

“Dynamic Bridge” Concept Paper

Ken Stillson
Mitretek Systems

Written for presentation to the
3rd Annual PKI R&D Workshop

Table of Contents

Problem Background	3
Solution Background	5
Dynamic Bridge Concept.....	6
Proposed Benefits	9
Challenges and Unresolved Issues.....	10
Request for Feedback.....	11
References.....	12

Abstract

Current cross-certificates based PKI trust mechanisms suffer from a scaling problem. Even given the topological simplification of bridge CAs, as cross certificate meshes grow in size and complexity, the number of possible routes between points increases very quickly, and the time required for path discovery can increase beyond a tolerable delay for real-time operation. This paper proposes a “dynamic bridge,” which is an automatically created transformation of a cross certificate topology, designed to reflect the same trust arrangements and constraints, but in a simplified structure. Creation of dynamic bridges should not require centralized coordination or infrastructure, and use of them for speed-enhanced validation should require clients to implement only a subset of standard path discovery and validation logic.

Problem Background

The standards that govern PKI, primarily [X509] and [RFC3280], envision a mechanism for a recipient, or “relying party” in one PKI domain to accept credentials from a sender or signer in another. The recipient has one or more “trust anchors.” These are certificate authorities (“CAs”) that the recipient trusts completely. These CAs can create cross-certificates, which indicate other CAs that this CA trusts. This process can be repeated multiple times, resulting in a chain of trust from the trust anchor to the sender’s certificate. [pathbuild]

This process involves three distinct areas: “path discovery,” “object location,” and “path validation.”

Path discovery entails finding the possible chain(s) of certificates between the sender and the trust anchor(s). Path discovery would be challenging enough even if all the certificates in the world were immediately available to select from. However, generally the only inputs to path discovery are the end-points: the sender’s certificate, available because it is included with the signed message being validated, and the trust-anchor(s), which are part of the relying party’s configuration. Path discovery therefore involves an iterative process sniffing out each “next possible link” – building the chain one link at a time.

Object location is the challenge of retrieving the certificates and cross certificates needed to feed the path discovery algorithm. Object location suffers from competition between several different mechanisms, none of which are very mature, and each of which assume they are the global solution. The separate mechanisms do not easily build upon each other. This challenge is not fatal, as the software of a relying party can support all of the contending mechanisms.

Path validation takes a candidate chain created by path discovery, and confirms that all the rules of trust transfer are followed within the chain. For example, cross-certificates can stipulate constraints that add requirements to the overall chain, or to parts of it some distance away from the certificate adding the constraint. Path validation checks all the constraints of each certificate against the others. Path validation also generally includes an “on-line status check” to confirm that no certificate in the chain has been revoked. Path validation is a computationally expensive process, but is well defined and reasonably well understood.

The focus of this paper is a scaling problem with path discovery.

Although developed independently, the concept is an extension of [Sun1], which envisions hierarchical root CAs issuing certificates directly to their n-level subordinates, thus flattening the hierarchical structure. The dynamic bridge extends this concept into the space of multi-domain PKIs linked by cross certificates, and handles the complexity of policy and constraint mapping introduced by such an extension.

In figure 1, a path discovery algorithm starts with the sender's certificate¹. An object location mechanism should easily find CA 1, as CA 1's name is stated in the sender's certificate. The next step will be for the discovery algorithm to ask the location mechanism to find all certificates issued to CA 1. This could result in any number of certificates, indicated by the multiple green arrowheads around CA 1. As we have the picture already laid-out, we can see that the link to CA 2 is the correct choice. However, there is no way for the discovery algorithm to know this. It may well select the link that leads into cloud X, and one can imagine that if cloud X contains a large and complex mesh, it may be some time before the path discovery algorithm realizes it made a wrong turn at CA 1, and tries the alternative path leading to CA 2.

