FORTINET®

Digital Certificates

in FortiGate

(SSL Inspection)

All You Need To Know

2 Complete Labs

in ManiPahlavanzadeh

Digital Certificates

in FortiGate

(SSL Inspection)

Mani Pahlavanzadeh
mani.pahlavan@gmail.com
ManiPahlavanzadeh

After completing this document, you will be able to achieve these objectives about FortiGate Methods of Firewall Authentication:

Part-1

- Digital Certification DefinitionDC Agent Mode
- Benefits of digital certification
- What Are the Types of Digital Certificates?
    - Transport layer security (TLS)/SSL certificate
    - Code signing certificate
    - Client Certificate
- Who Can Issue a Digital Certificate?
- Beneficial Features of Digital Certificates
- Differences Between Digital Certificates vs Digital Signatures

Part-2

- What is a digital certificate?
- X.509
- Certificate Signing Request
- Certificate Revocation List (CRL)
- Contents of a Digital Certificate
    - General Tab in detail
    - Details Tab in detail
    - Certification Path in detail

Part-3 ➔ Digital Certificate in FortiGate

- Why does FortiGate use Digital Certificates?
- Using Certificates to Identify a Person or Device
- How does FortiGate Trust Certificates?
- Encrypted Traffic with No SSL Inspection
- SSL Inspection Modes
- SSL certificate inspection
- Full SSL inspection (Deep SSL Inspection)
- Inbound or Outbound SSL/SSH Inspection
- SSL Inspection Profile Configuration in detail
- Exempting Sites from SSL Inspection
- FortiGate Self-Signed CA Certificate

- Full SSL Inspection – Certificate Requirements
- Applying an SSL Inspection profile to a Firewall policy
- Certificate Warnings during Full SSL Inspection
- Certificate Warnings on the FortiGate GUI
- FortiGate HTTPS Server Certificates
    - Download Private CA Certificate from FortiGate
    - Import Private CA Certificate into Endpoints
- Import a CA Certificate on FortiGate
- Import a Private Certificate on FortiGate
    - Generating a CSR in the FortiGate
    - Importing CA Certificate
- Import CRLs on FortiGate
- FortiGate Certificate Store
- Application and SSL Inspection
- Invalid SSL Certificate
- Untrusted SSL Certificates Setting
- Full SSL Inspection and HSTS
    - Visit Sites with HSTS Requirement

- **LAB-1: Configuring Full SSL Inspection on Outbound Traffic**
- **LAB-2: Dealing with Anomalies**

Fortinet Certificate Operations

In this document, you will learn why FortiGate uses digital certificates, and how to configure FortiGate to use certificates for SSL and SSH traffic inspection.

First of all, I want to explain Digital Certificate in detail since I have seen many folks getting confused about what is digital certificate, how to properly generate, and how to use digital certificates.

Digital Certificate – Part 1

Digital Certification Definition

A digital certificate is a file or electronic password that proves the authenticity of a device, server, or user through the use of **cryptography** and the **public key infrastructure (PKI)**.

Digital certificate authentication helps organizations ensure that only trusted devices and users can connect to their networks. Another common use of digital certificates is to confirm the authenticity of a website to a web browser, which is also known as a secure sockets layer or **SSL certificate**.

A digital certificate contains identifiable information, such as a user's name, company, or department and a device's Internet Protocol (IP) address or serial number. Digital certificates contain a copy of a public key from the certificate holder, which needs to be matched to a corresponding private key to verify it is real. A public key certificate is issued by certificate authorities (CAs), which sign certificates to verify the identity of the requesting device or user. DigiCert, GeoTrust, GoDaddy, Verisign, …

Benefits of digital certification

Digital certificates can be requested by individuals, organizations, and websites. To do so, they provide the information to be validated and a public key through a certificate signing request. The information is validated by a **publicly trusted CA**, which signs it with a key that provides a chain of trust to the certificate.
This enables the certificate to be used to prove the authenticity of a document, for client authentication, or to provide proof of a website's credential.

What Are the Types of Digital Certificates?

There are three different types of **public key certificates**:

- a transport layer security (TLS)/SSL certificate,
- a code signing certificate,
- and a client certificate.

TLS/SSL certificate

A TLS/SSL certificate sits on a server— such as an application, mail, or web server—to ensure communication with its clients is private and encrypted. The certificate provides authentication for the server to send and receive encrypted messages to clients. The existence of a TLS/SSL certificate is signified by the Hypertext Transfer Protocol Secure (HTTPS) designation at the start of a Uniform Resource Locator (URL) or web address. It comes in three forms:

Domain validated

A domain validated certificate is a quick validation method that is acceptable for any website. It is cheap to obtain and can be issued in a matter of minutes.

Organization validated

This provides light business authentication and is ideal for organizations selling products online through e-commerce.

Extended validation

This offers full business authentication, which is required by larger organizations or any business dealing with highly sensitive information. It is typically used by businesses in the financial industry and offers the highest level of authentication, security, and trust.

Code signing certificate

A code signing certificate is used to confirm the authenticity of software or files downloaded through the internet. The developer or publisher signs the software to confirm that it is genuine to users that download it. This is useful for software providers that make their programs available on third-party sites to prove that files have not been tampered with.

Client certificate

A client certificate is a digital ID that identifies an individual user to another user or machine, or one machine to another. A common example of this is email, where a sender signs a communication digitally and its signature is verified by the recipient. Client certificates can also be used to help users access protected databases.

Who Can Issue a Digital Certificate?

Digital certificates are issued by **CAs**, which sign a certificate to prove the authenticity of the individual or organization that issued the request. A CA is responsible for managing domain control verification and verifying that the public key attached to the certificate belongs to the user or organization that requested it. They play an important part in the PKI process and keeping internet traffic secure.

Beneficial Features of Digital Certificates

Digital certificates are becoming increasingly important, as cyberattacks continue to increase in both volume and sophistication. Key benefits of digital certificates include:

Security

Digital certificates encrypt internal and external communications to prevent attackers from intercepting and stealing sensitive data. For example, a TLS/SSL certificate encrypts data between a web server and a web browser, ensuring an attacker cannot intercept website visitors' data.

Scalability

Digital certificates provide businesses of all shapes and sizes with the same encryption quality. They are highly scalable, which means they can easily be issued, revoked, and renewed in seconds, used to secure user devices, and managed through a centralized platform.

Authenticity

Digital certificates are crucial to ensuring the authenticity of online communication in the age of widespread cyberattacks. They make sure that users' messages will always reach their intended recipient—and only reach their intended recipient. TLS/SSL certificates encrypt websites, Secure/Multipurpose Internet Mail Extensions (S/MIME) encrypt email communication, and document-signing certificates can be used for digital document sharing.

Reliability

Only publicly trusted CAs can issue recognized digital certificates. Obtaining one requires rigorous vetting, which ensures hackers or fake organizations cannot trick victims that use a digital certificate.

Public Trust

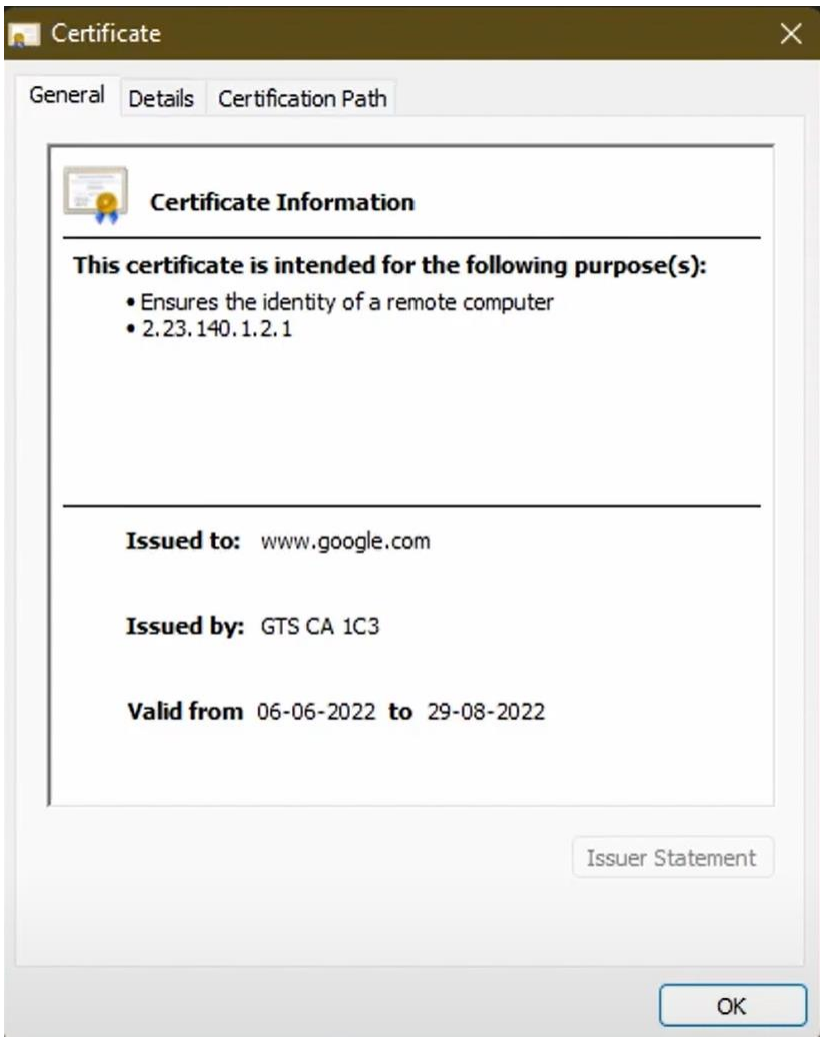
Using a digital certificate provides confirmation that a website is genuine and that documents and emails are authentic. This projects public trust, assuring clients that they are dealing with a genuine company that values their security and privacy.

Differences Between Digital Certificates vs Digital Signatures

- **A digital certificate** is a file that verifies the identity of a device or user and enables encrypted connections.
- **A digital signature** is a hashing approach that uses a numeric string to provide authenticity and validate identity. A digital signature is typically fixed to a document or email using a cryptographic key. The signature is hashed, and when the recipient receives it, it performs that same hash function to confirm that the information from the signer and has not been altered.

Digital Certificate – Part 2

What is a digital certificate?



- A digital certificate is simply a digital document used to trust ***someone*** or ***something*** on a network or internet.
- It is also known as a **public key certificate**.
- Digital Certificates follow **x.509** specification as described in RFC 5280.
- A certificate contains information which is always digitally signed; therefore, the name is **digital certificate**.
- They may include information about the entitiy a certificate is issued to such as **public key** associated with a **private key** of a certificate owner, information such as **organization name**, **location**, **email address**, and the **department**.
- A certificate will have a **date and time of issuance** including an **expiration date**.
- A certificate will also have information about their intended purpose.
- These certificates are distributed on the **internet** or **within a network** freely

- Certificates helps improving the identity of its owner or to prove their authenticity. for this reason, certificates are mostly used by a **web server for HTTPS**.
- you must have heard of **SSL/TLS**. If you are using a computer or any electronic device then the chances are there is a certificate in use somewhere.
- Most of your web browsing is secured by **HTTPS**. You may be using a smart card or a usb token which has a certificate stored in it. So, you could authenticate on a corporate computer.

X.509

X.509 is a standard by ITU (International Telecommunication Union) which defines the format of a Digital Certificate (Public Key Certificate).

X.509 has three versions: V1, V2, V3

X.509 was first introduced in 1988 as X.509 version 1. This version was later revised as version 2, and then version 3. Version 3 certificates are most common these days.

RFC 5280 specification of X.509 certificates also describe CRL (Certificate Revocation List) and Certification Path Validation. ➔ I'll explain what they are later in this document.

Versions of X.509 Certificates

There are three versions:

- **Version 1**
  Version 1 was introduced in 1988. They only had some basic information such as serial number, subject names, signature algorithm, issuer name, validity, and the public key. Version 1 certificates were found to be lacking some features, because of which it was revised as version 2.

- **Version 2**
  Version 2 certificates included a unique identifier for the issuer and the subject along with the properties of version 1. It was again revised as version 3 because of some features it lacked.

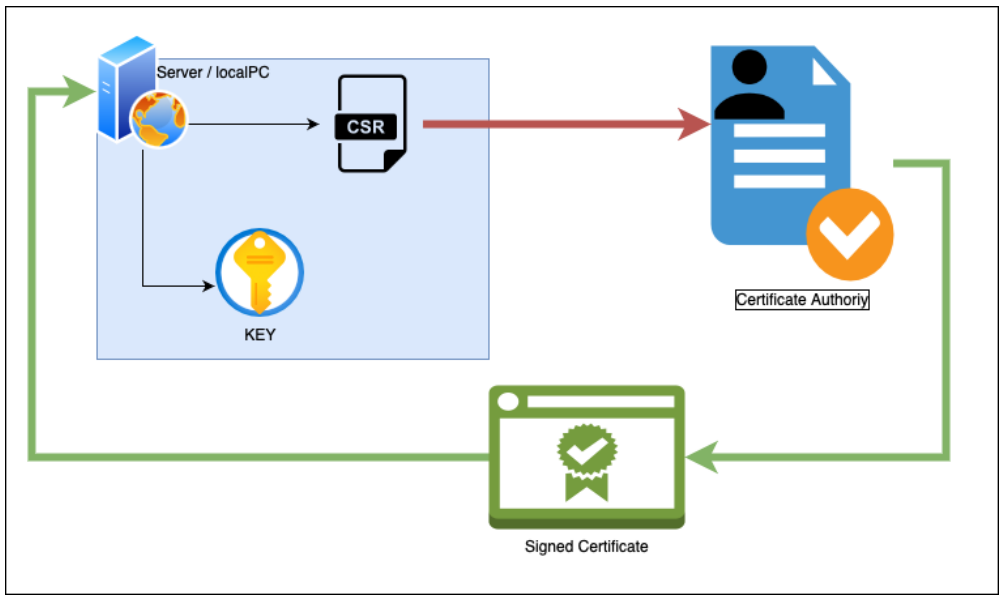
- **Version 3**
  version 3 certificates started using extensions along with the properties from version 1 and version 2.

Certificate Signing Request

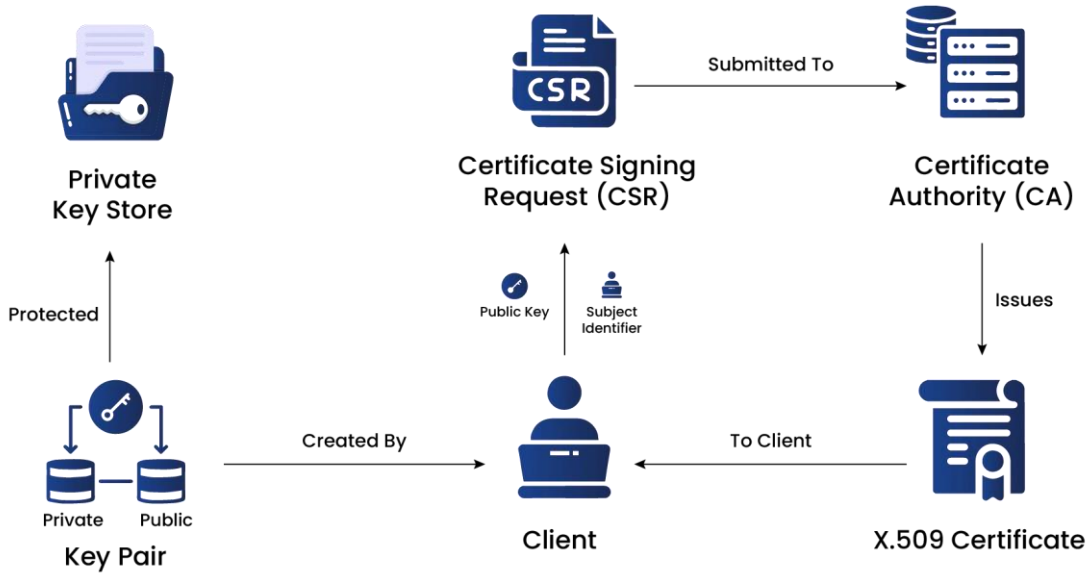
If I need SSL certificates for my website, I would have to apply for that. My initial step would be to generate a key pair using an Asymmetric algorithm such as **RSA** or **ECDSA** (The RSA algorithm uses significantly larger cryptographic keys than ECDSA). Once my keypair is generated, I need to apply for a certificate to get a Signed Certificate from a **CA**. I would have to request for it, and to do this, I would have to generate a **CSR**.

A **CSR** is also known as **Certificate Signing Request**. Imagine **CSR** to be just like an application form to get a signed certificate. It is a document just like a digital certificate. It would contain my public key along with some distinguished information about me. This information is signed using my private key and that signature is added to the CSR. Once my CA receives my CSR it would review it, so in the review process it would verify my signature using the public key that I included in the CSR.

If my CA can verify my signature, it would proceed with other verification process which it needs to perform as per their policy, and then they would finally issue a **Signed Certificate**.



The Comprehensive Lifecycle of Certificate Enrollment



- **Certificate Signing Request (CSR)**

  To initiate the certificate enrollment process, the entity generates a Certificate Signing Request (CSR). The CSR includes the public key and information about the entity that needs to be included in the certificate, such as the domain name for SSL/TLS certificates or the email address for S/MIME certificates.

- **Submitting the CSR to the CA**

  The CSR is submitted to the CA during the enrollment process. The CA verifies the identity of the entity and the information in the CSR. The CA may use various methods to verify the entity's identity, such as email verification, domain validation, or manual verification of legal documents.

- **Certificate Issuance**

  Once the CA has completed the verification process and is satisfied that the entity is legitimate, it issues a digital certificate. The certificate contains the entity's public key, identity information, validity period, and the CA's digital signature.

- **Certificate Delivery**

  The issued certificate is delivered back to the entity. Depending on the CA and the certificate type, the delivery may be done through email, a secure portal, or other methods.

- **Certificate Installation**

  The entity needs to install the issued certificate on the appropriate server or device where it will be used. For example, in SSL/TLS, the certificate is installed on the webserver to secure the website's connections.

- **Certificate Use**

  Once installed, the certificate is ready for secure communication protocols. Clients, users, or other entities interacting with the certificate holder can verify the certificate's authenticity through the CA's digital signature, ensuring a secure and trustworthy connection.

- **Certificate Renewal**

  Certificates have a limited validity period (typically 1-2 years). Before expiration, the entity must renew the certificate through a similar enrollment process to continue using it without disruption.

Certificate Revocation List (CRL)

CRL is Certificate Revocation List. A CRL is simply a list of all revoke certificate. A bunch of certificates that has been revoked by a CA.

every certificate has a validity period, however if CA needs to revoke a certificate it will add the serial number of that certificate into a list called CRL.

