Whitepaper

Controls for Using Certificates from Publicly-Trusted Authorities for Mutual Authentication

Table of Contents

Abstract 3

Assumptions and Scope 3

Background and Problem Statement 3

- Authentication 3
- Server Authentication 3
  - Using Certificates for *Server* Authentication 3
  - Using a Certificate from a Publicly-Trusted Issuer Solves the Problem of Pre-Configuring Trust 4
- Understanding Assurance 4
- Mutual Authentication 5
  - Using Certificates from an Internal Issuer for Mutual Authentication 5
  - Using Certificates from a Publicly-Trusted Issuer for Mutual Authentication 6

Solution 8

- Current Certificate Validation Controls 8
  - Require Unique Client Certificates 8
  - Disallow Wildcards in Client Certificates 8
  - Disallow Client Certificates that Have Long Validity Periods 8
  - Certificate Subject Name Checking 8
  - Validate Certificate Attributes 9
  - Pre-Associate Client Certificates 9
- Future Certificate Validation Controls 10
  - Certificate Transparency Framework 10

Conclusion 10

References 10

Abstract

Using digital Certificates from a publicly-trusted authority for Mutual Authentication exposes an organization to risks that require a well-defined strategy of compensating controls.

While the use of digital Certificates from a publicly-trusted authority for *server* authentication is ubiquitous – made possible by OS and browser updates which we rely upon to determine which Certificate authorities ("Issuers") to trust, relying upon that same framework of publicly-trusted Issuers for client certificates used in Mutual Authentication poses a risk of false-positive authentications and rogue Certificate usage.

Assumptions and Scope

This whitepaper assumes a general familiarity with:

1. Publicly-trusted Certificate Issuers such as Let's Encrypt, Sectigo (previously Comodo), DigiCert, GeoTrust, Entrust, GlobalSign, and
2. Digital certificate authentication for Server service applications.

The scope of this white paper is limited to the controls recommended for utilizing Certificates from publicly-trusted Issuers for Mutual Authentication.

The scope of this whitepaper does not include or address the use of digital Certificates from publicly-trusted Issuers for *user* authentication. Additionally, the scope of this whitepaper does not include or address core cryptographic details and recommendations such as X.509 standards, SSL/TLS fundamentals, algorithms, hashes, or cryptographic key strengths and lengths.

Background and Problem Statement

Authentication

The act of verifying a person or thing's identity is known as authentication. In cryptography, a digital certificate ("Certificate") is an electronic document used to prove that a specific public *key* belongs to the subject identified by the Certificate; Thus, verifying the subject's identity.

Server Authentication

Server authentication is the most common and familiar authentication scenario. In this scenario, a "Server" presents a Certificate to a "Client" to prove that **the Client can rely on the public key contained in the Server's Certificate in order to establish secure communications with the Server**.

Using Certificates for *Server* Authentication

In Server authentication, the Certificate's validity is verified in one or more ways - as determined by the Client's software (often a browser). This verification process can be as simple as verifying that the Certificate presented by the Server is not expired, is issued from an Issuer that the Client trusts, and has not been *revoked* by the Issuer.

A key requirement of the Certificate verification process is verifying that the Certificate comes from a *trusted* Issuer. If an organization is using an internal for Certificate issuance, the *trust* of that Issuer must be pre-distributed to all potential Clients. Inside an organization this trust

distribution can be easily automated and managed. However, pre-configuring that trust to Clients *outside* the organization poses an inconvenient problem of logistics.

Using a Certificate from a Publicly-Trusted Issuer Solves the Problem of Pre-Configuring Trust

If you wish to make your application services available outside of your organization, using a Certificate from a publicly-trusted Issuer solves the logistical problem of pre-configuration (of the trusted Issuer) because today's operating system and browser manufacturers handle the pre-distribution and update of publicly-trusted Issuers as part of their software update process(es).

