TECHNICAL WHITE PAPER

TLS encryption:
Is it really the answer to securing email?

egress®

The digital age has fundamentally changed the way all organisations operate, driving greater efficiency, scalability and the adoption of outsourced services, including those hosted in the Cloud.

At the same time, and as data breaches in the last few years have demonstrated, the more data we produce and process, the more difficult it is to secure and control. Consequently, it is more likely there will be some form of breach, which can result in potentially large fines and, more importantly for many businesses, reputational damage.

Unsurprisingly, steps have been taken to try to protect and secure data throughout its lifecycle. Of particular concern is the security of data sent via email between organisations. How is it possible to ensure the confidentiality and integrity of this information when sending to both established partners and unknown third parties?

> "...the more data we produce and process, the more difficult it is to secure and control."

Inevitably, businesses want solutions to this question that do not further complicate their IT infrastructure, operational workflows and end-user processes. Transport Layer Security (TLS) email encryption has for a long time been judged the ideal solution. After all, since it is already available in most mainstream mail servers and hosting providers, it seems like the obvious option. Additionally, an extension called STARTTLS provides a method of upgrading an existing insecure plaintext connection to an encrypted connection that uses TLS. STARTTLS also provides the further benefit that configuring a separate port for encrypted communication is not required, making TLS email encryption more straightforward to implement.

TLS in operation

TLS is a cryptographic protocol whose main goal is to provide privacy and security between computer systems communicating over a network. It utilises public and private key encryption to encrypt the transmitted data and thus set up a secure transport link between email servers on the Simple Mail Transfer Protocol (SMTP). It also optionally supports server authentication using digital certificates. The encryption keys used in a TLS connection are unique to each connection and are generated during an initiation process known as the handshake protocol.

Handshake protocol

When an email is sent between servers and TLS is enabled, the handshake protocol initiates. The first step is for the sending server (using STARTTLS) to send a request to the recipient server. This request and the following response by the recipient server establishes common security capabilities, choosing the protocol version, session ID, cipher suite, compression method and initial random numbers to use in the TLS session. Encryption keys are then exchanged and optionally the recipient server may send its digital certificate if requested. The sending server may also respond with its own certificate if requested. When the handshake protocol completes, an encrypted tunnel is established and the email is delivered securely.

While there is obviously some level of data security and assurance in this process due to the use of public key encryption and certificate authentication, the vulnerabilities of the TLS approach to email encryption become quickly exposed when considering the current level and sophistication of cyber threats. This white paper discusses some of the many concerns surrounding the practical and real-world uses of TLS encryption, and why it may not be the answer to all of an organisation's secure email needs.

Four problems with TLS encryption

Problem one: Reliance on DNS

The first issue with TLS is that because it functions alongside the SMTP, it also uses the Domain Name System (DNS) to look up the required recipient email server when attempting to deliver an email. DNS has well-known and widespread vulnerabilities that can be exploited, including with a classic DNS spoof:

An attacker gains access to the traffic via the same network, an Internet Service Provider (ISP), or even compromised or vulnerable network route firmware. With little effort, they can sit between the sending server and the internet, intercepting the DNS request and substituting the IP of the real recipient server with the IP address of a server under the attacker's control. A TLS tunnel is set up and the message delivered to the hacker's server with no issues reported to the sender.

This wouldn't be such a problem if certificates were required to be correctly validated, but in practice this is not the case. The technology used by TLS is the same as that used for Secure Sockets Layer (SSL) when visiting any secure web page. When a user views a secure website, a certificate validation takes place. For example, if a user attempts to connect to 'https://egress.com' but the server presents a certificate for 'https://evil.com', the user will be prompted with a message to say there is a serious security problem and, in many instances, prevented from proceeding. Yet with TLS on the SMTP, certificates are typically either poorly validated or not validated at all. Name mismatches are often allowed and email messages are permitted to be delivered. Reasons for this vulnerability will become clear in the discussion of 'Problem two: Real-world lockdown'.

