Search for Trust: An Analysis and Comparison of CA System Alternatives and Enhancements

Alexandra C. Grant
Senior Honors Thesis
Advisor: Charles C. Palmer

Dartmouth Computer Science Technical Report TR2012-716

Abstract. The security of the Public Key Infrastructure has been reevaluated in response to Certification Authority (CA) compromise which resulted in the circulation of fraudulent certificates. These rogue certificates can and have been used to execute Man-in-the-Middle attacks and gain access to users' sensitive information. In wake of these events, there has been a call for change to the extent of either securing the current system or altogether replacing it with an alternative design. This paper will explore the following proposals which have been put forth to replace or improve the CA system with the goal of aiding in the prevention and detection of MITM attacks and improving the trust infrastructure: Convergence, Perspectives, Mutually Endorsed Certification Authority Infrastructure (MECAI), DNS-Based Authentication of Named Entities (DANE), DNS Certification Authority Authorization (CAA) Resource Records, Public Key Pinning, Sovereign Keys, and Certificate Transparency. Provided are brief descriptions of each proposal, along with an indication of the pros and cons of each system. Following this, a new metric is applied which, according to a set of criteria, ranks each proposal and gives readers an idea of the costs and benefits of implementing the proposed system and the potential strengths and weaknesses of the design. We conclude with recommendations for further research and remark on the proposals with the most potential going forward.

Keywords. PKI, Certification Authority, Trust, Perspectives, Convergence, MECAI, DANE, CAA Resource Records, Public Key Pinning, Sovereign Keys, Certificate Transparency.

1 Introduction

Recent attacks on Certification Authorities (CAs) [1, 2] trusted by default in popular web browsers and operating systems have led many to question the state of the Internet trust infrastructure. These security breaches of Internet Certification Authorities (ICAs) or their agents have allowed fraudulent certificates to be issued for popular domains, leading to compromised security and privacy for users of those domains. These events have turned a lot of heads and there is now a renewed effort to examine the trustworthiness of Public Key Infrastructures (PKIs) and the ICA system as a whole and to determine, is it broken? And, if so, what can be done about it?

The PKI system relies on the trustworthiness of ICAs to ensure that certificates are only issued after it has been verified that the identity represented in the certificate is the one in control of the corresponding private key. This process, known as identity binding, should be as transparent as possible to allow the acceptance of the certificate by the broadest set of relying parties.

There are a variety of reasons why attackers would focus their efforts on ICAs (or their agents) as opposed to targeting an individual site or server and its corresponding certificate and associated private key. Compromising an ICA allows attackers to generate certificates on behalf of any person or entity that will be broadly trusted by the Internet community, providing many opportunities for further exploits. Unfortunately, in the eyes of the current Internet trust infrastructure, all ICAs are treated equally which, from a security perspective, is to say that they, and their certificates and records, are all trusted equally. However, such an assumption is far from the reality. Meanwhile, attackers take advantage of the blind trust in ICAs by searching for vulnerabilities in any of the ICAs in the trust community to identify the "weakest link." Some ICAs may be more trustworthy than others by using more secure systems and processes to protect their customers.

With the number of recent attacks on ICAs, some have advocated that it is time to reevaluate PKI and to propose an alternative or enhancement to the ICAs behind the SSL/TLS system. The proposals which have attracted the most attention are explored in this project. These included Perspectives[5], Convergence[4], DANE[3], Public Key Pinning[6] or HSTS pinning[10], CAA Records in DNSSEC[7], Sovereign Keys[8], MECAI [9], and Certificate Transparency [11]. However, before we can begin to seriously consider the benefit of the proposed

replacements or enhancements to the current SSL/TLS ICA system, we must have some means with which to assess and compare the security, efficiency and practicality of each alternative. This project has developed a means of comparing each system based on well-defined categories and criteria. In this way, we were able to effectively determine the best routes to take in ensuring a secure, scalable, and efficient process for establishing trust relationships on the Internet and provide insight into which, if any, of the proposals listed above are an appropriate replacement or enhancement to the existing ICA trust system used by SSL/TLS.

In the next section, we will provide an overview of each of the proposed replacements and/or enhancements to the SSL/TLS ICA system, a discussion of their components and policies, and an analysis of their perceived relative strengths and weaknesses. In latter sections we will define a new set of metrics for quantitative evaluations of the proposals, and draw conclusions about their suitability.

2 Proposals

2.1 Perspectives

Overview. The Perspectives [5] design uses a network of entities called *notary hosts* to observe a server's public key via multiple network vantage points. The reasoning behind this approach is that all clients, upon contacting the server with a specific query should receive the same public key in response. Any disagreement among the observed key values would indicate a potential problem. The keys gathered by the notary hosts are then compared to the unauthenticated key downloaded by the client and a decision is made to reject or accept the key based on the client's *key-trust policy*. While some network hosts can afford to pay the often high costs of certificate verification by relying on ICAs, Perspectives seeks to provide a more cost effective alternative for authenticating a public server's key.

Components. Perspectives relies on *notary authorities*, trusted third parties which are responsible for a group of notary servers. Their role is to publish the list of trusted notary server IP addresses and public key pairs to each notary server. In this way, the client can contact any notary server to download the list. This list is signed by the notary authority private key and each client will, by some out-of-band-method, have a copy of the notary authority's public key with which to verify the authenticity of the list.

Notary servers are responsible for keeping a record of server key data. For each entry (defined by a service type and a service ID), the notary server stores a record of all keys it has observed from this server along with a timestamp. When a client makes a query to a notary server, the notary server will look up the entry for the requested service in the database and then respond with the key history for that service.

The notary servers are comprised of two parts, the *probing module* which contacts services that use certificates and the *database module* which is a repository of all the services which are being monitored by the probing module. The probing module works by mimicking a client attempting to connect with a service It contacts the server and maintains the connection until it receives the server's public key, after which the connection is dropped. It is yet to be determined if notary servers will be given a default list of popular services to probe for key data or if the probing will be done only upon a request from a notary client. Once a notary server has contacted a server, it stores the service ID and details for contacting this service in the database module. The probing module will continue to periodically run through the list of known service IDs and download new key data. In this way, the notary server builds up a history of observations for each service over time. The repository is signed with the notary server's private key.

The Perspectives architecture requires that each user have a *notary client* which runs in the user's browser. This notary client contacts notary servers to request information on a server's public key. There are two possible scenarios which would prompt a notary client to contact a notary server: a user is accessing a site for the first time, or the previously verified certificate for a site has changed. In order to make a trust decision about the public key received from the server, the notary client will choose n notaries at random from its list of known, trusted notary servers. Each notary server will be contacted in parallel and the client will wait for responses from q of these notaries (where $q \leq n$, is defined by the user's key-trust policy). Once the client receives the observed key data from the notary servers, based on the user's key-trust policy (discussed below), the user will either be allowed access to the service or the public key will be rejected and the user denied access to the potentially malicious site.

The user's key-trust policy is designed to take into account, not only the consistency among notary servers for an observed key (defined as *quorum*) but also the duration for which that key has been seen (defined as *quorum duration*).

The role of this policy is to test the spatial and temporal consistency of an offered key with respect to notary data in order to determine whether to accept or reject the key. Each user has the ability to choose their desired level of security by adjusting the quorum and quorum duration threshold values. The higher the *q* and *d* thresholds, the more certainty that a key is valid. However, it must be noted that while a high *q* provides the strongest security, if any notaries are unavailable or compromised then it is easier for a valid key to be rejected. Likewise, if *d* is very high, then new or recently changed keys will be rejected.

Another aspect of the key-trust policy and notary design is data redundancy which provides added security and functions as a method for detecting comprised notary servers. This cross-validation mechanism requires that a certain number of *shadow servers*, say *r*, must corroborate a notary's observation for it to be considered valid. The client policy specifies this *r* and a user can customize this value as they desire. Notary authorities are responsible for publishing a list of the shadow servers responsible for each notary server. Each notary server serves as shadow server for several other notaries. Shadow servers store an immutable record of each observation made by another notary and with each query reply received the client will randomly pick *r* shadow servers to contact to determine that the reply is consistent with the information stored in these shadow servers. In this manner the observations recorded by a notary server are non-repudiable and inconsistencies in a notary's observations can be used to alert the notary authority of malicious behavior.

Analysis. Perspectives' strength lies in its ability to make trust decisions without having to rely solely on ICAs as a third party for certificate verification. This creates an alternative for services looking to potentially avoid the costs associated with traditional PKI since this service could leverage self-signed certificates instead. However, Perspectives is described by the authors as "not bullet-proof, but [it] provides a security trade-off suitable for many non-critical websites. [14]" The current PKI system is still recommended for sites that require strong security and high uptimes. From a realistic viewpoint, Perspectives has three potential use cases. It could provide validation for self-signed certs, adopt the role of a "low-end" CA certificate alternative, and/or provide additional security for CA root-signed SSL certs.

It is yet to be determined how the notary servers would be distributed and who would serve as notary authorities. As of right now, Carnegie Melon University runs a majority of the available notary servers, although ideally notaries would be distributed across the internet. There are still issues which must still be addressed for the notary infrastructure, including specifications for notary authority security policies (since a compromised notary authority could lead to widespread security issues for all clients relying on notary servers under that authority), and policies for creating this distributed architecture. As Adam Langley, a member of Google's Chrome browser team, remarked, in the event that Perspectives becomes widely adopted, "we [would] have a very strong interest for the notaries to function, otherwise Chrome stops working… that means Google would end up running the notaries. So the design boils down to Chrome phoning home for certificate validation. That has both unacceptable privacy implications and very high uptime requirements on the notary service."[12]

Along with the trouble of setting up a truly distributed notary infrastructure, Perspectives has yet to completely address issues surrounding user privacy. In the event that notaries expose [client_IP, service_name] pairs a user's browsing history would be revealed. In order to deal with this privacy concern, Perspectives recommends using a proxy if the user wants greater privacy protection. Convergence (see next section), an extension of Perspectives, proposes to remedy this by introducing "bounce" notaries as middlemen between the client and the notary servers.

Compared with other proposals and today's ICA system, Perspectives does offer more options to users for determining their own trust policy. However, the percentage of potential (what Perspectives calls) "power users" is definitely dwarfed by the percentage of non-expert users who will simply rely on out of the box default settings. Hence, any advantages gained by placing the trust decision in the hands of users will be lost for the majority. Also, as we see with today's browser security alerts, in the case that notaries do fail to verify a server's public key, most users will simply ignore the warnings and proceed to the site anyways.

Lastly, another concern that has been raised is that of "completeness." Currently, notary verification is only done for initial HTTPS requests and not for subsequent elements such as scripts or images. This would allow for malicious content to be downloaded after the public key has been verified, putting the user at risk for an attack.

Another aspect that is of concern is whether we would be swapping one set of Trusted Third Parties (TTPs) who have some potentially bad actors in their midst, but are at least regulated and audited according to published standards, for a second set of TTPs for whom no operational standards exist, nor audit regimes etc. It may be argued

that that is in fact a step backwards instead of a step forward unless regulation, transparency, privacy and audit are addressed.