Figure 1 - Server Authentication with Internal Certs

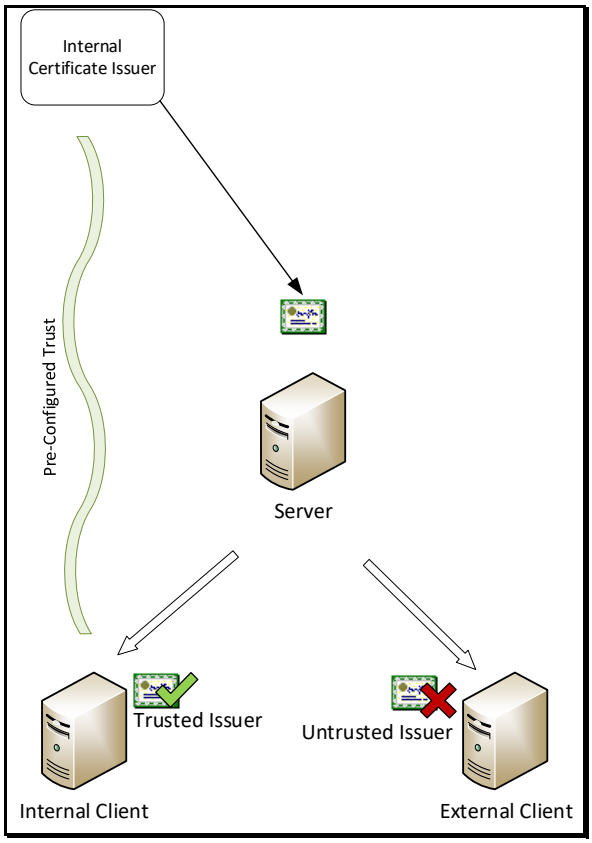
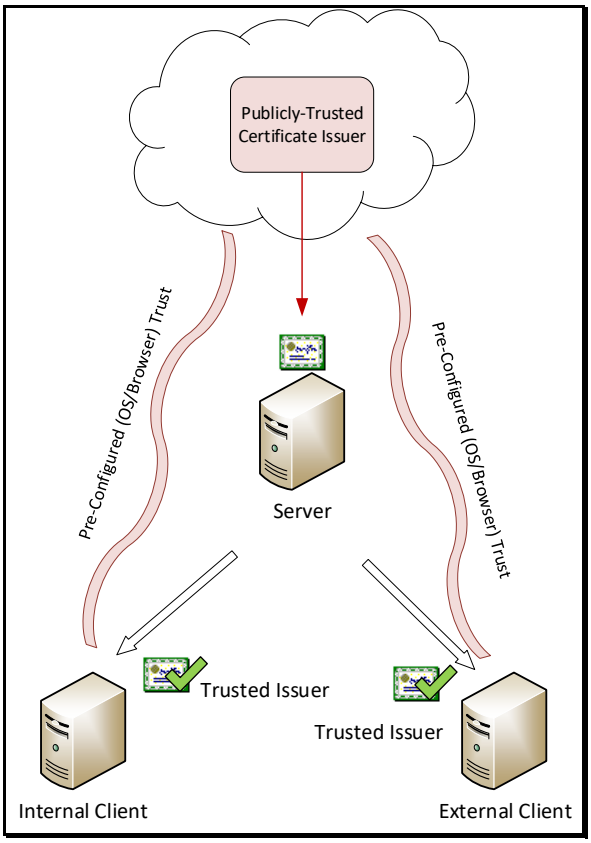


Figure 2 - Server Authentication with Publicly-Trusted Certs



Understanding Assurance

Assurance should be a core determiner of whether or not to trust a Certificate's Issuer. In this case assurance means the **rigor used to positively identify an entity or actor** before that entity or actor is issued a Certificate. With an internal Issuer, an organization can control and govern the policies and processes that define and support the rigor used to identify an entity. With a publicly-trusted Issuer an organization has little control over the rigor used to identify an entity. Most publicly-trusted Issuers only verify that a Certificate's name suffix (".organization.com" or "@organization.com") is owned by the organization that the entity claims to represent.

