MECAI - Mutually Endorsing CA Infrastructure

A proposal to improve PKI, OCSP, CAs.
Version 1
21 October 2011

Kai Engert
kaie@kuix.de – kaie@redhat.com

Abstract: We need solutions for compromised CA authorities, the OCSP privacy issue, the CRL bandwidth issue, the OCSP stapling limitations. Recent proposals for improvement suggested secondary authorities, being potentially secondary points of failure. This proposal asks that CAs take over responsibility and provide mutual notary services, because CAs are the ones who receive monetary benefits. The proposal is to introduce VAs - Vouching Autorities.

The problem(s)

PKI does not provide a builtin mechanism for revoking trust anchors (CA root certificates).

Current PKI revocation mechanisms have disadvantages.

CRLs are too big to download over mobile devices or in low bandwidth environments.

Both OCSP and CRLs require a side channel which might be (temporarily) unavailable, or introduces unwanted latency, or even fails to work completely in certain environments (e.g. when paying or authenticating to use a wireless network).

OCSP stapling avoids the side channel, but the initial design is insufficient because multiple OCSP responses are required.

The proposed enhancement for OCSP stapling (draft-pettersen-tls-ext-multiple-ocsp-03) solves this limitation, but it still has another limitation: It cannot tell whether the CA has been compromised.

With OCSP and CRLs, there is no redundancy for CA revocation infrastructure. If the CA's CRL/OCSP servers are unavailable, a client is unable to verify the state of certificates. Therefore a provider of Internet services might be reluctant to introduce a strict dependency on the availability of such services.

The Convergence and Perspectives proposals are great ideas, but they also rely on a side channel, and they introduce the dependency on secondary providers, an additional point of failure.

Proposed solution: Summary

The Certificate Authorities are the ones who generate profit based on PKI, so it seems natural that they should be required to provide any additional services necessary to make PKI work more reliably.

This proposal asks that all CAs run an additional service, a kind of vouching service, similar to the ideas used by the Perspective and Convergence projects. Each CA must be willing to contribute to the new MECAI system. CAs that are unwilling to participate will get banned from the list of accepted CAs by software and community projects.

This proposal suggests that the vouching information gets delivered to clients as part of the client-server protocols, similar to OCSP stapling, avoding the need for side channels.

When a client connects to a server, the client may pick a vouching CA (or list of candidate vouching CAs) that it trusts.

Proposed solution: Detailed description

This section lists highlevel descriptions for the required elements of the proposed MECAI infrastructure and their behaviour.

The MECAI system makes use of Vouching Servers (VS), in which a CA acts as a Vouching Authority (VA).

Whenever this document uses the isolated term server, it refers to a server providing any service wrapped using the SSL/TLS protocol.

TL: The name of a Trust List. An example of a trust list is the Mozilla CA list, as embedded in the NSS security library, as managed by Mozilla.org in accordance with the Mozilla CA policy. Additional examples are the lists of trusted CAs as embedded in various operating systems.

Infrastructure: In the beginning of the implementation of this idea, it will be sufficient if a few friendly CAs act as VAs and operate VSs. However, in the overall completion of MECAI, after an announced deadline, each CA must demonstrate its cooperation by running an accompanying VS, or they will be removed from TLs.

The list of VA servers (hostnames) shall be announced by CAs and embedded into software by applications supporting MECAI.

Vouching Data (VD) is a data object that contains a host name, one or two server certificate(s), a vouching statement by the CA that guarantees that recent revocation status information (not revoked) was obtained by the VS, and a timestamp indicating when this information was obtained. The VD carries a digital signature issued by the VA, including the VA certificate used for creating the signature. A VA certificate is an intermediate certificate that was issued by the CA. The VA certificate shall use the „designated OCSP responder“ key usage.

Protocol: Similar to the OCSP stapling protocol (RFC 4366 section 3.6), a new extension for the handshake between an SSL/TLS client and a SSL/TLS server must be defined. A client C requesting a connection to a server S will include additional properties in the client hello part of a SSL/TLS handshake. It will request MECAI vouching status in the context of a named TL. It will include a list of candidate VAs (at least one) that the client will accept as a VA for vouching for the server's certificate. The server S is required to include a signed VD object, in which the VD was issued by one of the candidate VAs requested by the client. The server S is allowed to return multiple VDs, one for each requested VA. A client shall be happy if at least one VD can be validated by the client. This can be used to enhance reliability in case a VA is having issues.

Client behaviour: A client initiating a SSL/TLS connection to a Server S will refer to its builtin list of VAs. The client may decide on its own which VAs it prefers for the connection to S. The client may have a preference that is based on the toplevel domain of S's hostname, e.g. based on the country. Alternatively, the client might simply pick two VAs at random.

Client behaviour: In this initial proposal, a client will never connect to a VS directly.

Server behaviour: Servers supporting the MECAI protol extensions will connect to a VS in order to obtain VD in advance. (This is similar to OCSP stapling, where servers are expected to fetch OCSP

data in advance for their own server certificate). In fact, a server is expected to connect to many VSs, in order to be prepared for all potential requests from clients. For example, for each VA that is known to the server, the server might connect to the VS every 6 hours, obtain recent VD and cache it for later use. (However, a server might optionally choose to dynamically connect to new VAs that the server learns about, by way of an incoming client connection.)

