# Contents

**Introduction** ....................................................... 3  
SSL/TLS Deployment Best Practices ........................................ 4
**Certificates and Private Keys** ........................................ 5  
Private Key Protection ................................................... 5  
Key Size ......................................................................... 6  
Signing Algorithm .......................................................... 6  
Certificate Validity Period ................................................ 6  
Certificate Revocation Status .......................................... 7  
Hostname Coverage ........................................................... 8  
Self-Signed Certificates .................................................. 9
**Server Configuration** .................................................. 11  
Valid Certificate Chains ................................................... 11  
Secure Protocols .............................................................. 12  
Secure Cipher Suites ..................................................... 12  
Renegotiation .................................................................. 13  
TLS Compression ............................................................ 13  
HTTP Compression .......................................................... 13
**Application Protection** ................................................. 14  
Malware .......................................................................... 15  
SQL Injection (SQLi) .......................................................... 15  
Cross Site Scripting (XSS) ............................................... 15  
Outdated and Vulnerable Applications ................................. 16  
Mixed Content .................................................................. 16  
Third Party Trust ............................................................... 17  
Secure Cookies .................................................................. 17
**Server Enhanced Security** ................................................. 18  
Perfect Forward Secrecy .................................................... 18  
Online Certificate Status Protocol (OCSP)  
Stapling ............................................................................ 19  
OCSP Must-Staple ............................................................ 20  
Elliptic Curve DSA (ECDSA) Private Keys ......................... 21  
HTTP Strict Transport Security (HSTS) .............................. 21  
HTTP/2 ............................................................................. 22  
Server Name Indication ................................................... 23
**Domain Protection** ...................................................... 25  
Certification Authority Authorization ..................................... 25  
Certificate Transparency .................................................. 26  
Public Key Pinning ............................................................ 28  
Certificate Reputation ..................................................... 29
**Advanced Certificates** ................................................... 31  
Multi-SAN Certificates ..................................................... 31  
Extended Validation (EV) Certificates .................................. 32  
Elliptic Curve Cryptography (ECC) Certificates .................. 33  
Private Trust Certificates .................................................. 34
**Reliable Certification Authority and Always-On SSL:**  
The Trusted Combination ................................................ 35  
Reliable CA ................................................................. 35  
Always-On SSL ................................................................ 35
**Tools** ............................................................................ 38
**Attacks** ....................................................................... 39
**About Entrust Datacard** ................................................ 40
**About Bruce Morton** ..................................................... 41
Introduction

There are many risks that come with the implementation of security using Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocol. Attacks can be formulated against the SSL/TLS protocol, the protocol may have been designed improperly by the server vendor, the certification authority (CA) could be attacked or you may implement SSL/TLS improperly on your server.

The Trustworthy Internet Movement surveys about 200,000 sites each month, grades them and ranks them by status through their SSL Pulse. Nearly a quarter of the protected sites receive an F.

These sites support SSL 2.0 and SSL 3.0, have insecure cipher suites, have small keys and support RC4.

The issues with the poorly graded sites are not system defects; they are the result of improper configuration and deployment.

SSL/TLS is deceptively simple. While it seems like it is easy to deploy, that’s not the way it works. Entrust SSL Experts have thoroughly reviewed the SSL/TLS deployment process and wish to offer you our best practices so you can spend the minimum amount of time possible on deployment and maintenance while still achieving the maximum results for your organization.

Pro Tip: Check out Entrust’s Best Practices Resources

Don’t forget to check out Entrust’s SSL/TLS Best Practices Page, part of our ongoing effort to provide you with the most up-to-date news on SSL/TLS vulnerabilities, improvements and resources.
Introduction

SSL/TLS Deployment Best Practices

Our goal with this guide is to provide you with tips and hints, expert deployment knowledge and thought leadership while encouraging you to take an Always-On SSL approach and recommending you acquire your certificates from a reliable CA.

The SSL/TLS Deployment Best Practices approach details all of the areas that should be deployed properly and regularly monitored and maintained when deploying SSL/TLS:

• **Private Key and Certificate** - Private Key Protection, Key Size and Signing Algorithms.
• **Server Configuration** - Valid Certificate Chains, Secure Protocols, Secure Cipher Suites, Renegotiation and Compression
• **Application Protection** - Malware, SQL Injection, Cross Site Scripting, Mixed Content, Third Party Trust, Secure Cookies
• **Enhanced Server Security** - Perfect Forward Secrecy, OCSP Stapling and HTTP Strict Transport Security (HTTPS)
• **Domain Protection** - Certification Authority Authorization (CAA), Certificate Transparency, Public Key Pinning, and Certificate Reputation
• **Advanced Certificates** - Extended Validation (EV), Multi-SAN, and Private Trust.
• **Always-On SSL** - protect all sites against common attacks (i.e., SSLstrip and Firesheep), provide security and privacy.

When deploying SSL/TLS it is great to have a CA who works with you as a partner. Consider a reliable CA which provides certificate management, certificate discovery and responsive CRL and OCSP responses. A reliable CA will also provide a wide variety of certificate types and flexible licensing models. There should also be great support including server installation, server certificate and website scanning.

Please review the SSL Best Practices and feel free to ask questions and provide comments.
Certificates and Private Keys

When it comes to security on the Internet, identity and authentication are important. Certificates are the ID cards of the security world, ensuring that a site is what it says it is. Private keys are the authentication, they let you in just like a key to your front door and the two work in tandem to ensure maximum security for your website.

Here’s a look at the specifics of Certificates and Private Keys, and the steps to follow to ensure your security is always top notch.

Private Key Protection

Server private keys must be protected and changed on a regular basis for maximum security.

Servers should always have physical and logical protection to keep attackers away. Each private key should also have password protection for backup storage.

In some cases, hardware protection of keys should be considered.

When a certificate is reissued, a new key should be generated. This will ensure old copies of the key no longer function. After a compromise or even a suspected security breach, be sure that a new key is generated and the old certificate is revoked.

Pro Tip: Check out Entrust’s Best Practices Resources

Don’t forget to check out Entrust’s SSL/TLS Best Practices Page, part of our ongoing effort to provide you with the most up-to-date news on SSL/TLS vulnerabilities, improvements and resources.
Certificates And Private Keys

Key Size

The recommended size for encryption keys is 2048-bit RSA. This is also the suggested maximum, as 2048-bit keys will continue to be endorsed for use until 2030.

There is no benefit in using a larger key size because it will decrease server performance. If you are planning to install your certificate on more than one server, consider using certificates with different keys. That way, if one key gets compromised then only the corresponding server will be compromised - not the others.

Learn more about the move to 2048-bit RSA keys.

Signing Algorithm

All servers and applications must support or be migrated to support SHA-2.

Microsoft and Google have implemented SHA-1 deprecation, meaning that sites still using the standard will no longer be viewed as trustworthy in their browsers. SHA-1 hashed certificates may already provide errors with Chrome.

In 2017, SHA-1 signed certificates will not be supported in Windows. Migration to SHA-2 will also require that the intermediate certificate also be signed with SHA-2. Most CAs will support SHA-2 and use the SHA-256 version of the hashing algorithm.

