An introduction to TLS

Jochen Kressin · 2018-08-20

Search Guard relies heavily on TLS (Transport Layer Security) to secure Elasticsearch and to meet compliance regulations like HIPAA, GDPR, PCI-DSS, SOX or ISO. This article explains the basics of TLS and why it is the best technology for securing and encrypting data at the moment.

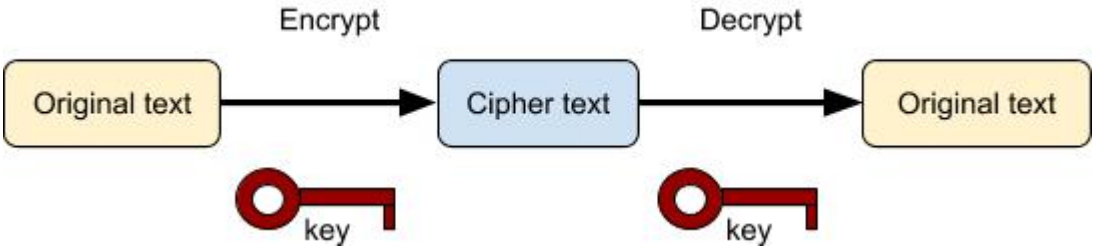
Secured inter-node communication is a must-have when protecting Elasticsearch clusters. We have explained that in an article "Security in Distributed Systems". Several things can go wrong when communicating plain text messages over a network:

- **Eavesdropping** - An attacker may read a message once it is sent over network.
- **Message Tampering** - An attacker may modify a message and a receiver may decode it without noticing that it has been changed. To protect against it, integrity of messages needs to be checked.
- **Message forgery** - It is the attacker who sends messages to a server but the server thinks it is a client.
- **Man-in-the-middle** - An attacker acts as a server to a client and as a client to a server. He gets messages from a client, communicates with server and sends some messages back to a client.

TLS (Transport Layer Security) is currently the best available technology that protects against those risks and of course is applied within Search Guard.

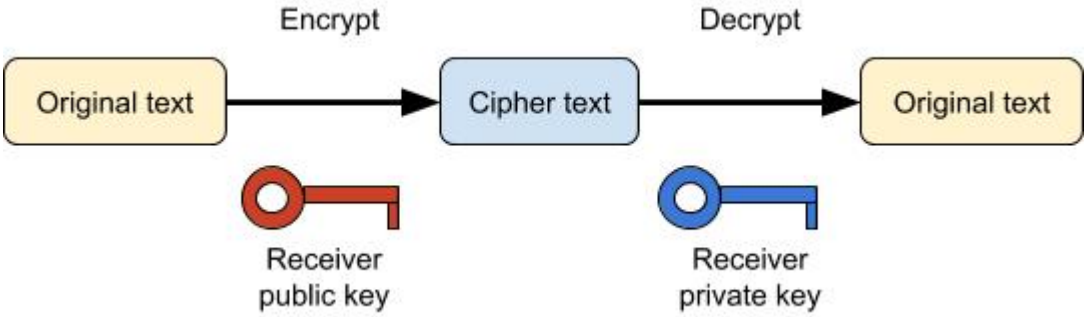
Symmetric and asymmetric cryptography

We start with symmetric and asymmetric cryptography. Symmetric cryptography assumes that both communication parties share the same key that is used to encrypt and decrypt messages.



Symmetric encryption algorithms perform best and are widely used in the industry. Unfortunately they require both sides to share the same key and provide no mechanisms to do so.

Here PKI (public key infrastructure) and asymmetric encryption comes in place. It assumes both sides of the communication have two keys: public and private. Public key is known to everyone while a private key of an actor is only known to him. Sender uses receiver's public key to encrypt the message and the receiver uses its private key to decrypt cipher text into original.



PKI can be used for both parties to agree on a symmetric key. Unfortunately it is still not enough as none of the sides is able to guarantee identity of the other. This can be only done by a third party that confirms identities in a form of certificates and is trusted by all actors.

Moreover we need to ensure that the receiver gets exactly the same message that came out of the sender. This can be done by checksums sent with each message that ensure message integrity.

What is TLS?