The reason why a CA might decide to revoke a certificate could be because the **private key** has been compromised or it could be because a certificate was found to be invalid or not in use anymore. For example, when an employee leaves an organization the CA for that organization would revoke all certificate used by that employee after they leave. Other scenario could be like an employee lost their smart card so CA of that company will revoke that employee's certificate as a precautionary measure.

A CRL is always **timestamped** and signed by CA.

They are distributed to a remote entity using CRL Distribution Point or in short CDP.

Time Stamping

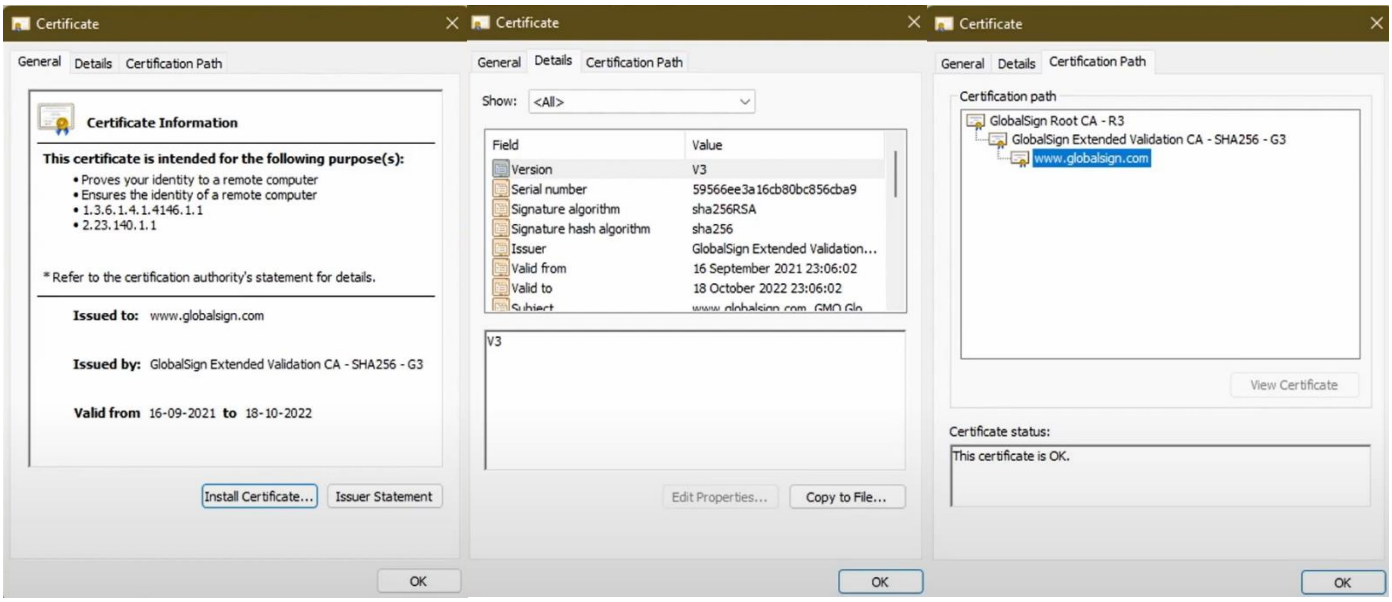
Time Stamping is a crucial part of **PKI**. it is used for proving the existence of a data at certain point in time. For example, I'm sure you must have signed a physical document or seen a document signed with a date and maybe a time. This way it's easy for anyone to prove that a document was signed on a certain date. The receiver of the document can't refute that the document was signed in some other date. Digital timestamping is exactly that. If you are a developer, you might want to sign and timestamp your software, so you could prove that your software was signed on a certain date and time.

To digitally timestamp a document, your document signing program calculates a hash of that document, and then sends it over to the **TSA** or **Time Stamping Authority**. This process is known as **Time Stamping Request**. The TSA would then add a timestamp with the current date and precise time to that hash. They would rehash that data and then sign it with their own private key. this signed data is then sent back to the requester who then adds that timestamp to their document. The response that you receive from TSA is called **Time Stamping Response**.

now i just need you

Contents of a Digital Certificate

Now I'm going to dissect a digital certificate, and explain all contents we see in it. For this exercise, I have already downloaded a global sign certificate. The three screenshots that you see are from my windows machine.



When you open a certificate file on windows, you should see three tabs:

- **General**
- **Details**
- **Certification Path**

General tab

The first thing you would see when you open a certificate file in windows, is the **general tab** and the first thing you see in the general tab is the **purpose of the certificate**. I have highlighted that portion with a red outline. You can clearly see that the purpose of the certificate is to prove identity of a client and the server.

Validity Period

We also see to whom the certificate was **issued to** and who was the **issuer**. It also includes the **start date** and **the end date** or you can just call it **validity period** of this certificate.

When a certificate expires, it will be considered untrusted. For example, I'm sure, you know what a passport is. A passport has an expiration date. It's valid for like 10 years for an adult and I guess it's for five years if you are a miner. You will not be allowed to travel using an expired certificate. Similarly, if your document signing certificate expires then your signed document will not be trusted by an operating system. In most cases, a document signing software would simply reject it. You will then have to contact your Certificate Authority to get your certificate renewed.

Install Cerificate

I downloaded this global science certificate on my computer as a p7b file. Microsoft operating system gives me the option to install the certificate into my certificate store if required.

Issuer Statement

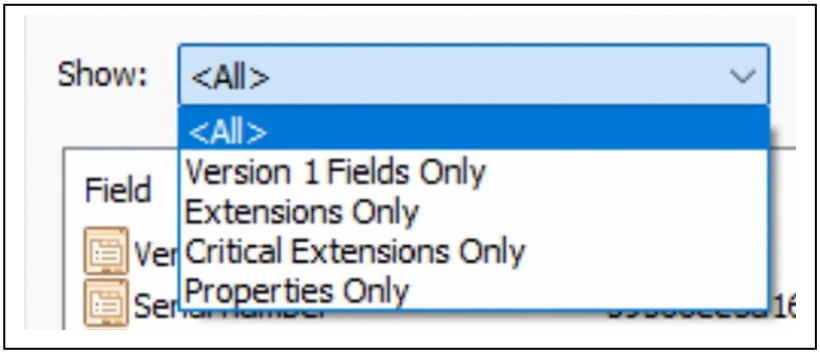
I also see **Issuer Statement**. clicking on Issuer Statement would open your web browser and display the Certification Practice Statement of the issuing CA. I will explain what a CPS is later on in this document.

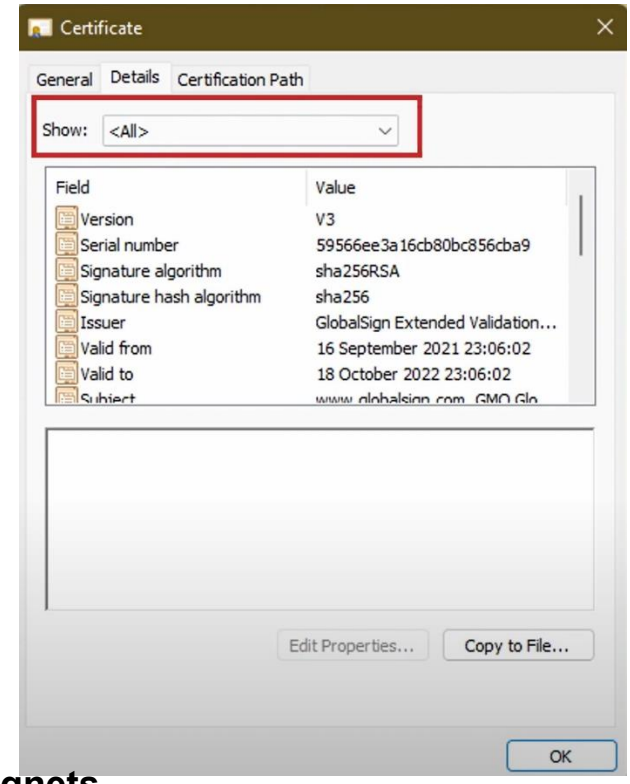
Details tab

Let's move on to **Details** tab. Image on the below shows what details tab looks like. In this tab, we see a drop-down list at the top with the label **Show**. The image on the left has all available options in the Show drop-down list:

- **Version 1 Fields Only** option shows only version 1 information from a certificate. This information includes information such as **version**, **serial number**, **signature algorithms**, **signature hash algorithm**, **issuer**, **valid to**, **valid from**, **subject**, **public key**, and **public key parameters**.
- **Extensions Only** would display all extensions which includes key usage, **extended key usage**, authority information access, certificate policy, CRL distribution point, **subject alternate names**, **authority key identifier**, **subject key identifier**, and **SCT list**.
- **Critical Extensions Only** displays extension that are marked as **Critical**.
- **Properties Only** would show the thumbprint.

Please don't worry I will explain each of these later in this document.

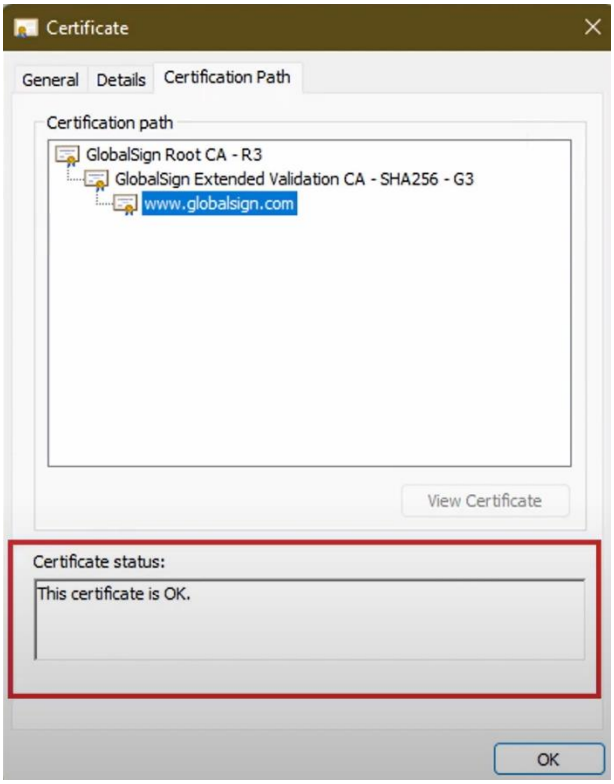




Certification Path tab

Certification Path tab shows the certificate chain. In this image:

- **www.globalsign.com** is the Subject for whom the certificate was issued to.
- It was issued by an intermediate CA with the subject name **GlobalSign Extended Validation CA - SHA256 - G3**
- And at the top we see the global sign root certificate which has **GlobalSign Root CA – R3** as a subject name.



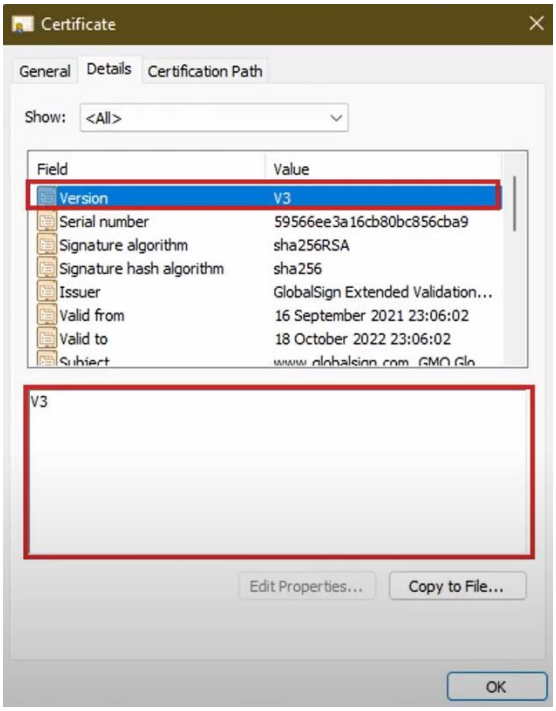
Certificate Status

You would also see **Certificate Status** down below. In this screenshot, you are seeing: **This certificate is OK.** Indicating that everything is good with this certificate. If there is a problem, you would see a message describing the problem that was found. For example, you might see an error message that says "**this certificate was revoked by its CA"** or you might see a message that says "**Issuer Certificate was not found"** or you could also see a message that says "**certificate is not trusted"**.

General tab in detail

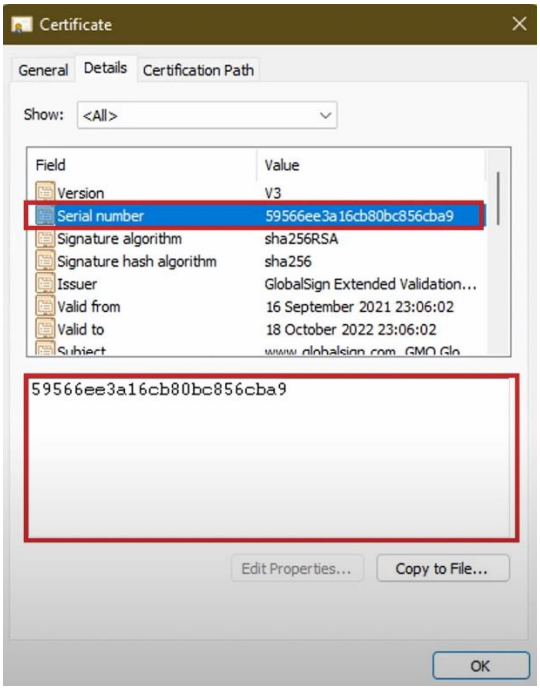
Let's go back to our details tab it has like so many options so let's discuss each of them.

Version Tab



Versions is the first option that you see in the details tab. Version shows **the version of x509 certificate** which is being used in this screenshot. We see version 3 as the value in the versions field.

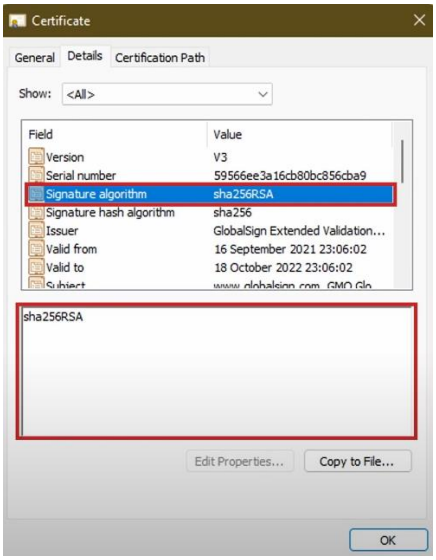
Serial Number



Next field after Version is a Serial Number which **shows a serial number of a certificate**. It is a unique positive long integer value assigned by a CA to each certificate. Now, some CAs may use a counter to generate these serial numbers but that's considered unsafe these days, so most CAs may use the **current time** so they would calculate the **epoch time** and they would use that as a serial number of a certificate. In some cases, CAs decide to generate a small hash value to be used as a serial number. The value that you see in this screenshot it's a hash value.

Signature Algorithm

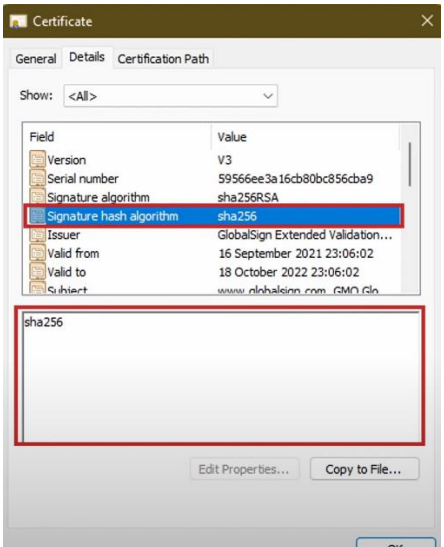
Next, we have a **Signature Algorithm** which **is the algorithm that was used to sign a certificate**.



In this screenshot, we see **sha256** with **RSA as the algorithm** which was used for signing. This means a **sha256** hash of this certificate was signed using **RSA Private Key**.

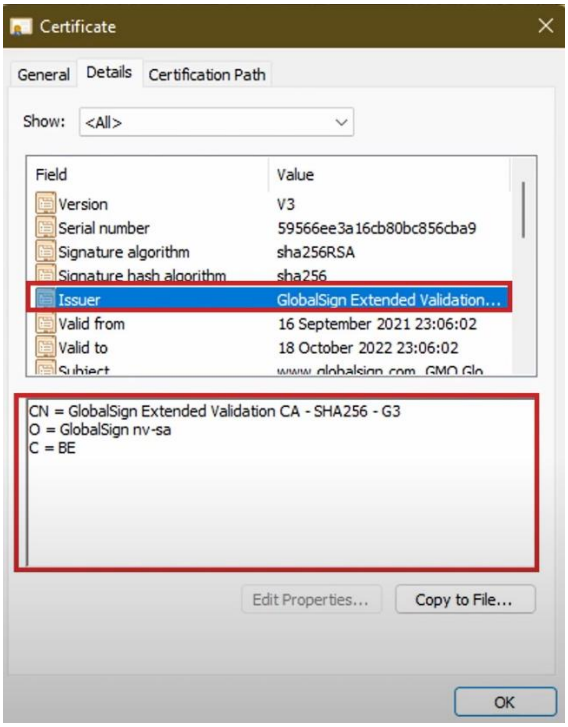
Signature Hash Algorithm

Signature Hash Algorithm is the hashing algorithm which was used for generating a hash of the certificate.



Issuer

Issuer is the name of the CA that signed a certificate request and issued the signed certificate. In this screenshot we see GlobalSign Extended Validation CA - sha256 - G3 as the common name for the issuing CA who issued the certificate.

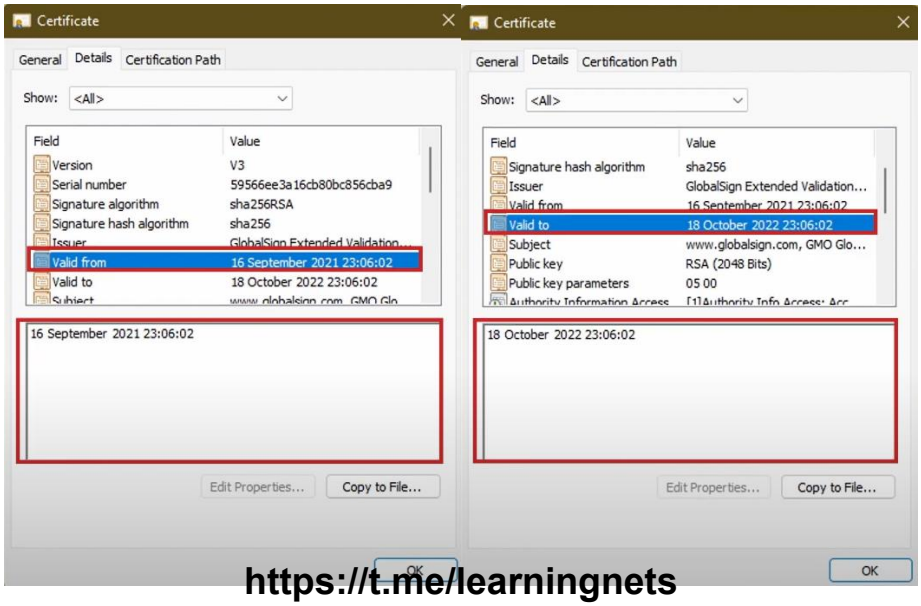


Valid from

Valid From shows a starting date of a certificate. Certificates have a validity date, and start date shows the date a certificate will be valid from.