Learn more about the SHA-1 to SHA-2 migrations.

Certificate Validity Period

The maximum validity period for Organization and Domain validated certificates is 39 months and for Extended validation (EV) certificates is 27 months. Short certificate validity periods help protect your private key. If your certificate validity is dropped to one year, then the key can be updated every year and the possible attack period is lowered.

How about 6 months or 3 months? Your CA should support you in choosing the certificate validity period to suit your application.
## Certificate Revocation Status

How does the browser check certificate revocation status?

<table>
<thead>
<tr>
<th>How</th>
<th>Definition</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Certificate Revocation List (CRL)</strong></td>
<td>A signed list of the serial numbers of all revoked certificates that were signed by the CA’s certificate.</td>
<td>A single point of reference for the status of all certificates issued by the CA’s certificate.</td>
<td>Over time, CRLs might become very large, resulting in unacceptable latency. An attacker may be in a position to block the CRL delivery.</td>
</tr>
<tr>
<td><strong>Online Certificate Status Protocol (OCSP)</strong></td>
<td>A signed response containing the status of one certificate.</td>
<td>An OCSP response is small and does not grow. As such, there is no latency due to size.</td>
<td>Browsers have to obtain an OCSP response for each certificate in the Web server’s certificate chain, requiring it to open additional connections, thereby impacting page load time. Privacy issues may be a concern as the CA can determine which websites a user is visiting. An attacker may be in a position to block the OCSP delivery.</td>
</tr>
<tr>
<td><strong>OCSP Stapling</strong></td>
<td>A signed response, fetched by the Web server, with the status of its certificate. The OCSP response is then provided by the Web server to the browser.</td>
<td>No privacy issues, as the CA does not know which user has asked for the OCSP response.</td>
<td>Need Web servers and browsers that support OCSP Stapling. An attacker may be in a position to block the OCSP delivery.</td>
</tr>
<tr>
<td><strong>Blacklist (for example, CTLs or CRLSets)</strong></td>
<td>A list of certificates that should not be trusted (whether or not they were revoked), distributed by the browser supplier.</td>
<td>The blacklist is distributed by the browser supplier as part of the browser executable. Any certificate on the blacklist can be rejected without any additional checks.</td>
<td>For practical reasons, the list is incomplete.</td>
</tr>
</tbody>
</table>
Certificates And Private Keys

Soft-fail vs. Hard-fail

The major concern for browser users is the policy of soft-fail versus hard-fail. The issue is that the certification authority (CA) CRL/Online Certificate Status Protocol (OCSP) response may not get delivered to the browser. This could occur as a result of a non-malicious failure somewhere in the infrastructure. But, it could also occur as a result of an attack.

Browser designers have determined that the former explanation is overwhelmingly more probable, so they have chosen a soft-fail policy. This means if there if the browser receives no response, then the certificate will be considered good, and the browser will allow access to the associated content.

A hard-fail policy, on the other hand, would mean that if the browser received no response, then the certificate would be assumed to be revoked, and the browser would block access to the content.

As of this writing, no major browser implements hard-fail.

The best practice is to work with a CA which provides CRL and OCSP responses in a reliable way. A timely response will help all users to quickly connect to your site securely with HTTPS.

In later sections we will discuss OCSP Stapling and OCSP Must-Staple.

Hostname Coverage

To guarantee maximum coverage, be sure that both the www and naked version of your domain resolve to the same web server and are protected with SSL.

You never know how a user will get to your site; therefore, ensuring both www.example.com and example.com are re-covered is vital.

While there is no functional difference between the two standards to the user, we are finding that more people are dropping www from their standard web surfing practices.
Certificates And Private Keys

Self-Signed Certificates

Self-signed certificates are not recommended for use by websites.

A self-signed certificate is an identity certificate that is signed by the same entity whose identity it certifies. In other words, it has not been verified by anyone other than the owner of the site. This means that users must trust that the owner is who they say they are, and the certificate provides no guarantee that proper security procedures have been followed.

Owners of self-signed certificates issue to their own policies, may not follow industry guidelines or best practices, and have not been audited.

Self-signed certificates do not receive the same trust certification in the browser as CA-signed certificates and are more prone to containing expired certificates.
Certificates And Private Keys

Self-Signed Certificate Model

- Owner says who they are
- Owner issues based on their own policy
- Owner is responsible for quality
- Owner may not follow industry guidelines
- Owner may not provide certificate status
- Compromised certificates may not be able to be revoked
- Owner is not audited
- Issuer of certificate may not be authorized by the domain owner
- Certificates may not be renewed if there are no reminders
- Self-sign certificate model does not provide trust and the browser provides a trust dialogue box to indicate such

Publicly-trusted CA-Signed Certificate Model

- CA verifies the owner of the domain and the certificate applicant
- CA issues to a policy in conformance with the requirements of the browser and operating system vendors. The requirements include the CA/Browser Forum Baseline Requirements, Extended validation (EV) Guidelines and recommendations from NIST.
- CA provides quality. Certificate checks are made to ensure no compromised keys are used while also following guidelines that ensure minimum key size, proper hashing algorithms, maximum validity period and proper certificate extensions are also used.
- CA updates policy based on industry best practices
- CA provides certificate status through CRL and OCSP
- Compromised certificates can be revoked
- CA is audited according to certificate issuing criteria such as WebTrust for CA, WebTrust for EV and SSL Baseline Requirements
- Certificate requesters for a Domain validated certificate are authorized by the owner of the domain. Requesters for Organization and Extended Validation certificates are authorized by a member of the organization specified in the certificate.
- CAs provide multiple reminders to ensure the certificates are renewed before they expire. CAs may also provide certificate discovery tools to find certificates on your systems which may not have reminders.
- Publicly trusted CA model is based on the CA being a trusted third party to the browser/OS vendor, the website certificate subscriber and the end-users of the website. The CA is obligated to meet the requirements of all three parties.

The best practice is to use an SSL/TLS certificate issued by a Reliable CA.
Server Configuration

Ensuring your server is properly configured is one of the best ways to guarantee its security. While SSL/TLS is a rock solid standard, an improper configuration can create holes that compromise security.

The following sections provide an overview of some of the main tools and solutions that can be used to ensure your server is configured correctly and that any potential holes are plugged.

Valid Certificate Chains

CAs are no longer allowed to issue SSL certificates directly from a root. The SSL certificates have to be issued from an issuing CA, also known as an intermediate or subordinate CA. The issuing CA may be one or more levels below the root CA. As such, when installing your SSL certificate, you must also install all of the issuing CA certificates.

*With all issuing certificates installed, then there will be a valid trust chain to the root.*

Over time, you might have to update your issuing CA certificate. For instance, the issuing CA certificate should always have a validity period longer than your SSL certificate. Also in the case of crypto migration, such as moving from SHA-1 to SHA-2, you will also need to install a SHA-2 issuing CA certificate.

You should not have to install a root certificate. The browser already has a set of trusted root certificates. Adding in another root certificate will only make your SSL handshake period longer, which will make the secure session longer to establish.