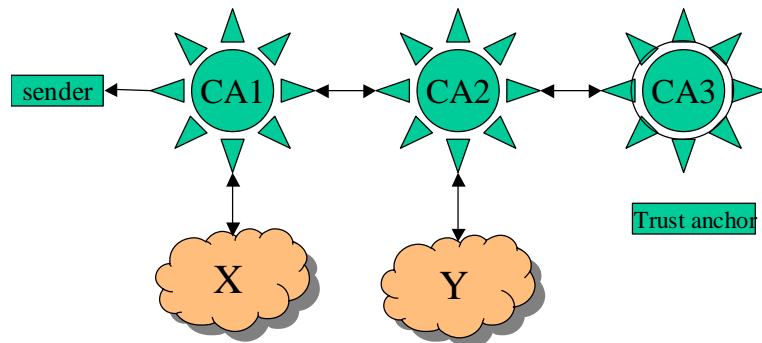


Figure 1

One response to this problem has been the implementation of PKI “bridge CAs.” [pathbuild][FBCA]. A bridge CA is like a trust “hub”; it re-organizes the cross-certificate “mesh” topology into a star shape, which shortens the number of links in chains.

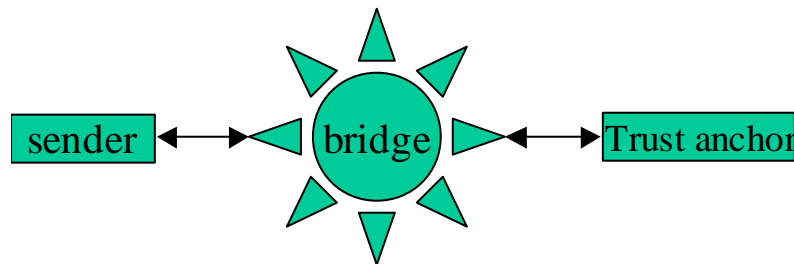


Figure 2

¹ There is an alternate technique [RFC3280] that begins at the trust anchor(s) and builds the path towards the sender. This difference is not relevant to the scaling problem that the example is building towards.

If an all-encompassing central bridge linked the entire world, there would be no path discovery problem. The cross-certificate from each trust anchor to the bridge would contain enough information to be located immediately from the bridge.

However, it has become clear that there will be no global bridge in the near future. No organization appears to have the combination of desire, funding, expertise, and ubiquitous acceptance that would be required. Rather, individual arenas are creating bridges that cover their natural scope. For example, the US Federal government has established the Federal Bridge CA [FBCA], Canada has a national bridge underway, and EDUCAUSE (a consortium of educational institutions) has created the Higher Education Bridge CA [HEBCA], etc.

These bridges are slowly being linked together. The result will likely be a cluster of large bridges, surrounded by a constellation of smaller bridges, surrounded by individual PKI domains. While this arrangement is an improvement on an unstructured mesh, in that the *average* path length will be lower, the problem discussed above in figure 1 remains. In fact, in figure 1, if CA1 and CA2 were both bridges, the number of possible “wrong” routes would likely increase drastically, compared to individual CAs.

Basically, path discovery suffers from a lack of a “sense of direction.” [PKI concepts]

Solution Background

A common approach to resolve the path discovery sense of direction problem involves a process scanning the entire cross-certificate mesh, and pre-processing the results in some way to make discovery of a specific path faster once the actual endpoints are known.

However, there is a general complication to this technique. One cannot “cache” pre-validated *partial* paths. This is because of the ability of any certificate in a chain to add constraints applying to other (non-neighboring) certificates. Specifically, if a chain from CA1 ↔ CA2 ↔ CA3 is found, and is believed to be a common component to complete chains, it would make sense to cache it as an available component for building larger chains. However, once CA4 is added, CA4 may contain a property forbidden by a constraint in CA2, or vice-versa. As this is true all the way through to the end-user certificate, no partial path is safe from elimination by constraints from certificates not included in the partial path.

This is a complication rather than a fatal flaw as it can be resolved by careful separation of path discovery and path validation. In other words, partial paths can be cached as “discovered,” so long as path validation is performed on the complete path once assembled, to make sure that constraints of the additional certificates are respected.