Notary lag, or increased latency because of delayed notary responses, is another concern with the Perspectives design. However, if caching is implemented it would decrease the number of client requests to notary servers and speed up the notary network.

Implementation. There are currently 8 notary servers running under the supervision of the CMU Perspectives team and code is available for users to set up notary clients to contact these servers.

CMU's Perspectives implementation uses 1369-bit RSA keys for service entry signatures[8] and store public keys as 128-bit MD5 fingerprints. It is estimated that a single service entry with several keys and timespans will consume approximately 250 bytes.

Notary requests are approximately 60 bytes and replies are 315 bytes (for standard key sizes). Estimates put the response rate for notary servers at about 20,000 requests per second. Bandwidth requirements for SSH were calculated to be approximately 1.5 KB for upstream bandwidth and 2.3 KB of downstream bandwidth and for SSL, 0.5 KB upstream and 2.0 KB downstream. These numbers are taken from the CMU Perspectives design paper.[5]

2.2 Convergence

Overview. Convergence [4] builds on Perspectives' design, extending the basic system to include more features as well as fixing some noted weaknesses in the Perspectives model. Convergence's mantra is one of "trust agility. [4]" This can be broken down into two goals which Convergence asserts are critical pieces missing from the existing system. First, a trust decision can be easily revised at any time and second, individual users can decide where to anchor their trust. The system includes Perspective's basic distributed design (see previous section – Perspectives, for the design overview) whereby a server's public key is viewed from multiple vantage points on the web by entities called notaries. The difference between the systems lies in the user's active participation in choosing which notaries to query for key information. Not only can users pick which notaries to add to the client's list but they can also remove notaries which they feel are no longer trustworthy. The system puts more control in the hands of the user than the Perspectives Project, by providing mechanisms which allow users to reject notaries (or potentially ICAs) which have proven themselves untrustworthy or to add entities ad hoc that they choose to trust.

Components. The components are the same as those described in the Perspective section. This includes the notary authority, notary servers, notary clients, and notary shadow servers. However, Convergence seeks to design notaries so that they are extensible, allowing for each to use a different backend setup. In this way, notaries can demonstrate trust agility by facilitating support for a range of trust infrastructures such as DNSSEC (DANE), the ICA system, signatures etc. As of now, the notaries use Perspectives. Convergence also introduces bounce notaries to improve user privacy (see next section - Improvements on Perspectives).

Improvements on Perspectives. Convergence addresses some of the challenges faced by Perspectives, notably completeness, privacy, and responsiveness.

Completeness. Perspectives only validates for the initial connection and not for subsequent elements such as JavaScript, images, CSS, etc. Convergence addresses this by checking all HTTPS requests through the notary mechanism.

Privacy. Perspectives has the disadvantage that a user's browsing history is stored on the notaries. Convergence remedies this by using notary bouncing. A notary is randomly picked by the client to be the *bounce notary*. This bounce notary knows the user's identity but not the service that they want to request. It passes the requests off to the other notaries which will be able to see what service was requested but won't be privy to the user's identity. The replies from these notaries are all sent back to the bounce notary which sends the information back to the user. For a user's privacy to be compromised, two notaries would have to collude with each other.

Responsiveness. Perspectives suffers from significant notary lag. Convergence handles this through caching and only querying if the key does not match the one in the cache or if the key does not exist in the cache.

Analysis. Convergence is a system built on the idea that trust should be placed in the hands of the users. The design was motivated by the argument that the current system has created a scenario where users are required to place their faith in the trustworthiness of the ICAs without any alternatives since removing a ICA from their trust database would severely limit the number of domains on the internet that a user could access. Ideally, Convergence empowers

the user although there is debate surrounding this idea. The user-configurability and flexibility of the system which is supposed to be regarded as Convergence's strongest attribute is also seen as a weakness, given the knowledge and experience level of the average user. It has been pointed out that most users would not change their default settings, so the default set of notaries included in the browser would remain unchanged, making those notaries subject to extremely high traffic loads, and potentially negating some of the responsive improvements.

While Convergence does offer solutions for Perspective's shortcomings, it shares the same concerns regarding user interaction. That is, any advantages gained by placing the trust decision in the hands of users will be a plus for power users but likely be lost entirely for the majority; also, the system has the potential to continue to train most users to simply ignore security warnings and proceed to the site anyways as they do with the current SSL/TLS system.

Other notable issues with the system are that Convergence does not support captive portals and, similarly to Perspectives, it suffers from what Moxie Marlinspikes describes as the "Citibank Problem" [4] i.e. there are sites such as Citibank where a different certificate is generated every time a client makes a HTTPS request to the server. This means that each notary sees a different certificate and so the user is blocked from the site. However, it was noted that Citibank is currently the only known site with this issue.

Convergence also suffers from the same unproven TTP issues that Perspectives does in that a user can essentially swap one set of TTPs (the ICAs) for a less regulated (today) set of TTPs with no defined set of standards for their operation, maintenance, and audit. There has also been no publication on who is expected to fund the operations of these notaries, and no business plan to date that makes their instantiation and long term operation and maintenance viable.

Implementation. Convergence is available as a plug-in for Mozilla's Firefox browser. As of current, there is no formal spec or documentation. More information can be found in the video on Moxie Marlinspike's BlackHat Conference talk [4]. We have found no published metrics comparing the claimed improvements of Convergence to those detailed for Perspectives, but we would anticipate incremental performance metrics if the responsiveness goals can be realized. The same implementation issues identified in the previous section for Perspectives also still applies to Convergence however.

2.3 MECAI

Overview. Mutually Endorsed Certificate Authority Infrastructure (MECAI) [9] proposes a system which, much like Perspectives and Convergence [4, 5], relies on the implementation of web notaries responsible for making statements about facts that can be discovered on the web. However, instead of doing away with the ICA system altogether, MECAI's notaries will be run by the ICAs which, on top of issuing server certificates as usual, will also be granted the power to issue shorter lived vouchers confirming facts seen on the web from the perspective of a notary. Clients connecting to a server will receive the certificate and they must also be issued a voucher from an ICA containing a certificate that can be matched with the one presented by the server. In this way the client can check the validity of the certificate along with other information including the revocation status.

Components. MECAI's proposed design includes *vouching authorities* (VAs), *vouching authority servers* (VASs), new short-lived certificates called *vouchers*, a voucher request protocol, and a proposal for a TLS handshake extension called *voucher stapling*. As previously mentioned, the VAs would be the ICAs which would be responsible for issuing the vouchers to clients. A VA could be a single ICA or a group of ICAs. The VASs are the collection of servers spread over the geographical area which serve clients and web servers with up-to-date vouching data managed by a VA. They are responsible for listening for client requests, obtaining information about different network viewpoints, and creating and signing vouchers which go to the client. Software vendors would be responsible for including the IP addresses of these servers in their client software. When a client wants to access a site, a VAS is responsible for giving the client a data object called a voucher which contains the hostname, the server certificate from the TLS handshake, a timestamp and a vouching statement from the ICA regarding revocation. Vouchers are signed by a VA so that the client can guarantee that a voucher is valid. With this information, a client can compare the server certificate received by the TLS handshake with the one included in the voucher they received from a VAS. If the certificates match and the revocation status is okay, then the client should be able to confidently proceed knowing that the certificate is most likely valid.

The vouching request protocol works as follows. In the case of a server which supports voucher stapling, during the initial setup of client-server SSL/TLS connection, the server includes a list of candidate VAs (at least

one) that the client can accept as a VA for vouching for the server's certificate. The server also includes the vouchers corresponding to the VA list, so that client can accept if at least one is valid. If no voucher is received from the server, the client must terminate the connection, giving an error and reporting the server with a feedback message. A server may request a voucher in advance from a VAS or the request may be made in real time. A client receiving a voucher will verify that it was indeed created by one of the candidate VAs the client has asked for, by validating the signatures against the respective ICA certificates that are built into the client.

In the case of a server which does not support MECAI, a VAS will receive a voucher which includes a hostname, a port number, the name of a *trust context identifier* (TCI), recent timestamp and the server's certificate as seen by the client. VAS will establish a handshake with the server and verify that all the tests pass, such as comparing certificates, checking that the voucher is reasonably fresh, and checking revocation status before returning the signed voucher to the client. Also, a VAS must keep a list of currently accepted root ICA certificates as accepted by each of the TCIs the VA supports. When the client receives the voucher, it then checks that it contains the expected TCI as part of an additional security check. In the case that ICA is compromised, VASs should be notified so that the VASs will stop vouching certificates issued by that ICA.

Analysis. MECAI's goals are to introduce a second trust opinion in order to ensure that a compromised ICA does not lead to clients accepting fraudulent certificates. Like Perspectives and Convergence, it relies on web notaries or vouching authority servers to provide information about a certificate during a TLS handshake. However, unlike these other systems, MECAI only requires feedback from one VAS which is to be randomly selected by the client from a list of trusted VASs.

The design places ICAs as the responsible party for the web notaries which avoids the creation of additional third parties but also requires a certain amount of cooperation from competitive ICAs. MECAI requires that the ICA who issued a voucher to a client be different from the ICA which issued the server certificate. To some effect, there is a conflict of interest and ICAs will be required to work with and trust other ICAs who they compete for business with. While this may encourage more responsibility among ICAs, it is unclear as to whether the ICAs would agree to such a system. It also places more requirements on ICAs who must additionally maintain their VASs and issue vouchers to clients in high quantity due to MECAI's 24-hour voucher freshness requirement.

MECAI also pushes for the creation of a TLS/SSL handshake extension in order to expedite the server certificate verification process. The proposal admits that this could introduce significant latency due to the increased size of the handshake. However, certain solutions have been proposed such as only using certificate fingerprints to decrease the amount of data which must be exchanged between client and server.

This extension would require that web servers upgrade to support the MECAI design which, from an implementation perspective, might be considered unreasonable unless there is sufficient incentive for sites to support it. Another implementation concern is that MECAI proposes that software venders build in VAS IP addresses into their products so that clients will be able to communicate with the MECAI notaries. It might be considered more reasonable for implementation purposes to allow this list to be downloaded by the client.

Currently, there is no implementation or formal spec for MECAI. The system is still very much within the initial concept design phase and there are still issues to be resolved and design decisions to be made. Due to the lack of detail it is difficult to give a fair estimate of the true practicality of the system and its potential for addressing concerns with the current system.