**Pro Tip: Check out Entrust’s Best Practices Resources**

Don’t forget to check out [Entrust’s SSL/TLS Best Practices Page](#), part of our ongoing effort to provide you with the most up-to-date news on SSL/TLS vulnerabilities, improvements and resources.
Server Configuration

Secure Protocols

The SSL/TLS protocol has evolved over the last 20 years to become an industry standard in security. However, there have recently been some important updates.

SSL 2.0 and SSL 3.0 should not be used.

New issues with SSL 3.0 were brought to attention with the POODLE vulnerability, so currently browsers do not support SSL 3.0.

When deploying SSL/TLS, be sure to implement TLS 1.0 and 1.1 for backwards compatibility. TLS 1.2 should also be implemented as this version does not have any known security issues.

Please note that some implementations of TLS 1.0, 1.1 and 1.2 were vulnerable to POODLE and upgrades should be implemented for mitigation.

Learn more about TLS 1.2 here.

Secure Cipher Suites

In order to secure data as it is communicated, the SSL/TLS protocol uses cipher suites. Each cipher suite is made up of algorithms to perform key exchange, authentication, ciphering and message authentication. Many cipher suites are configured on the server to support all browsers or applications which will address the server. Over time, the cipher suite may use algorithms which become insecure.

Cipher suites should use the minimum 128-bits for authentication and encryption.

RC4 cipher should not be used as it is susceptible to attacks and may bring trust indicators in browsers.

To mitigate FREAK, server administrators should disable support for any export RSA suites.

To mitigate the Logjam vulnerability, its recommended that server administrators disable support for all export cipher suites including DHE_EXPORT. Administrators are also encouraged to use either 1024-bit DHE with a freshly generated group, or deploy ECDHE as an alternative.

Please also review the researchers Diffie-Hellman deployment guide which discusses disabling export cipher suites, deploying elliptic-curve Diffie-Hellman (ECDH) key exchange, and generating a strong unique Diffie-Hellman Group.

Gibson Research and Mozilla have provided lists of recommended cipher suites.

- https://www.grc.com/miscfiles/SChannel_Cipher_Suites.txt
- https://wiki.mozilla.org/Security/Server_Side_TLS#Recommended_configurations
Server Configuration

Renegotiation

Renegotiation allows the server and browser to stop exchanging data in order to renegotiate a new SSL/TLS handshake. Client-initiated renegotiation is not required, and if enabled could create the opportunity for a denial of service (DoS) attack.

Renegotiation initiated by the server may be an issue if the server uses insecure renegotiation.

*With insecure renegotiation, there is no continuity between the old and new TLS streams.*

As such the renegotiated handshake could be done with a different client browser. Servers should be able to be configured for secure renegotiation. If secure renegotiation is not available, then renegotiation should be disabled.

TLS Compression

The CRIME attack showed that information leakage due to TLS compression could be used to uncover sensitive data.

**TLS compression should be disabled on the server.**

HTTP Compression

The TIME and BREACH attacks disclosed an issue with HTTP compression. HTTPS compression is very useful, so removal is not a practical option. The attack is difficult to implement, and as such, server changes are not recommended.

The TIME and BREACH attacks only make SSL 3.0 and TLS 1.0 vulnerable, so as we move to TLS 1.1 and 1.2, the attack will be mitigated.
Application Protection

Once your server is configured, you need to ensure your application is safe and protected. Consider Application Protection as your method for website security.

You need to be positive that the application running on the server is safe. It must not run malware and should not suffer from common mistakes such as cross-site scripting (XSS) or SQL injection (SQLi).

In some cases, while implementing Application Protection, you might find that your website has been search engine blacklisted. In other cases, the website may have mixed HTTP and HTTPS content or have not used secure cookies.

Both of these errors can be detected with Site scanning, so you can implement Application Protection.

Pro Tip: Use Trusted Layered Website Security

Why Layer Website Security?

• Use SSL encryption to secure websites and identities
• Find and remove malware
• Prevent domain blacklisting
• Eliminate website vulnerabilities
• Remove network weaknesses

Learn more about Entrust’s Website Security Bundles here.

Pro Tip: Check out Entrust’s Best Practices Resources

Don’t forget to check out Entrust’s SSL/TLS Best Practices Page, part of our ongoing effort to provide you with the most up-to-date news on SSL/TLS vulnerabilities, improvements and resources.
Application Protection

Malware

There are over 5 billion attempted malware attacks a year.

Malware can be injected using malicious JavaScript. A typical website can have hundreds or even thousands of potential vulnerabilities for malware injection because they are not easily detected.

Once malware is in place on a site, it can spread viruses, steal personal or financial information, and hijack computers or infect your customers’ computers after they visit your website. Ultimately, this negatively affects the site’s reputation and can result in lost business.

Sites should be scanned for malware on a daily basis.

If malware is detected on a website by search engines, it will result in blacklisting, which means wasted marketing and website design efforts. If your website is directly or indirectly linked to third party blacklisted site, then your site could also be blacklisted.

Blacklists should be monitored daily to ensure that your site does not appear on the list.

SQL Injection (SQLi)

SQL queries may be subject to manipulation. An attacker may be able to enter a malicious SQL statement into a query which could be executed.

We refer to this as an SQL injection (SQLi). The SQL strings could then be used to access sensitive information stored in the database.

Sites should be scanned daily to see if they are vulnerable to SQL injection techniques.

Cross Site Scripting (XSS)

Cross-site scripting (XSS) can be done in two classifications: reflected and stored.

Reflected, or non-persistent, XSS occurs when the script or coding that an attacker has created is sent via a third-party tool, such as an email.

A script is sent in the email asking the victim to click on a link and verify his or her logins, or to access a specific website. When the victim clicks on the link, the code will be sent to the web application and then returned to the victim, which executes the code or script.

Any information the victim enters can then be sent to the hacker and session cookies can be stolen.
Application Protection

Stored, or persistent, XSS occurs when the script or malware is stored directly on the web application.

Stored XSS attacks are the most devastating, as they affect all visitors to that specific page or link.

The Open Web Application Security Project (OWASP) provides methods to avoid XSS.

Sites should be scanned for cross-site scripting vulnerabilities on a daily basis.

Outdated and Vulnerable Applications

Over time, applications can become vulnerable to code weakness, outdated versions and weak policies. As applications age, attackers will find ways to compromise these programs. The impacts of these types of attack are wide-ranging, and greatly depend on the type of application that is attacked.

Application vulnerabilities can be:

• SSL/TLS Renegotiation DoS
• HTML injections
• Cross-site Scripting (XSS)
• SSH server type and version
• Robots.txt file
• ICMP timestamp request remote date disclosure
• Cookie injection
• SQL injection (SQLi)
• Web server info.php / phpinfo.php script detection

Website owners should consider scanning their applications on a daily basis to search for common vulnerabilities and unpatched software.

Mixed Content

If an HTTPS page includes content retrieved through regular, cleartext HTTP, then the connection is only partially encrypted.

The unencrypted content is accessible to sniffers and can be modified by man-in-the-middle attackers.

All content retrieved from other sites must also be encrypted to mitigate mixed content vulnerability.
Application Protection

Third Party Trust

If you use third-party services which are activated via JavaScript from another server, then, in essence, you must trust the third party. If the third-party is attacked or is negligent, then your site could be attacked.