Mutual Authentication

For increased security, a Server application can be configured to only communicate with Clients who have been validated in a way similar to Server authentication. When a Client is required to verify its identity to the Server application using a Certificate (so that the Server can likewise rely on the public key contained in the Client's Certificate in order to establish secure communications with the Client) it is known as "Mutual Authentication".

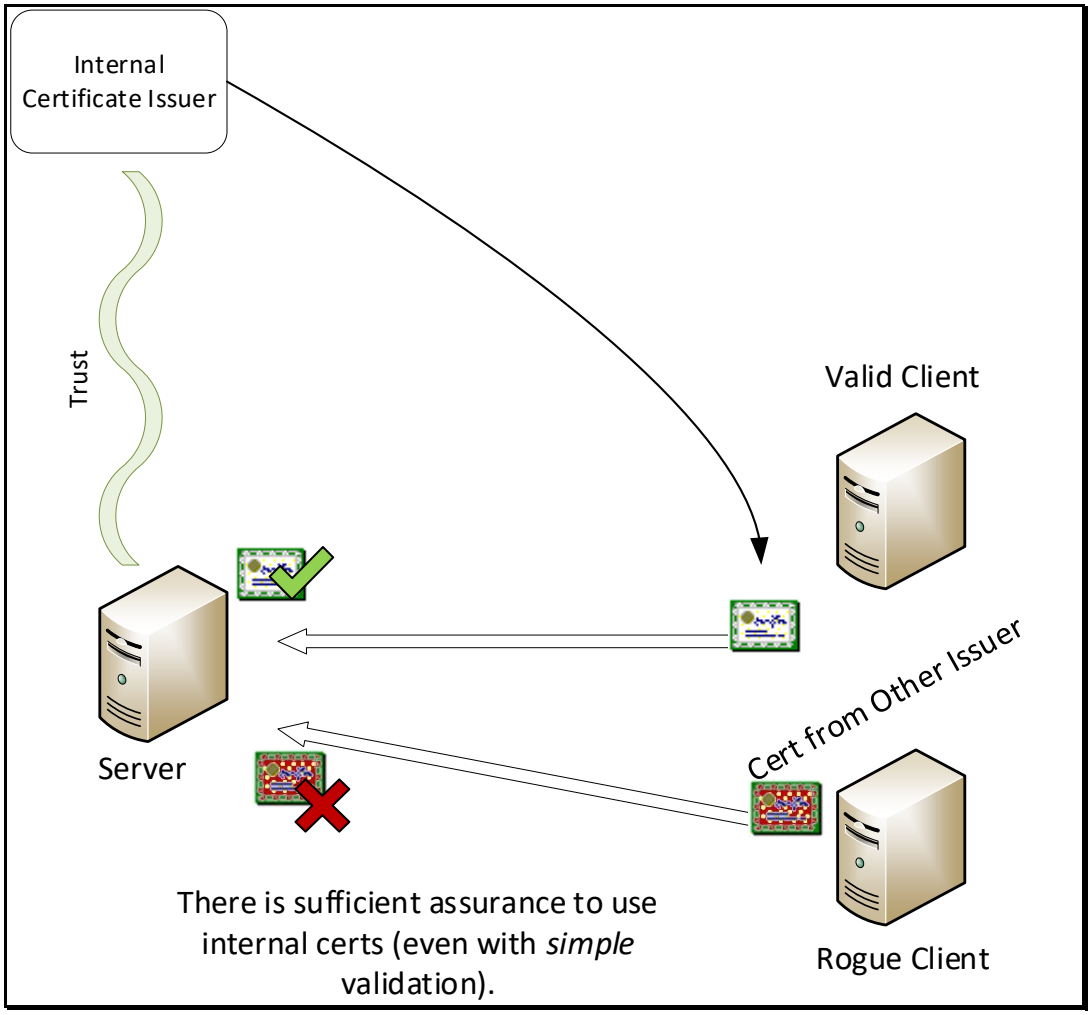
In many scenarios, a Server does not require any kind of authentication from a Client system. In some scenarios, a Server application may require that a Client system authenticate to a Server simply using basic authentication - using a pre-defined credential (user name and password) that is associated with an application service account.