An example of a path discovery assistance system is the “intermediate store solution,” envisioned and implemented in proof-of-concept by the FBCA’s “path discovery and validation working group (PD-VAL).”²

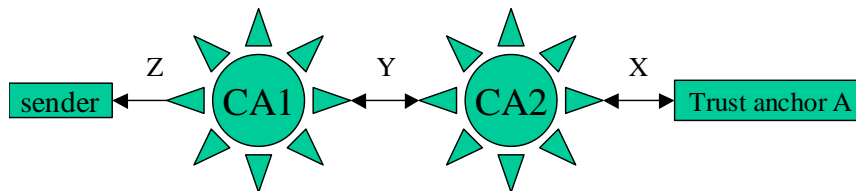
This intermediate store solution uses a “cross certificate spider” [HHSspider] to map out entire cross certificate topologies, and retrieve all the directly linked certificates and cross-certificates. This program essentially takes on the object location challenge, and retrieves all objects possibly needed to a server. The certificates are then stored into a distributable form (a PKCS7 file), and published to a web-server. A program running on end-user Microsoft workstations then regularly downloads the PKCS7 file, and adds its contents to the Microsoft Windows “intermediate store.” This removes the iterative piece-by-piece part of path discovery, leaving only the problem of selecting a valid chain given the pre-collected universe of certificates, a capability that is built-in to most modern versions of Windows.³

This approach shows some promise, in homogenizing differences between versions of Windows (its original purpose), solving the object location problem, and simplifying the path discovery problem. [CAI-POC] However, like most pre-caching solutions, it requires proprietary software running on the desktop to utilize the cache – in this case, the program that downloads and installs the PKCS7 cache file.

Dynamic Bridge Concept

The idea of the dynamic bridge is to actively search out paths with a path length greater than 1 hop, and “condense” them by creating direct cross-certificates to reduce the path-length to one.

For example we consider the path

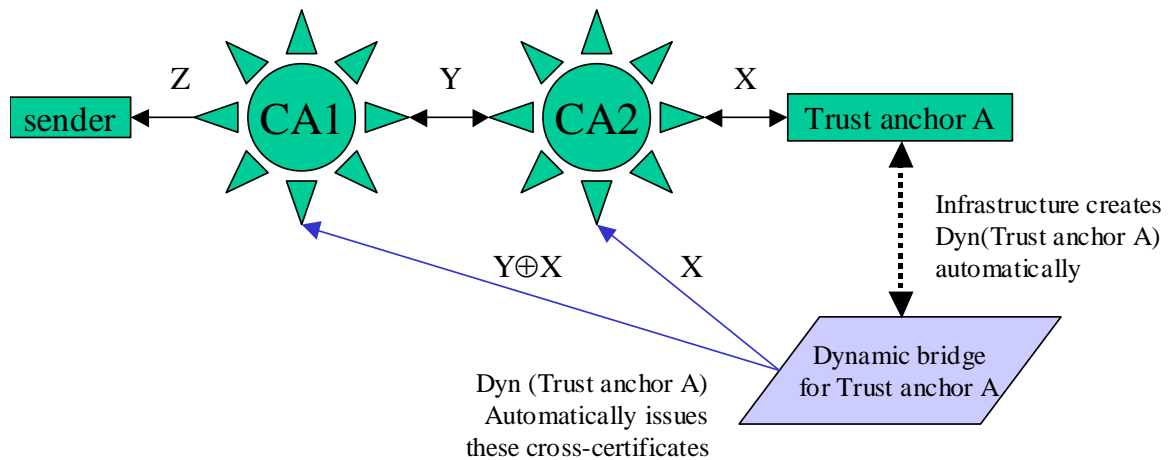


Here, X, Y, and Z are constraints, policies, policy mappings, and other attributes that effect a path’s validity.

² “PD-VAL” was established by the FBCA Operational Authority to research challenges on bridge-aware path discovery. PD-VAL members include representatives from NIST, the Federal PKI technical working group, various Federal agencies, PKI vendors, and the author.

³ Some versions of Windows do have the capability to perform iterative path discovery, but they rely on an object location solution that makes assumptions about certificate “AIA fields” that are not universally followed. By separating the object location function into the “cross-certificate spider,” which supports multiple object location mechanisms, this removes the Windows AIA requirement.

Dynamic bridge infrastructure would transform the above situation as follows:



Specifically– A new CA, “Dyn(A),” with its own self-signed certificate, has been created, and that new CA has issued a series of unidirectional cross certificates. The new cross certificates reflect exactly the same trust arrangements between their subjects and Dyn(A) as the initial CA, but they have been “flattened” to combine intermediate hops. All trust is transferred directly from Dyn(A) to the targets with new 1-hop cross certificates.

