Advances in Computational Sciences and Technology
ISSN 0973-6107 Volume 10, Number 5 (2017) pp. 671-686

https://dx.doi.org/10.37622/ACST/10.5.2017.671-686

Incorporating Trust in Public Key Infrastructure Certificates

Rahoof P P[1]*, Latha R Nair[2] and Thafasal Ijyas V P[3]

[1]*Division of Computer Science and Engineering, School of Engineering, Cochin University of Science and Technology, Kerala, India.*

[2]*Division of Computer Science and Engineering, School of Engineering, Cochin University of Science and Technology, Kerala, India.*

[3]*Dept. of Electrical Engineering, King Khalid University, Abaha, KSA.*

Abstract

The concept of trust is central to the formulation of an authentication mechanism for secure client-server communication. Recently, different trust models have been proposed based on varying definitions of trust. In this paper, the meaning of trust is explored in the context of public key infrastructures for certificate authentication. Also the articulation of trust in various prevalent trust models is analyzed. It can be seen that trust models have arisen in the context of different secure communication needs. We attempt to relate the implication of trust in such trust models with the sense of trust in public key infrastructures.

1. INTRODUCTION

One important issue in client-server communication is the privacy of the client and the server. Privacy is related to the role of third parties in the connection process. Third parties can encroach upon privacy in many ways. The client has to contact a third party domain name server (DNS) for domain name resolution [1]. This is essential in any communication. Security can be compromised due to the attacks on

the DNS. DNS attacks may be due to the flaws in the protocol implementation or may be a direct attack on the servers. DNS spoofing/DNS cache poisoning, ID hacking etc. belong to the first category. Bugs in the software implementation of the DNS server (like BIND, DNS Blast etc) may also be exploited to mount an attack on the server. Also, another prominent attack is Denial-of-Service (DOS) attack [2]. The Domain Name System Security Extensions (DNSSEC) is a suite of specifications capable of addressing many of the protocol issues [3]. Also, bugs are constantly monitored and software patches periodically released to address the latest issues arising in the security of client-server communication.

In a client-server communication, the client seeks a service from a server. The server is identified by a domain name [4]. Server authentication is the process of asserting that the server is the legitimate owner of the domain name. The Transport Layer Security (TLS) protocol is used for this purpose. The server presents a Public Key Infrastructure for X.509 (PKIX) certificate to assert its ownership over the domain name [5]. The PKIX certificates are issued by third party certificate authorities (CA). This digital certificate is an electronic document that proves the ownership of a public key by the server and contains the information about the public key. The certificate authorities that issue the PKIX certificates in turn require certificates for their authenticity. Thus a chain of trust of intermediate CAs are established till it ends in a root CA with a self-signed certificate. This chain is called a certification path. The certification path has to be validated before it can be used to establish the credentials of the server with which the client seeks secure communication. The process is shown in figure 1. This validation involves various checks on the certificates contained in the certification path. Typically, the signatures have to be verified and the revocation status of each certificate also has to be examined. Certification path validation algorithms are an important component of the PKIX system.

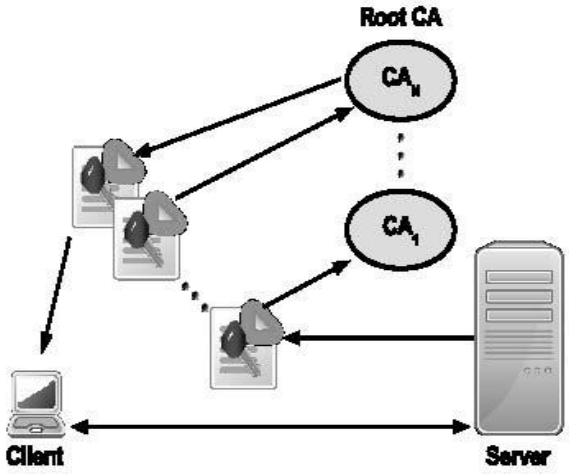

Fig. 1. Server Authentication

A vulnerability in this method of authentication is the CA itself. The certificate of the server should be issued by a trusted CA which is also called as a trust anchor. The

root CA's are currently required to pass the WebTrust for Certificate Authorities before thay are added to the root certificate store [6]. There is a root certificate store associated with each browser and operating system. But even the CAs with WebTrust validation can breach the trust, sharing their private key with third parties who may use it for fraudulent purposes. There is an option for users to distrust root CA's with questionable credentials.