However, due to credential vulnerabilities that include credential mismanagement, credential sharing, and insecurely storing passwords, some organizations choose to require that a Client use Mutual Authentication (verify its identity to the Server application using a Certificate) when access to the Server application is governed by regulatory requirements or is otherwise sensitive.

Using Certificates from an Internal Issuer for Mutual Authentication

Enabling or enforcing Mutual Authentication with Clients using an internal Issuer makes a lot of sense because an organization has **full control and governance over the issuance of internal Certificates** and can assume, with reasonable assurance, that a Mutual Authentication Certificate is coming from an **authorized** Client. In this scenario, the Server application may only require minimal configuration for basic validation of a Client Certificate for Mutual Authentication (for instance, *only* authenticate Clients that present a Certificate from the internal Issuer as seen in *Figure 3*).

Figure 3 - Simple Validation of a Certificate from an Internal Issuer May Be Equated to an Acceptable Level of Assurance



This configuration enables a potential for lower operational costs in scenarios where:

1. The assurance level of the certificate is used as a primary risk control for authentication, and
2. The primary Client base is internal.

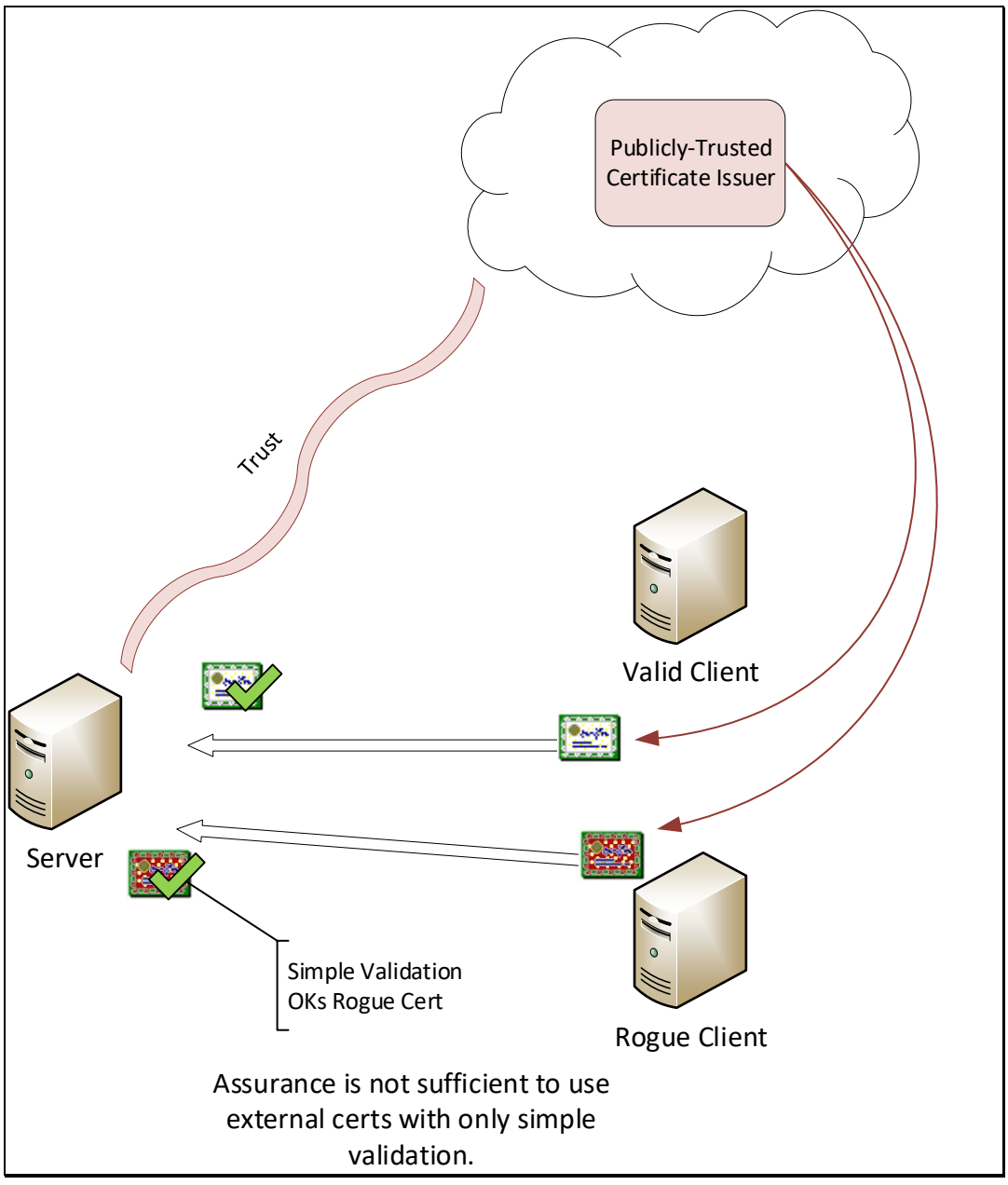
However, the operational overhead of internal Issuer Certificate distribution to external Clients has an operational cost generally proportional to the number of external Clients.