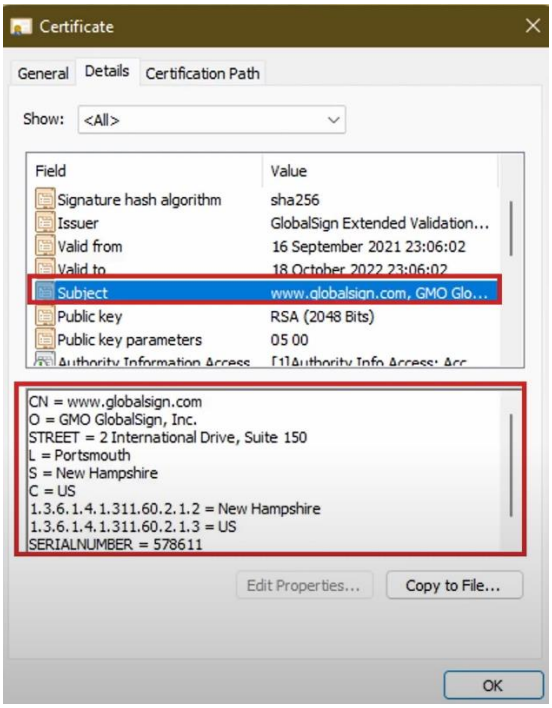
Valid from

Valid to is simply the date when a certificate would expire.



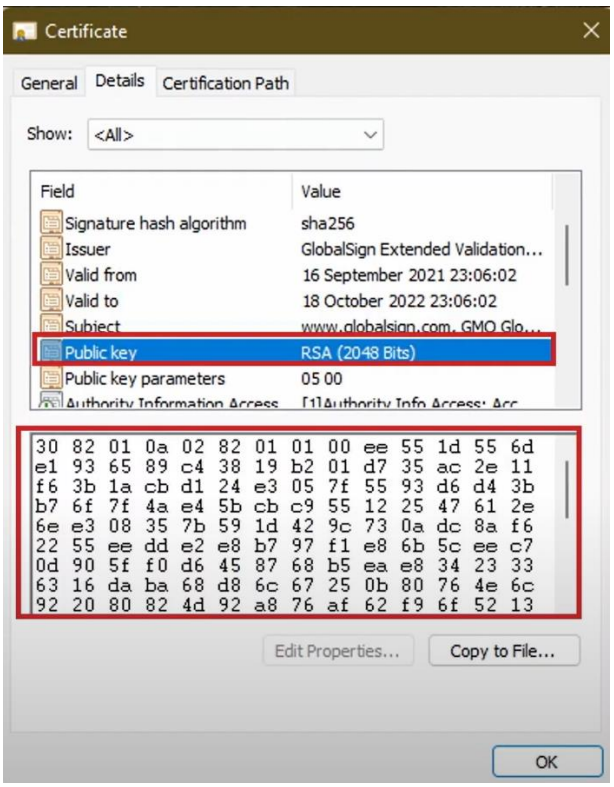
Subject

Subject field shows distinguished information about a subject certificate was issue to. Information contained in this field are **Common Name (CN)**, **Organization (O)**, **Organizational Unit**, **Country**, **Location**, **State**, **Email Address**, and few more.



Public Key

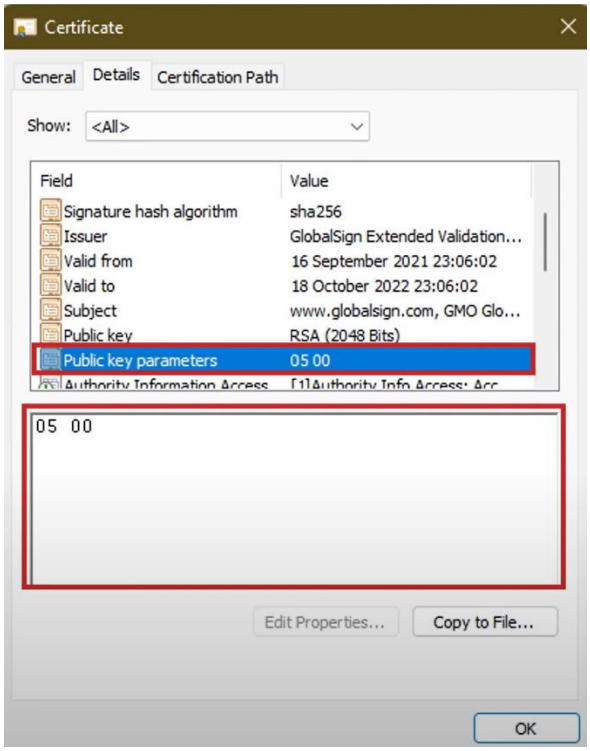
Public Key shows the ASN.1 encoded public key data, stored in the certificate in hex format.



Public Key Parameter

Public Key Parameters shows the parameters to be used to verify a certificate.

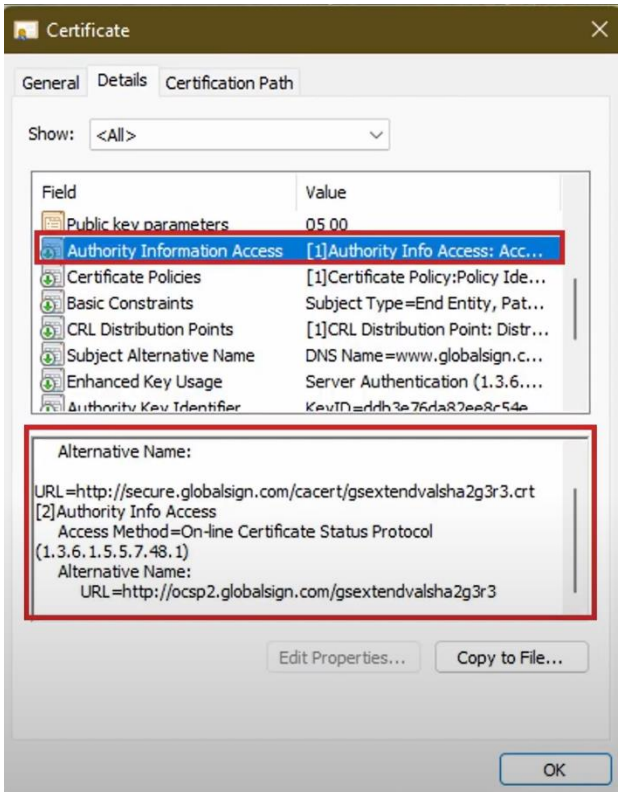
There are some signing algorithms that requires a parameter. For example, **RSA PSS** requires some parameter to sign and verify. You should see those parameters if such algorithms were used. In this image, we see **05 00** which means null. In case of easy dsa key you should see the curve in use access.



Authority Information Access

Authority Information Access gives information on how to get the issuer certificate. It should have a link from where the issuer certificate can be downloaded. It also has a link to **OCSP** from where the status of certificate can be verified. Like in this image, you can clearly see there's a URL **http://secure.globalsign.com/cacert/** and then we have the issuing CA certificate.

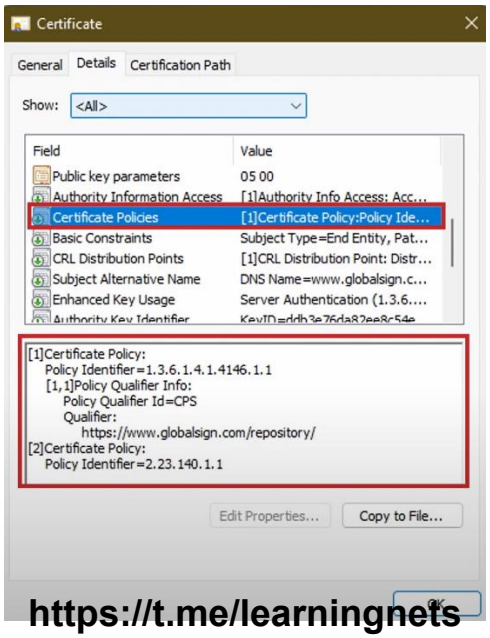
So, if you want you could use that link to directly download that certificate, and down below we have the link for **ocsp**. So, **http://ocsp2.globalsign.com** will be used to verify the status of this certificate.



Certificate Policies

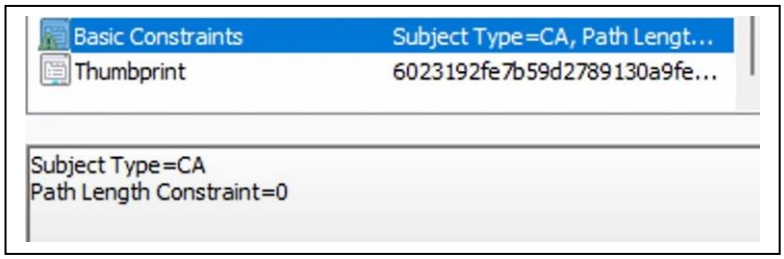
A **Certificate Policy** is simply a statement given by a Certificate Authority. It describes all security measures that must be followed by a Subject before a certificate can be issued to them. This statement by a CA clearly states how a subject would be verified, what they are allowed to do with those certificates, and under what circumstance a CA may decide to revoke their certificate. It also includes various security policies that Subject should implement.

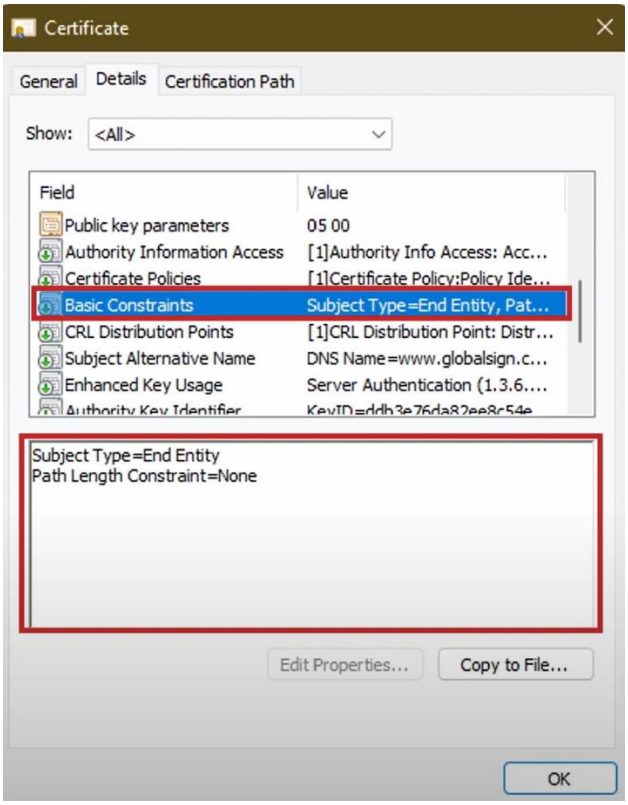
Along with certificate policy, a CA also includes something called **CPS** (Certification Practice Statement). A **Certificate Policy** discloses **what a subject should do**, and a **CPS** discloses **how they should do it**.



Basic Constraint

Basic Constraints is a property used to indicate whether a certificate belongs to a CA or some non-CA entity.





This extension has two attributes:

Subject Type

The first attribute is **Subject Type**. If you see Subject Type as **CA**, then it means it is a CA Certificate. Any other value apart from CA would mean it's a non-CA certificate.

Now the screenshot that you see on the right, shows a user certificate (**End Entity**). You can clearly see that it has a value of **End Entity** which means it is a non-CA certificate.

The screenshot that you see on the left, shows basic constraint and the basic constraint has Subject Type as CA, and that means that certificate is a CA Certificate.

Path Length Constraint

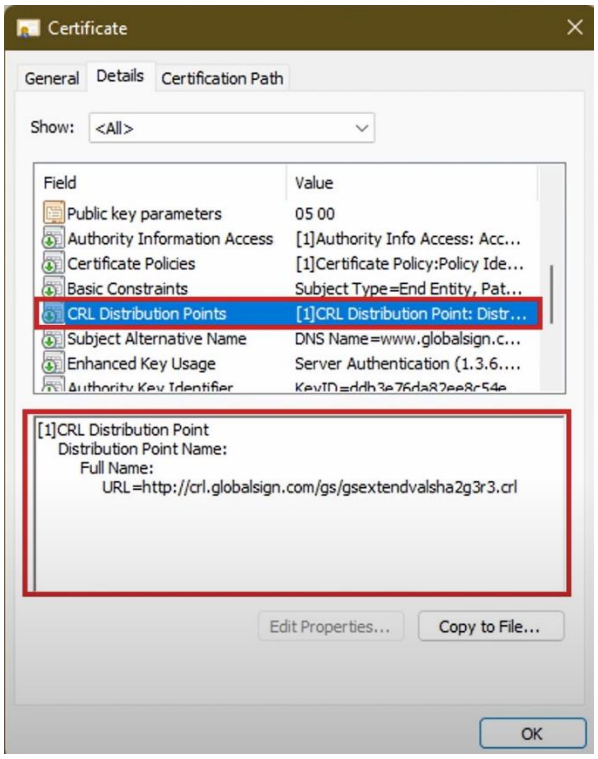
The second attribute that we have for basic constraint, is called **Path Length**.

Now this property defines how many sub CAs can exist under a CA. That is if that certificate you are looking at, is a CA Certificate. The screenshot that you see on the left, shows Path Length as **0** which means no sub-CA can exist under the CA.

For example, if there's a CA called CA1 with a path length of 2, then that would mean CA1 will allow a maximum of two sub-CAs in the chain. So, CA1 can have CA2 under it, and CA2 can have CA3 under it. If the Path Length is none and the subject type is CA, then that would mean that CA can have as many sub -CAs as required.

CRL Distribution Point

A **CRL Distribution Point** or **CDP** is the path from where a remote client can download a CRL (Certificate Revocation List). There's usually a URL mentioned in CDP which points directly to a CRL file. This CRL file can be used by the client to check the status of a certificate. If you look at the screenshot, you can clearly see there's a URL that says **http://crl.globalsign.com/gs** and then we have the CRL file. So, this URL can be used to check the status of a certificate.



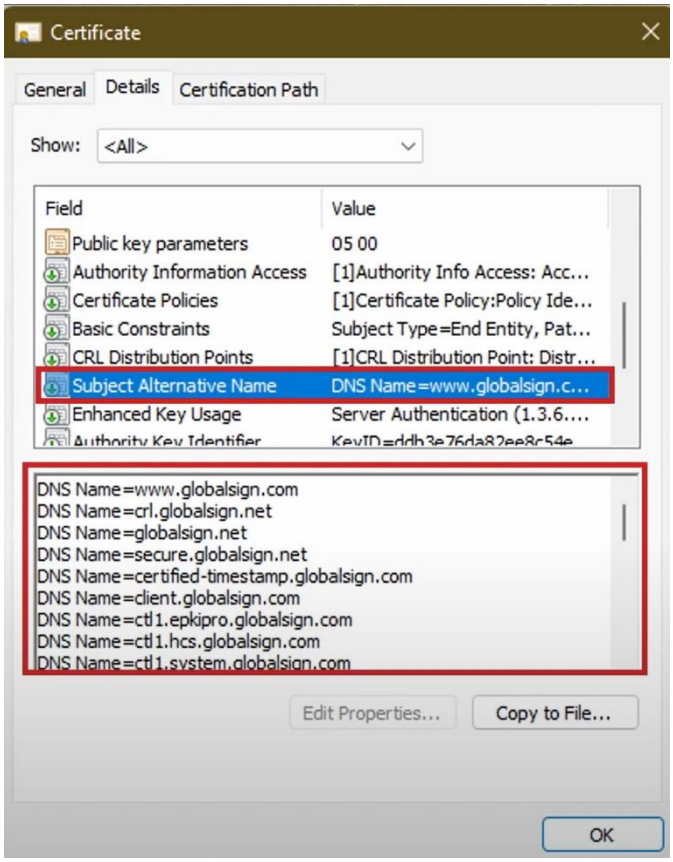
Subject Alternative Name

Subject Alternate Name or **SAN** has all alternate subject names for a certificate. Alternate names can be a **domain name**, **ip address**, or a **wild card**.

For example, just imagine that I have a site called Fortinet.com which can be accessed over **https**. A web browser would expect the Common Name of my SSL Certificate to be Fortinet.com, that is the Common Name should match with the Domain Name, if it does not match, then there would be an untrusted connection.

Now imagine that there are some sub domains such as videos.fortinet.com or blog.fortinet.com. Instead of creating multiple SSL Certificates for those domains, I can simply include those domains as a **Subject Alternate Name**. I can even mention the public ip address of Fortinet.com in SAN.

If I don't mention those domains in SAN, then accessing any of those sub domains would result in an untrusted connection.



Have a look at the screenshot. You can see that SAN has several sub domains for GlobalSign.

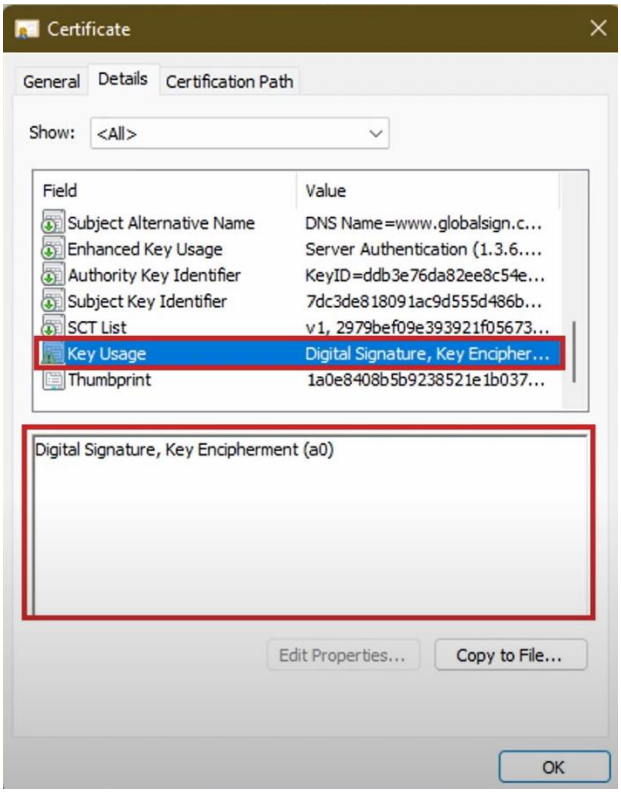
Here instead of using separate certificates for all those sub domain, GlobalSign decided to include all those domains in SAN. So, now if you try to access any of those domains, your browser will be able to establish a trusted **https** connection.