2.4 DANE (DNS-Based Authentication of Named Entities)

Overview. DANE [3] works by embedding certificate information into DNS records so that a client can receive authentication data directly from domain operators. The system also relies on DNSSEC to ensure a secure connection between the client and the DNS server. This system allows clients to query the domain operator about which certificates they should accept as credentials for that domain. The client validates the DNSSEC chain by verifying each record until it hits a root which it can identify as a trusted root. While there is currently no way to check which ICAs are supposed to be issuing certificates for a domain, DANE allows domain holders to put restrictions on which certificates should be accepted based on ICA and service type. This allows certificate misuse to be detected more easily because clients will be able to report certificates they are presented that do not belong to the set of trusted ICAs. The deployment of DNSSEC will also add another layer of security to web browsing and potentially eliminate MITM attacks which may occur whenever a client contacts the DNS server.

Components. As part of the DANE protocol, a new record type called TLSA (Transport Layer Security Associations) must be established. Its purpose is to relay information specifying which certificates are associated with a domain. The record is broken down into three fields. The *usage* field indicates which type of statement the record is making (CA constraint, service certificate constraint or trust anchor assertion). The *selector/matching* field indicates how a TLS certificate chain should be matched against this record (by public key, SHA-1 digest, etc.). And lastly, the *certificate for association* field contains the actual data which the TLS certificate chain will be matched against.

The CA constraints record provides a dynamic method for changing a client's trust anchor list. As DNS operators add or remove ICAs from their list of trusted certificate providers, they will send out TLSA records with CA constraint information to clients. Domain operators will take on new security responsibilities and will be responsible for asserting and revoking trust anchors for a domain. Since internally managed domain operators know exactly which ICAs they have requested certificates from and which specific certificates they have received, they are responsible for securely conveying this information to the client. For infrastructures that outsource domain operations, this is a little more problematic, and the domain host may have no preview into which ICAs are trusted and utilized by their client. DNS Servers will need to be upgraded to support DNSSEC. This will involve software changes and potentially hardware changes as well. Web clients will also need to be upgraded to support the authentication of web servers using DANE mechanisms.

Analysis. DANE takes advantage of the existing DNS infrastructure in order to validate server certificates. This extension of DNS function requires the implementation of DNSSEC as well as the cooperation of domain operators onto whom new security responsibilities will be placed. The system design localizes security issues and decreases the number of parties affected in the event that a compromise occurs. Domain operators have the power to advertise new trust anchors or revoke ICAs as they see fit which adds trust flexibility not seen within the current PKI system. This however lead to the question of how to ensure that domain operators are capable of securing themselves against attacks since they would then become the new targets in this proposed system and would face many of the problems currently seen by the ICAs. Also, while internal domain operations may have organizational trust preferences, any outsourced domain management will have to establish controls around this new trusted role.

There is also the issue of DNSSEC deployment to consider in implementation costs. Updates for servers would be required to support the new DNS information and client applications would also need to be configured or rebuilt to take advantage of the security offered by DNSSEC. While the instantiation of DNSSEC is something that can be seen as good for all it is still a considerable hurtle which must be met before DANE can be considered a viable alternative system. Some have pointed out that there may be added latency to the TLS connection process since the certificate chain must be verified using the DNSSEC signatures and a trust anchor identified.

There are a number of other considerations that must be taken into account. For instance, in the DANE implementation, trust is embodied into a single DNSSEC root – potentially creating a single point of failure for the entire system. The existing ICA system mitigates this by having multiple trust anchors – although some claim that too many is the problem here. Also, DANE can only certify trust for the domain – that is both an advantage and a disadvantage. It is limits the scope of trust just to the domain in question, but it is a potential disadvantage when compared with ICAs services that do more than just domain validation (e.g. EV and OV validated credentials).

Implementation. There exist some prototype deployment tools for DANE including a client-side implementation on Google's Chrome browser and server-side tools which generate DANE records and DNSSEC-stapled certificates. The difficulty will arise in building a prototype in an environment which as of yet does not support DNSSEC. More research must be done in exploring the costs necessary to implement these changes to the DNS infrastructure.

2.5 Public Key Pinning

Overview. Public Key Pinning [10] places web hosts in charge of informing clients (referred to in this section as User Agents or UAs) on which certificates they should expect to see in the host's certificate chain. When a UA receives response from the host, it validates the *pinning header* and, upon successfully meeting the required criteria (described in the next section - Components), stores the *pinning metadata* which is composed of the pin data and the max-age. Pin validation works by computing the fingerprints of the Subject Public Key Info (SPKI) structures in each certificate in the host's validated certificate chain, checking to ensure that the set of fingerprints intersects with the set of fingerprints in the host's pinning metadata. These procedures are intended to help hosts assert their cryptographic identity and to detect imposter pinned hosts with the goal of preventing MITM attacks. It also means

that domains will be allowed to white-list the public keys of specific ICAs. If a certificate is found in the chain that has been issued by an ICA not on the list (i.e. the pin is not found) then the UA will fail during pinning validation and the connection will be refused.

Components. Web hosts attach a new HTTP header which comes with SPKI structures containing the public key cryptographic hash for a certificate in the host's certificate chain and a backup public key crypto hash which is unique from any SPKI in the certificate chain. These SPKI hashes are called *certificate pins* because they pin the certificate (or more correctly, the public key contained in the certificate) to the site in question.

The SPKI is verified by the UAs using the certificate chain presented by the web host. The process includes the following steps and constraints: The UA must receive the public key pins in the header field over an error-free TLS connection. When the UA receives the pins, the connection will be authenticated if the SPKI structures in the certificate chain overlap with at least one of the given fingerprints. The set of pins must also contain at least one pin (the backup pin) which does not match a SPKI structure in the certificate chain. If any of these criteria are not met then the UA will discard any previously saved pinning metadata and the connection will be refused. However, if everything is validated successfully, the UA will set the host as pinned host and store the pins.

During the pin validation process itself, the UA computes the fingerprints of the SPKI structures in each certificate of the web host's certificate chain. It then checks that this set of fingerprints intersects with the set of fingerprints in the host's pinning metadata. If this intersection occurs, then the UA continues with the connection as normal. Otherwise, the UA treats this as a pin failure and the user is denied access to the site.

In the event of inadvertent pin failure, rather than deny the user access, the backup pin can be used. This backup pin is kept by the operator and it is a fingerprint for the public key of a secondary, not-yet-deployed key pair. This key pair is kept offline but a pin is set for it in the pinning header. In this way, if a failure occurs the operator can deploy the key pair and the UA can validate the backup pin since it was set in a previous valid pinning header.

Analysis. Public Key Pinning takes a different approach to validating cryptographic identities by hashing the public key (or more specifically, the SubjectPublicKeyInfo) rather than hashing the certificate. This has an obvious advantage over hashing the certificate because ICA certificates are often reissued and different versions may be substituted for the one you expect. Since public keys are reused in certificates it makes more sense to hash this information which you know must be correct. However, to prevent attackers from using the public key in ways where the client would interpret it incorrectly, it is necessary to hash the SPKI structure rather than the public key bit string which takes into account type, parameters, and extensions as well as the public key itself.

There are still some use cases to be addressed with Public Key Pinning as noted by Chrome team developer Adam Langley. The client can't pin to a cross-certifying root without potentially erroneously rejecting or validating pins [6]. As an example, suppose DigiCert's root is cross-signed by an early Entrust root to facilitate trust in older clients. Those legacy UAs which don't recognize the DigiCert root are still able trust those certificates because it can trace a trust path back to the legacy Entrust root. However, you wouldn't want to pin the legacy Entrust root because newer clients will stop their chain at the trusted DigiCert root.

Another issue with the system is a bootstrap problem. Browsers can be fooled if the user is visiting a website for the first time because they have no stored pinning metadata. So it is necessary in this case that all pins are preloaded with the browser. However, as noted by the authors of the system, this creates scaling issues and, as of the latest draft, this scalability problem is still an outstanding issue that is yet to be dealt with.

Public Key Pinning locks out users in the case of failed validation which, in the event of an actual attack, does prevent users from simply clicking through security warnings. While it is easy to see how this default is beneficial to users and how it will protect them from attackers, it does lead to lock-out scenarios. In the case of poor management from hosts or attacks which cause users to pin SPKI when they don't have the corresponding private key pair, pin validation failures make it impossible for the user to access the site. In the latter situation, the user becomes locked out until the max-age expires and the UA no longer recognizes the pinned host.

Lastly, the authors raised a valid concern when they observed that deploying the system will require "operational and organizational maturity" since hosts will need to keeping track of which SPKIs they are pinning and whether or not the pins are still valid [10].

Implementation. Google's Chrome browser has experimented with shipping user-extendable embedded pin sets. The Chrome 13 release also supports HTTPS pins for most Google properties and has released a list of whitelisted ICA public keys currently accepted by the Chrome browser.

2.6 Sovereign Keys

Overview. The Sovereign Keys [8] system implements a persistent, secure association between Internet domain names and public-keys called Sovereign Keys. Sovereign public keys can be registered for a particular service under a particular domain. Clients using that service must verify the operational public keys (such as those at the end of X.509 certificate chains) have been cross-signed by the Sovereign Keys. If verification fails, the client will terminate the connection after informing the user with an appropriate message.

Components. Sovereign Keys uses *timeline servers* which are responsible for storing the mappings between domain names and sovereign keys. They are read-and-append-only data structures so the history of sovereign keys for a domain name is maintained and previous claims cannot be overwritten or changed. Timeline servers maintain public and private key pairs which are used to authenticate themselves and are shipped with the client software.

Issuing a sovereign key for a domain requires control in the DNS (either an ICA signed certificate or a key published by DANE for example) and timeline servers need an OCSP response to verify the status of such ICA signed certificates before adding a sovereign key to that domain name. Clients cannot query timeline servers directly; rather they use *mirrors* which maintain copies of the records on timeline servers.

Mirrors keep full, up to date copies of all the trusted timelines and are identified by IP address, port number and a public key. Clients can query any of the mirrors and should cache the response and must check the consistency of the response. Monotonicity must be preserved meaning every signed entry with higher serial number must have higher timestamp than the previous entry.

When a domain owner registers a new name, it is recorded in one timeline server which will contact other timeline servers so that they can incorporate a copy of the registration. This registration is cached by the servers in *reference entries*. To add a sovereign key, as mentioned previously, an ICA signed certificate or some form of name-domain association must be presented so the sovereign key can be issued. In the case of a certificate, a timeline server will verify its revocation status via OCSP before issuing the sovereign key.

The key also contains protocol, port, and wildcard fields which are used to indicate which protocol the service is meant for (HTTPS, SMTPS, etc.), what port it is working on, and whether or not this key is valid for any subdomain. In order for a sovereign key to be revoked, the key holder must present the appropriate serial number, sovereign key name, and sovereign key.

Sovereign Keys also provides methods for detecting compromised mirror or timeline servers. A server will be identified as compromised if a mirror or client sees two different signed entries on the same timeline, if an entry violates the monotonicity property, or if an entry for a new sovereign key is not signed by one of the timeline's root ICAs or the DNSSEC root (if DANE is being used). *Renegation* of this timeline server would occur, meaning that trust in the server would be revoked. A copy of the contradictory timeline statements would be stored, clients would cease to trust that timeline server, and the reneged information would be added to a list of bad timeline servers which can be queried by other clients.