This “condensation” of multiple hops into a single one is similar to [Sun1], except that rather than flattening hierarchical chains, the condensed hops here include cross-certificates between distinct PKI domains. This creates a significant new complication – trying to condense the constraints of the cross certificates.

In the case of the trust transfer from A -> CA2, with the constraint X, nothing has changed. However, in the case of the path A -> CA2 -> CA1 with the constraint X between A and CA2, and the constraint Y between CA2 and CA1, this path of length two has been reduced to a path of length one, directly from Dyn(A) to CA1, with the new constraint “Y⊕X”, which is the ordered combination of constraints Y and X.

For example, if Y requires policy1, and X requires policy2, then Y⊕X would require both policies in a single constraint. If Y maps policy1 to policy2, and X maps policy2 to policy3, then Y⊕X map policy1 directly to policy3. It is asserted that all the constraints possible in cross certificates can be combined by a fully implemented ⊕ function, and that therefore the types of certificates and chains that can be condensed is not limited.

A dynamic bridge infrastructure would automatically create Dyn(A) by using a cross certificate spider to locate all paths that lead back to A. Once this process is completed, a user that previously used A as their trust anchor would change their trust anchor to Dyn(A). They would now find that all paths leading back to A are now immediately available directly from Dyn(A) with path length 1.

Furthermore, it is proposed that all cross certificates created with the \oplus function also add a path length constraint indicating that at most one further hop is permitted. This would effectively prevent iterative path discovery. If a user uses only Dyn(A) as a trust anchor, then the process of path discovery is reduced to selecting the cross certificate from the issuer of the sender's certificate to the trust anchor.

Walking through a typical path discovery algorithm: in typical implementations, discovery starts with the sender's certificate, which contains the name of its issuer. An object location query is executed for all certificates whose subject matches the issuer of the sender's certificate. In this case, only CA1 will be returned. CA1 is not a trust anchor, so the algorithm iterates. An object location query is run for all certificates whose subject is CA1. This will return numerous certificates, including the one issued directly from the dynamic bridge. As the dynamic bridge is a trust anchor, path discovery is concluded. The key is that the path discovery algorithm never had to make a "guess" as to which certificate to select. The very first query that returned multiple certificates including a direct link to the trust anchor. This avoids the need for a "sense of direction."⁴

There is an interesting implication for revocation checking. The dynamic bridge path has "short circuited" CA2. A correctly implemented \oplus function would ensure that CA2's constraints are respected, however in the original topology, CA2 also has the capability to revoke its cross certificate to CA1, thus breaking the path. The dynamic bridge path does not pass through CA2, and thus removes CA's revocation capability.

There is a solution. The expiration date for the dynamic bridge's certificate from Dyn(A) to CA1 should be set to the earliest CRL "next update" time for any of the CA's that have been "flattened" from the path. i.e. The earliest CRL update time has become the cross certificate expiration time. In this way, a dynamic bridge cross certificate is valid only up until the time that a CRL update could have invalidated part of the consolidated path.

The dynamic bridge must continuously re-generate its cross certificates as they expire, and obviously will re-perform full path validation before re-consolidating paths, thus giving the intermediate CA's the opportunity to revoke paths.

Clearly the dynamic bridge is going to be a busy system. Its internal database conceivably consists of certificates for every CA in the world, it must continuously scan for new CA's (new paths to add), and perform the above re-validation of existing paths each time a CRL expires. However, a clever implementation could filter on changes to previous queries to detect additions, and queue re-validation carefully for only expiring paths. The assertion is made that this implementation is feasible.

⁴ Note that it is possible that multiple paths existed through the original mesh between the sender's CA and the trust anchor. In this case, there would be multiple single-hop cross certificates from dynamic bridge to the issuer's CA. Although there is a "choice" as to which of these multiple paths will be selected, this does not change the claimed advantage. Whichever single hop cross certificate is selected by the path processor, it results in immediate path to the trust anchor with no further iteration, there is no "heading off in the wrong direction" regardless of which certificate is selected.