Key Usage

Key Usage is used to define what a certificate is allowed to do. Think of it this way: imagine you have a Certificate with RSA Public Key. Now, you could use that public key to **encrypt data** or **encrypt a key**. You could also use that public key to **verify a signature of a data or a CRL**.

Now, what if you want a Public Key to be restricted to just verifying signature offer data?

After all, these certificates are distributed publicly. So, depending on who these certificates are issued to, I want to restrict what they can do with it. I can do this using **Key Usage extension**. **Key Usage** allows a key to be restricted to some specific type of operations. These operations are indicated using **Usage bits**. There are total of **9** usage bits starting from usage bit **0** which is **DigitalSignature** and all the way till usage bit **8** which is **decipherOnly**.



Key Usage Bits

0- DigitalSignature

The first usage bit that we have is usage bit **0** which is DigitalSignature. Using this usage bit, I can restrict a Public Key to **only verify assigned data and nothing else**.

1- NonRepudiation

The next usage bit that we have is usage bit **1** which is for NonRepudiation. I can use it to **prevent a signing entity from falsely denying some action**.

2- KeyEncipherment

Usage bit **2** is for keyEncipherment which will **allow a Public Key to encrypt another key for key transportation purpose**. For example, if you have an AS key and if you want to share that AS key with your colleague, you could simply encrypt it using the the Public Key that they provided, and share the encrypted data and they should be able to decrypt it using their own private key.

3- DataEncipherment

We have usage bit **3** which is DataEncipherment. **This allows a public key to encrypt some data**.

4- KeyAgreement

Next, we have usage bit **4** which is for **KeyAgreement**. **This allows a Public Key to be used for key agreement purpose**, that is if you're using a supported algorithm such as "Diffie-Hellman key exchange calculator" for example.

5- **KeyCertSign**

Usage bit **5** is KeyCertSign. **this allows a public key to verify signature of a certificate**.

6- **cRLSign**

Usage bit **6** is cRLSign. **This allows a public key to verify signature of a CRL**.

7- **encipherOnly**

We have usage bit **7** which is encipherOnly. This bit only works when KeyAgreement bit is also set. **This allows a Shared-Secret-Key derived using key exchange to be only allowed for encryption purpose**.

8- **decipherOnly**

Usage bit **8** is decipherOnly which is the opposite of encipherOnly. This usage bit will also work if KeyAgreement bit is set, and **it will only allow us Shared-Secret-Key which was derived you during key exchange to only decrypt data**.

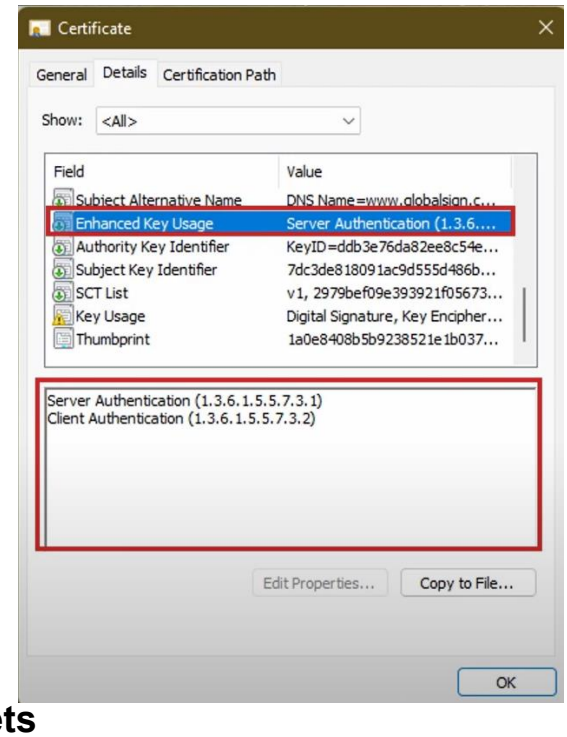
Now, depending on what you want to do, you could set multiple Key Usage set. For example, a **DigitalSignature** Usage Bit can be set along with **CRLSign** and **KeyCertSign**. This allows a public key to verify signature of a data. It will also allow verifying signature of a CRL and verifying signature of a certificate.

Enhanced / Extended Key Usage

Next, we have **Enhanced Key Usage** which is also known as Extended Key Usage or EKU In short. EKU can be used to specify for what purpose a certificate is actually meant for. Extended Key Usage may seem like Key Usage properties. The key difference is that EKU is very specific.

- **EKU specifies the purpose of certificate**
- **whereas Key Usage specifies what a certificate can do**

- **ServerAuthentication**
- **ClientAuthentication**
- **CodeSigning**
- **EmailProtection**
- **TimeStamping**
- **OCSPSigning**
- **anyExtendedKeyUsage**



For example, a Certificate with **DigitalSignature** set for Key Usage can be used for verifying signature of a data. Now, if I add **CodeSigning** as an Extended Key Usage, then I'm restricting that Public Key of a Certificate to only verify signature of a code or an application.

The key uses specified in RFC5280 are as follows:

- **ServerAuthentication**
  The first one is ServerAuthentication which is used for server authentication.

- **ClientAuthentication**
  We have ClientAuthentication which is meant for client authentication

- **CodeSigning**
  We have CodeSigning which is used for signing binary such as **Executable (.exe)** files or **jar files** or **dll files** or maybe **RPM files**

- **EmailProtection**
  EmailProtection is used for protecting your email like signing email or encrypting your email

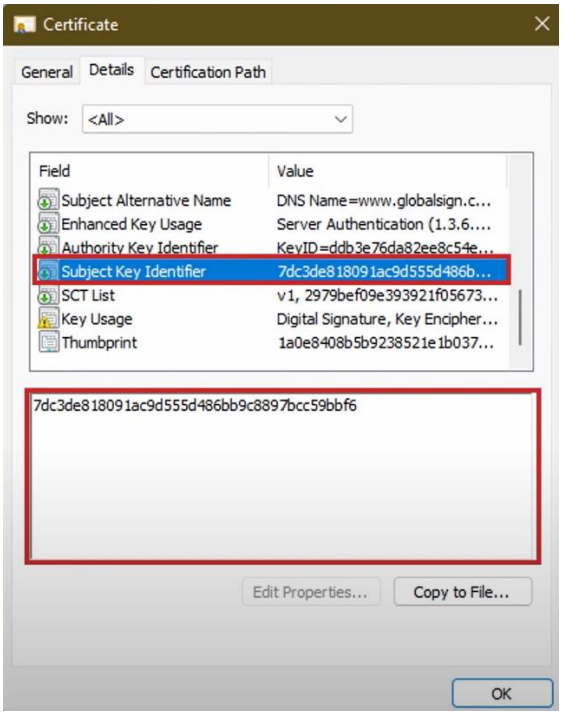
- **TimeStamping**
  We have TimeStamping which is used for Time Stamping purpose

- **OCSPSigning**
  OCSPSigning is used for signing an OCSP Response.

- **anyExtendedKeyUsage**
  And we have anyExtendedKeyUsage which will allow any kind of usage. It is totally unrestricted.

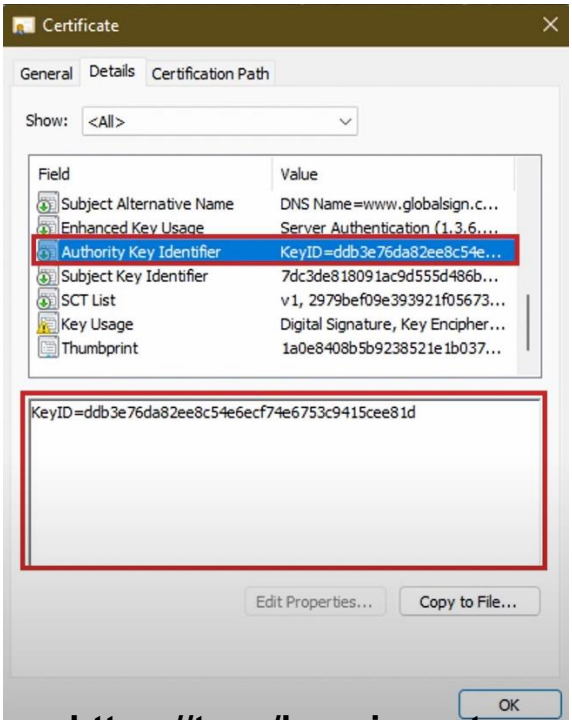
Subject Key Identifier

A **Subject Key Identifier** is used for identifying a certificate that contains a particular Public Key. This identifier is simply a **SHA1 hash** of a Public Key.



Subject Key Identifier is necessary for **constructing a Certificate Path**. Let me explain how certificate path is constructed.

We talked about **Subject Key Identifier** which is a **SHA1 hash** of a Public Key. The main use of Subject Key Identifier is to identify a Certificate that Contains a Particular Public Key. I also mentioned that Subject Key Identifier is used for building a Certificate Chain. Let me explain how:



Authority Key Identifier is a **Subject Key Identifier of a CA** that issued a Certificate. When a certificate request is signed by CA, the Subject Key Identifier of that CA is embedded in the Signed Certificate as Authority Key Identifier. This makes it easier to identify who signed a certificate. Every operating system has a trust store which contains a bunch of CA certificates. When a Signed Certificate is used, the Authority Key Identifier embedded in it, is used to identify the CA that signed that Certificate. This process is repeated multiple times till a **Root CA Certificate** is found. This is how an operating system builds a Certificate Chain.

SCT List

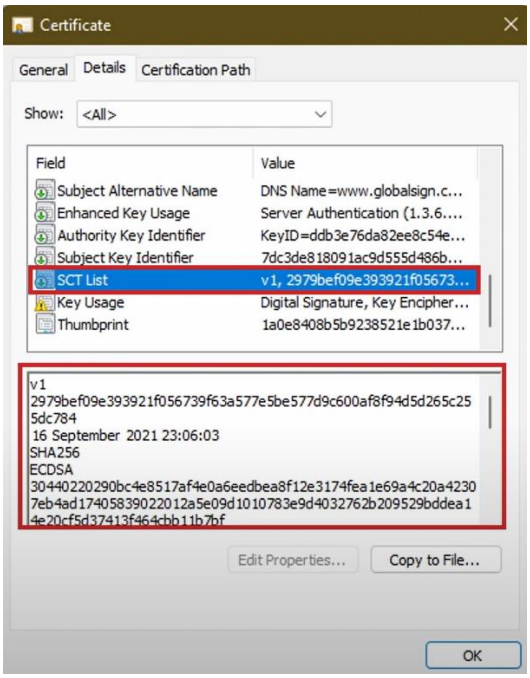
SCT is a short form for "Signed Certificate Timestamp". Before I explain what SCT is, I think I should talk about **CT Log** which is also known as "Certificate Transparency Log".

A Certificate Transparency (CT) is a new protocol which is used for publicly logging all Certificates signed by CA. This log is publicly accessible and makes it easy for anyone to audit their CA. The idea is to eventually enforce it on all clients. I believe **google chrome** browser already has this enforced.

A client can simply refuse to accept a certificate that does not appear in the CT Log. This will eventually force all certificate authorities to publish their signed certificates into CT Log.

CT Logs are appended only, which means a Certificate Authority is only allowed to add entries to it. They can't make changes to it, or remove an entry. The purpose of CT Log is to avoid misuse of a certificate. So, if anyone finds out about a certificate which is being misused, they can act on it as they see fit.

When a new log is appended to CT Log, a Signed Certificate Timestamp (SCT) is returned and SCT is the proof that a certificate is signed by a CA and has been added to a CT Log.



Digital Certificate – Part 3

Why does FortiGate use Digital Certificates?

Why does FortiGate use Digital Certificates?

FortiGate uses digital certificates to enhance security in multiple areas.

Traffic Inspection

FortiGate uses digital certificates for **inspection**, **mainly outbound or inbound traffic inspection**.

- If FortiGate trusts the certificate, it permits the connection.
- But if FortiGate does not trust the certificate, it can prevent the connection.

How you configure FortiGate determines the behavior; however, other policies that are being used may also affect whether FortiGate accepts or rejects connection attempts. **FortiGate can also inspect certificates to identify people and devices (in the network and on the internet)**, before it permits a person or device to make a full connection to the entity that it is protecting.

Privacy

FortiGate uses digital certificates to enforce privacy. Certificates, and their associated private keys, **ensure that FortiGate can establish a private SSL connection to another service**, such as FortiGuard, or a web browser or web server.

Authentication

FortiGate also uses certificates for **authentication**. Users who have certificates issued by a known and trusted CA can authenticate on FortiGate to access the network or establish a VPN connection. Administrator users can use certificates as a second-factor authentication credential to log in to FortiGate instead of tokens. (Two-Factor Authentication methods: Token, Certificate)

Privacy

FortiGate Uses SSL for Privacy

- SSL features:
  - Privacy of data
  - Identifies one or both parties using certificates
  - Uses symmetric and asymmetric (public key) cryptography
- Symmetric cryptography
  - Uses the same key to encrypt and decrypt data
  - Need safe way to exchange the single key
  - Faster than asymmetric cryptography
  - Used by FortiGate for exchange with other managed devices, for example, FortiManager
- Asymmetric cryptography
  - Uses two keys, one public and one private
  - Only the public key is shared with peers
  - Slower and more resource intensive than symmetric cryptography
  - Widely used, for example, HTTPS traffic

FortiGate uses SSL for Privacy to:

- **Ensure that data remains private when connecting with servers, such as FortiGuard, and with clients, such as a web browser**.
- Another feature of SSL is that **FortiGate can use it to identify one or both parties using certificates**.

SSL uses two ways of cryptography to establish a secure session between two points:

- **Symmetric**
- **Asymmetric**

It is beneficial to understand the high-level process of an SSL handshake in order to understand how FortiGate secures private sessions.

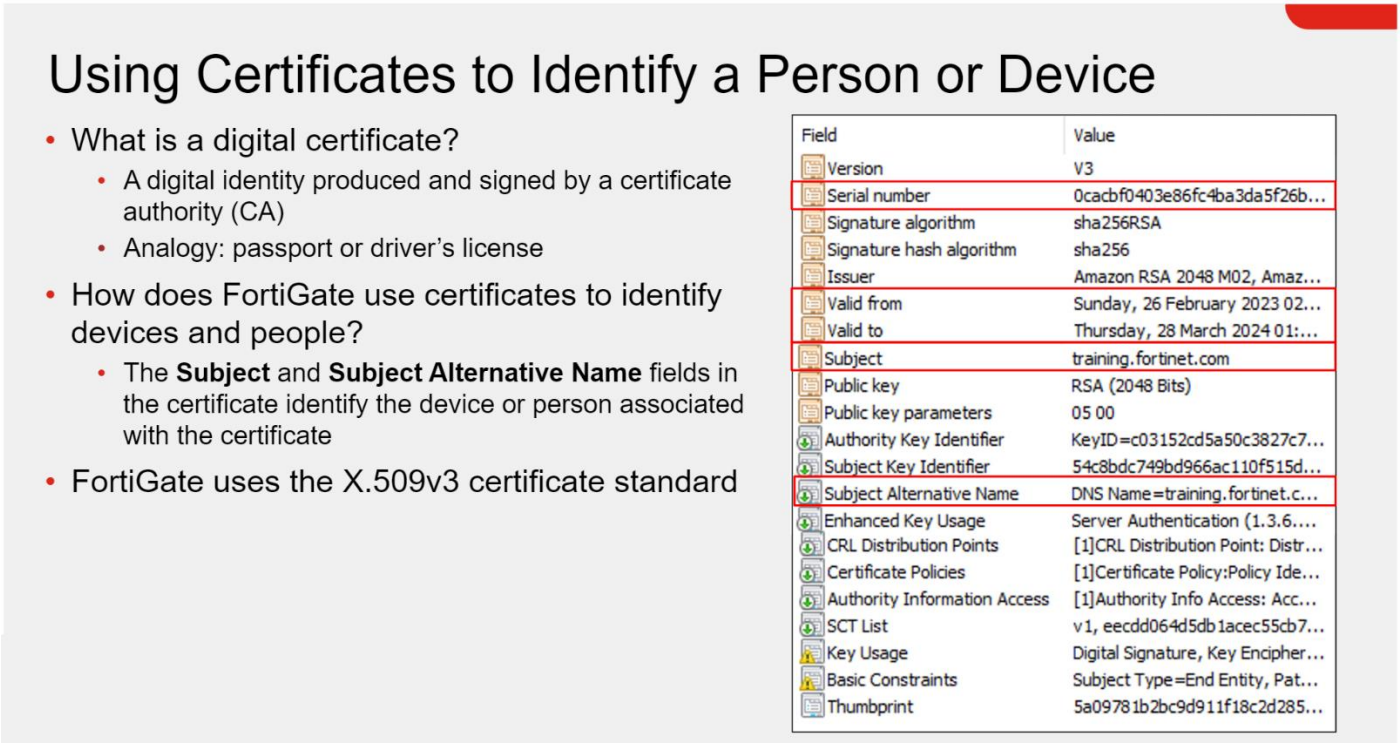
Symmetric Cryptography

For symmetric cryptography, **the same key is used to encrypt and decrypt the traffic**. This process requires fewer computing resources and is faster than asymmetric cryptography. However, one drawback is the requirement to share the key between participating devices in a safe way. When FortiGate establishes an SSL session between itself and another device, it must share the symmetric key (or rather the value required to produce it—usually the password you configure), so that data can be encrypted by one side, sent, and decrypted by the other side.

Asymmetric Cryptography

Asymmetric cryptography uses a pair of keys: One key performs one function, and the other key performs the opposite function. When FortiGate connects to a web server, for example, it uses the web server public key to encrypt a string known as the premaster secret. The web server private key decrypts the premaster secret.

Using Certificates to Identify a Person or Device



What is a digital certificate?