Please consider the source of your third-party services and ensure you trust the third-party before implementation.

Secure Cookies

Non-secure cookies are subject to man-in-the-middle attacks.

The server should always use secure cookies.
Server Enhanced Security

Having enhanced server security in place is critical to defending your website from the most devastating attacks, exploits, and other security threats.

By deploying techniques such as Perfect Forward Security, OCSP Stapling, ECDSA Private Keys, HTTP Strict Transport Security (HSTS), and HTTP/2, you can increase performance and ensure that your websites are always protected.

Perfect Forward Secrecy

**Perfect Forward Secrecy (PFS)** ensures there is no link between the server’s private key and each session key.

If both client and server support PFS, they use a variant of a protocol named Diffie-Hellman (after its inventors), in which both sides securely exchange random numbers and arrive at the same shared secret. It’s a clever algorithm that prevents an eavesdropper from deriving the same secret, even if the eavesdropper can view all the traffic.

All browsers support PFS when using ECDHE cipher suites with RSA and Elliptic Curve Cryptography (ECC) SSL certificates. All browsers except IE also support DHE with RSA certificates.

You can also test your browser to see if it supports PFS. If your website needs to support older browsers that may not support PFS, you’ll have to configure your web server to also offer non-PFS suites.

Implementation of PFS will help mitigate Pervasive Surveillance.

**Pro Tip: Check out Entrust’s Best Practices Resources**

Don’t forget to check out Entrust’s SSL/TLS Best Practices Page, part of our ongoing effort to provide you with the most up-to-date news on SSL/TLS vulnerabilities, improvements and resources.
Online Certificate Status Protocol (OCSP) Stapling

Digital certificate status is provided by the certificate revocation list (CRL) and online certificate status protocol (OCSP).

The CRL is a list of all certificates that have been revoked. If the certificate serial number is not on the list it is assumed to be good. OCSP provides a response for all certificates. In layman’s terms, the response is either good or bad.

There are debates as to which method is the most valuable, however, it comes down to the certification authority (CA) and the user’s site access methodology. If the CA revokes a lot of certificates and thus manages a large CRL, then using CRL is bad because the large file takes a long time to download.

On a mobile device, the user may not want to download a large file and may not have the room to store the file. As such, in the mobile world, OCSP is the favored methodology.

For more information on OCSP, please see this pair of memos from the Internet Engineering Task Force (IETF): RFC 5019 and RFC 6960.

OCSP responses can be provided in two ways.

1. The most common method is for the CA to operate an OCSP service. When a browser wants to find out the status of the certificate, it finds the OCSP site from an extension in the certificate and checks to see if the certificate is good or bad. This requires the browser to rely on a service being provided by the CA. Unfortunately, some CAs are not good at providing their OCSP responses. In some cases, there is no service and in other cases the service is just way too slow. Slow service means it provides latency on the website as it tries to load up.

2. The second alternative is OCSP stapling. In OCSP stapling, the Web server obtains the OCSP response from the CA. When a browser comes to the site, the OCSP response is stapled to the SSL handshake. This means there is no extra connection to the CA’s OCSP service. The result is less latency and a faster loading website. It also allows the website owner to manage their own performance by increasing the throughput of their servers as their website gets more popular. There is also an upside for the CA, as it does not have to compensate for additional performance for highly active websites.

If you are running a website and want to decrease latency, consider implementing OCSP stapling. You will have to find out if your server supports stapling. The following servers support OCSP stapling:

- Apache 2.3.6+
- Microsoft IIS 7+, where stapling is enabled by default
- NginX 1.3.7+
Server Enhanced Security

**OCSP Must-Staple**

The OCSP Must-Staple solution can help resolve the OCSP problem. If the Web server could securely tell the browser that it supported OCSP Stapling, then the browser would know to expect an OCSP-stapled response. And if no response was received, the browser could hard-fail.

The website administrator has to determine if their site will support OCSP Must-Staple. First, they will have to have their website support OCSP stapling, then they must add the OCSP Must-Staple flag. The design is not finalized, but the OCSP Must-Staple flag can be implemented in two ways:

1. **Must-Staple Assertion in the SSL Certificate**
   
   In this case, the website administrator has to advise its CA that it wants OCSP Must-Staple. The CA will put an object identifier (OID) extension in the SSL certificate indicating Must-Staple. When a user goes to the website, the browser will review the certificate and see the OCSP Must-Staple indicator. It will then require an OCSP-stapled response from the Web server. If no response is received, then the browser will hard-fail.

   There is a draft proposal for an IETF RFC on OCSP Must-Staple.

2. **Must-Staple Assertion in the SSL Header**

   A more immediate solution to OCSP Must-Staple would be to include the flag in an HTTP response header. Mozilla developers are currently working on this solution.

   In this case, the Web administrator will add a Must-Staple response header to their Web server responses. The header will include a max-age specification, which will tell the browser that the Must-Staple flag is valid for a certain period of time. The browser will then cache the Must-Staple information.

   The next time the browser goes to the website, it will know that this is a Must-Staple site. If no OCSP staple is received, then the browser will hard-fail.

   This solution does have a “first visit” problem (i.e., Trust On First Use or TOFU). This means that until a browser has visited a site, it will not have the Must-Staple information. This would allow an attacker to interfere with the browser’s first visit to a website. This makes the solution good, but not perfect. There is also the possibility that the attack can be mitigated using a preloaded list of Must-Staple sites.

Here is a summary of OCSP Must-Staple:

<table>
<thead>
<tr>
<th>How</th>
<th>Definition</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCSP Must-Staple</td>
<td>The flag is implemented as a specific object identifier (OID) extension in the SSL certificate</td>
<td>No “first visit” problem - all connections to the Web Server carry the Must-Staple flag.</td>
<td>Web server needs a certificate issued with the OCSP Must-Staple flag.</td>
</tr>
</tbody>
</table>
**Server Enhanced Security**

<table>
<thead>
<tr>
<th>How</th>
<th>Definition</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCSP Must-Staple (assertion in HTTP Response)</td>
<td>The flag is implemented as an HTTP Response Header</td>
<td>Works with existing SSL certificate.</td>
<td>“First visit” problem</td>
</tr>
</tbody>
</table>

OCSP Must-Staple removes most of the issues with traditional revocation checking and allows the browsers to implement a hard-fail policy. Although there are some cons listed, these are basically items that will be resolved as the deployed browsers and Web servers support OCSP Stapling and Must-Staple.

Currently, all of the new desktop browsers support OCSP stapling. Regarding Web servers, Microsoft IIS by default supports OCSP Stapling and versions of Apache and Nginx can be configured to support OCSP Stapling. Other servers such as F5 will soon support OCSP Stapling as well.

**Elliptic Curve DSA (ECDSA) Private Keys**

Almost all servers use RSA keys. We mitigate private key attacks by increasing the key size. For instance, we have moved from 1024 to 2048-bits. However, increased size will decrease performance.

Elliptic Curve Cryptography will provide strong security with smaller keys. As a result, servers are now starting to support ECDSA keys. The issue is the browser software must also support ECC.