Analysis. The Sovereign Keys design draft ignores some important details with regards to organization and setup. It is not clear who will pay for and manage the timeline servers and mirrors and what kinds of additional hardware and software will be required in order to support the high availability service on which internet trust will be dependent. This could involve the cooperation of ICAs or potentially the formation of another TTP. The set of timeline servers is assumed to be small which gets away from the decentralized model and marks the timeline servers as prime targets for attacks. Because of this, most of the client's trust will be placed in the party or parties responsible for managing the timeline servers.

The proposal's goals are in line with addressing concerns about the current system. Namely, the Sovereign Keys model strives to reduce the number of attack points so that compromise can be more easily detected and quickly dealt with. Rather than relying on ICAs, automatic security mechanisms will identify security breaches and compromised servers and ultimately take actions to protect clients and alert TTPs.

While Sovereign Keys has these aforementioned advantages and while it decreases a user's reliance on ICAs, it requires many changes to current SSL/TLS implementations. Moxie Marlinspike, the creator of Convergence noted that deploying Sovereign Keys would "require a major internet migration, changing both the way that every webserver deploys SSL today, as well as the way that every SSL client processes server certificates. [16]" Sovereign Keys would require the integration of new client technologies which browsers may be unwilling to implement. It would also necessitate changes on the side of the web hosts who would have new commitments and responsibilities.

Sovereign Keys does not negate the need for the use of ICA run OCSP responders since timeline servers will need to check the revocation status of the certificates. It would also appear that DNS updates would be needed in order to know how to request and utilize data from timeline servers.

Mirrors are to be used to improve performance and reliability of the Sovereign Keys system. They are expected to be bootstrapped using a bulk data distribution mechanism such a BitTorrent. There is some concern about the size of the mirror server response which is expected be fairly large. As a result there would be increased latency in the TLS connection.

Along with querying mirrors about sovereign key information, clients will be expected to query servers for a list of the mirrors and timelines which have been reneged (i.e. are no longer trustworthy). Renegation has its own set of issues and renegation tracking and synchronization will need to be carefully addressed to ensure clients are not trusting compromised servers.

As with Perspective and Convergence, because Sovereign Keys uses mirrors as third parties for verification there is the problem that these mirrors are in the position to learn the client's IP address the service or site which is being queried. A proxy is recommended to prevent the possibility of mirror leaking a user's browsing history.

Other concerns include the embedded keys for timeline servers in clients which may lead to update issues as timeline servers are added, changed or reneged. The description does not seem to provide an easy way to manage or trust timeline servers which are the backbone of Sovereign Key's trust architecture.

2.7 CAA Record in DNSSEC

Overview. The Certificate Authority Authorization (CAA) DNS Resource Record allows a domain owner to choose which ICAs should be authorized to issue certificates for that domain. Under this system, ICAs must follow the policies defined in the *certificate policy statement* in order to issue certificates. The CAA records will be the basis for an ICA's validation requirements. The system mimics DANE's design in that a domain publishes a record in DNS restricting which ICAs can issue certificates for it. However, rather than supplying this information to a client, the record is intended for use by the ICAs who will be responsible for checking the record and only issuing a certificate if their name is on the list. This setup would require the ICAs to honor the CAA records and behave correctly. This system also relies on web clients trusting ICAs.

Components. Before a certificate can be issued, *certificate authority processing* is necessary whereby an ICA checks for a CAA record and verifies that the certificate request is consistent with the parameters of the record.

The *CAA resource record* publishes two pieces of property information about the domain, the *issue* property and the *iodef* property. The issue property grants authorization either to the domain holder or an authorized party acting under the domain owner to issue certificates. The iodef property specifies a URL which may be used to report violations of certification policy or issues regarding certification practices. The records are published at the public delegation point, meaning that the policy covers all subordinate domains as well. Additional properties may be included which could, for example, specify the certification policy or authentication process to be used.

Each CAA record contains a [tag, value] pair where each tag represents a property of the record and the value is the value specified for that particular property. Multiple properties may be specified for a domain and a property may even be specified more than once. The CAA uses the hash of the certificate and can specify that any certificate issued for a domain must be rooted by the hashed certificate.

While DNSSEC is recommended for additional security it is not a required part of the CAA record implementation. Along with increased security, DNSSEC would also allow for an issuer to have non-repudiable proof of their authorization. Where DNSSEC data is available, the issuer should archive all records for audit purposes.

Analysis. While CAA records seek to apply constraints to which ICAs may issue certificates for a domain, this goal cannot be sufficiently enforced and misuse cannot be adequately prevented. ICAs may ignore the records and publish certificates anyways. However, it is indicated by the authors that this would be looked down upon and may motivate others to exclude the ICA from their list of trusted roots. It should also be noted that using CAA records does nothing to prevent misuse by an ICA who is authorized to issue certificates. In order to prevent against this type of misuse, it is advised that domain holders take care to choose ICAs who employ appropriate security measures and are in good standing. This puts extra work on domain owners who must make good trust decisions when choosing which ICAs to authorize.

Other concerns to be addressed are CAA record spoofing which could allow an attacker to obtain a certificate for a domain from an unauthorized ICA. This would have prolonged consequences since certificates can be valid for up to several years. DNSSEC is recommended as a security measure to prevent spoofing but this falls back on the same predicament faced by systems such as DANE.

Overall, the CAA records proposal has similar aims to DANE with the advantage that DNSSEC is not required (though it is recommended) for deployment. However, while allowing domains to restrict which ICAs are authorized to issue certificates does decrease the opportunity for misuse or at least limit its scope, the system lacks an effective means of enforcing CAA processing. It also puts more pressure on domain owners to make objective decisions on which ICAs are more trustworthy and less open to compromising attacks. ICAs which are already acknowledged by the community to have weaker security would most likely ignore CAA record constraints given that it would be unlikely that there would be negative consequences. Since, CAA records basically require that ICAs check themselves (whereas DANE leaves this task to the client), domain owners are required to trust on good faith that ICAs will follow through with the recommended CAA processing before issuing certificates.

Assuming that ICAs do follow the guidelines of the system, CAA records must still wrestle with cases of record spoofing and certificate misuse. In the event that an authorized ICA does become the victim of certificate misuse, domains would have the option of excluding the ICA in the future but the system does little to prevent ICA compromise in the first place. Instead, domain owners must guess at which ICAs are most trustworthy and then hope that other ICAs will respect the restrictions of the published records.

One possible enhancement to the use of CAA records is to develop a service where clients periodically verify an ICA whitelist to identify which ICAs are known for following the guidelines. This whitelist would require some entity to systematically check CAA against certificates issued in the referenced domains to discover a breach of protocol and then have a mechanism of publishing this to the whitelist. This could be an optional addendum for the protocol, as it requires establishment of a protocol, operation of a new service, and modification to web clients in order to take advantage of it. Alternatively, browser vendors could provide this service to their customers by making audit and compliance a required aspect of their ICA trust distribution mechanism.

One strong advantage of this approach over others, is that (assuming the optional whitelist service as detailed in the paragraph above is not used), then there are no changes required by web clients and CAA can be adopted immediately. However, this approach still relies upon trust in ICAs and not all of them have demonstrated a capacity to consistently implement secure processes in the best interest of the entire community.

2.8 Certificate Transparency

Overview. Certificate Transparency is the system proposed by Google to prevent certificate abuse and it relies on the idea that every certificate should be published in an audit log which is publicly available. There is a certain amount of cooperation between parties which is necessary with such a system since ICAs must have published accompanying audit proofs for each certificate they issue. The incentive for them to publish this information stems from the fact that clients will not accept certificates if they do not collude with any of the ICA's audit logs. The system strives to ensure that no certificate will be issued for a domain without the domain owner's knowledge. It also eliminates the client's need to solely trust ICAs since a certificate may be recognized as fraudulent using information from the logs.

Components. Certificate Transparency relies on the idea that certificates should be published and has proposed a system where audit logs, audit proofs, and Merkle trees are used to look up and crosscheck certificate information to ensure their validity.

The *audit logs* would serve as method for cross validation so that a client, after receiving a certificate, can check with each ICA's audit proof to ensure that the certificate has been published. There are three possible scenarios if a client encounters a bad certificate. In the case that a client cannot find the certificate information published in the ICA's log or in the public log, it will reject it. If the client finds that the certificate has been published, it will accept it but if the domain owner sees that the certificate is not valid they can take measures to have it revoked. In the case that the client finds that the certificate colludes with an ICAs audit log but the certificate is not published on the public log, the client will accept the certificate. While it is not immediately obvious that the certificate is not valid, it should soon become apparent that the ICA's audit proof does not correspond with the public log and the certificate can be revoked.

The logs would be read-and-append-only and with each entry the log is signed using a signed Merkle tree. Browsers will check the Merkle proofs to verify that certificate is valid (that is, that the certificate has been published in the audit log). Each certificate is also accompanied with an *audit proof* which includes the Merkle

signature on the top hash along with the list of hashes from the top of the Merkle tree down to the particular certificate. The Merkle tree is a type of hash tree where nodes higher up on the tree are the hashes of their children. In this case, the leaves of the Merkle tree will be the hashes of the certificates which need to be signed and the root will be the hash over all the children. For each certificate, the signature includes the root hash value along with the path from the particular signature up the tree to the root.

Analysis. Certificate Transparency requires additional accountability and cooperation from ICAs in order to ensure that certificate information is available publicly. This keeps domain owners aware of the certificates which have been issued for their domain and helps protect users from accepting fraudulent certificates. However, these goals do beg the question of how to ensure that ICAs will publish the certificates that they issue and how much information this public log should hold so as to be useful but not overwhelming.

The proposed idea is that ICAs will have the incentive to publish their certificates in the case that, with the widespread implementation of the Certificate Transparency system, clients would otherwise reject the certificates if they are not found in the audit log. As a result, domains would favor ICAs which published their certificate information. ICAs which fail to publish would risk losing business to those ICAs which follow the system's model.

The other component to consider is the audit logs. Research is being done now on determining the appropriate durations for each timeline segment in the log proofs. It is also yet to be decided if each ICA should have its own public audit log or if there should only be one log for all certificates. It is also uncertain as to who will be responsible for the logs, whether it be the ICAs or an independent party. There is some expectation that a third party will be created to monitor the logs which raise the question of how this entity will be organized and how we can ensure that they are trustworthy.

The system also relies on the expectation that domain owners will monitor the logs for certificates which should not have been issued for their domains. This places more responsibility on domain owners and requires that clients trust that domain owners won't be negligent.

Revocation is another issue to be considered in light of the shortcomings of the current system. Certificate Transparency does not have any specific revocation scheme defined though it has been suggested that a separate audit proof be created which contains a list of revoked certificates.

However, the system does address the concerns of recognizing fraudulent or unpublished certificates. It should be noted that there are cases where a bad certificate may be accepted over a short period of time, specifically in the case that a log provides a fake audit proof to an ICA. Although the hope of the project group is that the opportunity for this type of compromise is severely limited due to the fact that once a log is recognized as compromised it will not be trusted in the future.