2. DOMAIN VALIDATION

CAs validate the digital certificates of servers through various means like domain validation (DV) [7]. DV is based only on the consent of the applicant and the proof of the applicant's control over the domain. The approval of validation is done through email, telephone call, pseudo-random nonce or DNS TXT records. In DV, even though the consent of the domain owner is obtained, there is no mechanism for verifying who the domain owner is. Domain validated certificates are vulnerable to phishing and man-in-the-middle (MitM) attacks [7]. For example, a serious flaw in the DNS discovered by Dan Kaminsky allows an attacker to redirect network clients to alternate servers of his own choice [8]. A more secure alternative to DV is extended validation (EV)[9]. In fact, DV certificates can be upgraded to EV certificates. In addition to the encryption and other features as is used in DV, EV incorporates identity validation also. For obtaining EV certificate, the requesting server's identity has to be verified by a CA. The EV requirements comprise of manual checks of all the domain names requested by the requesting server. They are checked with reference to governmental sources, independent information agencies, and through phone calls to the applicant company. On acceptance, the government-registered serial number of the company and its physical address are stored in the EV certificate. Thus there is an additional warranty on the legal status and identity of the business organization.

But recent incidents have indicated that even EV certificates are not immune to attacks. Advanced Persistent Threats (APT) are invoked as the cause of these attacks [10]. In an APT, advanced or sophisticated techniques employing malwares are used to exploit the vulnerabilities in systems. The victim systems are persistently monitored and data extracted to build the attack. Sophisticated data and security analysis techniques have to be used to counter APT. The issue of APT is nowadays raised in the context of attacks on CAs. Current experience shows that even trusted root CAs can be broken into with the intention of issuing fraudulent certificates to other servers and CAs. Such fraudulently-issued certificates can be used to infiltrate into mission-critical systems and implant malwares with devastating power. Due to these reasons, the security compromise of root CAs has become an important issue in network security. Once a root CA is compromised, it can create an avalanche of certificate forgery for not just domain servers but also to subordinate CAs.

In this context, it becomes important to reassess the whole mechanism for enforcing security in communication. It becomes clear that, ensuring the security of the root CAs does not solve the problem. Even with the root CAs, the author in [11] demonstrates that given the proliferation of CAs, even a low probability of breakage

into root CAs accumulate into a very high probability of security compromise. The only viable solution seems to be to prevent compromised CAs from issuing certificates to any applicants whatsoever. Here we squarely face the issues of certificate authorization and certificate revocation.

3. CERTIFICATE AUTHENTICATION METHODS

The certificate authentication problem essentially revolves around the issue of who is authorized to issue certificates and declare them valid to an entity. Certificate path validation is an important component of this. Certificate path validation can take place in two directions. In forward path validation, the path is built from the target entity to a trust anchor (typically a root CA) while in reverse path validation it is done in the opposite direction. In [12], the authors prove that the reverse path validation is more effective than the forward path validation for general trust models. Examples are the 'name constraints' and 'policy processing' extensions in X.509 certificates. Both these extensions can be used to filter out many alternative paths that do not validate, if we make use of a reverse strategy. But these extensions are not capable of doing away with the power of an impostor CA issuing fraudulent certificates [11]. The only option seems to be dynamically updating a list of trusted CAs. But given the large number of CAs, this is not viable unless it is a corporate environment requiring a small list of CAs.

The Online Certificate Status Protocol (OCSP) can be used to check the revocation status of compromised CAs [13]. But confirmation of certificate validity requires contact with another third party. Man-in-the-Middle (MitM) attacks are also possible amidst OCSP queries. When adopted popularly, CA servers of high traffic websites will be flooded with a large volume of OCSP queries. This is also another disadvantage because it slows down the connection process. Stapling also called as Certificate Status Extension is proposed as an alternative to this. In stapling, the certificate holder itself queries the OCSP server at regular intervals to obtain a signed stamped OCSP response. This stamped response is provided to the connection seeking client. One disadvantage is that, in a certificate chain OSCP stapling has to be done for each link in the chain. Multiple Certificate Status Request Extension of TLS [14] or multi-stapling as the name indicates can be used to provide the status of the requested server as well as the intermediate certificates. The size of the multi-stapled data in a handshake is a problem that need to be addressed. This may create significant delays in the handshake.