In the short-term, if you want to deploy ECDSA, you will likely also have to support RSA keys for backwards compatibility. Some servers will allow you to have both an RSA and an ECDSA key. This will allow for migration away from RSA.

**HTTP Strict Transport Security (HSTS)**

HTTP Strict Transport Security (HSTS) is a great way to secure your website. Implementation of HSTS is an extension of the Always-On SSL policy.

For each website you want to protect with HSTS, you must first deploy an SSL/TLS certificate (if you haven’t already), and configure that website to be accessible only via HTTPS, not via HTTP. Then you convey to HSTS-enabled browsers that your site is only available with HTTPS by sending the HSTS header value. Supporting browsers will automatically change any HTTP query for your website into an HTTPS query.

**If there is no HTTPS version available, then the browser will provide a trust dialogue to the user.**

HSTS is defined in the IETF RFC 6797 and is being deployed in most browsers. Browsers that do not support HSTS will ignore the HSTS header value, so website administrators do not have to wait for full browser support.
Server Enhanced Security

Mitigated Risks

HSTS will help to mitigate attacks using the sslstrip tool, which will take a request for an HTTPS domain and change it to a request to a similar site with an HTTP domain. With HSTS, the name change from HTTPS to HTTP will be detected by the browser.

HSTS will also mitigate the following security issues:

- User bookmarks or manually typed http domains will be redirected to HTTPS for the target domain
- An HTTPS site that inadvertently contains an HTTP link will be redirected to HTTPS for the target domain
- In the case where a man-in-the-middle attack redirects the user to an invalid certificate, HSTS will not allow the user to override the invalid certificate message

Even if your site does not host or require any sensitive data, protecting your website with HTTPS provides a consistent look and feel to all parts of your website, and ensures that user communication is secured.

Support

HSTS is supported on Chrome, Firefox, Opera and Safari. It will also be supported by Windows 10 when it is released in 2015. You can check “Can I use Strict Transport Security?” for the browsers that are HSTS-supported.

What About an HTTPS Redirect?

Most website administrators are quite familiar with redirects and it’s natural for them to suggest using this technique to force connections from HTTP to HTTPS as an alternative to HSTS.

Unfortunately, the redirect requires an initial connection to the server over HTTP that is susceptible to an attack.

An adversary could obtain non-secure session cookies when the browser initially connects to the site, and then can also inject malicious content - such as a fake login page - in the non-encrypted response. Once the browser is aware that a site requires HTTPS via HSTS, it is more secure than a simple redirect.

HTTP/2

HTTP/2 is the next major version of the network protocol used by the web.

HTTP/2 is based on Google’s protocol SPDY and has been implemented on Chrome, Firefox, Opera, Safari, and other browsers. The goal of HTTP/2 is to allow client/servers to choose a protocol, maintain compatibility with HTTP/1.1, decrease page load latency and support common existing uses of HTTP.
Server Enhanced Security

Both Google and Mozilla development teams have announced their intention to support HTTP/2 over TLS only. Other browsers are expected to follow, making the standard mandatory. As such, to get the greatest benefits of HTTP/2 it must be deployed with TLS.

Learn more about HTTP/2 Security or review http2 explained here.

Server Name Indication

A dilemma may arise where you want to continue to deploy your web service but are running out of IPv4 addresses. You consider deploying multiple virtual servers that will use the same IP address. However, your thought is that you can only have one SSL certificate per IP address. How will you make your service secure?

Server Name Indication (SNI) is an extension to the SSL/TLS protocol that allows the browser or client software to indicate which hostname it is attempting to connect. SNI is defined in RFC 6066.

By supporting SNI at the server, you can present multiple certificates and support multiple servers at the same IP address. Since the client indicates the hostname, the server can select the correct certificate to complete the SSL handshaking process.

For SNI to be effective it must be implemented by the majority of browsers, which thankfully is the case today. If a browser does not support SNI, then a default certificate will be presented. If the certificate does not support the domain, then a certificate warning will be seen. Of course, there will be no warning if the certificate is a wildcard supporting the requested subdomain.

SNI can abort a connection in some man-in-the-middle (MITM) attacks. For instance, if the browser requests a connection to the server with a specific domain name, but receives a certificate that does not support the proper name, then a certificate warning will appear which could indicate a MITM attack.

SNI can be deployed with many certificates or one certificate. Many certificates would just mean that individual certificates would be installed to support each domain name. Alternatively you might want to group related sites together and leverage one certificate; a wildcard or multi-domain certificate for example. The multi-domain certificate allows many domain names to be added to the subject alternative name (SAN) field with a light performance cost due to the size of the certificate. As more domains are added, the multi-domain certificate can be updated, or new certificates can be added.

The disadvantage is SNI is not supported by all clients (such as Windows XP). Consequently, Internet Explorer running on XP will not work, and although Windows XP is no longer officially supported, Microsoft does offer special long-term support for enterprises willing to pay. Alternative browsers will help mitigate the XP problem as Chrome, Firefox and Opera will all support SNI on XP.
Server Enhanced Security

An alternative to supporting security with fewer IP addresses is not to deploy SNI, but use a single multi-domain certificate. This solution is limited if the domain names are owned or controlled by different entities. There can also be a certificate management issue as the multi-domain certificate will have to be updated every time a new domain needs to be supported.

The advantage of SNI is scalability. SNI will allow you to deploy SSL with fewer IP addresses and fewer servers. It will allow unique certificates to be used for different sites, identities and brands which may improve security and trust. As we see Windows XP users fade away, SNI will be the best alternative to support many domains with fewer IP addresses.
Domain Protection

Many CAs can issue certificates for your domain. There are a few ways to protect or monitor certificates for your domains so you can be sure you’re safe from a mistaken implementation or an attack.

• Certificate Authority Authorization (CAA)
• Certificate Transparency (CT)
• Certificate Reputation
• Public Key Pinning

Pro Tip: Check out Entrust’s Best Practices Resources

Don’t forget to check out Entrust’s SSL/TLS Best Practices Page, part of our ongoing effort to provide you with the most up-to-date news on SSL/TLS vulnerabilities, improvements and resources.

Certification Authority Authorization (CAA)

One of the issues of having many public CAs is that any or all can issue SSL certificates for any domain. This would be upsetting to a subscriber that has reviewed the SSL industry and has chosen a CA that they can trust and work with. Another CA can issue a certificate for their domain at the same time.

This happens all the time, but in an honest way. When an enterprise user wants a certificate and doesn’t know that the company has already negotiated a pre-existing relationship with a CA, he just orders the certificate online from a CA of his choosing. Unfortunately, the certificate could be issued by the CA to an attacker. This is what happened in 2011 at both Comodo and DigiNotar. In both cases, the attacker found a way to get a CA to issue certificates for domains even though the owners of the domains were not their customers.

The result is that many industry experts have been looking for ways to stop CAs from issuing certificates for a domain that the registrant did not authorize. The solution to this problem is Certification Authority Authorization (CAA). CAA is specified through RFC 6844.
Domain Protection

CAA is a DNS resource record that allows a DNS domain name holder to specify one or many Certification Authorities authorized to issue certificates for that domain. The specification can help you limit trust to one CA, or may help you take advantage of a volume discount negotiated with your preferred CA. CAA records can also provide the CA with contact information to allow the CA to notify the domain owner when they received a request which did not match the CAA record preferences.