Proposed Benefits

It is believed that this arrangement would lead to benefits in each of the three primary areas of cross certificate-based trust building.

For path discovery, the need for iterative discovery is removed. The path length constraints added by the \oplus function ensure that no “wild goose chases” will occur by the path discovery algorithm taking a wrong turn; as no “turns” are allowed.

For path validation, the process of validation has become simpler, both because path lengths are reduced, and because the cumulative effects of the chained constraints have been pre-calculated.

For object location, the dynamic bridge process has already collected and consolidated all the objects that the relying party will require. Assuming the dynamic bridge’s cross-certificates are all stored in a single directory, the recipient’s object location system needs only to be directed to that location.⁵ If the dynamic bridge’s object location mechanism supports multiple of the competing retrieval techniques (e.g. DN searching against X.500, AIA, LDAP referral trees, etc), the dynamic bridge’s user’s software need only support the mechanism that leads it to the dynamic bridge’s consolidated repository.

Construction of a dynamic bridge requires no particular privilege, nor the cooperation of the mesh participants, other than CAs posting their cross-certificates into locations that the dynamic bridge’s object location techniques can find. Any enterprise, or even end-users, can establish their own dynamic bridge(s). If multiple trust anchors are in use, either multiple independent dynamic bridges can be constructed (if selection as trust anchors must be separately selectable), or a dynamic bridge could merge multiple meshes by seeding the “condensing” process from multiple trust anchors.

While only those that utilize the dynamic bridge’s trust anchor will receive the above benefits, the existence of the dynamic bridge is non-harmful to those that do not utilize it.

Finally, utilization of a dynamic bridge does not require any specialized software. Any standards-compliant path discovery system will be able to gain the advantages of a dynamic bridge just by re-selecting their trust anchor(s). In-fact, only a small subset of the standards-required capabilities are needed; some PKI software that is not fully compliant now (due to lack of iterative path discovery, or limited object location capabilities) would be made fully capable by using a dynamic bridge.

⁵ Another interesting possibility is that the dynamic bridge could set AIA and/or SIA fields in the cross certificates that it generates. An example of a possible advantage -- the dynamic bridge will have multiple object location techniques that it may utilize when searching the cross certificate mesh, storing the technique that worked in an SIA field of the cross certificate could simplify object location of the target certificate for path building techniques that start with the trust anchor and work towards the sender.

Challenges and Unresolved Issues

Firstly, while the dynamic bridge does not create new key pairs⁶, it does automatically create cross certificates which its users will trust. Frequently, CAs are kept “offline” to improve security, but due to the automated and continuous nature of its processing, the dynamic bridge must be “online.”⁷

Secondly, the entire concept hinges on the \oplus function – the ability to take a series of constraints in separate certificates along a path and condense them into a single set of constraints, stored in a single X.509 compliant cross certificate constraint. Intuitively, this should be possible. A sample “permitted and forbidden sub-trees” algorithm is specified in [RFC3280], and although that algorithm is designed to be run iteratively during path construction, it should also be possible to run it during the cross certificate crawl. Furthermore, it appears that the constraints specification system is adequately expressive that transitive combinations of constraints can be simplified to a single constraint, but until \oplus is successfully implemented, caution is needed.

Thirdly, Microsoft Window’s build-in path discovery system (“CAPI”) remains a challenge with respect to object location. CAPI does not support specification of a “default directory” or an “AIA⁸ of last resort,” which could be used to point to the dynamic bridge’s directory. CAPI builds from the sender towards the trust anchor, so addition of AIA fields to dynamic bridge certificates does not help, as the sender’s certificate’s AIA will not lead to the relying party’s dynamic bridge directory. Again, this could be overcome by loading the dynamic bridge output into the relying party’s intermediate store, but this would require proprietary software on the desktop.

Finally, there is an issue with respect to the object location algorithm used by the spider process that builds the dynamic bridge. The crawl must start at the known trust anchor(s), and spread outwards. This direction is “the hard way” for object location.

When AIA fields are not present, the object location problem is generally solved via an LDAP search for DN’s matching the subject field of the desired objects. When building a path from the sender’s certificate towards the trust anchor(s), each certificate contains the DN of it’s issuer, so the next possible steps can be queried by searching for certificates with the subject that matches the issuer of the current DN. However, certificates do not have an “issuee” field, so this technique cannot proceed in the opposite

⁶ The creation of cross certificates involves signing an existing key with an existing key, it does not generate new key pairs.

