was Teaching Assistant for Linux and Open Source class, Network Security Analysis. Currently, he is joining Asosiasi Forensik Digital Indonesia (AFDI) membership in Research and Development Division.

Mr. Noviansyah's awards include Top Ten Graduate Student in Department of Defense Cyber Crime Center (DC3) Digital Forensics Challenge 2013, and one of the best quantitative Master Thesis in Heinz College 2014.