Publication of CAA resource records will allow public CAs to implement additional controls to reduce the risk of unintended certificate issuance. The benefit to the CA is as follows:

- Increases the reliability of a validated domain name
- Decreases chance of miss-issuing a certificate to a domain with a high phishing risk
- Reduces CA’s attack profile

Although implementation of CAA is not mandatory, all CA’s were obligated to disclose their CAA policy through their Certificate Policy of Certification Practice Statement by April 2015.

Consider protecting your domain name using CAA.

Certificate Transparency

Certificate Transparency (CT) is another proposed method to resolve the issue of many public CAs having the ability to issue SSL certificates for any domain. CT has the following goals:

- To make it impossible (or at least very difficult) for a certification authority to issue a certificate for a domain without it being visible to the owner of that domain.
- To protect users as much as possible from mis-issued certificates.

This is achieved by creating cryptographically assured, publicly auditable, append-only logs of certificates. Every certificate will be accompanied by a signature from one or more logs asserting that the certificate has been included in those logs. Browsers, auditors and monitors will collaborate to ensure that the log is honest. Domain owners and other interested parties can monitor the logs for mis-issued certificates.

The goal of CT is to log all SSL certificates in many publicly available logs. Trust would only be provided to logged certificates. The logs would be auditable for reliance, and also monitored to detect when a certificate was issued for any specific domain name.

This will be a huge benefit for the Internet as the solution scales for all domains and all domain name owners, regardless of their size or the use of their site. Domain name owners will be allowed to monitor the logs, which will probably be a service offered to them by a third party such as a big search engine firm or their CA.

Currently, CT has been implemented for Extended Validation (EV) SSL Extended Validation (EV) SSL certificates. CT is supported by Google Chrome 41. For older EV certificates, the CA may have submitted these certificates to Google to be whitelisted for Chrome. EV certificates which are not logged or whitelisted will lose their EV indication in Chrome 41 and later versions.
Domain Protection

For an EV certificate which has been logged, you will see the following indication in Chrome.

![Image of EV certificate logged in Chrome](image)

When you select “transparency information”, you will see time-stamps from each log.

![Image of signed certificate timestamps](image)

Most certification authorities (CAs) will not support certificate transparency for non-EV certificates, so you will still see legitimate SSL certificates where Chrome will state “does not have public records.” If certificate transparency is successful, it will likely be extended to all SSL certificates.

With certificate transparency, the logs can be monitored which will indicate that all EV SSL certificates which have been issued for a given domain. This will allow unauthorized certificates to addressed and revoked.
Domain Protection

Requirements for CT are defined in experimental RFC 6962. The IETF is currently working on a new RFC standard.

Public Key Pinning

The current browser-to-certification authority (CA) trust model allows a website owner to obtain its SSL certificate from any one of a number of CAs. That flexibility also means that a certificate mis-issued by a CA other than the authorized CA chosen by the website owner would also be accepted as trustworthy by browsers.

Public key pinning or HTTP Public Key Pinning (HPKP) allows the website owner to make a statement that its SSL certificate must have one or more of the following:

- A specified public key
- Signed by a CA with this public key
- Hierarchical-trust to a CA with this public key

If a certificate for the website owner’s domain is issued by a CA that is not listed (i.e., not pinned), then a browser that supports HPKP will provide a trust dialogue warning. Please note that website owners can pin multiple keys from multiple CAs if desired, and all will be treated as valid by the browsers.

The website owner trusts that its chosen specified CAs will not mistakenly issue a certificate for the owner’s domain. These CAs often restrict who can request the issuance of a certificate for the owner’s specific domains, which provides additional security against mis-issuance of certificates to an unauthorized party.

Implementation of HPKP is tricky to get right, and as a result, it is mostly used by a handful of high-profile, security-sensitive sites. If you decide to use HPKP, you should start with a very short max-age value and gradually increase it if you don’t have any problems.

You can detect problems with HPKP reporting, which started to be provided with Chrome 46. For HPKP reporting you can start by sending the Public-Key-Pins-Report-Only header instead of the Public-Key-Pins header. When your site sends such a header, Chrome will verify if the current connection matches the pins, and sends a report to the report-uri if not. Chrome will not block requests based on the pins in a Report-Only header, so this is a safe way to try out HPKP and see if it causes problems for your users without running the risk of bricking your site.

When you roll out the real Public-Key-Pins header to start enforcing your pins, you can include a report-uri value in that header as well, so that you will continue to get reports if any problems occur.
Domain Protection

Certificate Reputation

One of the advantages of the SSL industry is that certificates can be issued from most trusted CAs. This allows certificate customers flexibility in choosing their CA or deciding to use a number of CAs. The disadvantage is the end-user does not know if the CA was authorized to issue the certificate and there could be a chance that the certificate is fraudulent.

Security experts have come out with proposals to allow domain owners to authorize CAs (Certification Authority Authorization), allow the Web server to state which public key is trusted (Public Key Pinning) or allow the owner of a website to monitor certificates that have been issued for their domain (Certificate Transparency).

Microsoft is proposing a solution to improve trustworthiness of certificates: Certificate Reputation. In Internet Explorer (IE) 11, Microsoft will extend the telemetry collected by SmartScreen Filter to include SSL certificate presented by websites. They will create tools to build intelligence about all certificates issued by every trusted root CA.

The goal is to flag potential man-in-the-middle (MITM) attacks using publicly trusted certificates. Examples of flags are:

- Website using a subordinate CA certificate
- Website presents a different certificate for only certain regions
- Significant change in the fields of the certificates that a CA issues, such as the OCSP responder location

Certificate Reputation has the following advantages:

- Privacy – When a certificate subscriber purchases a certificate for its internal domain name, this domain name will not be available publicly. Data will also be sent encrypted and no personally identifiable information is retained.
- Certificate Monitoring – Domain owners could be notified by email when new certificates are issued with their domain names.
- Scalable – The solution will scale without requiring cooperation from third parties such as website operators.
- Deployment – Certificate Reputation should be easy to deploy as it will only require efforts from Microsoft. The solution will not rely on changes being performed by third parties such as CAs, subscribers, Web server developers and OCSP developers.
Domain Protection

Security experts also say there are some disadvantages:

- **No Public Log** – Microsoft will own the database and it will not be made publicly available, nor available for audit.
- **Sensitivity** – Attacks that are highly targeted will be difficult to detect.
- **All Certificates Not Covered** – The solution will rely on the telemetry gathered by the use of IE 11 (and later). This means it is targeted at certificates that browsers use and not other applications. There is also the opt-out issue, where an organization might not provide data back to Microsoft; in this case, the solution will be deprecated for those sites.

There is also the issue of Certificate Transparency (CT) versus Certificate Reputation. Google is pro-CT and is looking to require the CAs to support CT for EV SSL certificates in 2015. With the proposal of Certificate Reputation, it would appear that Microsoft does not support the CT proposal. This could be a disadvantage to CAs that use Windows PKI that might not support CT.