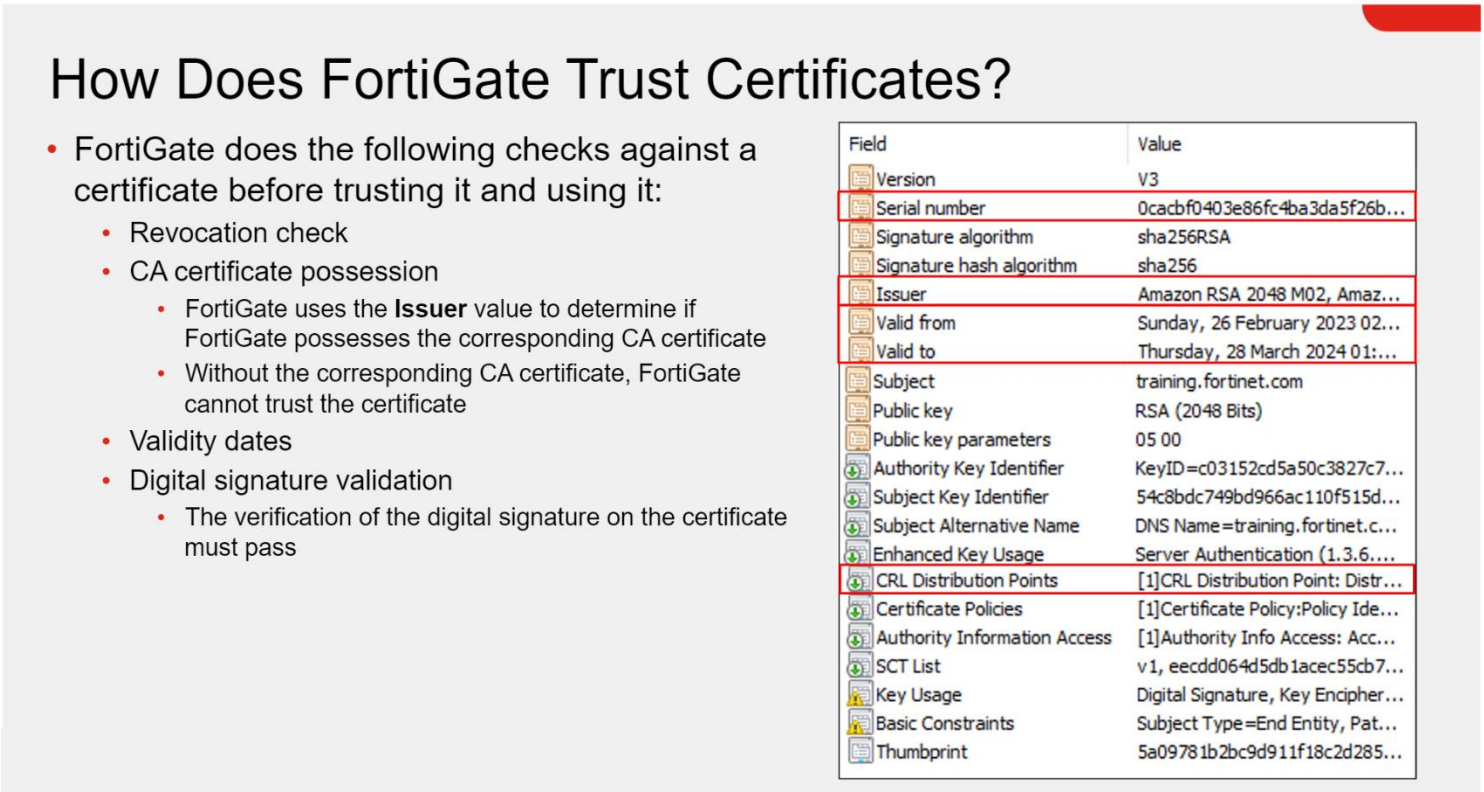
A digital certificate is a digital document produced, and signed by a Certificate Authority (CA).

It identifies an end entity, such as **a person** (for example, Joe Bloggins), **a device** (for example, webserver.acme.com), or **thing** (for example, a Certificate Revocation List (CRL)).

How does FortiGate use Certificates to identify devices and people?

FortiGate identifies the device or person by reading the Common Name (CN) value in the **Subject** field, which is expressed as a distinguished name (DN). FortiGate could also use alternate identifiers, shown in the **Subject Alternative Name** field whithin the certificate, whose values could be a network ID or an email address, for example. FortiGate can use the **Subject Key Identifier**, and **Authority Key Identifier** values to determine the relationship between the issuer of the certificate (identified in the Issuer field), and the certificate. FortiGate supports the **X.509v3 certificate** standard, which is the most common standard for certificates.

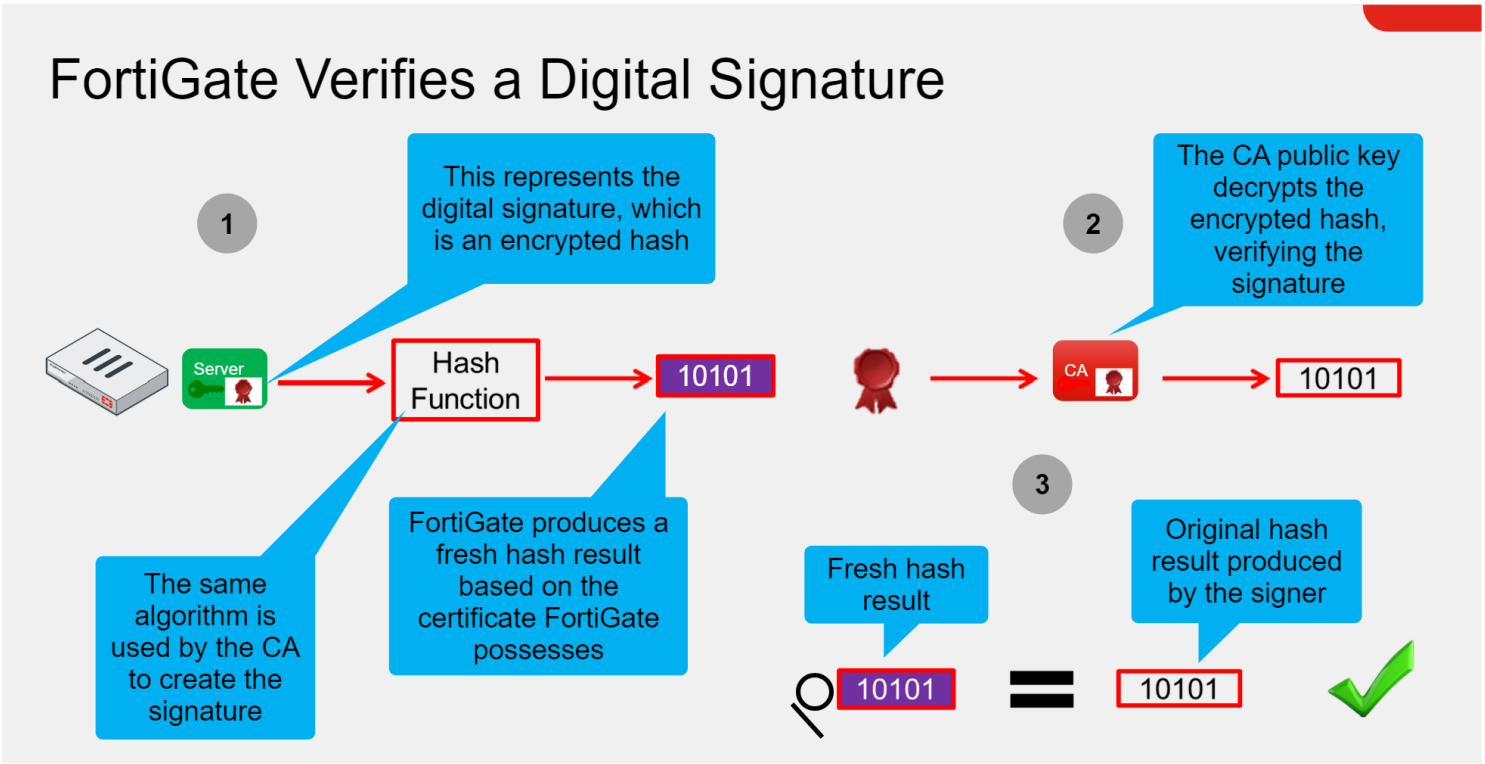
How does FortiGate Trust Certificates?



FortiGate runs the following checks before it trusts the certificate:

- Checks the Certificate Revocation Lists (CRLs) locally on FortiGate to verify if the certificate has been revoked by the CA.
  - FortiGate can download the relevant CRLs, and check if the serial number of the certificate is listed on the CRL. If the certificate is listed, it means that it has been revoked, and it is no longer trusted.
  - FortiGate also supports the Online Certificate Status Protocol (OCSP). When FortiGate uses the OCSP, it interacts with an OCSP responder (FortiAuthenticator acts as the OCSP responder) to check if the certificate is still valid.
- Reads the value in the **Issuer** field to determine if it has the corresponding CA certificate. Without the CA certificate, FortiGate does not trust the certificate.
- Verifies that the current date is between the **Valid from** and **Valid to** values. If it is not, the certificate is rendered invalid.
- Validates the **signature on the certificate**. The signature must be successfully validated.

FortiGate verifies a Digital Signature



FortiGate Validates the **signature on the certificate**. The signature must be successfully validated.

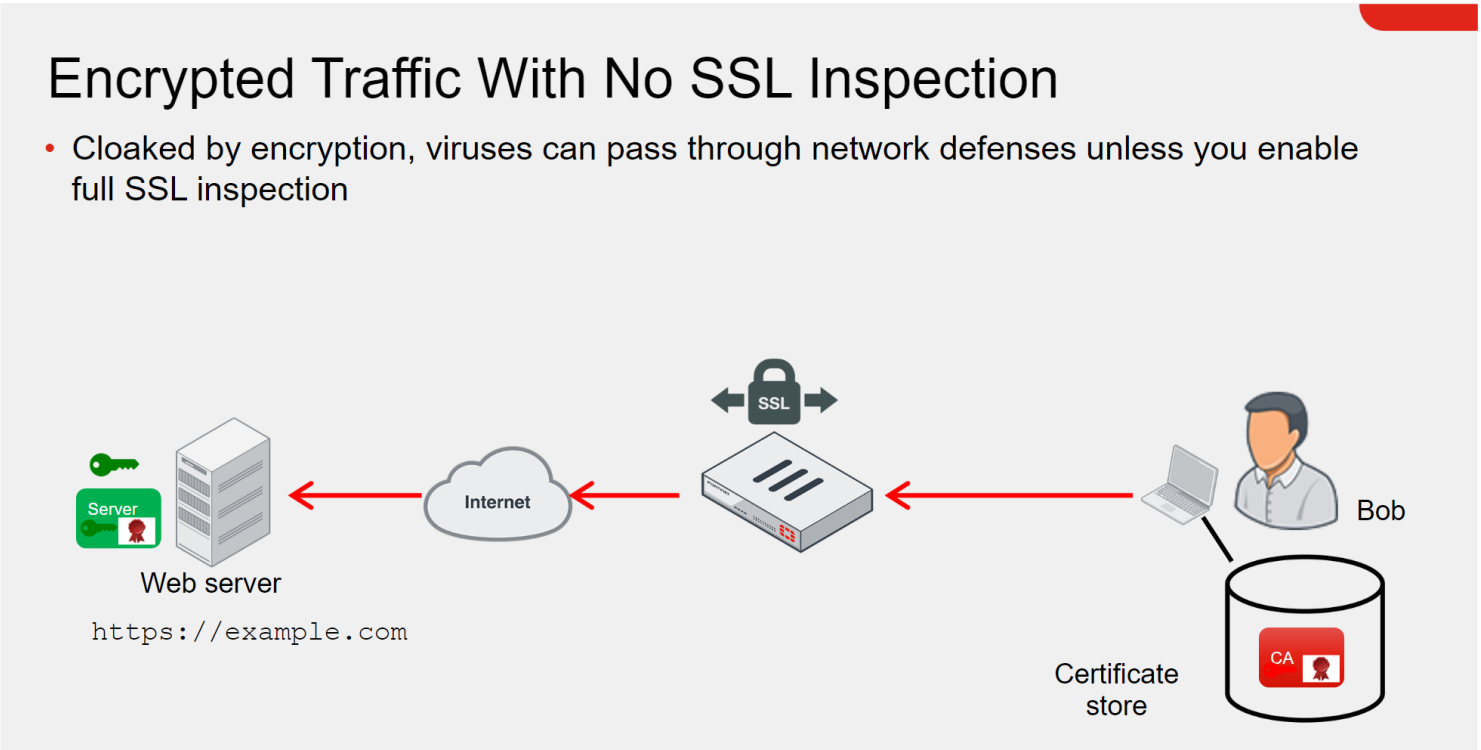
Before it generates a digital signature, the CA runs the content of the certificate through a hash function, which produces a hash result. The hash result, which is a mathematical representation of the data, is referred to as the original hash result. The CA encrypts the original hash result using its private key. **The encrypted hash result is the digital signature**.

Step 1) When FortiGate verifies the digital signature, it runs the certificate through a hash function, producing a fresh hash result. FortiGate must use the same hash function, or hashing algorithm, that the CA used to create the digital signature. The hashing algorithm is identified in the certificate.

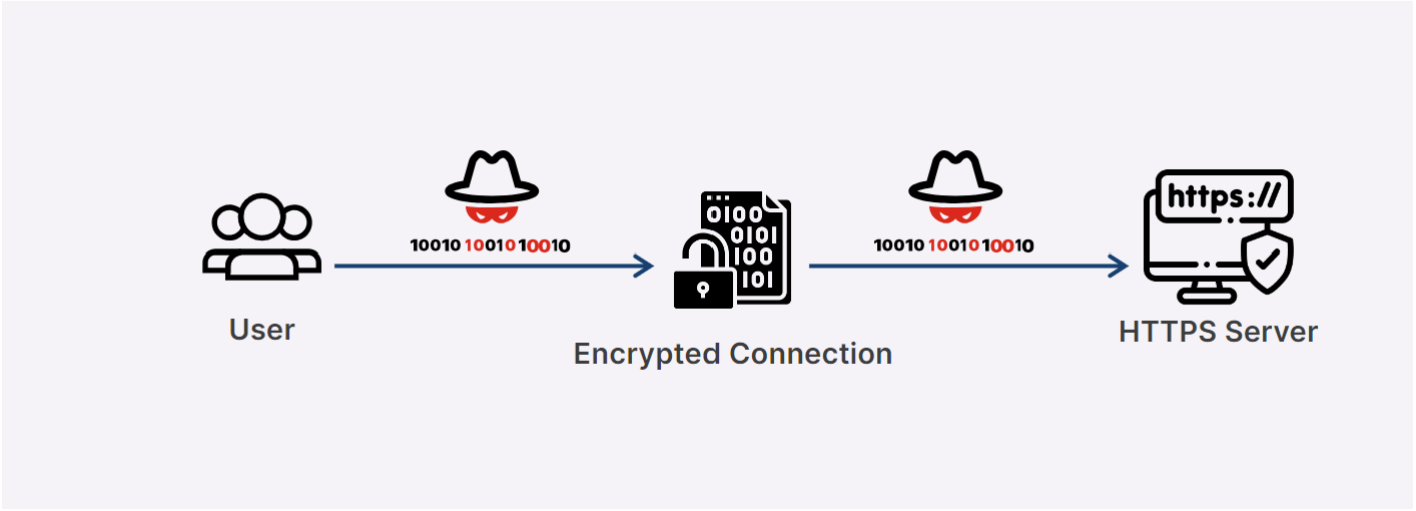
Step 2) FortiGate decrypts the encrypted hash result (or digital signature) using the CA public key and applying the same algorithm that the CA used to encrypt the hash result. This process verifies the signature. If the key cannot restore the encrypted hash result to its original value, then the signature verification fails.

Step 3) In the third, and final, part of the verification process, FortiGate compares the fresh hash result to the original hash result. If the two values are identical, then the integrity of the certificate is confirmed. If the two hash results are different, then the version of the certificate that FortiGate has is not the same as the one that the CA signed, and data integrity fails.

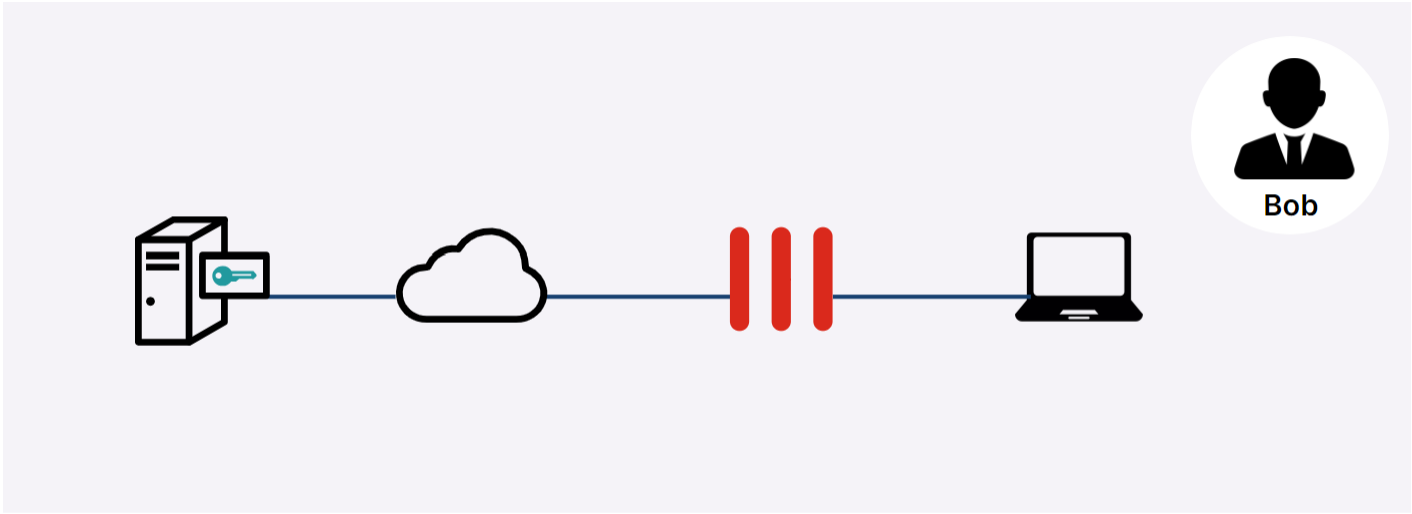
Encrypted Traffic with No SSL Inspection



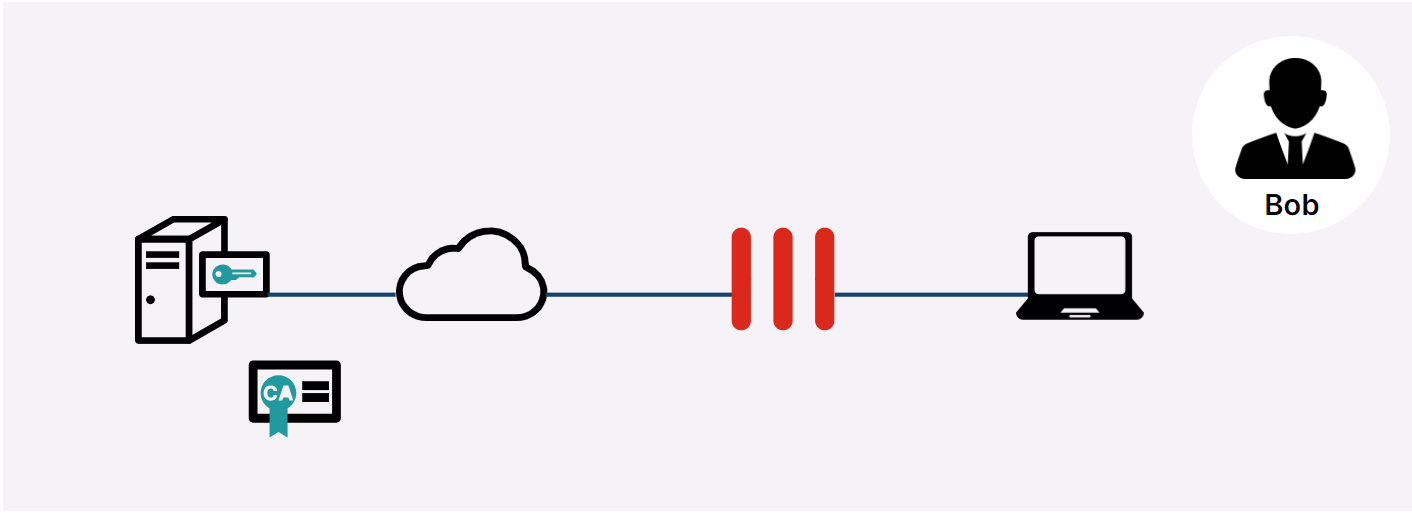
HTTPS offers protection by applying encryption to web traffic; however, it also introduces a potential security risk because attackers may attempt to use encrypted traffic to get around your network's normal defenses. For example, if a session is encrypted when you download a file containing a virus, the virus might get past your network security measures.