Scaling will be another consideration to remember as the project progresses. This issue is tied in with the size and availability of the logs and whether or not it is reasonable to assume that they can be sufficiently monitored.

There is currently no mention of how self-signed certificates will play into this design as ICAs are the ones responsible for publishing certificate information.

Implementation. The Certificate Transparency team is currently working to provide code for the system but it is still very much a work in progress. For now it is helpful to monitor the project's wiki page for updates and information on a working prototype [11].

3 Metric for System Analysis

3.1 Metric Overview

As part of our research into these proposed alternatives, one of the goals was to construct a new metric which would allow us to make a fair comparison between the systems based on well-defined criteria of high importance. This section will define each category for comparison, apply the metric to the individual proposals, and conclude with a summary of the outcome as well as a proposed path for the future of PKI. In addition, the proposals will be judged based on current PKI practices and concerns, evaluating how well the system addresses these existing problems and what trade-offs, if any, must occur to remedy them.

The ranking system will be applied to each category and the scores fall on a scale of 1 to 10 with a mark of 10 indicating that the system successfully meets all the requirements for the category and a mark of 1 indicating that the system either did not address the use-case or that it failed to meet any of the goals defined for the category. The scores will be accompanied by a detailed description outlining the reasons for the given score.

3.2 Categories

Each of the proposals can be organized into four distinct categories on the basis of trust infrastructure. That is, the requirements of the systems are broken down by examining the fundamental changes which must be made to the existing architecture and identifying the trust base mechanism for each proposal. Using this guideline, we define the following categories:

3.3 Categories based on Trust Infrastructure

I. **Trusted Third Parties.** The proposals outlined by **Perspectives**, **Convergence**, and **MECAI** require the implementation of an additional trusted third party to the PKI system, to either work alongside or replace ICAs. These three systems have architectures built around "notaries" or "notary authorities" which provide cross-validation of a server's public key.

II. **DNSSEC.** The proposals outlined by **DANE**, **Sovereign Keys**, and **CAA Records** each rely on DNSSEC deployment as either a critical or highly recommended component of their trust architectures. These systems rely upon the DNS hierarchy which ultimately rests upon the DNSSEC root in order to derive trust. Responsibilities for asserting trust and binding identities would shift from the ICAs to the domain operators in the case of DANE and Sovereign Keys but remain with ICAs for CAA Records.

III. **Pinning.** The proposals outlined by **Public Key Pinning** and **Sovereign Keys** rely on web service hosts to provide an alternative form of attestation in addition to certificates issued by ICAs.

IV. **Transparency.** The proposals outlined by **Certificate Transparency** and to a lesser extent **Sovereign Keys,** rely on additional methods of trust verification such as published audit trails. These systems work with the existing ICA infrastructure to develop additional measures for preventing and detecting certificate misuse.

3.4 Criteria

1. **Resource Availability and Defined Business Case**. There must be resources available with a defined business case to form and operate the trusted third parties (TTPs).
2. **Changes for Deployment and Operation**. The proposed system should minimize changes to the experience of actors within the existing system (ICA practices changes are more favorable than web host changes, which are more favorable than web client changes).
3. **TTP Trustworthiness and Security**. TTPs must be trustworthy and employ good security practices.
4. **Scalability**. The system must scale.
5. **Latency.** The security mechanisms of the system must not cause significant latency.
6. **Compromise Detection and Response Quality**. Clients must be able to identify compromise and act accordingly.
7. **Trust Revocation and Flexibility**. Clients must be able to revoke trust and users should have more control over their trust anchors.
8. **Protection Offered by System Defaults**. Default implementations should improve the flexibility/capability/protection of the majority of web users.
9. **DoS Prevention and Distributed Trust Architecture**. The system must guard against DoS attacks in the event that a TTP is compromised or unresponsive to client requests. It should also not create a single point of failure.
10. **MITM Attack Prevention, Identification and Response**. The system should address the MITM problem of the current system by reducing the probability of this event or increasing a user's likelihood of identifying when they are under attack.
11. **User Privacy**. User privacy must be protected.

4 Trusted Third Parties Score Evaluation

4.1 Perspectives.

1. **Resource Availability and Defined Business Case:** Perspectives will need to acquire more notary servers before the system can be widely used. There are currently 8 notary servers available for public use with Carnegie Mellon acting as the notary authority for the group. Setting up more servers will have both software and equipment costs with no indication of who will pay for the servers (most likely the notary authority). **Score: 4**
2. **Changes for Deployment and Operation:** Perspectives requires organizational changes and the establishment of third parties. Client browsers will need to have lists of notary IPs and software changes as part of the deployment process. **Score:4**
3. **TTP Trustworthiness and Security:** With Perspectives, anyone can set up a notary server. However, it is up to the Notary Authorities to advertise trustworthy notary servers to clients. We cannot count on individuals who are running the servers to use good security practices. Instead, users must rely on the Notary Authorities as well as feedback from the system to make good trust decisions and identify compromised notaries. There is a fair amount of uncertainty surrounding the trustworthiness of the system and much of the trust policy rests on the user's shoulders – not an ideal design for the average user. **Score:2**
4. **Scalability:** Perspectives notary servers would ideally be highly distributed among the internet. There is the concern that the notary servers may end up being run by only a few large companies or organizations which leads us back to the problem of we face with ICAs, where instead we are forced to trust the notary authorities. There is also the question if the system will scale. The number of notary servers will need to grow as the user base grows otherwise this will put a lot of stress on existing notary servers and they may be unable to deal with the number of client queries. **Score: 4**
5. **Latency:** Communication between clients and notary servers may add latency to the connection. *n* Notaries will be queried in parallel with only *m* replies needed ($m<n$) so as to decrease wait time. There is also an opportunity for caching to be used so that notary servers would be able to send a cached observation of a server's public key instead of having to open a new connection to the site each time a different client sends a query for the same site. **Score: 6**
6. **Compromise Detection and Response Quality:** Notary servers will be looked after by Notary Authorities. Each server keeps an append-only log which can be cross-verified by log copies stored on other servers (shadow servers). This provides a method for auditing notary servers and detecting compromise. Clients will be responsible for notifying Notary Authorities in the case that the logs do not collude. The Notary Authorities themselves are currently treated as trusted entities and there are no details on attack possibilities which might originate at this level. **Score: 5**
7. **Trust Revocation and Flexibility:** Notary Authorities are also responsible for sending out information about which notaries are trustworthy. Once the Notary Authority is alerted of a comprised notary server, they can send out this information to the clients and the notary's IP will be removed from the list of notaries a client can query. The timeframe for this information to be relayed to clients is dependent on whether or not a notary server pushes this information to the client or if the client must query a notary server for the updated list. Meanwhile, users have the option to alter their trust policy by altering the quorum and quorum duration values (defined in the Perspectives section above). However, they do not have the capability to specifically indicate to the client which notary servers or notary authorities to trust. **Score: 6**
8. **Protection Offered by System Defaults:** The defaults for perspectives are intended to be adequate for the average user. The quorum and quorum duration settings are not aggressive and there need be no interaction on the user's part to make trust decisions. The client makes observations based on notary server responses and takes appropriate action to ensure that the server's public key is valid before allowing the user to connect to the site. **Score:7**
9. **DoS Prevention and Distributed Trust Architecture:** Perspective's design is dependent on notary feedback which means it is sensitive to DoS attacks in the event that notaries are compromised or become unresponsive. For client settings which require a high amount of notary collusion (high quorum), unresponsive notaries can be crippling and will keep users from accessing legitimate sites. In the case that a few large Notary Authorities become responsible for a majority of the notary servers (which would seem likely despite Perspective's vision of a distributed architecture), we are essentially back to the problem where users are required to put their trust in a select few. So these Notary Authorities, much like the ICAs of the current system, would become the targets and a compromised Notary Authority would lead to compromised notary servers and potentially widespread DoS. **Score: 3**
10. **MITM Attack Prevention, Identification and Response:** In the event that an attack is detected, the connection will be closed. Rather than displaying a warning dialog the page will fail to load and instead the

user will see an error message. This prevents the user from bypassing the warning. Perspectives does not so much reduce the probability of a MITM attack as it increases the chance of such attacks between detected. **Score:5**

11. **User Privacy:** Notary servers will be sent requests from clients about specific sites so servers will have access to both client IP data and the services requested. This opens up the possibility that compromised notary servers could reveal a user's browsing history. The proposal recommends the use of proxies if the user wants privacy protection. **Score:2**

4.2 Convergence

1. **Resource Availability and Defined Business Case:** Because Convergence is an extension of the Perspectives design, it has the same third party concerns as described in the Perspectives scoring section. **Score:4**
2. **Changes for Deployment and Operation:** Convergence requires the same basic changes in organization and trust architecture as Perspectives. While Convergence does advocate that the ICA party is retired in favor of the notary system, it does indicate that Convergence can be made to work alongside ICAs. **Score:3**
3. **TTP Trustworthiness and Security:** The TTP security concerns for Convergence are identical to those outlined for Perspectives. **Score: 2**
4. **Scalability:** Convergence seeks to achieve the same highly distributed architecture as the Perspectives project. There are no remarkable differences between the systems in this category so the scores are the same. **Score:4**
5. **Latency:** Convergence and Perspectives share the same notary query protocol with the exception that Convergence introduces bounce notaries as middlemen for privacy reasons. This additional step in the protocol will slow down the response speed of the notaries. However, notary servers in the Convergence design will use caching which will decrease response time and improve on the latency foreseen in the Perspectives project. **Score:7**
6. **Compromise Detection and Response Quality:** Convergence uses the Perspectives protocol for identifying compromise (i.e. comparing logs with shadow notaries). **Score:5**
7. **Trust Revocation and Flexibility:** Convergence is concerned with "trust agility" and so the user is in complete control over which notaries are in the client's trust list. This allows the user to revoke trust of any notary server at any time. If a user decides that they don't want to trust notary servers run by a specific Notary Authority they may choose to remove them from their list of trusted notary servers. With the exception of some power users, it seems highly unlikely that the average user would be well-informed, or interested enough to evaluate the security of a Notary Authority or individual notary server. In can be argued that Perspective's advantage is that it will never rely on the user to know which notaries to trust but rather employs mechanisms to report suspicious notary behavior and to remove compromised notaries from the client's trust list. Convergence's advantage is that third parties which may be considered less trustworthy can be manually taken off the user's trust list giving third parties incentive to employ better security practices. The usability is still a major concern, however and in the event that notary servers are run by a handful of Notary Authorities, blacklisting one will severely decrease the number of notary servers available to a user. **Score:4**
8. **Protection Offered by System Defaults:** Convergence expects its users to take a more active role in defining their trust policy and picking their trust anchors. Simply relying on the default implementation means that most web users will not take advantage of the trust flexibility offered to them by the system. Instead it will function much like the Perspectives systems. **Score:6**
9. **DoS Prevention and Distributed Trust Architecture:** Convergence's DoS concerns are identical to those faced by Perspectives. **Score:3**
10. **MITM Attack Prevention, Identification and Response:** Like Perspectives, Convergence looks to raise awareness of MITM attacks. Rather than decreasing the likelihood of such attacks, the notary system should allow for better attack detection. The current documentation for the system does not specify how Convergence will interact with the user when an attack is detected (i.e. if the user will be allowed to bypass the warning, if the attack with by automatically circumvented, or if the detected attack will cause the connection to simply fail). **Score: 4**
11. **User Privacy:** Convergence seeks a solution to Perspectives' privacy issue and address the problem by introducing bounce notaries. These bounce notaries prevent notary servers from being able to view both a

client's IP and the service they requested. The bounce notary will know the client's IP whereas the notaries querying the server for the public key will only know the service/site that was requested. **Score:7**