A concern among website owners is that an attacker can have a certificate issued for their domain name.

In March 2015, Microsoft deployed Certificate Reputation. Through the use of Windows, Internet Explorer and other applications, certificate data for all types of SSL certificates is collected and provided to Microsoft. Microsoft does not collect any information that could be used to identify the user, and as such, privacy can be maintained.

The certificate data can only be viewed by users who can confirm ownership of the domain. The data is provided through [Bing Webmaster Tools](https://www.bing.com/webmaster) and shows data similar to the image below.

The data includes identity information such as the name of the server, the name of the entity, and the name of the CA. It provides data on how long the certificate has been available and its validity. It allows the user to download the certificate and report fraudulent certificates to Microsoft.

The advantage of Certificate Reputation is that it works for all types of SSL certificates and not just EV. It works for all CAs, as the CAs do not need to participate in the Certificate Reputation program. Certificate Reputation is also available to all administrators as Microsoft is providing the information through a portal. The disadvantage is that it only provides data from Windows and its applications.

**Fraudulent certificates are an increasing issue, as such it is recommended that domain owners use Certificate Reputation to monitor their domains.**
Advanced Certificates

For the highest possible security for businesses, there are a number of advanced certificates available that provide enterprise-level protection. These include Multi-SAN Certificates, Extended Validation (EV) Certificates, Elliptic Curve Cryptography (ECC) Certificates, and Private Trust Certificates.

Multi-SAN Certificates

This certificate is sometimes called Unified Communications Certificate (UCC), Multi-Domain Certificate or Multi-SAN certificate.

The unique feature of the UC certificate is that it takes full advantage of the subject alternative name (SAN) field. In doing so, the issuer allows the certificate subscriber to request many domain names be included in the SAN fields.

The result is the UCC provides many distinct advantages:

- **Flexibility** - One certificate can protect multiple domains and sub-domains owned by the certificate subscriber. This supports virtual hosting over SSL on a single IP address and allows the certificate to protect more than one name, such as https://www.example.com and https://example.com.

- **Compatibility** - Some applications, such as Microsoft Exchange, require more than one domain to be protected and will also only allow one certificate to be used.

- **Security** - UC certificates protect only the domains specified by the website owner. The certificate does not protect an unknown site name that an attacker has set up. This is a big advantage over a wildcard certificate.

- **Verification level** - UCC can be issued using domain validation (DV), organization validation (OV) or the most secure extended validation (EV). This allows to website owner to select the level of identification that they want to present to their site users.

- **Price-effectiveness** - As many domains are allowed in the certificate, the cost is generally less than requesting an individual certificate for each domain

For multi-SAN certificates, order the SANs with the most popular domains first to reduce browser connection time.
Advanced Certificates

**Pro Tip: Powerful SSL-based security for today’s communication platforms.**

Entrust UC Multi-Domain SSL Certificates secure multiple domains, sub-domains or hostnames with a single SSL certificate — saving you time and money when compared to buying individual certificates.

Learn more about Entrust UC Multi-Domain Certificates here.

**Extended Validation (EV) Certificates**

In the past, the browsers and the certification authorities (CAs) did not have standards for SSL certificate validation or management. The result was there were occasions where certificates were mis-issued. This enabled attacks such as a phishing site to get a legitimate certificate.

The CAs and browsers formed the CA/Browser Forum and created the standard for the extended validation (EV) SSL certificate. The validation process strengthens how the owner is identified and how the issuance of the certificate authorized. The result is the browser will provide the identity in the browser status bar and will provide a green EV indication.

Over the years, the websites with a high number of phishing attacks have moved to EV SSL. Their customers expect to see their name in the green bar. This trust can be extended to websites where the owner can qualify to meet the EV criteria.

**Pro Tip: The highest level of assurance for Web security possible.**

Entrust EV Multi-Domain SSL Certificates are supported by the most complete validation processes available, including certificate transparency for the Google Chrome browser. These certificates take advantage of the added visual cues in today’s popular browsers — a clear indicator to your customers that your website is secure.

Learn more about Entrust EV Multi-Domain SSL Certificates here.
Advanced Certificates

Elliptic Curve Cryptography (ECC) Certificates

Content provided by the Certificate Authority Security Council

Elliptic Curve Cryptography (ECC) is an approach to public-key cryptography which can replace RSA or DSA.

ECC is a fundamentally different mathematical approach to encryption than the RSA algorithm which uses an elliptic curve as an algebraic function (y² = x³ + ax + b). ECC is based on a one-way property in which it is easy to perform a calculation but infeasible to reverse or invert the results of the calculation to find the original numbers.

ECC can be used for both digital signatures using Elliptic Curve DSA (ECDSA), and in key exchange using Elliptic Curve Diffie-Hellman (ECDH). Digital certificates can be signed with ECDSA instead of RSA. ECC can also be used in the SSL/TLS handshake when the server and client are negotiating session keys using ECDH.

The benefit of ECC is the keys are stronger for a given bit length. For instance, an ECC key size of 256-bits is equivalent to a 3072-bit RSA key and about 10,000 times stronger than a 2048-bit RSA key. This means as the RSA keys get weaker, we can migrate to ECC keys where there will be no requirement to accommodate much larger RSA keys. ECC is much faster during the SSL/TLS handshake which reduces site latency and requires servers with less processing power. In addition, ECC supports Perfect Forward Secrecy.

A disadvantage of ECC is that it is not supported by all client software.

If you plan to migrate to ECC certificates, then you may want to implement a server which will support both an RSA key and an ECC key at the same time.

This approach will support both old and new clients.
Advanced Certificates

Private Trust Certificates

Private Trust SSL Certificates enable the use of non-registered domain names that are currently owned so long as the root certificate is also distributed.

Entrust Private SSL Certificates provide the following:

• Non-registered domains - Domains will be registered for one customer, so no other customer will get the same domain. This will mitigate the security issues.
• Registered domains - Certificates will also be allowed to include names that you have registered
• Multiple domain names - Certificates will support one or more domain names
• Unlimited server licensing and reissues - Certificates can be placed on more than one server and can be reissued as required
• Same features as our FQDN certificates - Certificates will require the same key size, signing algorithm, validity period and CA protection as all of the other certificates Entrust issues

Pro Tip: Get privately trusted SSL certificates that enable the continued use of non-registered domain names.

This type of certificate provides the same key sizes, signing algorithms, validity periods and CA protection as Entrust’s proven publicly trusted SSL certificates — but issued via a private shared CA that protects you from possible impersonation attacks by ensuring no two certificate names are alike.

Learn more about Entrust Private SSL Certificates.
Reliable Certification Authority and Always-On SSL: The Trusted Combination

When it comes to ensuring your website is secure, it’s always best to not to cut corners. That means trusting a Reliable CA and deploying Always-On SSL.

