CONTENT PROVIDED BY

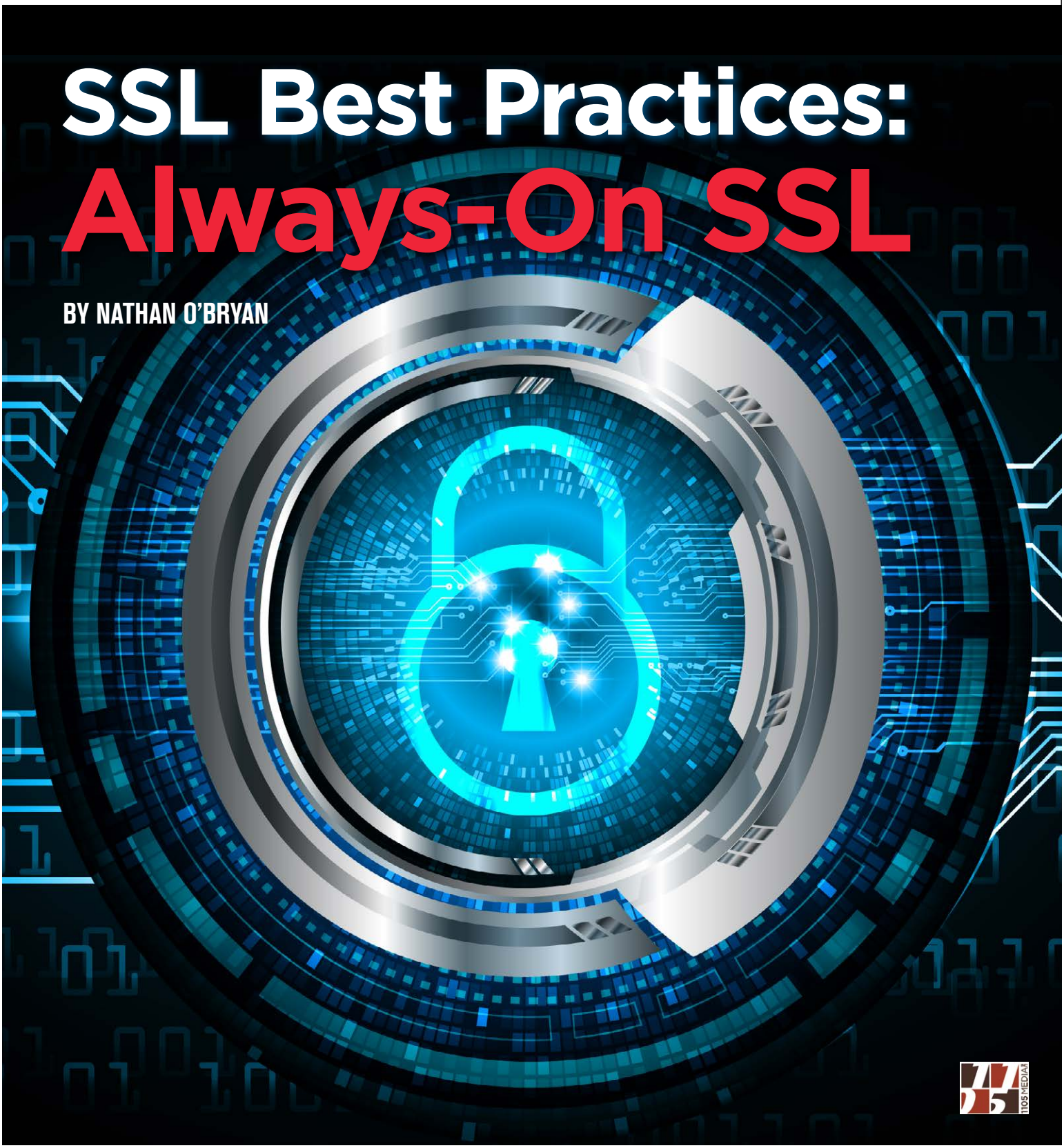
Redmond
THE INDEPENDENT VOICE OF THE MICROSOFT IT COMMUNITY

SPONSORED BY

Entrust®

SSL Best Practices: Always-On SSL

BY NATHAN O'BRYAN



1105 MEDIA

It's clear that information is the currency of the future. The internet has become the backbone of our economy, and keeping our data secure is one of the major challenges we need to face. Connecting any computer to the internet carries the risk of exposing any data on that computer to whomever can hack it. Our job, as IT professionals, is to do everything we can to make our systems as secure as possible. That means ensuring that all connections to our servers are secured with Secure Socket Layer (SSL) encryption.

Secure Socket Layer encryption

Secure Socket Layer encryption was developed by Netscape for web browser in the 90s. While the SSL protocol has been replaced by a newer protocol called TLS, we still refer to public key encryption on the internet as SSL.

While the SSL protocol has been replaced by a newer protocol called TLS, we still refer to public key encryption on the internet as SSL.