Using Certificates from a Publicly-Trusted Issuer for Mutual Authentication

Enabling or enforcing Mutual Authentication with Clients using a Certificate from a publicly-trusted Issuer **requires a well-defined strategy for Client Certificate validation** because of the risks introduced by the lower level of assurance with which Clients are able to obtain Certificates from the public Issuer. This lower assurance is a result of the inability for an organization to control the rigor used to positively identify the subject of the Client Certificate.

If you choose to allow Mutual Authentication with Certificates from a public Issuer, it is not enough for you to rely on only basic validation of Client Certificates. In this scenario, a Server must not be configured to *only* validate requests that come from a Client that presents a Certificate from a publicly-trusted Issuer because **doing so would mean trusting *all* Client Certificates issued from that publicly-trusted Issuer** as seen in *Figure 4*.

Figure 4 – Simple Validation of a Certificate from a Publicly-Trusted Issuer Does Not Equate to an Acceptable Level of Assurance



Solution

A combination of compensating validation controls must be put in place <u>at the Server application</u> if an organization plans to allow the use of Certificates from publicly-trusted Issuers for Mutual Authentication.

Current Certificate Validation Controls

A combination of the following controls may be implemented at the Server application to lower the risks of allowing the use of Certificates from publicly-trusted Issuers for Mutual Authentication. It is recommended that as many of the following controls be implemented as is possible.

Require Unique Client Certificates

Each Client must be issued a unique Certificate for Mutual Authentication, and Certificates must not be shared between Clients.

Disallow Wildcards in Client Certificates

A wildcard certificate is not unique. Therefore, the Server application must be configured to reject the authentication for any Client that presents a Certificate that uses wildcards as any part of its identity (*.organization.com).

Disallow Client Certificates that Have Long Validity Periods

The longer and more often any Certificate is used, the more susceptible its private key is to loss or discovery. Therefore, it is important that a Client periodically obtain new Certificates for Mutual Authentication. This control can be implemented passively - as Client acceptance of formal organizational policies, or actively by the Server application as a validation measure during authentication.

> **Note:** The length of the validity period should be determined by an organization's risk and compliance requirements. The CA/Browser Forum has specified that as of March 2018, the maximum validity period for an SSL/TLS certificate is 2 years.
>
> Further, Google has sponsored a new ballot with the CA/Browser Forum that, if passed, will reduce the validity period of SSL/TLS certificates as of March 2020 to 1 year.