HTTP Public Key Pinning (HPKP) is another approach [15], in which a host is pinned or associated with its own X.509 certificate or public key. If the certificate is known to belong to a host at the time of the development of an application, this certificate or public key can be preloaded to it. Otherwise it has to be added during the first encounter. Preloading is preferred because it is offline while a privileged attacker can foil pinning in a first encounter. The advantage of pinning is that it eliminates the need to contact third party servers and CAs during the connection. Backup public keys are preferred for sites in case they lose the key or control over key and thus are not able to

authenticate during a handshake. An attacker may pin a wrong key during the first encounter thus blocking a site from a client permanently. Thus pinning puts a lot of responsibility upon the user applications.

DNS-based Authentication of Named Entities (DANE) binds certificates to domain names using DNSSEC [16]. Thus it can be viewed as a generalization of pinning through incorporation of DNS. DANE protocols can augment the security of the conventional PKIX-based system and allows domain holders to assert certificates for themselves, without involvement of third-party CAs. But contrary to pinning this increases the responsibility and role of DNS operators. DNSSEC allows a client to securely verify the credentials of a zone or domain from its operator. For security, the operator may specify trust anchors or provide constraints on CA or certificates. TLSA records in the DNS resource records are used for this. DANE makes use of the knowledge of the domain operators about the CAs from whom they have received certificates. Thus the problem of mis-issue of certificates is solved. DANE deployment at the client side is a more challenging issue than on the server side due to the need for application developers to reframe DNS support according to the DNSSEC specifications. Validating the chain of DNSSEC signatures in a connection also imply several round trip delays and may not be desirable in a real-time communication scenario. The increased role of DNS operators also opens up newer possibilities of server attacks.

It is clear that the problem of mis-issue of certificates and the related security hazards on the one hand demands reduced role for third party systems in authentication. But this put more load and responsibility on the client and user applications. Preloaded security information without any update may lead to permanent lockup or blockage of websites. There is a trade-off involving many factors which demands development of new security trust models. Radical solutions can emerge if such trust model are formulated. A trust model has to be distinguished from merely a set of security mechanisms adopted in a security architecture. It comprises of a security mechanisms in conjunction with a security policy addressing business, technical and regulatory requirements [17]. It would be of interest to examine certain trust models that have been proposed recently.

4. TRUST MODELS

An important trust or security model is Trust On First Use (TOFU) or Trust Upon First Use (TUFU) [18]. Examples of TOFU are SSH and HPKP that we have discussed above. In this model, when the client wants to establish a relationship with a server, it will first look for its public key or some identifier in its own local database or store. If such an identifier does not exist there, then a mechanism for accepting the key submitted during the first encounter with that particular server will be initiated. This may typically involve the human user himself. The human involvement is a strength as well as a weakness of this paradigm. Involvement of a human being in every validation lacks scalability and is prone to induce a higher probability of human

errors. Also, acceptance of any key during the first encounter implies that the clients and users are vulnerable to attacks by entities lying in the path of communication.

To deal with these weaknesses in the TOFU approach, another alternative was proposed which is called PERSPECTIVES [19]. Rather than depending on a model of a single unassailable third party server or self-signed certificates, this model transfers the trust on a set of so-called 'semitrusted', 'network notary' servers. Security is based on the 'perspective' accrued by the client from the data obtained from these diverse network vantage points over a span of time. The system has a simplified deployment model without the need of any CA's for verification and issue of certificates. The basic philosophy is to address the weakness of current authentication schemes, which is complete trust on client and server.

Perspectives has also many weaknesses. One is attack on multiple notaries that may prevent the client from forming the necessary quorum in the perspective. Similarly, the caching of the responses from servers may make the response from a notary invalid in the case of a switching of the certificate of the server. This would be detected by the notary only through its periodic check on the site and creates a notary lag. The 'Convergence' model avoids this problem [20]. It compares the information with the user with that of the notary cache. The server is contacted in case of a mismatch and thus avoids the problem of notary lag due to this problem. Notary bounce is another radical concept in the convergence model. One notary will be assigned the role of a bounce or a dumb proxy through which communication to other notaries is conducted. The bounce does not know the content of the communication and the notaries does not know the identity of the client. Thus this two-tiered approach to notaries has considerable augments the privacy of communication.

One important aspect in the formulation of a trust model, is the type of attack that is sought to be addressed through the model. Realistic attack scenarios are important for analyzing the security performance of the model. For example, in [19], the authors analyze the performance of the model using the MitM attack. The attack scenarios specifically demonstrate the vulnerabilities in the system. A network model is proposed in [21] which captures the high-level structure underlying the design of concrete attack-resistant networks. In [22], an information-theoretic perspective on security is formulated by modeling the client-server architecture under active attacks as a binary-erasure wiretap channel. It is proposed that the secrecy capacity of the equivalent wiretap channel can be used as a metric for the design of attack-resilient architectures.