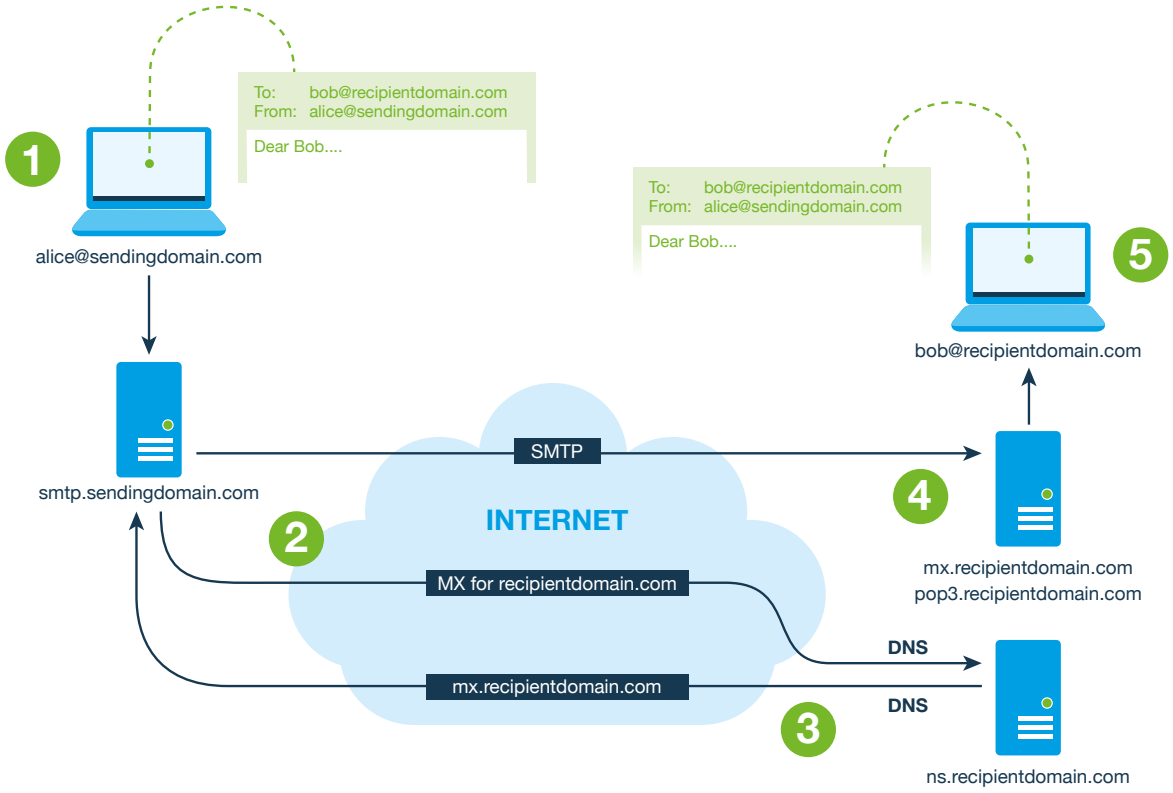

Fig. 1: The sequence of events that takes place during email sending on the SMTP

Problem two: Real-world lockdown

While there are significant technical issues with the implementation of TLS, there are operational concerns to contend with as well. With the large-scale adoption of hosted email content filtering and anti-spam protection, the ability to conduct accurate DNS name validation and certificate validation becomes a significant hurdle. This necessitates a sending server ignoring mismatched domain names and certificates in order to send an email to a recipient with outsourced email services.

"This necessitates a sending server ignoring mismatched domain names and certificates in order to send an email to a recipient with outsourced email services."

If, for example, ABC Inc. (abc.com) utilises the services of a third-party anti-malware or content inspection service (e.g. inspectionmailservices.com), the third party will host their MX email record (a DNS resource that specifies a mail server responsible for accepting email messages on behalf of abc.com). When a server attempts to send an email to bob@abc.com it will request the email server location via DNS. DNS will return a list of servers (e.g. mail1.inspectionmailservices.com) that have no reference to the abc.com domain. To ensure email is delivered, the sending server will by default ignore the domain name and associated certificate mismatch. This, as discussed, creates the conditions for potential man-in-the-middle attacks.

So in a practical example to demonstrate the various vulnerable stages of TLS email encryption, the following could happen:

1. alice@sendingdomain.com decides to send an email to bob@recipientdomain.com, so the sendingdomain.com server attempts to send securely using STARTTLS

2. DNS sends an unencrypted request to find the MX record for sending to bob@recipientdomain.com. A response is generated that is totally unrelated to the original domain, for example mail1.inspectionmailservices.com. This could have as easily replied with evil.server.com and the sending server would have no opinion on this, hence leaving the process vulnerable to a man-in-the-middle attack.

3. A second DNS query is then made to ascertain the IP address of mail1.inspectionmailservices.com. This represents another vulnerability that could be exposed as part of an attack.

4. If the attacker leaves the DNS request alone but suppresses the '250-STARTTLS' from the server's reply, the message sends in plaintext. However, even with the TLS connection established, there is a certificate validation problem. Alice wants to send to bob@recipient.com, yet it connects to mail1.inspectionemailservices.com, which presented a certificate for mail1.inspectionemailservices.com.