4.3 MECAI

1. **Resource Availability and Defined Business Case:** The MECAI proposal places ICAs in the role of Vouching Authorities. ICAs will be responsible for the notaries or Vouching Authority Servers. The proposal makes use of existing resources and organization within the current system. However, it will require that ICAs take on additional responsibilities. Vouchers will have to be issued to clients in high quantity by the CAs and the CA issuing the voucher must be different from the one who issued the server certificate. CAs will be required to work with and trust other ICAs in the voucher verification process. So while ICAs have the resources to implement the infrastructure, the business case to do so is not well defined from the ICAs perspective. **Score:6**
2. **Changes for Deployment and Operation:** MECAI's design will require ICA practice changes as well as the establishment of vouching authority servers. The proposal indicates that vouching authorities could be made up of a group of ICAs which would require ICA collaboration and cooperation. Browsers will need software changes to incorporate the MECAI client and there may be changes needed to website servers as well in order to handle voucher information that will be passed along during the handshake. There are some significant organizational challenges to overcome as well as software changes on the client-side and server-side. **Score:3**
3. **TTP Trustworthiness and Security:** Because ICAs would be acting as VAs and would have control over the notaries, this design is more advantageous from a trust perspective than the Convergence and Perspectives systems. However, compared to today's system, the ICAs' level of trust would remain the same. **Score:5**
4. **Scalability:** MECAI does not have any significant scaling issues from an implementation perspective. However, because a new voucher must be issued for every client request to a VAS, the sheer numbers of vouchers which will have to be continuously issued may prove to be a difficult task for the ICAs who will be responsible for running the VASs. **Score:5**
5. **Latency:** The handshake extension proposed by MECAI could cause significant latency in the connection. Clients must also be able to obtain a voucher during the client-server SSL/TLS connection which requires ICAs to have fresh vouchers readily available for clients to validate the server certificate. This requires that ICAs (acting as VAs) keep up with the demand for vouchers. Between ensuring that vouchers will be readily available, transmitting the voucher to the client and checking the voucher certificate information with the one received by the server, MECAI's design struggles with a considerable challenges regarding latency. **Score:2**
6. **Compromise Detection and Response Quality:** In the event that the client receives a fraudulent or expired certificate from a server, the voucher verification process will indicate that the site's certificate does not match with the expected certificate (the one indicated in the voucher). MECAI also uses Vouching Authority Servers which listen for client requests and create and sign vouchers. VASs will be notified in the even that a ICA is compromised so that they will stop vouching certificates issued by that ICA. MECAI does not provide any protocol for detecting a compromised VAS. However, clients will maintain a list of trusted VAS IP addresses, information which is provided to them by the Vouching Authority. In the case that a VAS is no longer trustworthy the client's trusted VAS list must be updated. **Score:6**
7. **Trust Revocation and Flexibility:** As mentioned in the previous section, in the case that a ICA is compromised, VASs will have to be notified so they will no longer vouch for certificates by that ICA. This detection process may involve a domain owner reporting a rogue certificate to an ICA realizing that fraudulent certificates have been issued based on voucher verification failures. There will be a delay which is dependent on the validity period of the vouchers until client software can be updated and trust revoked. During this period, until the rogue certificates have been revoked, vouchers will no longer be produced for certificates issued by the affected CA. This means that clients would not be able to access sites for whom the ICA has issued certificates until the rogue certificates are revoked. **Score: 4**
8. **Protection Offered by System Defaults:** The system does not require user interaction or feedback for making trust decisions. While users will not be able to pick and choose their trust anchors, MECAI's protocol will essentially blacklist ICAs which have been compromised and quickly revoke fraudulent

certificates once they have been detected. The system provides users with protection without requiring/recommending user interaction like Perspectives and, to a greater extent, Convergence. **Score:7**

9. **DoS Prevention and Distributed Trust Architecture:** Since clients are dependent on vouchers from the VASs, in the event that the servers are unresponsive or compromised a client would be unable to perform certificate verification and the connection would be closed. Users will also be denied access to sites in the event that an ICA which is responsible for issuing certificates for those specific sites, is blacklisted by the VASs due to compromise. MECAI has yet to develop an emergency backup plan for such a scenario. **Score:3**
10. **MITM Attack Prevention, Identification and Response:** As with Perspectives and Convergence, MECAI's notary system works better at detecting MITM attacks rather than preventing them. ICAs will still be open to the same attacks they face with the current system. MECAI also seeks to prevent users from simply clicking around the security warnings and instead kills the connection and fails to load the page. **Score:5**
11. **User Privacy:** MECAI does not discuss any issues with regards to user privacy. From the description, it seems valid to explore if VASs, which receive client requests for specific services (i.e. they have both the client IP and the request information), would raise privacy concerns regarding a user's browsing history. **Score:1**

4.4 DANE

1. **Resource Availability and Defined Business Case:** The role of domain operators in DANE's proposal requires them to take on additional responsibilities, including advertising trust anchor information to clients. ICAs would be required to work alongside domain operators to detect rogue certificates and compromise. There is some concern as to the difficulty of establishing domain operators in this new role and DANE does not clearly define the policies, regulations or business case of this new organization. **Score:3**
2. **Changes for Deployment and Operation:** The shift of trust from ICAs to domain operators will most likely be met with resistance. DNSSEC deployment will also mean many changes must be made to both client-side and server-side software. **Score:2**
3. **TTP Trustworthiness and Security:** Domain operators would be expected to be able to guard themselves against attacks and to detect certificate misuse. As well as these security considerations, other issues arises in the case that domain operations are outsourced to another entity. DANE would require a reorganization of the current domain operator policies and potentially the creation of new regulations to ensure good security. The security practices of the ICAs however, will not be much different than they are in the current system. **Score:3**
4. **Scalability:** DANE is entirely dependent on DNSSEC deployment. All valuable information and security would be lost if the system were to encounter servers or clients only using standard DNS software. As it stands, the deployment challenges are considerable for DANE because of its reliance on DNSSEC. **Score:2**
5. **Latency:** It has been pointed out that there may be added latency to the TLS connect process since the certificate chain must be verified using the DNSSEC signatures and a trust anchor identified. We can expect, with the additional verification requirements that this protocol will potentially slow down the connection. **Score: 6**
6. **Compromise Detection and Response Quality:** In the DANE design, domain operators will know exactly which ICAs they have requested certificates from and which specific certificates they have received. Domain operators will be responsible for identifying rogue certificates and for advertising new trust anchor information to clients in the event that an ICA is compromised. **Score:5**
7. **Trust Revocation and Flexibility:** Domain operators will be responsible for revoking client's trust anchors. The user will not have control over picking their trust anchor. **Score:4**
8. **Protection Offered by System Defaults:** DANE gives domain owners the flexibility/capability to choose which ICAs can issue certificates for that domain. Domain owners have the ability to revoke this trust at any point in time, and this information is advertise to client through the use of TSLA records. Users are still forced into trusting ICAs and now domain operators to make good decisions and to keep themselves secure. **Score:5**
9. **DoS Prevention and Distributed Trust Architecture:** With DANE, trust is embodied into a single DNSSEC root – potentially creating a single point of failure for the entire system. **Score:2**

10. **MITM Attack Prevention, Identification and Response:** DANE verifies a server's public key using the DNSSEC trust chain until it identifies a trust anchor. While DANE does not necessarily decrease the chance of a MITM attack, it makes detection possible. Since domain operators will be in charge of knowing the certificates issued to their domain they should also be able to more easily and quickly identify rogue certificates than ICAs in the current system. **Score:7**
11. **User Privacy:** DANE's proposal does not address user privacy concerns. However, since DANE uses DNSSEC, the information passed between the client and the server would be secure. User privacy should not be a concern. **Score:9**

4.5 CAA Records

1. **Resource Availability and Defined Business Case:** CAA Records relies on the existing ICA infrastructure as well as domain operators. The resources are available however both parties will required to adopt additional responsibilities and practices to support the system. **Score:7**
2. **Changes for Deployment and Operation:** ICAs will be required to check CAA resource records using a procedure called Certificate Authority Processing before issuing certificates for a domain. ICA practice/procedure changes will need to be made so that ICAs follow the record constraints published by domain owner. Domain owners will need to take a more active role by publishing CAA resource records to grant authorization to specific parties for certificate issuance. Because ICAs are responsible for checking record properties before issuing certificates, no client changes will be necessary. **Score:6**
3. **TTP Trustworthiness and Security:** Domain owners will need to use good security practices in order to prevent fraudulent records from being published. ICAs will also need to take responsibility for checking CAA records and adhere to the property information and constraints published for the domain. Essentially ICAs must use self-regulation since CAA records does not force the ICAs to do the record processing procedure. Domain owners must hope that ICAs are trustworthy and will follow the rules laid down by the system. **Score:3**
4. **Scalability:** CAA Records also highly recommends the deployment of DNSSEC to provide additional security. While it is not necessary, it would provide much needed security for the DNS system on which CAA records relies. Other than this deployment, CAA Records faces no other scaling concerns. **Score:5**
5. **Latency:** CAA Records does not require any additional protocols or security checks during the client/server handshake so there is no additional latency. **Score:9**
6. **Compromise Detection and Response Quality:** Domain operators will be responsible for ensuring that ICAs follow the published record information by keeping track of what certificates have been issued for their domain. However, it will be difficult to discern between rouge certificates and ICA record negligence. CAA Records does not propose a solution for record spoofing either so these are issues which still need to be addressed. **Score:2**
7. **Trust Revocation and Flexibility:** Domain operators have the power to publish records about domain properties, essentially creating an ICA whitelist. In the event that a domain operator decides that certificates from a specific ICA should no longer be accepted by clients they can publish new records to that revoke trust. However, this does not guarantee that ICAs will follow the published records. Clients do not have the ability to choose their trust anchors. It will be up to the domain owners to make good trust decisions. **Score:3**
8. **Protection Offered by System Defaults:** CAA Records provides flexibility and capability for domain orders to limit the ICAs which can issue certificates for their domain. It does not provide much protection since the system relies on ICAs to honor the records. As for web users, they are dependent on the domain owners and ICAs to practice good security, honor the records, and keep an eye on the certificates being issued. Users have no way of flexibility when it comes to choosing their trust anchors and must put their faith in the ICAs and domain owners. **Score:3**
9. **DoS Prevention and Distributed Trust Architecture:** The CAA Records system is no worse and no better in preventing DoS since, with the exception of domain owners' ability to whitelist ICAs, the system is identical to the one currently in place. However, in the case that a domain owner only accepts certificates from a single ICA and that ICA becomes compromised, revoking trust would mean that there would be no ICAs authorized to issue certificates for that domain and users would be denied access to sites in the domain until the compromise was dealt with and the domain owner reauthorized the ICA or another ICA was authorized in its place. **Score:5**