4.1. Meaning of Trust

Trust is a central concept to be engaged in the development of a trust model. Even though, the meaning and sense of trust will change depending on the nature of applications, certain salient features have to be understood. The end-user cannot be expected to possess the experience and expertise to recognize the potential risk in each connection. Hence there should be a mechanism to assess the trustworthiness of an entity to which the user is connecting. There are two aspects to the challenge of

secure communication. One is that, the communication of valuable and private information between two entities should be secure from all forms of leakage, corruption and eaves-dropping. This can be ensured through cryptographic methods. The second is that, the authenticity of the entity with which connection is sought by the user should be ensured. Modeling of trust always presupposes a malicious player or players trying to compromise the security and authenticity of the connection and related communication. Thus in the model we have to accept the presence of useful and harmful entities. Useful entities include the client itself, the actual entity to which the connection is sought or made, all intermediate agencies required to establish this connection, and other agencies that verify and guarantee the authenticity of the useful entities and thwart the attempt of malicious entities. Harmful entities include actual mischief-seekers as well as the resources they may implant in the useful entities to subvert the trust, security and authenticity. A more careful viewpoint has to include the vulnerabilities in the hardware and software configurations of the connection also as part of the malicious aspect of total trust model. Thus we cannot clearly demarcate the useful and malicious parts in the model. This is much more so, in our context in which even the trust of root CAs themselves are compromised.

4.2. Measuring Trust

In a trust model, there are many important issues like measuring trust. This not an easy thing because trust is a soft measure and also a predominantly social concept [23]. This prevents from creating a clear-cut metric of trust. Transitivity of trust in a trust chain is also problematic [24]. If the trust is transitive, it can be extended to all domains that a trusted domain trusts in. It can be represented as follows:

$$(\mathbf{A}\ \textit{trusts}\ \mathbf{B}) \,\&\, (\mathbf{B}\ \textit{trusts}\ \mathbf{C}) \Rightarrow \mathbf{A}\ \textit{trusts}\ \mathbf{C} \qquad (1)$$

Without transitivity, chains of authentication and authorization cannot be formed. A method to formalize trust is to define trust classes as is elaborated in [25]. Trust classes are defined based on functions, tasks and reasons for potential distrust. The authors developed a framework to define and express trust requirements and to analyze the trust in authentication protocols. A similar approach in [23] makes use of 'trust categories' defined on the basis of the tasks for which different entities are trusted upon and also 'trust values' reflecting the level of trust for each entity. Direct trust and recommender trusts are given different semantic categories each with an integer trust value. Through this approach, trust can be calculated through different recommendation paths and the average trust calculated. Various other approaches to formulate trust as a metric and to evaluate it has also been proposed [23]. An interesting approach is formulated in [26], which makes trust an information-theoretic metric. In a relation between two entities, there is uncertainty (consequently entropy) that one entity will behave in a desired manner by the other entity. This is quantified into a trust value. The trust model proposed in [27] distinguishes between what is called as integrity trust and competence trust and defines trust as environment-

specific. A competent entity need not possess good integrity. However, in certain real-time scenarios, competence may have to be prioritized. This approach draws heavily from social science conceptions of trust to develop a computationally-dynamic trust model. In defining trust, the authors in [28] considers the trade-off between trust and anonymity. From these works it is obvious that through specific mathematical frameworks trust, though essentially qualitative, can be quantified into metrics. In the next subsection, we will briefly examine how these quantified metrics of trust are used to manage trust relationships.

4.3. Incorporating Trust

We examine how trust values are incorporated into a valid framework for the propagation of trust. Based on the knowledge of an entity about another entity, entropy based trust values and probability (of a particular action) values can be assigned [26]. These are subjective to each entity. These values can be used to create a concatenation of entities for trust propagation. In the case of such multiple paths of propagation existing for trust they can be combined using techniques like maximal ratio combining or data fusion [29]. In [22], instead of the trust value, secrecy capacity of the assignment matrix between access points connected to the clients and targets connected to the servers is used as an optimization metric. Through this the optimization of the design of the assignment matrix and thus the client-server architecture can be achieved. In [25], the trust classes and the related trust rules are used to design protocols whose execution can reflect particular trust circumstances. The trust model in [27], contains two types of agents, namely trusters and trustees, a related database of trust information which also includes context of trust and rules for dynamically updating the trust of each truster about a trustee.