Client behaviour: If no VD was received from the server, the client may stop the connection with an error. A client receiving VD 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 CA certificates that are built into the client. If the verification fails, or if the VD signature was created using a different CA, the client will stop the connection with an error. The error message shown by the client will name the VAs that were requested to provide the vouching information. This way users are able to notice and report that a particular VA is frequently unavailable.

VS behaviour: A VS will listen for Vouching Requests (VR) and reply with VD objects. A VR includes a hostname, a port number, the name of a trust list context (TL), a very recent timestamp and the server's certificate. In order to reduce the likelyhood of denial of service attacks, the VR must be signed using the server's certificate. After receiving the VR, the VS will attempt to validate the received certificate (including obtaining revocation information). If it can't, the connection will be dropped. If the timestamp is not recent, the connection will be dropped. If the VS cannot verify the digital signature, the connection will be dropped. Next, the VS will initiate a separate connection from the geographic location of the VS. It will use the DNS information available at the location of the VS to query the IP address of the requested hostname, and it will initiate a SSL/TLS connection to the indicated port number of this IP address. (This connection will operate without the MECAI extension.) The VS will use this connection to retrieve the server's certificate from the handshake. The VS will compare the server certificate retrieved with the server certificate contained in the VR. If they differ, the VS will drop the connection (to the original requesting client). However, if all tests have passed, the VS will create the VD and return it to the requesting client.

Server behaviour: Some providers of internet services might run a large number of servers. The use of a single hostname might actually be run by a large number of servers. From time to time a service will be required to update its infrastructure from an old certificate to a newer one. It is impossible to perform such a transition in the fraction of a second. This means, there will be times where more than one certificate is actively being used for a single hostname. The MECAI system must support this scenario. If a server sends a VR, it may actually send two VRs. Both VRs must be identical, except the certificate and the signature. When such a combination of two VRs is received by the VS, it will require that both signatures are valid. There is an additional requirement to ensure that the entity sending the VR is really in posession of the private keys for both certificates. (This additional complication can be avoided if both certificates contain the same public key.) Each VR must contain a first signature, wrapped with an outer signature. In other words, the VR containing the older cert must be signed by the older cert, wrapped with a signature made by the newer cert. The second VR containing the newer cert must be signed by the newer cert, wrapped with a signature made by the older cert. If the VS fails to verify any of the 4 signatures, it will drop the connection. The VS will then perform its usual probing, by obtaining the server certificate on its own, by starting connections from one or more geographical locations to the hostname. The VS will require that each probed server uses either the old or the new certificate. If all probes were successful, the VD produced will include the list of both certificates (allowing clients to accept either certificate as valid).

VS behaviour: A VA might choose to perform the hostname and server certificate query from more than one geographical location, in order to ensure they are identical from all „perspectives“.

Client behaviour: The SSL/TLS handshake might be enhanced with an additional feedback message. A

client that is unable to validate any of the VDs provided by a server, prior to dropping the connection to the server, might send a feedback message to the server. This could allow a server to learn if there are problems with the use of certain VAs. Also, if a client decides to continue the connection, because at least one of the VDs were valid, the client might still send a feedback message to the server, informing the server about VAs that cannot be verified.

VS behaviour: A VS is required to keep a list of the currently accepted root CA certificates (trust anchors) as accepted by each of the TL the VA supports. A VS is required to be in active human contact with the people that maintain the various trust lists (TL). Whenever the maintainers of a TL decide to revoke the trust of a root CA certificate and communicate that to the VA, the VA shall immediately reconfigure the configuration used by all its VS to remove the revoked certificate. This way the VAs will no longer vouch for certificates issued by the revoked CA.

Variation of this idea

If the Certificate Authorities cannot be convinced to participate in MECAI, the alternative is that the community provides this system on its own. For example, several independent organizations could step up to run the VA/VS service. For example, Mozilla.org could develop a policy how to approve vouching organizations, issue signing keys to them, and allow any applications to embed the public signing keys for performing the described verfification of VD.

Benefits of MECAI

Assuming the ideas presented in this proposal does not contain flaws, or only minor design flaws that can be fixed, the following benefits should be achieved by MECAI:

- Clients will eventually be able to require availability of valid VD data from servers
- information about revoked toplevel CA certificates need to be communicated to a limited number of systems, and will have the result that VAs are no longer willing to vouch for servers from compromised CAs, having the effect that clients will refuse to connect to systems using compromised certificates
- the need of side channels for CRL and/or OCSP is eliminated, which simplifies the checking of current trust information
- the load for CRL download servers and OCSP servers will be reduced, because clients no longer connect to them – only servers will
- it's no longer sufficient to compromise a single CA. A hacker obtaining a fraudulent cert faces the additional problem that the hacker must be able to present vouching data for the fraudulent certificate, which was produced by a different CA chosen by the client