Reliable CA

A reliable CA provides the peace of mind and security of knowing that your website is being protected by professionals. Here are some of the advantages of a trusted CA:

• CA verifies the owner of the domain and the certificate applicant
• CA operates to a policy in conformance with the requirements of the browser and operating system vendors. The requirements include the CA/Browser Forum Baseline Requirements, Extended validation (EV) Guidelines and recommendations from NIST and the IETF.
• CA provides quality to the certificate. Checks include compromised keys, minimum key size, ensuring hashing algorithms, maximum validity period and proper certificate extensions.
• CA updates policy based on industry best practices
• CA provides certificate status through CRL and OCSP
• Compromised certificates can be revoked
• CA is audited to certificate issuing criteria such as WebTrust for CA, WebTrust for EV and SSL Baseline Requirements
• Certificate requesters for a Domain validated certificate are authorized by the owner of the domain. Requesters for Organization and Extended Validation certificates are authorized by a member of the organization specified in the certificate.
• CAs provide multiple reminders to ensure the certificates are renewed before they expire. CAs may also provide certificate discovery tools to find certificates on your systems which may not have reminders.
• Publicly trusted CA model is based on the CA being a trusted third party to the browser/OS vendor, the website certificate subscriber and the end-users of the website. The CA is obligated to meet the requirements of all three parties.
Reliable Certification Authority And Always-On SSL: The Trusted Combination

In short, while cheaper options exist, none are as safe, trusted and effective as a Reliable CA. In the end, saving a few dollars up front could cost thousands down the road to recovery from a breach or complete data loss.

Always-On SSL

Always-On SSL is an approach to securing your website to mitigate attacks against your users. Always-On SSL involves three concepts: SSL across your entire site, SSL deployed to the best practices, and SSL with leading technology.

SSL across Your Entire Site

The approach to Always-On SSL is to avoid hijacking of the sessions with your users. This means not just securing the sensitive parts or financial transactions, but your entire site. The goal is to ensure your users start their sessions with SSL and never turn it off. This mitigates the hijacking attacks of common tools.

SSL Deployed to Best Practices

SSL is only as strong as its implementation. Often attacks aren't a result of a clever hacker, but rather a simple implementation error that left a website wide open for exploits. Because SSL is deployed on thousands of different servers/software configurations, ensuring your site is following best practices is key.

Here are some common configuration tips:

1. Always ensure your DNS is configured correctly. This means having your www and naked domain addresses pointing to the same server. If your www.example.com address points to one server, while example.com points to another or doesn’t resolve at all, there can be problems.

2. Having different sites on ports 80 and 443. Consider configuring your secure and non-secure websites with the same content.

3. Using self-signed certificates conditions your users to accept trust dialog boxes. When a real trust issue appears, it is ignored. Replace self-signed certificates with certificates issued by a reputable public CA.

4. Ensure SSL servers are properly configured. Many deployments rely on the web server default settings. Unfortunately, these defaults may be wrong or insecure. Use the Entrust Server Test Tool to identify these problems, and then tune your server as appropriate.

5. Don’t use incomplete certificates. A user types and expects to see the same site as on, but get a certificate error. This can be fixed by using a multi-domain SSL certificate that supports both example.com and www.example.com in the Subject Alternative Name (SAN) field.

6. Don’t mix SSL and plain text on a site. Mixed content can lead to man-in-the-middle attacks.

7. Use SSL for more than just authentication. Sites that use SSL only for authentication are vulnerable to session hijacking. Use SSL for authentication and then for the rest of the session.
Reliable Certification Authority And Always-On SSL: The Trusted Combination

8. Always use Secure Cookies. Non-secure cookies are subject to man-in-the-middle attacks.

9. Always use secure content. Using mixed page content can compromise security. A single plain-text link is enough to compromise the entire secure SSL site. Don’t mix secure and unsecure content on a page.

SSL with Leading Technology

If you have SSL on your entire site and it is secure by mitigating the poor practices, then perhaps you can upgrade your security by using leading technology such as:

• EV SSL Certificates
• HTTP Strict Transport Security (HSTS)
• OCSP Stapling
• Perfect Forward Secrecy
• SHA-2 Hashing
• TLS 1.2

These items and many others are discussed throughout this guide.
Tools

**Entrust SSL Server Test** is a great first step how your site stacks up to recommend configurations and mitigation of known attacks.

Here are some helpful links to help with your SSL deployment:

**Certificate Discovery** is a comprehensive solution that helps find, inventory and manage digital certificates across diverse systems and environments.

**SiteLock** will help you perform regular scans to help you defend your website from malicious malware and common vulnerabilities.
Attacks

Website attacks are getting more common and more complex all the time. While a comprehensive list of all exploits would take days to read, here are some of the most common, most infamous exploits from recent years.

**BEAST**: Short for Browser Exploit against SSL/TLS, BEAST performs what’s known as a chosen plaintext-recovery attack against AES encryption in SSL 3.0 and TLS 1.0. The technique exploits an encryption mode known as **cipher block chaining (CBC)**, in which data from a previously encrypted block of data is used to encode the next block.

**CRIME**: The attack exploits an SSL/TLS feature that is used to implement HTTPS and affects all versions of SSL and TLS. The attack is performed by an agent that needs to be loaded on the victim’s browser. The attacker must also be able to sniff the victim’s HTTPS traffic.

**FREAK**: FREAK is a man-in-the-middle (MITM) vulnerability and stands for “Factoring RSA-EXPORT Keys.

**Heartbleed**: Heartbleed is an implementation exploit that allows an attacker to read the memory of a system over the Internet and compromise the private keys, names, passwords and content. Attacks that exploit Heartbleed are not logged, so they are not detectable. Attacks can be from client to server or server to client.

**Logjam**: a security vulnerability against a Diffie–Hellman key exchange ranging from 512-bit (US export-grade) to 1024-bit keys.

**Lucky Thirteen**: This exploit uses a known timing attack previously believed to be impractical. There is a subtle timing bug in the way that TLS data decryption works when using the (standard) CBC-mode ciphersuite. Given the right set of circumstances, an attacker can use this to decrypt sensitive information, such as passwords and cookies.

**POODLE**: The **POODLE attack** (Padding Oracle On Downgraded Legacy Encryption) allows items such as “secure” HTTP cookies or HTTP Authorization header contents to be stolen from downgraded communications.

**RC4 Attack**: The bytes coming out of the RC4 aren’t quite random-looking as they have small biases. By getting many different encryptions of the same message using different keys, an attacker can use the small deviations to figure out what was encrypted.
About Entrust Datacard

Consumers, citizens and employees increasingly expect anywhere-anytime experiences - whether they are making purchases, crossing borders, accessing e-gov services or logging onto corporate networks. Entrust Datacard offers the trusted identity and secure transaction technologies that make those experiences reliable and secure. Solutions range from the physical world of financial cards, passports and ID cards to the digital realm of authentication, certificates and secure communications. With more than 2,000 Entrust Datacard colleagues around the world, and a network of strong global partners, the company serves customers in 150 countries worldwide.

For more information, visit www.entrustdatacard.com

Contact & Social Links

www.entrust.com
Entrust@entrust.com
888.690.2424
About Bruce Morton

Bruce Morton has worked in the public key infrastructure and digital certificate industry for nearly 20 years and has focused on SSL and other publicly trusted certificates. He is an active member of the CA/Browser Forum and the CA Security Council. Bruce oversees the governance and compliance of Entrust’s publicly trusted PKI.