5. TRUST IN PKIs

Public-Key Infrastructures (PKIs) were developed as a mechanism for articulating and managing trust relationships. The clients make use of PKI certificates to determine whether it is secure to accept the keys provided by an entity. This in effect makes use of the concept of the transitivity of trust as is mentioned above in the form of certificate chains. But recent security breaches and flaws have led to the erosion of trust in the PKI itself. Trust anchors are a central concept in PKIs. Certificate chains and certificate path validation to the trust anchors are important mechanisms in the PKI mechanism. Trust as is discussed above in terms of various variables can be incorporated in the different fields which are part of the certificate. The data in the certificate fields are used for path validation. However all information in the certificate fields may not be useful for path validation. There may be information which have a limited scope and also those which are shared by many certificates. A typical example of certificate fields is shown in the table below:

Table 1 Certificate fields and their functions

Certificate Field	**Function**
Issuer Distinguished Name (DN)	Identity of the CA
Subject DN	Identity of the certificate holding entity
Public Key	The shared public key
Authority Key Identifier (AKI)	SHA-1 hash of the public key held by the signer of the certificate
Subject Key Identifier (SKI)	SHA-1 hash of the public key included within this certificate

In the above list, Subject DN and Issuer DN is not unique while the Public Key, AKI and SKI are unique to the certificate.

In a PKI trust model, there are two important processes that are central to the proper management of trust. They are:

1. Path construction
2. Path validation

Path construction is a top-down process starting from the self-signed root CA downwards. Validation is bottom-up. The Issuing DN and AKI certificate fields provides an efficient search mechanism in the path validation process. It can be seen that path construction is more complex than path validation.

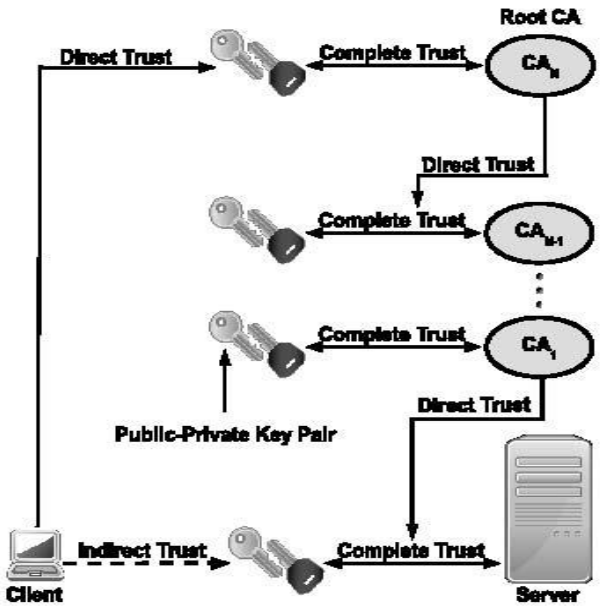

Figure 2. PKI Trust Structure

5.1. Path Validation and Trust in PKIs

A validated path cannot give 100% guarantee about the authenticity of the certificate. In identity spoofing, a malicious player deludes the CA to issue a PKI certificate by masquerading as the actual owner. Thus trust is not the same as the validation of a PKI certificate. A certificate can only be seen as a means of providing some amount of trust, however large or small it may be. The core issue is that A is ready to do a transaction with *B*, if it is convinced of the identity of *B*. This transaction has to be executed over a network, and the nature of the transaction demands it. Authentication of B also has to be undertaken over the network itself. These are the givens of the problem. PKI is a method for implementing this.

Trust is incorporated in PKI trust models through hierarchical structures starting with root CA. A client seeking a service from a server has to first establish an out-of band, direct trust relation with the root CA. An out-of-band channel for establishing the trust means an extra-protocol channel. This may be in the form of preloading of certificate or public key during the development of an application or web browser as is mentioned earlier or some other mechanism or channel of communication. In the PKI trust model, the intermediate CAs and the end users (servers) seeking certificates should also have trust in the CAs for their authentication in relation to clients. The root CAs, intermediate CAs and the end-user clients should have complete trust in their own public private key pairs. In a CA chain, the root CA generates its own