⁷ Technically, only a border directory containing the cross certificates must be fully on-line. The dynamic bridge must be able to push cross certificates onto its border directory, but other than this action, can be well isolated. While an “air gap” around CA’s is “better,” it is not unusual for production CA’s to have a live one-way connection to their border directories.

⁸ AIA stands for “authority information access,” and in this context, gives a location to obtain all certificates whose subject matches the issuer of that certificate. AIA fields may be used by object location algorithms to obtain the “next step” in path discovery. An “AIA of last resort” specifies a general technique for finding any certificate’s issuers given it’s DN. Use of this type of mechanism is one of the competing object location solutions referred to in the first section.

direction. [X509] defines the “SIA” extension (the converse of the AIA field) for this purpose, but SIA is not widely populated.

There is a saving grace – bi-directional trust. When A issues a cross certificate to B, the matching reverse cross certificate is usually issued by B to A. The first iteration of the “subject that matches the issuer” technique will locate the reverse certificate, thus revealing the DN of the CA one step further from the trust anchor. The next iteration of the “subject that matches the issuer” technique will then return the forward cross certificate, allowing the crawl to spread outwards. This procedure has been shown to work in an implemented spider [HHSpider], but it does rely on bi-directional trust (or SIA fields) to discover the entire mesh.

Request for Feedback

Mitretek Systems is presenting the dynamic bridge concept to the PKI community without intent to assert patent protection, in hopes that its utility may be assessed, and discussions started concerning the possibility of implementation.

Those interested in providing feedback, or joining discussions on the topic, are asked to contact the paper’s author, Mr. Ken Stillson of Mitretek Systems, at stillson@mitretek.org, or 703-610-2965. If there is sufficient interest, a mailing list or similar discussion mechanism will be established.

References

- [CAI-POC] K. Stillson, *HHS CAI Proof-of-Concept Results*. Prepared by Mitretek Systems for the Dept. of Health and Human Services, Sept. 2003
[Available upon request from HHS]
- [FBCA] CSRC/NIST *Federal Bridge Certification Authority*
See <http://csrc.nist.gov/pki/fbca/welcome.html>
- [HEBCA] Higher Education Bridge Certification Authority, web site.
see <http://www.educause.edu/hebca/>
- [HHSspider] K. Stillson *The Certificate Spider – A Simplified Technical Description*
Mitretek Systems. June, 2003. Prepared for Mark Silverman of the
department of Health and Human Services
[Available upon request from HHS]
- [Pathbuild] M. Cooper, Y. Dzambasow, P. Hesse, S. Joseph, R. Nicholas,
Internet X.509 Public Key Infrastructure: Certification Path Building
(Internet Draft) IETF PKIX Working Group. December 2003. See
<http://www.ietf.org/internet-drafts/draft-ietf-pkix-certpathbuild-03.txt>
- [PKI concepts] K.D. Stillson *Public Key Infrastructure Interoperability: Tools and
Concepts* The Telecommunications Review, Mitretek Systems,
December 2002. See
http://www.mitretek.org/publications/2002_telecomm_review/stillson_07.pdf
- [PKIX] Stephen Kent, Tim Polk (co-chairs) *Public-Key Infrastructure IETF
Working Group*. See <http://www.ietf.org/html.charters/pkix-charter.html>
- [RFC3280] R. Housley, W. Polk, W. Ford, D. Solo *Internet X.509 Public Key
Infrastructure Certificate and Certificate Revocation List (CRL) Profile*
IETF Network Working Group. April 2002
<http://www.faqs.org/rfcs/rfc3280.html>
- [Sun1] R. J. Perlman, Sun Microsystems, Inc *System and method for
shortening certificate chains* U.S. Patent Office, application
#20020147905 October 10, 2002. See
<http://appft1.uspto.gov/netahhtml/PTO/srchnum.html> (#20020147905)
- [X509] ITU-T (formerly CCITT). *Recommendation X.509: The Directory—
Public-Key and Attribute Certificate Frameworks*. 2000.