Certificate Subject Name Checking

With this control, a Server application is configured to check the Subject name (and Subject Alternative Names) of any Client Certificate being presented for Mutual Authentication against either:

- **A name constraining whitelist** (a list containing acceptable names and name patterns)
- **A name constraining blacklist** (a list containing names and name patterns that are unacceptable)

This control does not normally result in excessive operational overhead as Client Certificates are renewed because Certificate names commonly remain unchanged.

Note: Active Directory (or another LDAP directory) may be used for whitelist checking where Client Certificates are validated only if their name matches an object (or object UPN) already defined in the directory.

Validate Certificate Attributes

In addition to the typical attribute validation checks (Subject, Subject Alternative Name, Valid From, Valid To, Issuer, Revocation status) certain other Certificate attributes may be checked during a Server application's authentication (or authorization) validation. Specifically, the Server application may be configured to verify one or more of the following Client Certificate attributes:

- **Enhanced Key Usage** (verify that the Certificate has been configured for a custom purpose as defined by your organization)
- **Application Policies** (verify that the Certificate is configured to conform to one or more custom policies of your organization)

With proper planning this control can be easily defined and implemented, and when paired with other controls (such as name checking) can serve to significantly reduce the risk of rogue Certificates being validated during Mutual Authentication.

Pre-Associate Client Certificates

A common control for Mutual Authentication is pre-configuration of the application Server with Certificate thumbprints (or public keys) of each Client *before* that Client attempts to authenticate. In this way, an organization effectively pre-authorizes only Clients that are known. This control, common in federation scenarios, is optimal for risk mitigation when using Certificates from publicly-trusted Issuers for Mutual Authentication.

Certificate Pinning

To further mitigate risk, an organization may also associate (pin) a Client Certificate's unique name, thumbprint, or public key with a *specific Client IP address* for more robust version of pre-association.

For flexibility, you may choose to pin a Client Certificate (that is otherwise validated with proper controls) to a Client's IP address *after* the Client's first successful authentication. This approach locks the future use of the Client Certificate to a single IP without the need to have previously defined the Client Certificate information at the Server.

Note: Pinning a Certificate to a unique Client IP should only be used where Client IP addresses are expected to remain relatively static.

Any type of pre-association controls (including Certificate pinning) generally result in increased operational overhead as Client Certificates are renewed because Certificate thumbprints and public keys change with Certificate renewal.

Note: In order for this control to be of value with publicly-trusted Issuers, the pinning *must* take place at the end-entity (leaf) certificate level and not any part of the Issuer's Certificate *chain*.

Future Certificate Validation Controls

Certificate Transparency Framework

In the industry certificate pinning associated with Server validation (from a Client's perspective), is being deprecated in favor of a validation process known as Certificate Transparency. This process can also be implemented in Mutual Authentication scenarios from the perspective of the Server application.

In simple terms, with this control you configure your application Server to leverage discrete software modules that allow the Server to validate a Client's Certificate against an open framework for monitoring and auditing SSL certificates. The validation is done using a cryptographically assured, publicly auditable (append-only) log or journal. This validation framework is underpinned by blockchain concepts.

Note: Certificate Transparency is a newer control that is not yet widely implemented.

Conclusion

With a well-defined strategy of compensating controls, an organization may be able to reduce or mitigate the security risk of accepting Certificates from a publicly-trusted Issuer for Mutual Authentication.

References

- Google's Certificate Transparency Project
  - <http://www.certificate-transparency.org/>
- RFC 5280: Format and semantics of certificates and certificate revocation lists (CRLs) for the Internet PKI
  - <https://tools.ietf.org/html/rfc5280>
- Catalin Cimpanu, *Google wants to reduce lifespan for HTTPS certificates to one year*, ZDNet, 17 Aug 2019
  - <https://www.zdnet.com/article/google-wants-to-reduce-lifespan-for-https-certificates-to-one-year/>
- CA / Browser Forum
  - <https://cabforum.org/>