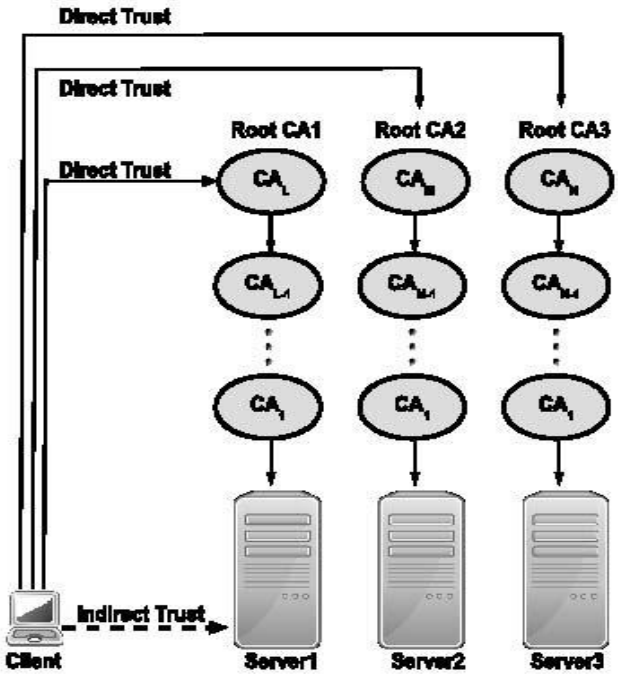

Figure 3. PKI Trust Structure with Multiple PKIs

self-signed certificate. The root CA then authenticates the next intermediate CA and establishes the binding between its identity and public key. This is how direct trust is established between the root CA and the first intermediate CA. Similarly, this intermediate CA also establishes a direct trust relation with the next CA in the chain and so on to the end-user server who seeks a certificate. The trust between a client and the server is an indirect trust. All these trusts can be represented as shown in figure 2. We will examine the threats to each of these trusts.

5.2. Direct Trust between Client and Root CA

The PKI structure depicted in figure 2 requires only a single root CA. Hence it is easy to distribute its public key through a secure out-of-band channel to a client. Thus this trust is optimal unless there are issues with the honesty, integrity and reliability of the root CA. These issues are out of scope of the fundamental definition of trust in a PKI infrastructure. In addition to this, the trust structure in figure 2 does not scale and is relevant to the operation of an organization having ownership of the root CA and all connected intermediate CAs. In a more general model, there are multiple root CAs with each CA having its own PKI structure. This is shown in figure 3. In this case, the secure out-of-band distribution of the root CA certificates becomes problematic. One solution is to code all the root certificates in the web browsers. Many of these pre-installed certificates have a limited expiry period. For example, in one browser that I am using and was installed this year, many of the pre-installed certificates have expired two or three years before. To handle this problem, there is a provision for dynamically adding and deleting root CA certificates in the web browser-based PKI model. But there is a threat of

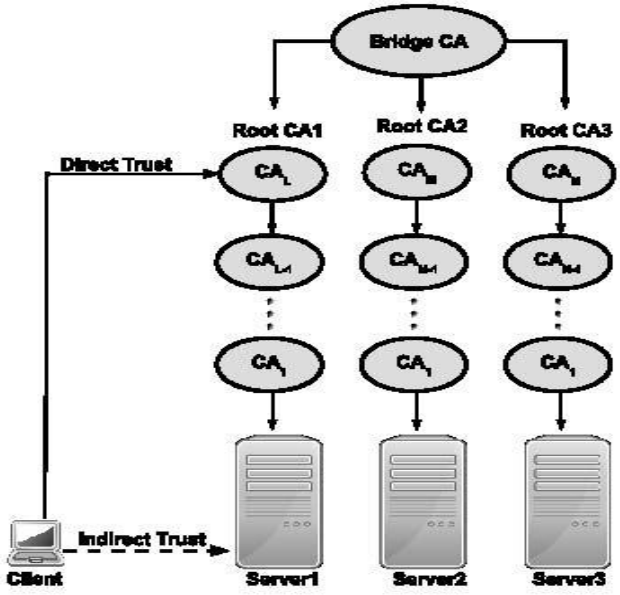

Figure 4. PKI Trust Structure with Multiple PKIs and a Bridge CA

spoofing and APT in this case which severely compromises the security. The direct trust of a client on a root CA can be based on the cross certification of the multiple root CAs of the certificates of each other. However, this also does not scale well due to the almost exponential increase in the number of cross certifications required when the number of root CAs increases.

Using a bridge CA, as shown in figure 4, the required number of cross certifications can be decreased to the number of root CAs. All root CAs cross certify only with the bridge CA. The client needs to obtain only one certificate from the root CA connected with the server from which the client seeks service.