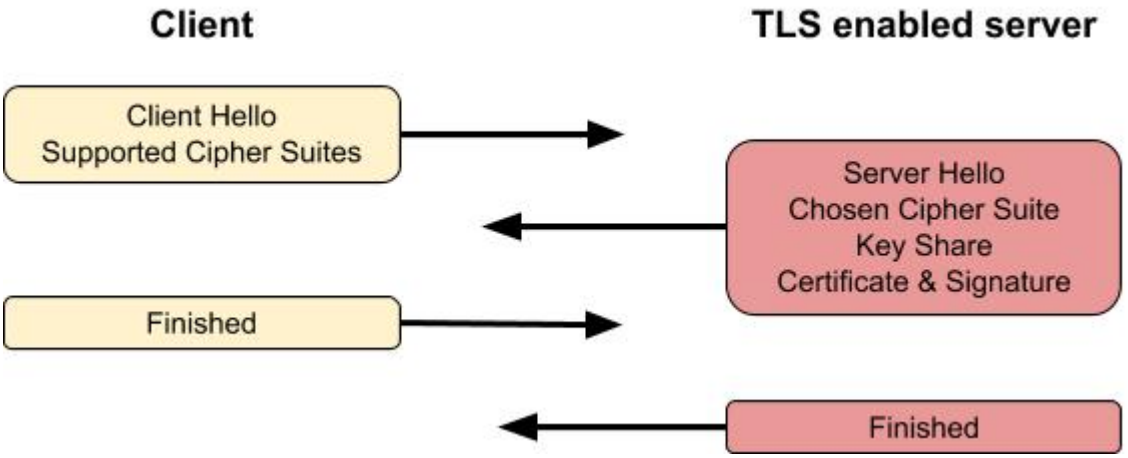
Public key infrastructure, symmetric encryption, certificates and checksums... It is quite a lot. Fortunately there is a standard that applies it. TLS (Transport Layer Security) is a framework that uses symmetric and asymmetric encryption algorithms and protects communication against risks we have mentioned earlier. It allows multiple encryption algorithms and allows client and receiver to agree on a one that is supported by both.

TLS Handshake

Let us have a look at TLS handshake, the key feature of TLS that allows both sides to negotiate security mechanisms used for securing communication channel:

- Client starts with *"Client Hello"* message that contains the highest TLS version it allows and a list of cipher suites and compression methods it supports.

- Server responds with *"Server hello".* It sends chosen cipher suite, its key that will be used for an encryption and a certificate that depends on a chosen cipher suite.
- Client checks certificate of a server in a third party CA (Certificate Authority) and sends an encrypted "*Finished*" message which contains checksum of all previous unencrypted messages.
- Server sends encrypted "*Finished*" message.



TLS Implementation

In order to make TLS work one should configure client and a server so that TLS Handshake succeeds with "Finished" message. Both sides should have:

- certificate signed by a CA or an intermediate certificate,
- public and private key.

Certificates and keys can be stored or transferred in different formats:

- **X.509** - A standard that defines format of public key certificates.
- **PEM** (Privacy-enhanced Electronic Mail) - You may have noticed it at the end of emails. It can store certificates, public and private keys. Values are stored in a text format generated using base64. PEM format is widely used for exchanging keys.
- **PKCS8** - A standard syntax for storing private key.
- **PKCS12** - An archive for storing multiple cryptography objects in a single file. The common file extension is .p12 or .pfx
- **Java KeyStore (JKS)** - Java repository for security certificates.

Summary

No matter how good you try to protect Your Elastic Search, it is worth nothing once communication between cluster nodes is not secured. At SearchGuard we understand it deeply and have put a lot of effort to make TLS work and secure our clients' data. The decision has been made based on our extensive experience and compliance regulations (GDPR, HIPAA, PCI, SOX) that our clients need to meet. Meeting them requires transmission security and integrity controls. TLS is the best on the market technology to do so.

Where to go next:

- Generate your own TLS certificates by using our online TLS service, the offline TLS Tool or our sample PKI scripts
- Read the next article in this series, an introduction to TLS certificates
- Read about the Search Guard TLS configuration options in our documentation
- If you have problems configuring TLS, read our TLS troubleshooting guide or ask for help on our forum.

Image: shutterstock / Profit Image

This article was published on the Search Guard blog: https://search-guard.com/blog/elasticsearch-searchguard-tls-introduction/