10. **MITM Attack Prevention, Identification and Response:** CAA Records has no client-side changes so the user's experience will be the same as in the current system. MITM attacks will be no more detectable with CAA Records than they are now. Attacks however may be less likely in the ICAs honor the published records. If an attacker compromises an ICA the effect will be contained and affect only those domains who have chosen to trust that specific ICA. **Score:5**
11. **User Privacy:** This is not applicable to the system because CAA Records' design does not create privacy concerns. **Score:10**

4.6 Public Key Pinning

1. **Resource Availability and Defined Business Case:** Public Key Pinning will require good organization from web hosts since they will be responsible for attaching the HTTP header and keeping track of the pinned SPKIs. Other than preparing web hosts to assume this responsible which will require a well-organized model no other resources are needed. **Score:6**
2. **Changes for Deployment and Operation:** ICAs will not need to change their practices however web client changes will be necessary to incorporate pinning capabilities. The system also requires web host changes as they will be responsible for the pinning data and the HTTP headers. These changes are extensive and will affect web clients and web hosts which will be more trouble from an implementation perspective than changes in ICA practices. **Score:3**
3. **TTP Trustworthiness and Security:** The web hosts must be able to successfully keep track of SPKI pins and ensure that pins are not spoofed or that inadvertent pin failures are dealt with quickly. This will require organization and operational maturity from the web hosts. **Score:5**
4. **Scalability:** Public Key Pinning does present some scaling problems because browsers must be preloaded with pins. This is because initially browsers have no stored pinning metadata so browsers can be fooled if the user is visiting a website for the first time. This is an issue which is still in the process of being addressed. **Score:3**
5. **Latency:** Clients will need to check the Public-Key-Pins response header field for certain criteria (outlined in the Public Key Pinning description in the previous section). During this pin validation process the client will have to computer the fingerprints of all the SPKI structures in the certificate chain which may slow down the connection process. **Score:6**
6. **Compromise Detection and Response Quality:** In the event that the SPKI fingerprints do not intersect with the set of fingerprints in the host's pinning metadata the connection is closed and the user is not allowed to proceed. If pin validation process fails then the client will alert the user indicating that there was a connection error. As for compromise on the server-side, web hosts will be responsible for keeping track of SPKI pins and identifying any issues such as spoofed pins. **Score:6**
7. **Trust Revocation and Flexibility:** If the header field does not meet the criteria outlined by the system proposal then the client will discard any previously set pinning metadata for the web host. So if a client senses that something is not right the pinning metadata is no longer trusted and thrown out. However, users will not be able to choose their trust anchors explicitly. The control falls into the hands of domain owners who can using Public Key Pinning to whitelist ICAs. **Score:6**
8. **Protection Offered by System Defaults:** Pubic Key Pinning does not allow for improved flexibility or capability on the part of the user. It does however increase protection for web users. The client will do all the pin validation automatically and will close the connection if an attack is detected, preventing clients from undermining any warning messages and putting themselves at risk. **Score:5**
9. **DoS Prevention and Distributed Trust Architecture:** The Public Key Pinning design will cause users to become "locked out" in the event of inadvertent pin failure or failed validation. The proposal seeks to remedy this issue by introducing backup pins which can be validated to allow the user access to the site. The duration of the lock-out is determined by the max-age variable; users must wait for this to expire before the client can try connecting again. As such, the system is very sensitive to DoS attacks. **Score:3**
10. **MITM Attack Prevention, Identification and Response:** Public Key Pinning will decrease the number of MITM attacks since it allows host to whitelist ICAs. In the advent that an ICA is compromised, only users connecting to hosts which accept certificates issued by that ICA will be at risk. When attacks are detected (i.e. validation fails at some step during the connection process) the user is denied access to the site rather than allowing them to click through warnings. This has the advantage that the user will be protected but it

also has the side-effect that any sort of pin failure will lock out the user for a certain period of time. **Score:7**

11. **User Privacy:** User privacy is not a large concern for this system. However, pinning metadata will be stored on the client for each site the user tries to access. **Score:8**

4.7 Sovereign Keys

1. **Resource Availability and Defined Business Case:** The system will require resources to set up the timeline servers and the mirror servers. Timeline servers will also need to be able to query OCSPs for revocation information. There is no formal description on who should manage the timeline servers but it is recommended that they should be "chosen to provide some diversity of jurisdictions, operational philosophies and security implementations. [8]" This makes it difficult to judge whether or not a business case is feasible and whether or not there are resources available to build the architecture for Sovereign Keys. A more detailed description of the management and distribution of responsibilities will be required. **Score:2**
2. **Changes for Deployment and Operation:** Continuing from what was enumerated in no. 1, because much of the design is still in progress it is hard to assess the extent of the changes which will be necessary for the system implementation. The TTP(s) running the timeline servers will be the most critical actors in the system as the timelines servers are crucial to the Sovereign Keys trust architecture. Sovereign Keys will definitely require client-side changes and web host changes in order to incorporate the sovereign key protocol. ICA issued certificates will also have to be published to timeline servers. Domain owners will be responsible for registering sovereign keys for a new service. **Score:3**
3. **TTP Trustworthiness and Security:** One concern is the trustworthiness of the party or parties that ends up running the timeline servers. This is potentially a significant point of weakness in the system so good security practices are essential. Sovereign Keys will need to address this as the design progresses since it has so many implications for user security. **Score:3**
4. **Scalability:** Sovereign Key's largest scaling issue is its proposed use of TOR to circumvent MITM attacks and redirect users. There is also the issue of deciding how far back timeline servers should keep records of claims history. The number of timeline entries should be kept to a reasonable number to avoid putting stress on the servers. **Score:6**
5. **Latency:** Because of the fairly large size of the query responses from mirrors there may be an increase in latency of the TLS connections. **Score:4**
6. **Compromise Detection and Response Quality:** Clients will be responsible for querying the mirror servers about reneged servers in order to remove the IP addresses of potentially compromised mirrors from their trust list. Clients may also help detect a compromised server by checking that cached responses from mirrors are consistent with any replies they receive from the mirror servers. In the case that there are any suspicious or missing entries then the client will stop trusting the mirror and take actions to get the mirror reneged (most likely by alerting the timeline server managers). There are synchronization issues tied up in this process and the potential for false alarms. Timeline servers are crucial to the system and since they all trust each other, if a compromised timeline server goes unnoticed this opens up the possibility of attacks on users. **Score:4**
7. **Trust Revocation and Flexibility:** Clients will have the ability to revoke their trust in mirror servers if the replies from a mirror do not pass the cross-validation checks with the client's cached responses. Clients however, will still be forced to trust the managers of the timeline servers since they are such a critical piece of the system. **Score:5**
8. **Protection Offered by System Defaults:** The Sovereign Key design automates many of the necessary security mechanisms including identifying comprise and security breaches, preventing MITM attacks by circumventing the attack and redirecting the user, and revoking trust in mirror servers when they are reneged due to suspicious behavior. The system increases the protection of web users but it does not offer them any flexibility in terms of trust decisions. **Score:5**
9. **DoS Prevention and Distributed Trust Architecture:** The Sovereign Keys system does not provide for a very distributed trust architecture and this leaves it open to certain DoS concerns. Attackers could flood the timeline server with registration requests or flood the mirrors with queries. These false registrations would be a concern in the event that an ICA is compromised since Sovereign Keys places its trust in the ICAs for new sovereign key registration. This leaves still leaves ICAs more vulnerable to attack which means the

proposal fails to adequately address issues with the current system design. The timeline servers also provide another attack vector since clients and mirrors treat this relatively small number of servers as trustworthy entities. **Score:3**

10. **MITM Attack Prevention, Identification and Response:** Sovereign Keys should provide robust protection against MITM attacks by verifying that a server's public key has been cross-signed by a sovereign key. In the event that a MITM attack is detected, warning messages will be used to discourage the user from continuing to the site. The design draft also proposes that attacks can be circumvented by using TOR to connect to a "hidden service" address. However, it is not reasonable to expect that this method of circumvention could be used on a large scale. **Score:7**
11. **User Privacy:** Sovereign Keys shares the same privacy concerns as Perspectives because mirrors will have the ability to learn the client's IP and service they are querying about which opens up the possibility that a user's browsing history will be revealed. As with Perspectives, Sovereign Keys recommends the use of a proxy to prevent this and alleviate user privacy concerns. **Score:4**

4.8 Certificate Transparency

1. **Resource Availability and Defined Business Case:** Certificate Transparency will require the creation of audit logs which need to be managed and monitored by some third party. The proposal points to the ICAs as potential candidates for this task. Domain owners will also need to monitor the public log. The business case requires further attention and more detail although the resources needed for the system appear to be available. **Score:5**
2. **Changes for Deployment and Operation:** The ICAs would be required to publish information to the logs about every certificate they issue so ICAs would need to change their practices to incorporate this step. Clients would need to check the audit proofs and the logs to ensure that the certificate is valid so the system will require changes from the web client. Domain owners would also be responsible for monitoring the public log for certificates published for their domain and revoking ones which are not valid. **Score:4**
3. **TTP Trustworthiness and Security:** The ICAs would remain the TTP in the Certificate Transparency system and they would have to secure their audit logs and expand their practices to include careful monitoring of the logs. As with the current system, there is no way to ensure that ICAs will have equally strong security (i.e. that they are all equally trustable). **Score:4**
4. **Scalability:** Scaling is another concern for the system. The size and the ability of the logs will be tied to whether or not they can be sufficiently monitored. If the logs are too large it will be difficult to detect problems within them. **Score:3**
5. **Latency:** There is not enough documentation available to assess whether or not latency should be a concern for the Certificate Transparency system. Clients will have to use cross-validation by checking the ICAs' audit logs and/or the public log. One would assume this would mean the client would have to query a server which stores the logs and wait for a response to validate the certificate. This may slow down the connection process. **Score:5**
6. **Compromise Detection and Response Quality:** The client is responsible for checking the public log and the ICAs' audit proofs in order to verify the server's public key and potentially identify a rouge certificate. In the case that the certificate cannot be found in either the ICA log or the public log, the client will reject the certificate. If the certificate is only published in the public log it would be up to the domain owner to see that the certificate is not valid. If the certificate only appears in an ICA's audit proof then it will be up to the ICA to identify this and revoke the certificate. While this cross-validation technique is effective for identifying compromise it also requires the efforts of many parties and careful monitoring. **Score:6**
7. **Trust Revocation and Flexibility:** Users will not be able to customize their list of trust anchors. They will still need to rely on the ICAs no matter their trustworthiness. The audit proofs and public log are intended to encourage better and more secure practices on the part of the ICAs and to hold them accountable for any compromise that results in rouge certificates. **Score:2**
8. **Protection Offered by System Defaults:** The changes in the PKI system will revolve around practice changes implemented by the ICAs and the domain owners and so users will not be able to choose which ICAs they wish to trust. The creation of the logs will afford users more protection from attacks, or at least allow fraudulent certificates to be detected more quickly. However the capabilities of web users will not be any different from what they are now. **Score:5**