The issue of direct trust in a root CA can be eliminated completely by making use of non-hierarchical trust models like Pretty Good Privacy (PGP) [30] and the open source, Gnu Privacy Guard (GPG) [31]. Typically in GPG, there are different trust levels for the public keys as shown in the table 2:

Table 2 Trust Levels in GPG

Trust Level	**Explanation**
-	No owner trust assigned / not yet calculated
e	Trust calculation has failed; probably due to an expired key
q	Not enough information for calculation
n	Never trust this key
m	Marginally trusted
f	Fully trusted
u	Ultimately trusted

*5.3. Direct Trust between CA*s

Direct trust of a CA at a higher level in a CA or entity immediately below it is established on two things:

- Verification of the identity of the lower entity in the certificate chain.
- Authentication of its public key.

This trust, though direct, cannot say anything about the reliability and honesty of the entity. Yet this is critical since, the underlying entity is entrusted with the responsibility of issuing certificates and other critical services. The VeriSign [32] and DigiNotar [33] incidents have demonstrated the vulnerabilities in the hierarchical PKI model that compromise the direct trust between entities in a PKI trust structure. Certificate masquerading is a serious issue in a web-based PKI with many root CAs and intermediate CAs [34]. It can be said that the Browser PKI is only as secure as the

weakest of each separate PKI it contains, and each separate PKI is only as strong as the weakest of each CA member it contains [35].

5.4. Complete Trust between CA and Server

In PKI trust models servers can generate their own public-private keys. But in certain trust models, the CA generates the public-private key pair for the server. When the server itself generates the key pair, the CA must verify the binding between the key and the server. It is possible that a CA generating the public-private key pair for a server may have a malicious intent and masquerade as the server itself, thus compromising security. This is a vulnerability in the complete trust essential for the binding between an entity and its key pair. It could be said that the direct trust in the relationships between the CAs in a PKI tree till the server is the basis of the complete trust of an entity in its own key pair, when the keys are generated by a CA.

5.5. Indirect Trust between Client and Server

The indirect trust of the client in the server is a function of all the above-mentioned trusts. Ultimately, this trust and the communication or transaction based on it is the raison d'etre of the whole trust model. Elimination of dedicated CAs as in PGP, reversal of the direction of certificate chain as in the case of the SPKI/SDSI trust model [36] etc. generate new semantics of trust. Trust in a PKI cannot cover the honesty or integrity of an entity. Direct trust in general presumes a secure out-of-band channel for establishing trust. In many cases there is no clear specification of this. Cost-effective distribution of certificates may be contrary to the spirit of the out-of-band channel.

6. CONCLUSION

Trust plays an overwhelmingly important role in the development of authentication mechanisms in secure communication. In the different types of connections between entities in a secure communication model, trust takes on different meanings and implications. An exact delineation of this typology will go a long way in developing better trust models and hence more secure communication. In this paper, we have explored the issue of trust in secure client-server communication. We have examined the methods of domain validation and certificate authentication. Recent trust models are also explored. More than all, the incorporation of trust in its various forms in a PKI hierarchy is given ample consideration in this work. The nature of the trust between different entities in a PKI and their weaknesses are also examined. Hopefully, the issues covered in this paper can serve as a basis for analytic formulations of the vulnerabilities of trust in PKIs and the development of novel techniques to address them.

REFERENCES

[1] https://tools.ietf.org/html/rfc882

[2] McDowell, Mindi, "Cyber Security Tip ST04-015 -Understanding Denial-of-Service Attacks", United States Computer Emergency Readiness Team, November, 2009.

[3] https://tools.ietf.org/html/rfc3833

[4] https://www.ietf.org/rfc/rfc1034.txt

[5] "Certification Path Validation",. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, Network Working Group, 2008.

[6] https://www.sslshopper.com/article-what-is-webtrust-for-cas-certification-authorities.html

[7] https://www.sslshopper.com/best-domain-validated-ssl-certificates.html

[8] https://lwn.net/Articles/289138/

[9] https://cabforum.org/about-ev-ssl/

[10] Colin Tankard, "Advanced Persistent threats and how to monitor and deter them", Elsevier Network Security Volume 2011, Issue 8, August 2011, Pages 1619

[11] Rolf Oppliger, Certification Authorities Under Attack: : A Plea for Certificate Legitimation, 2014, IEEE Internet Computing, January/February 2014

[12] Y. Elley et al., Building Certification Paths: Forward vs. Reverse, Proc. 2001 Network and Distributed System Security Symp. (NDSS 2001), ISOC, 2001, pp. 153160.