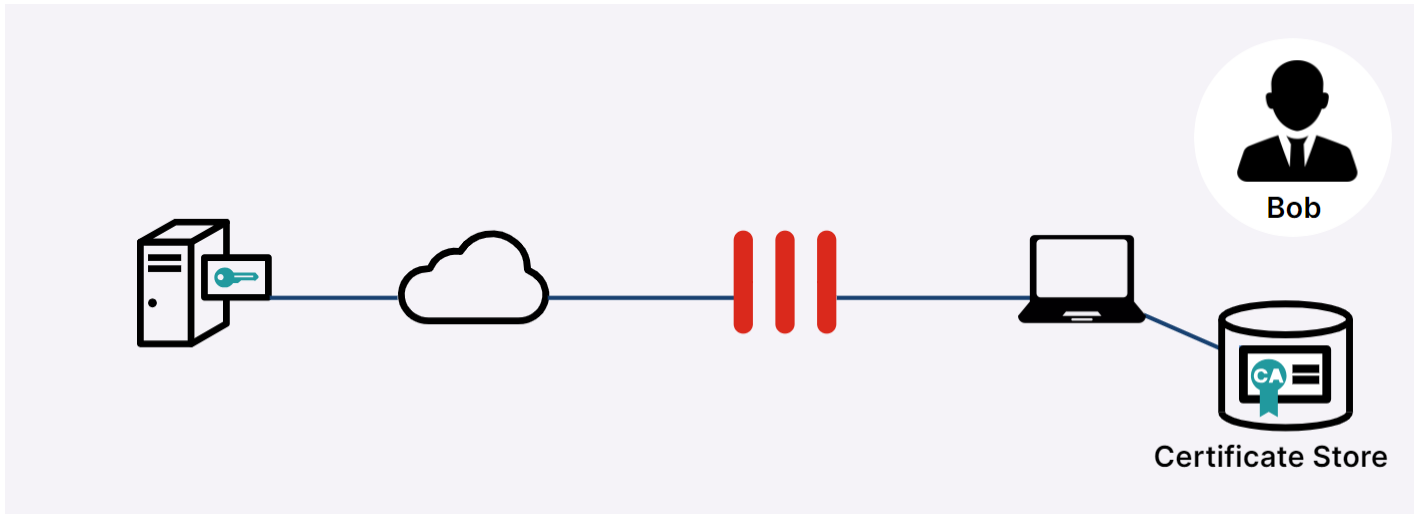


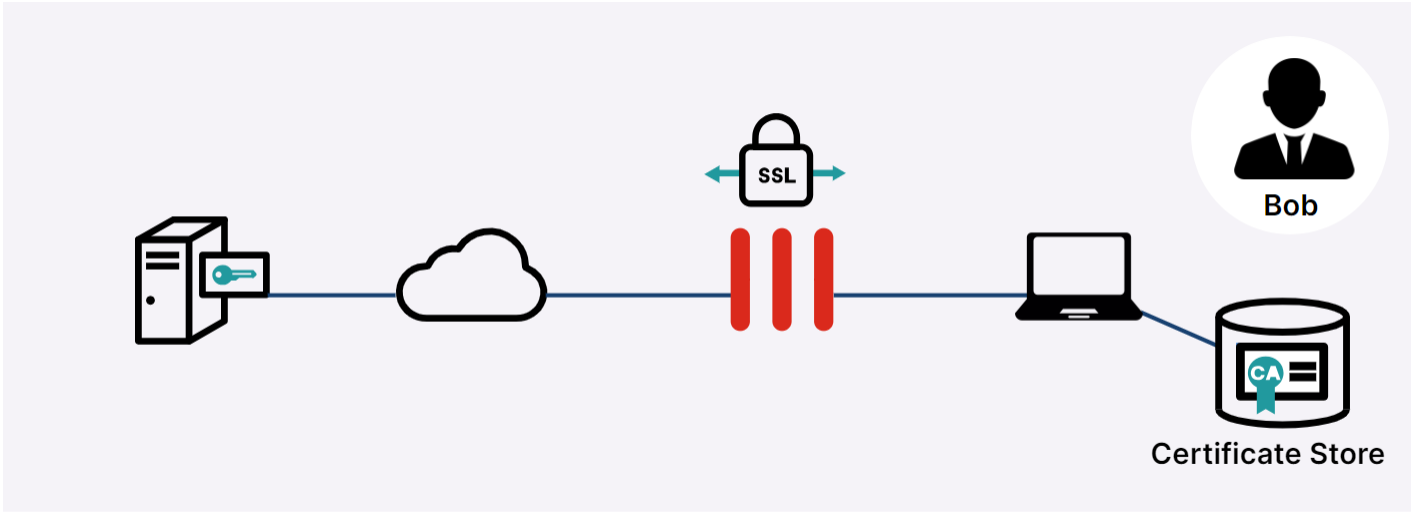
In the example shown on this slide, Bob connects to a site with a certificate issued by a legitimate CA.



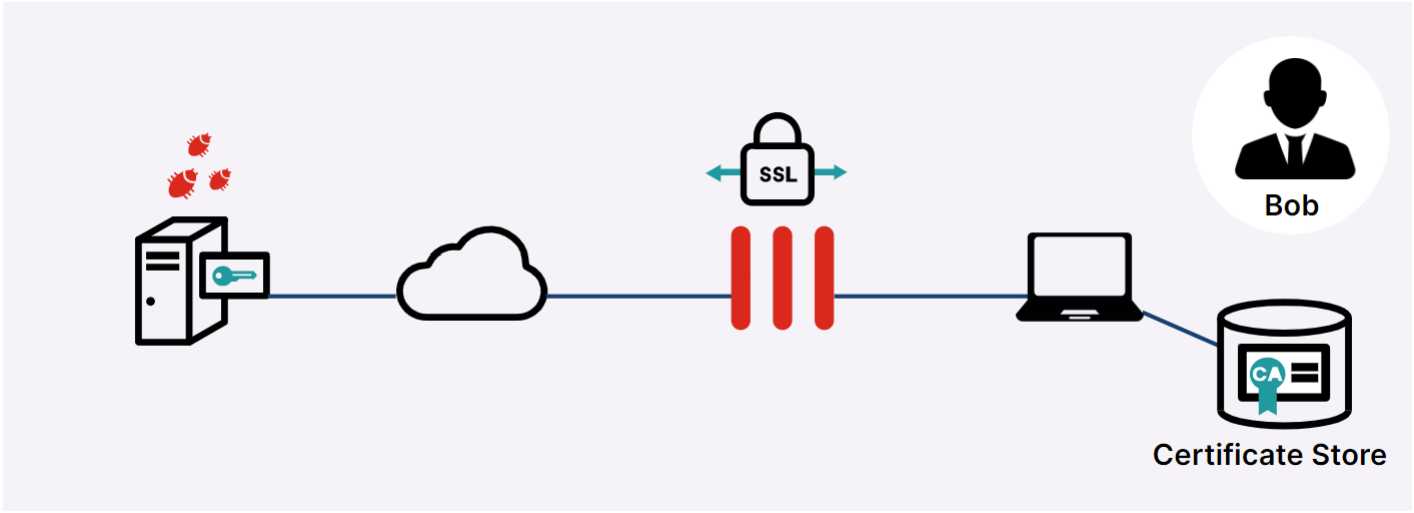
Because the CA is an approved CA, the CA verification certificate is in Bob's certificate store, and Bob's browser is able to establish an **SSL session** with the `example.com` site.



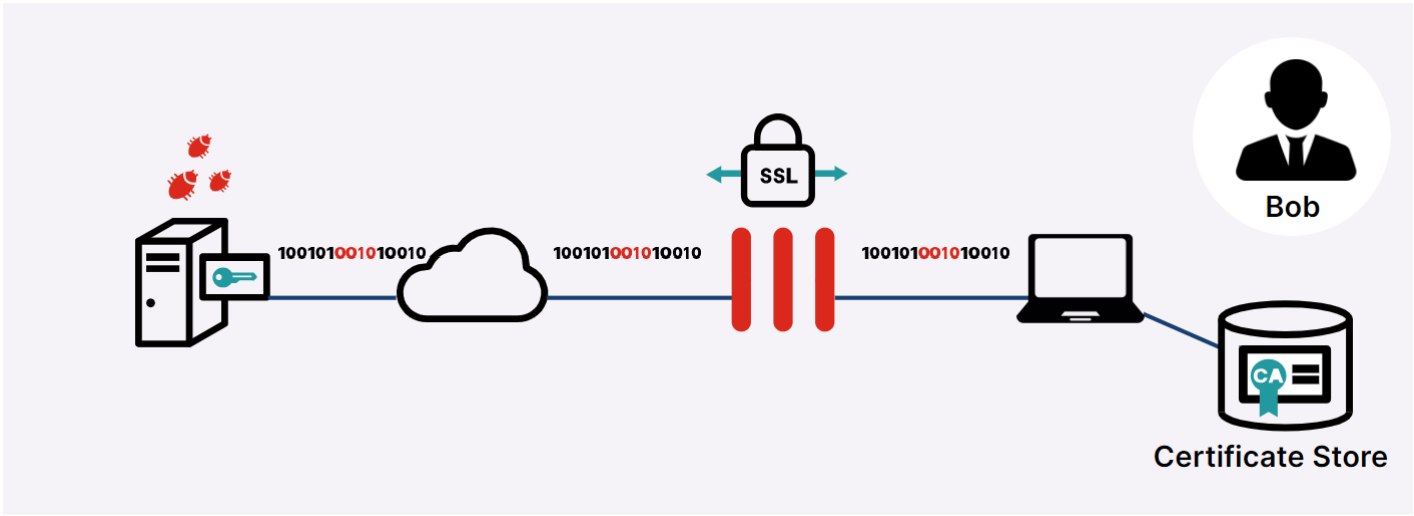


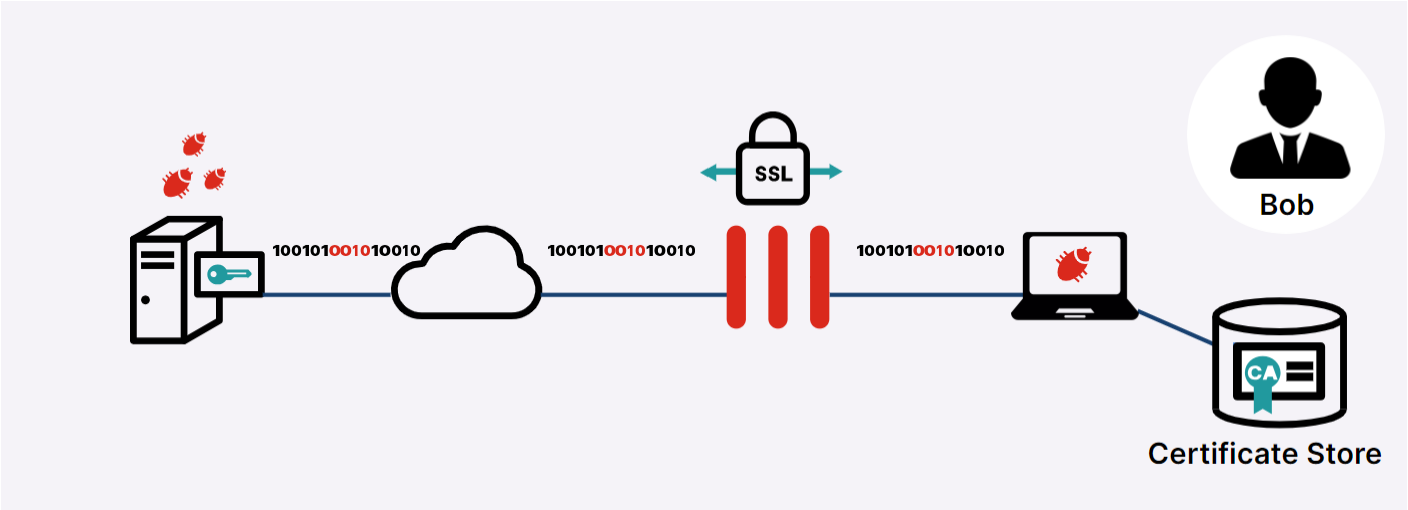


However, unknown to Bob, the `example.com` site has been infected with a virus.



The virus, cloaked by encryption, passes through FortiGate undetected, and enters Bob's computer. The virus is able to breach security because full SSL inspection is not enabled.



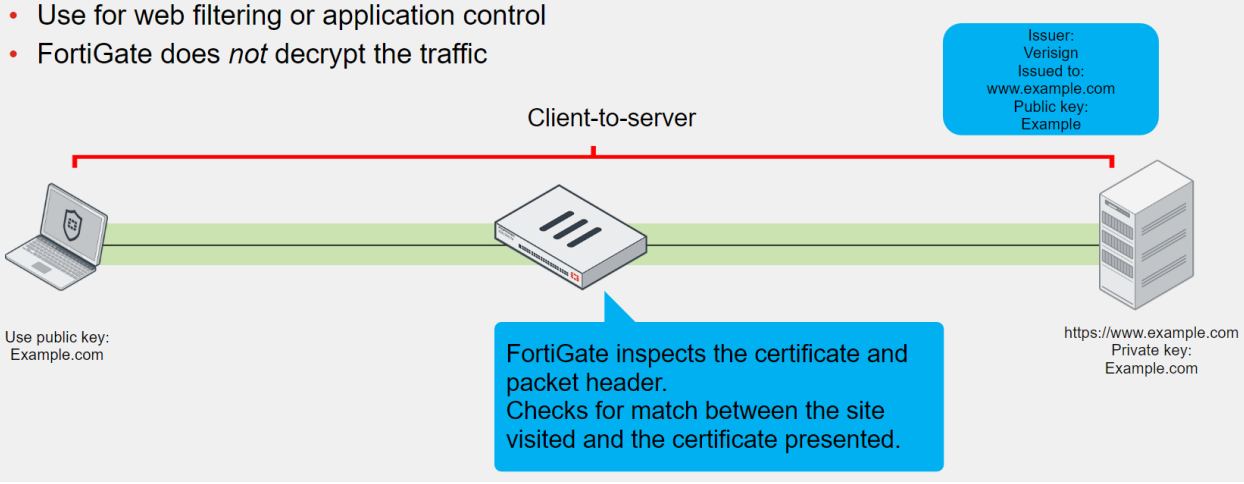


You can use **full SSL inspection**, also known as **deep inspection**, to inspect encrypted sessions.

SSL Inspection Modes

SSL Inspection Modes

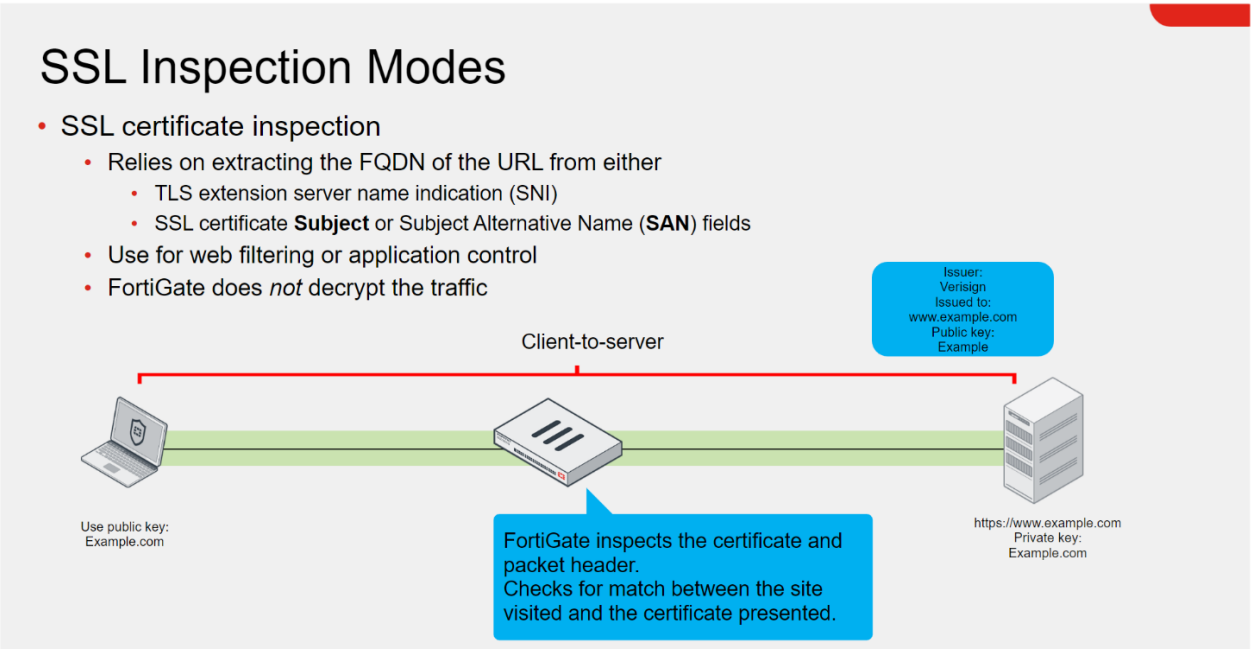
- SSL certificate inspection
  - Relies on extracting the FQDN of the URL from either
    - TLS extension server name indication (SNI)
    - SSL certificate **Subject** or Subject Alternative Name (**SAN**) fields
  - Use for web filtering or application control
  - FortiGate does *not* decrypt the traffic



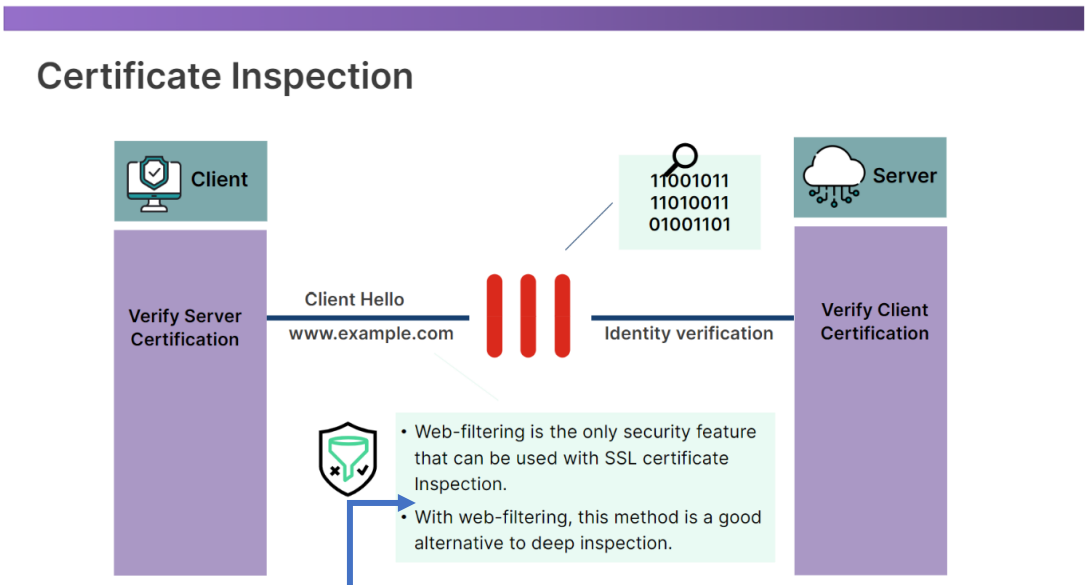
There are two SSL inspection modes:

- **SSL certificate inspection**
- **Full SSL inspection (deep inspection)**

SSL certificate inspection



When you use SSL certificate inspection, FortiGate inspects the SSL/TLS handshake when a session begins. By doing this, FortiGate verifies the identity of the web server and makes sure that the HTTPS protocol is not used as a workaround to access sites you have blocked using web filtering.