SSL was developed to allow websites to establish secured connections with web browsers so that secured data could be transmitted over the public internet securely. This, in turn, allowed for all manner of transactions to take place over the internet. Whether we're talking about online banking, buying digital music, or doing your Christmas shopping from the comfort of your own living room, SSL is what makes it possible. Without SSL, we'd all have to go back to the dark days of needing to physically leave our houses every time we wanted something. I'm not sure how society functioned before the internet.

Where do certificates come in?

All web browsers have the ability to communicate to web servers via encrypted SSL. For that to work, however, the web server needs to have a SSL certificate installed.

The SSL certificate is a special file that contains some specific information to allow this SSL encrypted connection to work. The certificate contains:

- A public key to encrypt the data
- A private key to decrypt the data
- A subject name that identifies the website owner
- Dates for which the certificate is valid
- The digital signature of the issuing Certificate Authority

With this information, any web browser is able to negotiate a secure connection with your web server.

OK, but how does the certificate create this secure connection?

When a web browser attempts a secure connection to a web server secured by a SSL certificate, the browser and the server start a process called an "SSL Handshake."

During this SSL Handshake process, the browser uses the public key from the certificate to encrypt information including a session key and rules for the secure communication session. The web server then uses the private key from the certificate to decrypt this blob of information and establish a secure communication session using the session key.

The recommendation is to use a key size for encryption of 2048-bit RSA.

Here is a more precise walk though of the SSL Handshake process:

- Browser connects to a web server (website) secured with SSL (https). Browser requests that the server identify itself.
- Server sends a copy of its SSL Certificate, including the server's public key.
- Browser checks the certificate root against a list of trusted CAs and that the certificate is unexpired, unrevoked, and that its common name is valid for the website that it is connecting to. If the browser trusts the certificate, it creates, encrypts, and sends back a symmetric session key using the server's public key.
- Server decrypts the symmetric session key using its private key and sends back an acknowledgement encrypted with the session key to start the encrypted session.
- Server and Browser now encrypt all transmitted data with the session key.

What key size should I be using?

The recommendation is to use a key size for encryption of 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 with no appreciable gain in security.

If you are planning to install your certificate on multiple servers, then you should consider using certificates with different keys. That way, if one key gets compromised then only one server will be compromised – not the others.

Are some signing algorithms more secure than others?

Absolutely.

All servers and applications must be migrated to SHA-2 or better to be considered secure. Microsoft and Google have implemented SHA-1 deprecation. Sites still using the SHA-1 standard will no longer be viewed as trustworthy in their browsers.

All servers and applications must be migrated to SHA-2 or better to be considered secure.

In 2017, SHA-1 signed certificates will no longer be supported in Windows. Migration to SHA-2 or better 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.

Is it OK to use a self-signed certificate for Always-on SSL?

No.

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.

The best practice is to use an SSL certificate issued by a reliable public certificate authority.

Is HTTP Strict Transport Security something I should consider as well?

Yes.

HTTP Strict Transport Security (HSTS) is the next step beyond Always-on SSL to secure your website.

Once your site is secured with Always-on SSL, 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.

The importance of encrypting everything

There are lots of known threats that can compromise a server connected to the internet. While known issues are a challenge, the hard part of security is not protecting against the threats you know about. The hard part is always securing your servers against the next threat. The first person to figure out a way to secure internet servers against unknown exploits is going to be a very rich person. Until then, we need to make the attack surface on our servers as small as possible. That means protecting every single bit of information about our systems we can.

There are lots of known threats that can compromise a server connected to the internet.

It seems that the one common thread in nearly every story of a hack is the small bit of information the hackers were able to obtain that got them started. Always-on SSL is about ensuring that every bit of data coming into and going out of your servers is as secure as it can be.

But how does encrypting everything help?

Let's compare this strategy to something that we've all accepted as security doctrine, the "deny all" last rule on a firewall.

The last rule on every firewall in existence is "deny all." This means that if the connection attempt is not explicitly allowed by one of the rules above, then that connection is not allowed.

Always-on SSL is much the same idea. If you want to talk to this web server, then that conversation is going to need to be secured. This make all connections to the web server more secure.

If not for security

If you've read this far and you are still unconvinced, it's likely that you are not going to be convinced by any security based arguments put forward here. That, in turn, means you are either in marketing or management. This is not a problem, because I know how to speak your language.

Google has recently announced it will factor HTTPS in as a ranking signal. So if security is not enough of a reason for you to implement Always-on SSL, then maybe SEO is.

Wrapping things up

Always-on SSL is the best practice for all web servers. There is no reason that any unsecured connection should be made to any web server, no matter how mundane the data in question. It is neither difficult nor costly to ensure that all connections to your web servers are secured, and it just might help prevent the next major data breach. R

Nathan O'Bryan is a Microsoft Certified Solutions Master in Messaging and an Office 365 MVP. He is available on Twitter at @MCSMLab or online at www.mcsmlab.com.