Problem three: TLS fails open

STARTTLS is intended to 'fail open' in the event of failure or certain errors occurring, rather than 'failing safe' (whereby if errors occur, the email fails to send at all). In the event of failure, STARTTLS falls back to normal plaintext SMTP and messages are sent in clear text. It is for this reason that the standard form of TLS is known as 'Opportunistic TLS': only encrypting if the conditions are right and sending in clear text if not. Attackers can exploit this design by sending certain types of packets to trigger a fail open error, thus transmitting messages in clear text. Email delivery has been perceived as more important than the security and confidentiality of the messages being delivered; users do not accept non-delivery.

Additionally, once they have access to the email infrastructure, attackers can suppress '250-STARTTLS' from the server's reply, convincing the sending server to not encrypt the submission at all, or even disable support for TLS.

> "In the event of failure, STARTTLS falls back to normal plaintext SMTP and messages are sent in clear text."

Problem four: Lack of auditing

Another significant limitation of TLS is that there is no easily accessible audit information or proof of transmission. If an email is delivered to a non-verified server, a server with incorrect certificate or is sent in plaintext there will be no notification to the sender or recipient and no easily accessible audit trail for the system administrators to access. Being able to quickly identify and remedy IT security failures is crucial in the defence against modern network attacks and yet neither TLS nor STARTTLS provides the auditing capabilities to do so. End-users would never know if TLS failed to protect their email communications and no special notifications are produced if TLS is suppressed or fails, leaving IT administrators similarly in the dark. While text logs of network communications are produced, there is no intelligent way of organising or identifying key points. In a practical sense, there is a severe lack of auditing information, and therefore a severe lack of knowledge and control of communications security.

What about Forced TLS?

There are some steps organisations can take to mitigate the issues associated with Opportunistic TLS. Configuring a Forced TLS policy setting adds an additional security measure to the TLS process. It requires that the recipient email domain be authenticated as a trusted source before email is sent to it via TLS encryption, otherwise the email isn't sent at all. By forcing TLS encryption in this way the 'fail open' issue is avoided, as emails simply won't send if they cannot be sent via TLS. It also guarantees that all emails to and from specific domains are transferred securely. Forced TLS furthermore provides the option to require that certificates be issued by a trusted certificate authority, making man-in-the-middle attacks less likely to succeed.

Again, however, the issue is one of practicality. It is easy to provide simple-sounding solutions such as the implementation of Forced TLS policies, but the crucial factor is whether these solutions are practical to implement and coherent with the way in which email communication occurs.

Forced TLS is cumbersome to implement, requiring complex configuration that includes running PowerShell scripts, and the creation of various rules and policies in conjunction with the recipient email domain. Hence, Forced TLS is only plausible between institutions; email between organisations and customers, end-users or smaller organisations has to rely on Opportunistic TLS with all its previously mentioned shortcomings. Data is at its most vulnerable when it leaves an organisation and Forced TLS is of no use in these cases.

Forced TLS also does not solve the aforementioned problems with Opportunistic TLS. It is still reliant on vulnerable DNS and certificate validation remains a problem. Certificates can be issued by a trusted certificate authority whilst necessarily failing to match the domains involved, owing to the use of third-party mail services (as discussed in 'Problem two: Real-world lockdown'). Validation errors such as these don't prevent organisations from being formally compliant, even though they potentially lead to security issues.

Summary

When TLS via SMTP was first invented, it was not originally designed to provide the levels of message confidentiality or integrity now demanded by organisations. Its reliance on subsequent extensions, including STARTTLS and domain-based message authentication, leaves it more vulnerable to external attack. TLS does not work in the same way as SSL, since with SSL even the oldest web browsers would highlight certificate and domain mismatches, and prevent any potential vulnerability. Yet it is perfectly normal in 2016 for SMTP, managed by a major SMTP specialist, to ignore all the aforementioned issues and deliver email. Further, even if a particular SMTP server (e.g. that of alice@sendingdomain.com) was very strictly configured, it would be unable to deliver mail securely to a huge number of recipients using STARTTLS without ignoring issues at the recipient side.

"Since the fail-safe non-delivery of email is unacceptable to most users and businesses, this leaves only one option to provide true security of email: implementing message-level encryption as well."

All of these findings suggest that even as businesses and email providers continue to deploy STARTTLS, there is no guarantee that the email will be encrypted as it travels from one server to another (from sender to receiver). Since the fail-safe non-delivery of email is unacceptable to most users and businesses, this leaves only one option to provide true security of email: implementing message-level encryption as well. Opportunistic and forced TLS have their uses, but as a supplement to message-level security, not a replacement.

Egress Software Technologies Ltd

Egress Software Technologies is the leading provider of hosted and on-premise encryption services designed to secure all forms of electronic information and delivered to customers in both the Public and Private Sectors via a single platform: Egress Switch.

The award-winning Switch portfolio of products includes Secure Email, Secure File Transfer, Secure Web Form and the latest online collaboration offering, Secure Workspace.

www.egress.com

info@egress.com
0844 800 0172
@EgressSwitch

egress