[13] https://tools.ietf.org/html/rfc6960

[14] https://tools.ietf.org/html/rfc6961

[15] https://tools.ietf.org/html/rfc7469

[16] Richard L. Barnes, "DANE: Taking TLS Authentication to the Next Level Using DNSSEC," IETF Journal October 2011, Volume 7, Issue 2

[17] Donna Andert, Robin Wakefield, and Joel Weise, "Trust Modeling for Security Architecture Development,"SunBluePrints™ OnLineDecember2002, http://www-it.desy.de/common/documentation/cddocs/sun/blueprints/1202/817-0775.pdf

[18] Gabor X Toth and Tjebbe Vlieg, "Public Key Pinning for TLS Using a Trust on First Use Model," Available at http://tg-x.net/code/certpatrol

[19] D. Wendtandt, D.G. Andersen, and A. Perrig, Perspectives: Improving SSH-Style Host Authentication with Multi-Path Probing, Proc. Usenix 2008 Annual Tech. Conf. (ATC 08), Usenix Assoc., 2008, pp. 321334.

[20] Thoughtcrime Labs/IDS: Convergence, http://convergence.io (2011).

[21] T. Bu, S. Norden, and T. Woo, "Trading resiliency for security: Model and algorithms," in Proc. 12th IEEE Int. Conf. Network Protocols, Berlin, Germany, 2004, pp. 218227.

[22] Matthieu Bloch, Rajesh Narasimha, and Steven W. McLaughlin, "Network Security for Client-Server Architecture Using Wiretap Codes", IEEE Transactions on Information Forensics and Security, VOL. 3, NO. 3, September 2008

[23] A. Abdul-Rahman and S. Hailes, "A distributed trust model,"in Proc. 1997 New Security Paradigms Workshop, 1998, pp. 4860.

[24] Audun Jφsang, "The right type of trust for distributed systems," In Proceedings of New Security Paradigms '96 Workshop, 1996.

[25] Raphael Yahalom, Birgit Klein, Thomas Beth, "Trust Relationships in Secure Systems A Distributed Authenrication Perspective", In Proceedings of IEEE Symposium on Research in Security and Privacy, 1993

[26] Yan Lindsay Sun, Wei Yu, Zhu Han, and K. J. Ray Liu, "Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks", IEEE Journal on Selected Areas in Communications, VOL. 24, NO. 2, FEBRUARY 2006.

[27] Yuhui Zhong, Bharat Bhargava, Yi Lu, and Pelin Angin, "A Computational Dynamic Trust Model for User Authorization", IEEE Transactions on Dependable and Secure Computing, VOL. 12, NO. 1, January/February 2015

[28] Li Lu, Yunhao Liu, Lei Hu, Jinsong Han, and Lionel M. Ni, "Pseudo Trust: Zero-Knowledge Authentication in Anonymous Peer-to-Peer Protocols", Technical Report TR 2006-10

[29] D. L. Hall and S. A. H. Mcmullen, "Mathematical Techniques in Multisensor Data Fusion", Norwood, MA: Artech House, 2004.

[30] Michael W. Lucas , "PGP & GPG : Email for the Practical Paranoid", No Starch Press Inc., 2006. ISBN: 978-1-59327-071-1

[31] J. Callas, L. Donnerhacke, Finney H., D. Shaw, and R. Thayer, "RFC 4880 - OpenPGP Message Format," IETF, November 2007. Available at: http://www.rfc-editor.org/.

[32] Microsoft Security Bulletin MS01-017 (March 22, 2001): Erroneous VeriSign-Issued Digital Certificates Pose Spoofing Hazard. http://www.microsoft.com/technet/security/bulletin/MS01017.asp, 2001.

[33] Elinor Mills, "Fraudulent Google certificate points to Internet attack," http://news.cnet.com/, August 29 2011.

[34] James M. Hayes (1998), "The problem with multiple roots in web browsers - certificate masquerading," In 7th Workshop on Enabling Technologies,

Infrastructure for Collaborative Enterprises (WETICE 98), pages 306313. CAUSA Proceedings, IEEE Computer Society, Palo Alto, June 17-19 1998

[35] Audun Jφsang, "PKI Trust Models," In Atilla Eli et al. (editors), Theory and Practice of Cryptography Solutions for Secure Information Systems (CRYPSIS). IGI Global, May 2013. ISBN13: 9781466640306

[36] D. Clarke, J.-E. Elien, C. Ellison, M. Fredette, A. Morcos, and R. L. Rivest, "Certicate chain discovery in spki/sdsi," J. Comput. Security, vol. 9, no. 4, pp. 285322, 2001.