Web Filtering and Application Control

When using SSL certificate inspection, **FortiGate is not decrypting the traffic**. During the exchange of Hello messages at the beginning of an SSL handshake, FortiGate parses the Server Name Indication (SNI) from client Hello, which is an extension of the TLS protocol. The SNI tells FortiGate the hostname of the SSL server, which is validated against the DNS name before receipt of the server certificate. If there is no SNI exchanged, then FortiGate identifies the server by the value in the **Subject** field or **SAN** (Subject Alternative Name) field in the server certificate.

So:

- First, FortiGate tries to get the URL from the SNI field. The SNI field is a TLS extension that contains the complete URL that the user is connecting to. It is supported by most modern browsers.
- If the SNI field is not present (because the web client may not support it), FortiGate proceeds to inspect the server Digital Certificate to get information about the URL or the domain.

The only security features you can apply using SSL certificate inspection mode are **web filtering** and **application control**.

SSL certificate inspection allows FortiGate to identify **the website visited** or **the application in use** and categorize it. You can, therefore, use it to make sure that the HTTPS protocol isn't used as a workaround to access sites you have blocked using web filtering.

Note that while offering some level of security, certificate inspection does not allow FortiGate to inspect the flow of encrypted data.

This method does not introduce certificate errors and can be a useful alternative to deep SSL inspection when you use web filtering and application control.

Certificate inspection

FortiGate supports certificate inspection. The default configuration has a built-in *certificate-inspection* profile which you can use directly. When you use certificate inspection, the FortiGate only inspects the headers up to the SSL/TLS layer.

If you do not want to deep scan for privacy reasons but you want to control web site access, you can use *certificate-inspection*.

When a firewall policy is in flow-based inspection mode, ***SSL Certificate Inspection* does not validate the certificate**. *Untrusted SSL certificates* and *Server Certificate SNI* checks are not performed. If these features are needed, use **proxy-based inspection mode**.

Inspect non-standard HTTPS ports

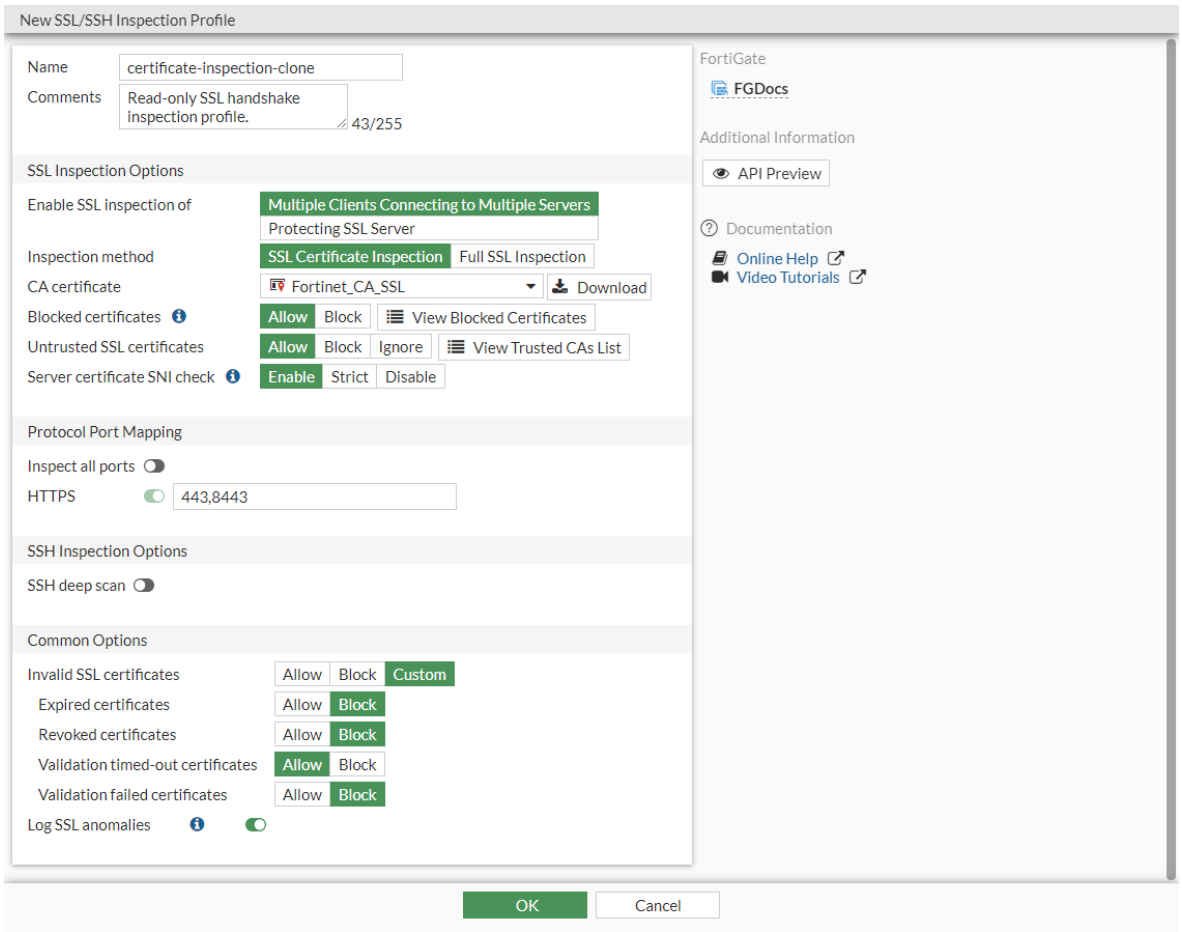
The built-in *certificate-inspection* profile is read-only and only listens on port **443**. If you want to make changes, you must create a new certificate inspection profile.

If you know the non-standard port that the web server uses, such as port **8443**, you can add this port to the *HTTPS* field.

To add a port to the inspection profile in the GUI:

1. Go to ***Security Profiles*** **>** ***SSL/SSH Inspection*****.**
2. Create a new profile, or clone the default profile.
3. If you do not know what port is used in the HTTPS web server, under ***Protocol Port Mapping*** enable ***Inspect All Ports***.

   If you know the port, such as port 8443, then set *HTTPS* to *443,8443*.

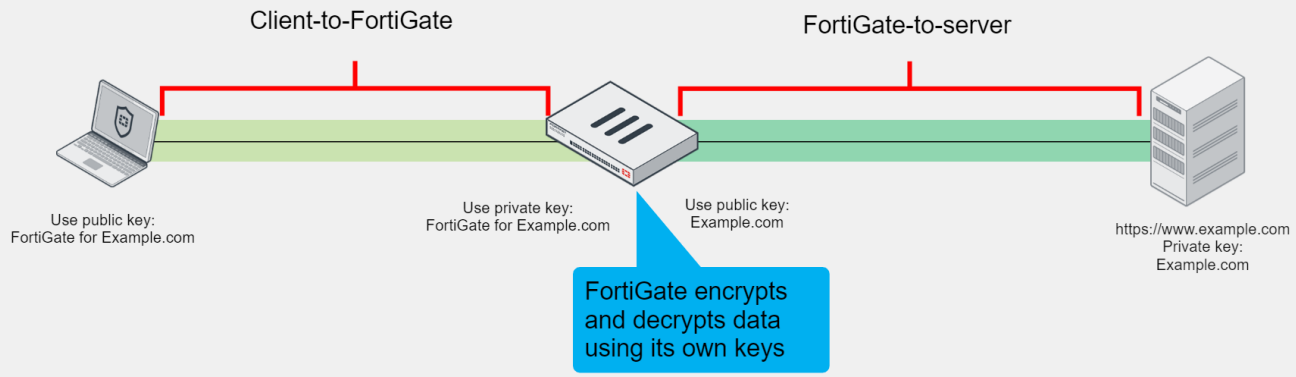


4. Configure the remaining setting as needed.
5. Click *OK*.

Full SSL inspection (Deep SSL Inspection)

SSL Inspection Modes (Contd)

- Full SSL Inspection
  - FortiGate acts as a man-in-the middle proxy
  - Maintains two separate SSL sessions—client-to-FortiGate, and FortiGate-to-server
  - FortiGate encrypts and decrypts packets using its own keys
  - FortiGate can inspect the traffic



You can configure full SSL inspection to inspect all of the packet contents, including the payload. FortiGate performs this inspection by proxying the SSL connection. It means FortiGate acts as a man-in-the middle proxy.

Two SSL sessions are established:

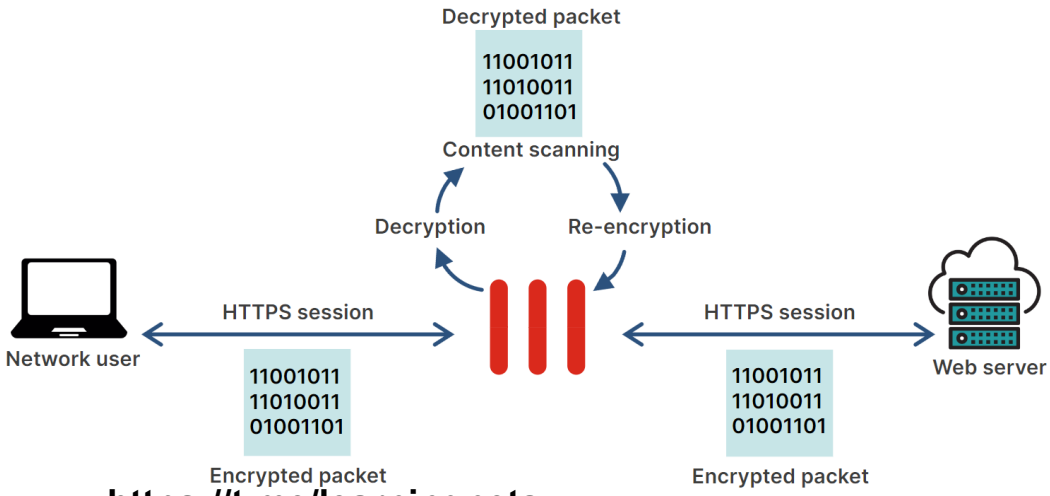
- Client-to-FortiGate
- FortiGate-to-Server

The two established sessions **allow FortiGate to encrypt and decrypt packets using its own keys**, which allows FortiGate to fully inspect all data inside the encrypted packets.

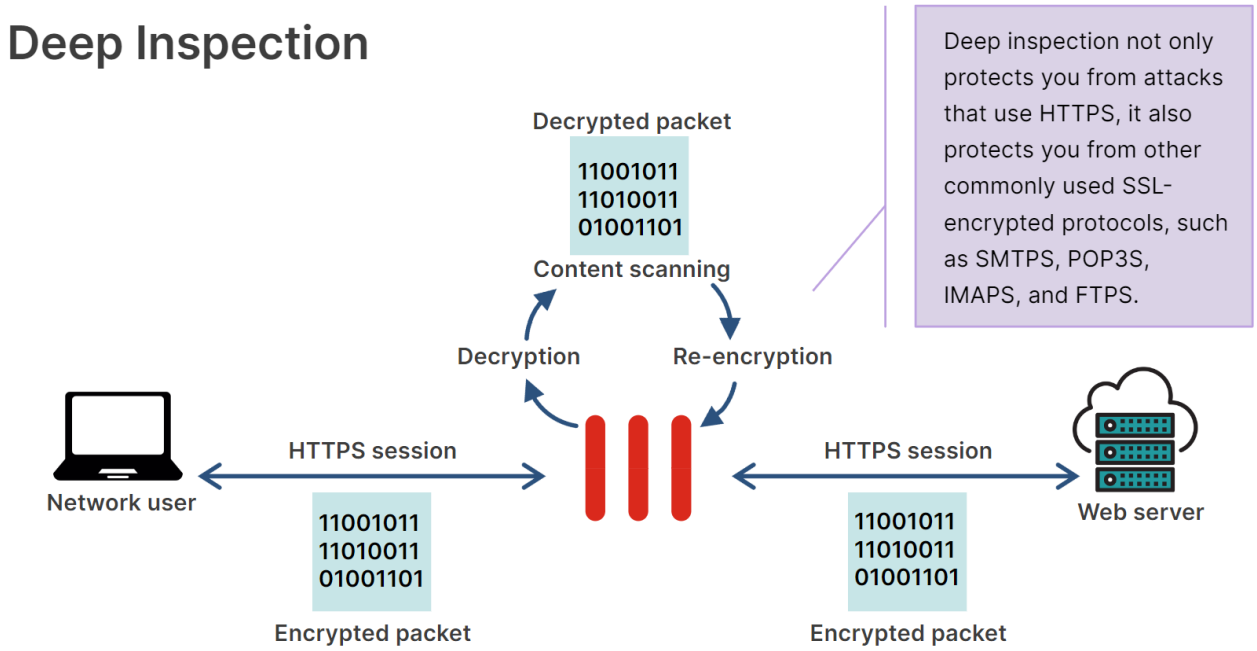
When you use deep inspection, FortiGate impersonates the recipient of the originating SSL session, and then decrypts and inspects the content to find threats and block them. It then reencrypts the content and sends it to the real recipient.

FortiGate impersonates the recipient of the originating SSL Session:

- Impersonates – Decrypts
- Inspects – Block Threats
- Re-encrypts and sends to real recipient



You can apply **all types of security scanning** with SSL deep inspection, including **web filtering**. Deep inspection not only protects you from attacks that use **HTTPS**, it also protects you from other commonly used SSL-encrypted protocols such as **SMTPS**, **POP3S**, **IMAPS**, and **FTPS**.



Protocol port mapping

To optimize the FortiGate's resources, the mapping and inspection of the following protocols can be enabled or disabled:

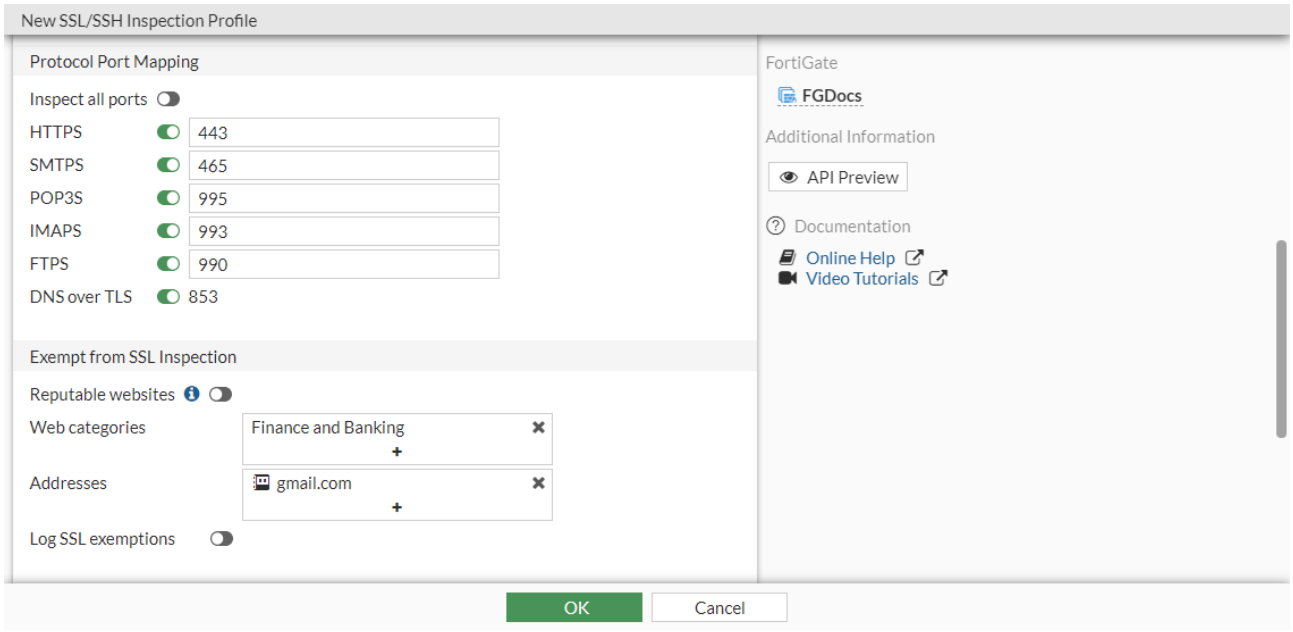
- HTTPS
- SMTPS
- POP3S
- IMAPS
- FTPS
- DNS over TLS

Each protocol has a default TCP port. The ports can be modified to inspect any port with flowing traffic. The packet headers indicate which protocol generated the packet.