9. **DoS Prevention and Distributed Trust Architecture:** In the event that the public log because unavailable (due to compromise or failure), clients will be unable to validate certificates and will be unable to connect. Users will also be denied access to sites if the ICAs' audit logs become unavailable but this will only affect sites who use certificates issued by those specific ICAs so the extent of the DoS is somewhat mitigated. If the Certificate Transparency design chooses to have a single public log it could potentially be the single point of failure for the system. **Score:4**
10. **MITM Attack Prevention, Identification and Response:** Certificate Transparency seeks to make publicly available a list of valid certificates which would allow sites to know what certificates have been issued for their domain and increase the chance that rouge certificates will be identified before users become the victim of MITM attacks. The system improves fraudulent certificate detection but does not necessarily decrease the number of attacks. It may however work to make ICAs more responsible since they will be required to have an audit proof for each certificate. Clients will also have a way of cross-validating certificates and making users aware of attacks. **Score:7**
11. **User Privacy:** The Certificate Transparency design is still in progress and there has been no mention of privacy issues. However, if clients have to query a server about a specific certificate then that server will be aware of the client's IP and the certificate for the site the user wants to access. This could potentially lead to cases where a user's browsing history could be viewed by a malicious party. **Score:6**

4.9 Table Summary of Scores

	Convergence	Perspectives	MECAI	DANE	CAA	Pinning	SK	CT	Current System
1. Resource Availability &Define Business Case	4	4	9	3	7	6	2	5	9
2. Changes	3	4	3	2	6	3	3	4	10
3. Trustworthy & Secure Trust Source	2	2	5	3	3	5	3	4	5
4. Scalability	4	4	9	2	5	3	6	3	8
5. Latency	7	6	2	6	9	6	4	5	7
6. Compromise Detection	5	5	6	5	2	8	4	6	6
7. Trust Revocation	8	4	4	2	3	6	5	2	6
8. Improved protection/flexibility/capability of web users	7	7	7	5	3	5	5	5	5
9. DoS/Failure Prevention	3	3	3	2	5	3	3	4	6
10. MITM Attack Prevention & Response	4	5	5	7	5	7	7	7	4
11. User Privacy	7	2	1	9	10	8	4	6	7
Totals	**54**	**46**	**54**	**46**	**58**	**60**	**46**	**51**	**73**

5 Conclusion

After evaluating each of the proposals based on the metric's criteria, the results show that there is no cure-all for the weaknesses identified in the existing ICA infrastructure. However, we can conclude by looking at the scores given to the proposed alternatives that the current ICA architecture is arguably the strongest and most scalable system.

By making comparisons with the existing system the resulting scores predict that implementing any of the proposed alternatives would result in a degradation of the current infrastructure. In their efforts to mitigate the MITM threat and improve revocation, additional burdens would be placed on the system. Theses scores also suggest

that the proposals which will perform best are those which are enhancements to the existing trust infrastructure rather than replacements for it.

It is advisable that we first look to implementing measures which would strengthen and improve the existing system, rather than opting for a completely new design. Initiatives have been taken by the Certification Authority and Browser (CAB) Forum to update and secure the existing system by identifying four specific points for improvement: 1) Common minimum security practices, 2) Improved revocation processing, 3) Mitigation of MITM attacks and 4) Better audit. There is hope that by combining these initiatives with some of the recommendations from existing proposals, practical changes can be implemented to strengthen the current system.

The four different trust architecture groups – TTP, DNSSEC, Pinning, and Transparency – are a preliminary step in organizing the proposals for further research. Changes can be grouped into one of the four categories based on how they seek to improve trust, although overlap is certainly possible. A common theme of all of the proposals is that significant changes will have to occur in at least one of these four architecture groups. For TTP changes, this means the establishment of additional third trusted parties or significant changes to practices and/or additional responsibilities for ICAs. The DNSSEC category requires that trust come from the DNS layer and changes will have to be made to incorporate additional security and verification mechanisms between clients and DNS servers and to further harden the DNS servers and their support infrastructures. Pinning relies on changes to deployed web servers, giving them the ability to create whitelists for ICAs which can be checked by clients before accepting certificates. Lastly, Transparency relies on the use of additional auditing procedures to provide methods for cross-verification and to create more visibility on the part of ICAs and the certificates they issue.

Addressing the current weaknesses by any means will come with certain costs and such non-trivial changes are always difficult with a system that has been the broadly accepted solution for some time now. Thus, the conclusions derived from this work are an assessment of which systems will provide the most benefit and improvement in light of their current weaknesses and concerns, while also addressing the practicality, feasibility, scalability of these alternatives.

There are also the implementation difficulties to consider with any enhancements or new system proposals. Aside from purely technical challenges, most of the alternative systems will be met with resistance from the parties involved in the proposals' changes. ICAs, domain operators, and web hosts are all mentioned in the proposals as parties that would have to make modifications to their current infrastructures and/or practices. A critical part of the adoption decision will involve understanding how many additional responsibilities these parties are willing and capable of accepting and what kind of business models can be adopted which will benefit everyone involved.

We make these assessments while mindful of the fact that all of these alternative systems are relatively new proposals and that their designs are still in progress. As they evolve it may be necessary to reevaluate how effective they are at addressing the goals defined in the metric offered in this paper.

The metric described in this paper provides a starting point for understanding the criteria that must be taken into account when evaluating any proposed changes to the current PKI. Other measurements may be added to this list of 11 criteria. However, it addresses the major concerns and weak points of the existing system as compared to the proposed alternatives and deals with the most important issues surrounding implementation. A useful extension of our work would be to understand how the criteria can be weighted to address the specific needs and concerns of different entities. Using this baseline evaluation, groups can gain a better understanding of which alternative system would be the best fit for them and provide data to support those recommendations.

In this paper we've addressed the reality of the challenges faced by implementing a new system. In evaluating all of the currently proposed alternatives, we've provided a solid baseline analysis, using metrics that evaluate each option objectively, to determine the most feasible path going forward. While more research is needed to expand upon our results and further explore the options available, the research has identified the proposals having the greatest scalability and feasibility. Particularly, due to proven technology, scalability and built-in economic models, we should focus on strengthening the existing system based on CAB Forum recommendations. This analysis and metric are the first steps towards clarifying the nature of the challenges involved and organizing future efforts to improve trust in the PKI system.

6 Acknowledgements

I would like to thank my advisor, Charles Palmer, for all of his support and guidance during this long process. Next, I would like to thank Scott Rea at DigiCert who first approached me about this topic. The work in this paper would not have happened without his help. I'd also like to thank Massimiliano Pala at NYU Poly and Sean

Smith and the other professors in the Dartmouth Computer Science department who helped me along as I was researching and writing this paper. Finally, thank you to my friends and family for all the encouragement.

7 References

[1] F. Rashid. (2011, Mar 28). *Comodo Hacker Exploited Insecure Passwords to Generate SSL Certs*[Online]. Available: http://www.eweek.com/c/a/Security/Comodo-Hacker-Exploited-Insecure-Passwords-to-Generate-SSL-Certs-736334/

[2] D. Fisher. (2011, Aug 30). *DigiNotar Says Its CA Infrastructure Was Compromised* [Online]. Available: http://threatpost.com/en_us/blogs/diginotar-says-its-ca-infrastructure-was-compromised-083011

[3] P. Hoffman et al.(2012, Jan 14). *Using Secure DNS to Associate Certificates with Domain Names For TLS* [Online]. Available: http://datatracker.ietf.org/doc/draft-ietf-dane-protocol/?include_text=1

[4] M. Marlinspike. (2011, Aug 18). *Convergence* [Online]. Available: http://www.youtube.com/watch?v=Z7Wl2FW2TcA

[5] D. Wendlandt et al. (2011, Jul). *Perspectives: Improving SSH-style Host Authentication with Multi-Path Probing* [Online]. Available: http://perspectivessecurity.files.wordpress.com/2011/07/perspectives_usenix08.pdf

[6] A. Langley. (2011, Mar 4). *Public Key Pinning* [Online]. Available: http://www.imperialviolet.org/2011/05/04/pinning.html

[7] P. Hallam-Baker et al. (2011, April 21). *DNS Certification Authority Authorization (CAA) Resource Record*[Online]. Available: http://tools.ietf.org/html/draft-hallambaker-donotissue-00

[8] P. Eckersley. (2011). *Sovereign Key Cryptography for Internet Domains* [Online]. Available: https://git.eff.org/?p=sovereign-keys.git;a=blob_plain;f=sovereign-key-design.txt;hb=master

[9] K. Engert. (2001, Oct 21). *MECAI – Mutually Endorsing CA Infrastructure*[Online]. Available: https://kuix.de/mecai/

[10] C. Evans et al. (2011, Sept 21). *Certificate Pinning Extension for HSTS*[Online]. Available: http://tools.ietf.org/html/draft-evans-palmer-hsts-pinning-00

[11] B. Laurie, A. Langley. (2012). *Certificate Transparency*[Online]. Available: http://www.certificate-transparency.org

[12] A. Langely. (2011, Sept 7). *Why not Convergence?*[Online]. Available: http://www.imperialviolet.org/2011/09/07/convergence.html

[13] P. Hallam-Baker et al.(2012, Mar 7). *DNS Certification Authority Authorization (CAA) Resource Record*[Online]. Available: http://tools.ietf.org/html/draft-ietf-pkix-caa-05 (CAA Records)

[14] D. Wendlandt. (2011). *Perspectives: Can Host Authentication be Secure AND Cheap?*[Online]. Available: http://perspectivessecurity.files.wordpress.com/2011/07/perspectives_nanog.ppt

[15] C. Evans et al.(2011, Dec 9). *Public Key Pinning Extension for HTTP*[Online]. Available: http://tools.ietf.org/html/draft-ietf-websec-key-pinning-01

[16] L. Constantin. (2011, Nov 22). *EFF Proposes New Method to Strengthen Public Key Infrastructure*[Online]. Available: http://www.pcworld.idg.com.au/article/408227/eff_proposes_new_method_strengthen_public_key_infrastructure/