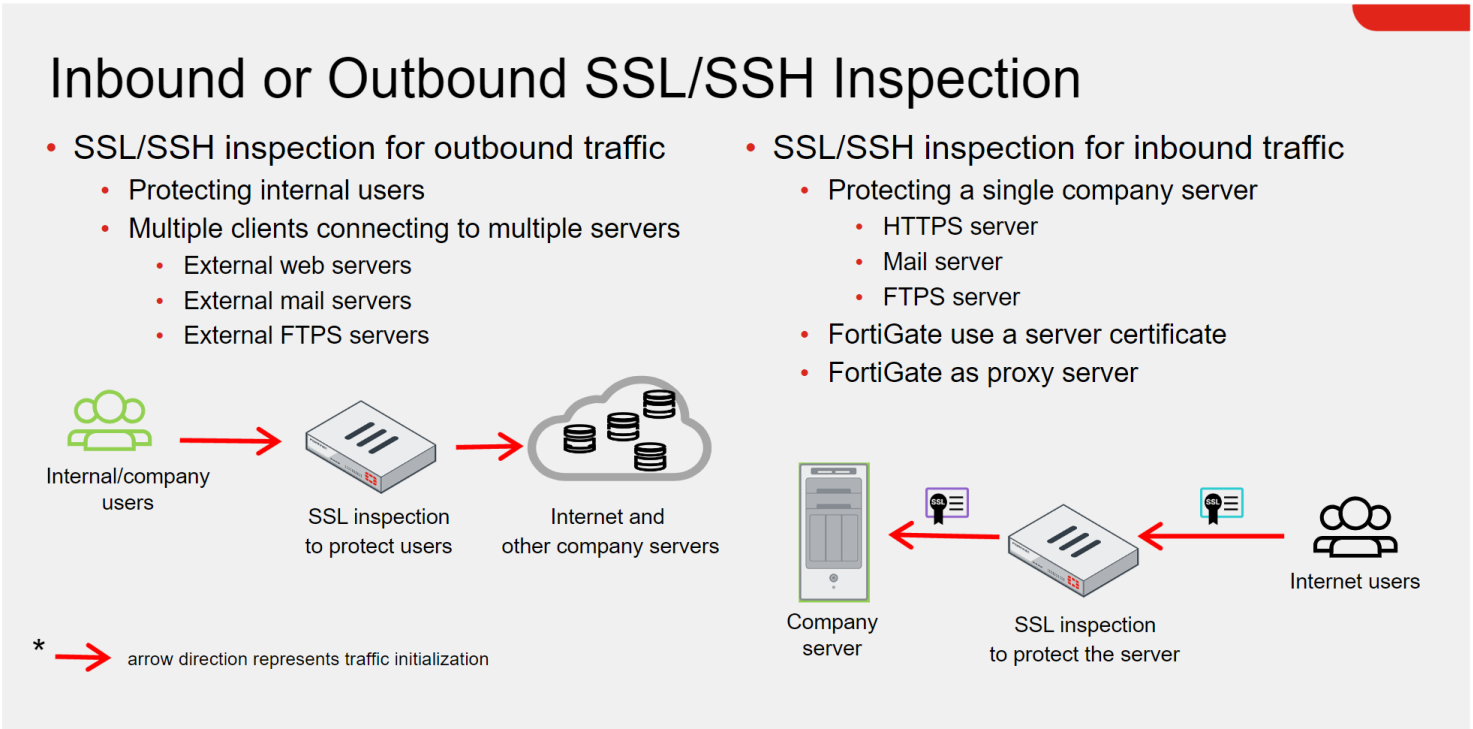
Protocol port mapping only works with **proxy-based inspection**. Flow-based inspection inspects all ports regardless of the protocol port mapping configuration.

Protocol Port Mapping

Inspect all ports with the **IPS engine** by enabling **Inspect all ports**. If Inspect all ports is disabled, specify the port through which traffic will be inspected in the field next to the listed protocols. Traffic of that protocol going through any other port will not be inspected.



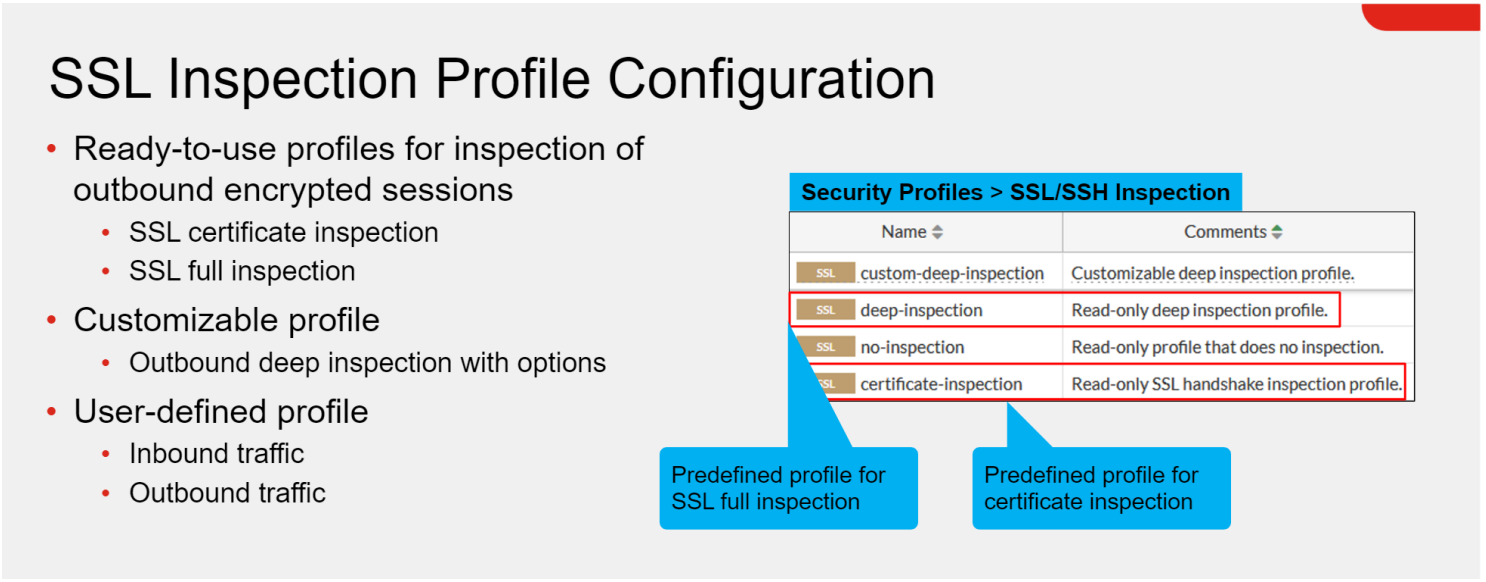
Inbound or Outbound SSL/SSH Inspection



FortiGate can proceed to SSL/SSH inspection for inbound traffic. Usually, this is the traffic initiated by local users bound for web servers on the internet. FortiGate protects users from traffic received from outside servers.

Conversely, you can use FortiGate to protect the company servers. Typically, you will protect the company web server from the outside world. For this purpose, FortiGate acts as a proxy server and presents the server certificate to internet users.

SSL Inspection Profile Configuration



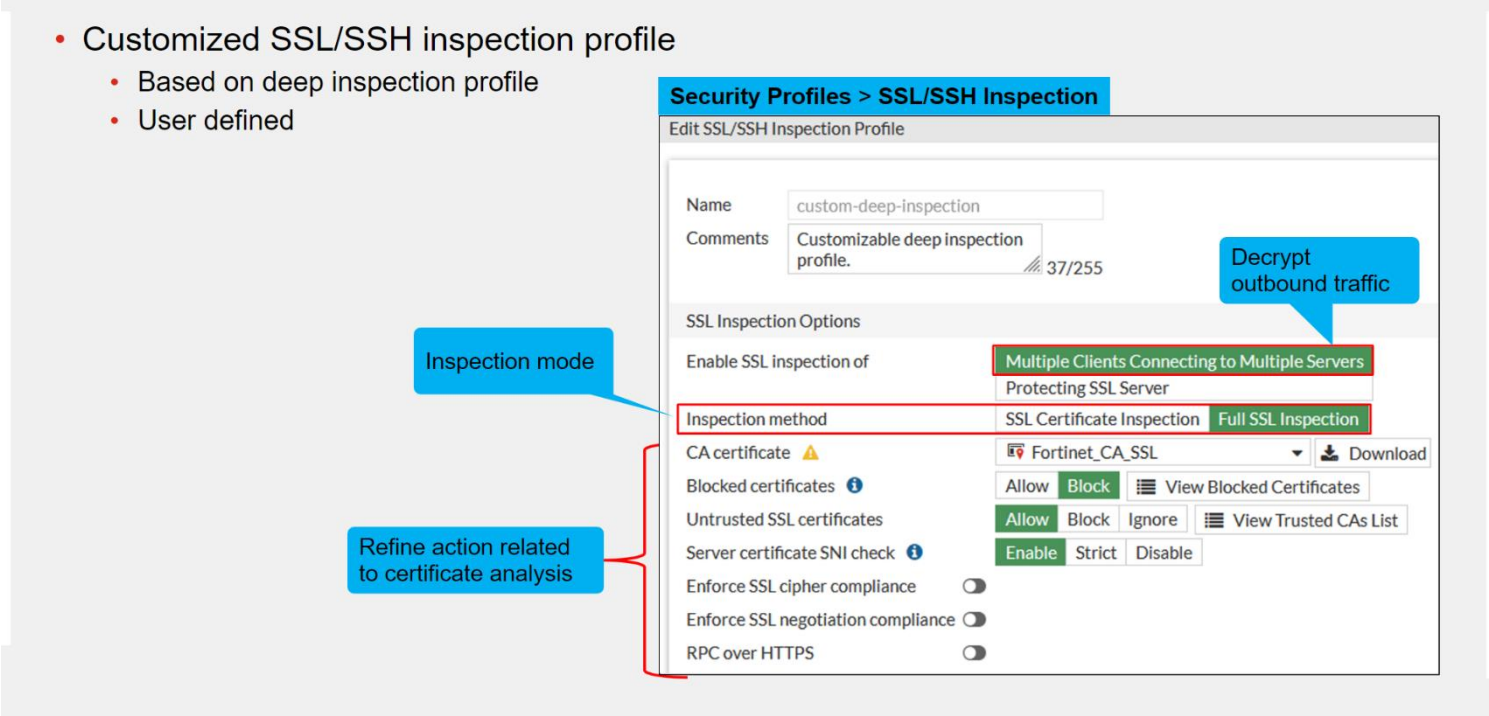
To use FortiGate SSL inspection, you can apply an SSL inspection profile to the **firewall policy**. FortiOS includes four pre-defined SSL inspection profiles, three of which are **read-only**:

- **certificate-inspection**
- **deep-inspection**
- **no-inspection**

You can edit the fourth preloaded profile, **custom-deep-inspection**. You can also **clone** any of the read-only profiles or **create** your own custom inspection profile.

If you define an inspection profile for **inbound traffic**, or use some specific options for an **outbound inspection profile**, you can adjust the custom-deep-inspection profile or create your own profile.

The profile applied by default when you create a new firewall policy is the self-explanatory **no-inspection** profile. Other predefined profiles available are **certificate-inspection**, and **deep-inspection**, which applies full SSL inspection to the outbound traffic.



The predefined **certificate-inspection** and **deep-inspection** profiles are read-only. If you want to adjust the profile parameters, you can use the predefined custom-deep-inspection profile, or create a new, user-defined profile. **(Security Profiles > SSL/SSH Inspection)**

When you define a custom SSL/SSH profile:

Name:

Enter a unique name for the profile.

Comment:

Enter a comment. (Optional)

Enable SSL Inspection of:

- You can enable SSL inspection for output traffic with the parameter **Multiple Clients Connecting to Multiple Servers**,
- or for inbound traffic with the parameter **Protecting SSL Server**.

Inspection Mode:

You can choose between **SSL Cerificate Inspection** or **Full SSL Inspection** based on your needs.

- ***SSL Certificate Inspection*:** Only inspects the certificate, by way of the headers up to the SSL/TLS layer, and not the contents of the traffic.
- ***Full SSL Inspection*:** Inspects the SSL/TLS encrypted traffic payload.

CA Certificate:

You can select the CA certificate **used for traffic re-encryption between FortiGate and the destination**. By default, FortiGate uses the preloaded `Fortinet_CA_SSL` certificate.

Blocked Certificate:

Block or **Allow** potentially malicious certificates. Select **View Blocked Certificates** for a detailed list of blocked certificates, including the listing reason and date.

Untrusted SSL certificates:

Configure the action to take when a server certificate is not issued by a trusted CA.

- ***Allow*:** Allow the untrusted server certificate. This is the default value.
- ***Block*:** Block the session.
- ***Ignore*:** This option is for ***Full SSL inspection*** only. It re-signs the server certificate as trusted. When configured in the GUI for certificate inspection it has no effect and the setting is not saved.

Click ***View Trusted CAs List*** to see a list of the factory bundled and user imported CAs that are trusted by the FortiGate.

Server certificate SNI check:

Check the SNI in the Hello message with the CN or SAN field in the returned server certificate:

- ***Enable*:** If it is mismatched, use the CN in the server certificate for URL filtering.
- ***Strict*:** If it is mismatched, close the connection.
- ***Disable*:** Server certificate SNI check is disabled.

I explain this option in detail later in this document.

Enforce SSL cipher compliance:

Enable/disable SSL cipher compliance. This option is for Full SSL inspection only.

Enforce SSL negotiation compliance:

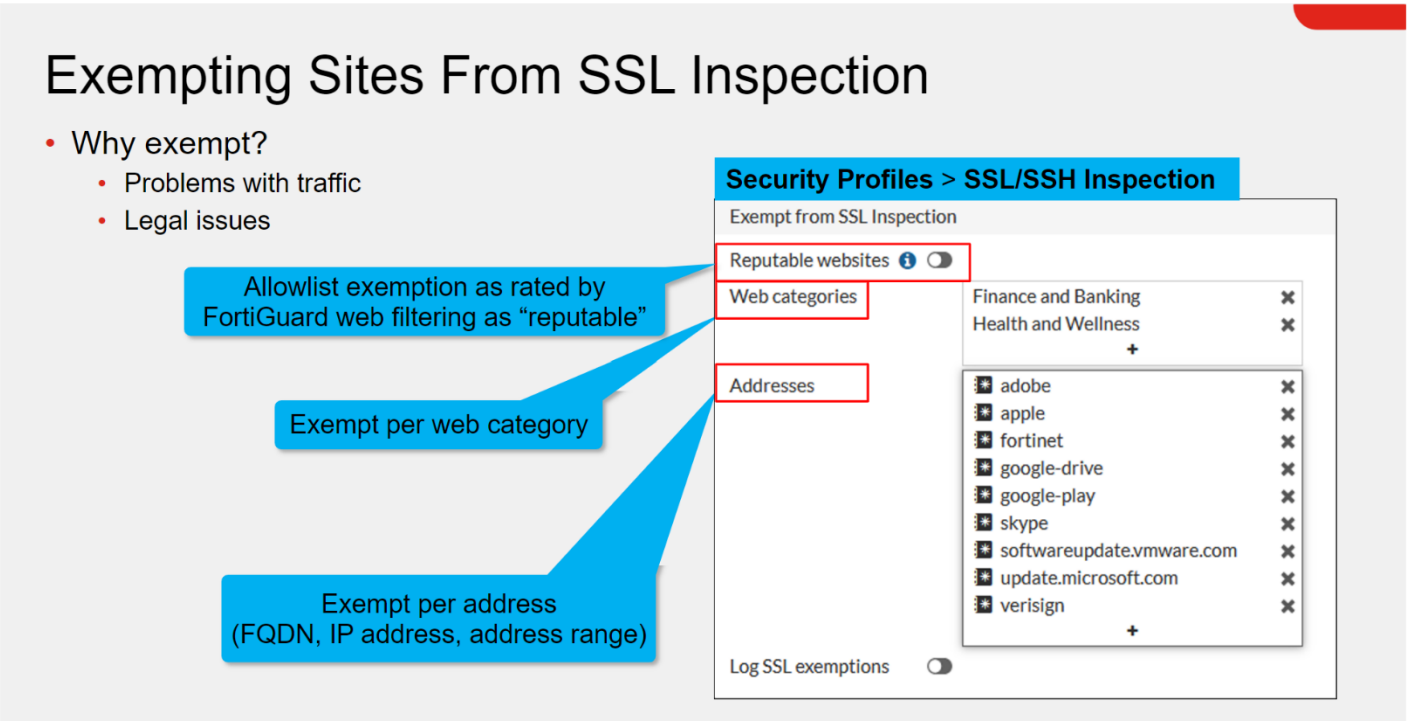
Enable/disable SSL negotiation compliance. This option is for Full SSL inspection only.

RPC over HTTPS:

Enable/disable inspection of Remote Procedure Calls (RPC) over HTTPS traffic. **This option is for Full SSL inspection only**.

Exempt from SSL Inspection:

Exempting Sites from SSL Inspection



Within the **Full SSL inspection** profile, you can also specify which SSL sites, if any, you want to exempt from SSL inspection. You may need to exempt traffic from SSL inspection if it is causing problems with traffic, or for legal reasons.

Problem with Traffic:

Performing SSL inspection on a site that is enabled with HSTS, for example, can cause problems with traffic. Remember, the only way for FortiGate to inspect encrypted traffic is to intercept the certificate coming from the server and generate a temporary one. After FortiGate presents the temporary SSL certificate, browsers that use HSTS refuse to proceed.

("HTTP Strict Transport Security" (HSTS - RFC 6797) is an HTTP header that a web server can use to inform clients (such as web browsers) that the particular website can only be accessed using HTTPS (with SSL) rather than in clear text.)

Legal Issues:

Laws protecting privacy might be another reason to bypass SSL inspection. For example, in some countries, it is illegal to inspect SSL bank-related traffic. Configuring an exemption for sites is simpler than setting up firewall policies for each individual bank.

You can exempt sites based on:

- Their **web category**, such as Finance and Banking,
- or you can exempt them based on **their address**.
- Alternatively, you can enable **Reputable websites**, which excludes an allowlist of reputable domain names maintained by FortiGuard from full SSL inspection. This list is periodically updated and downloaded to FortiGate devices through FortiGuard.

The predefined deep-inspection and custom-deep-inspection profiles exclude some web categories—Finance and Banking, and Heath and Wellness—and some FQDN addresses such as google-play, skype, or verisign. When using the custom-deep-inspection profile, you can add or remove sites from this list.

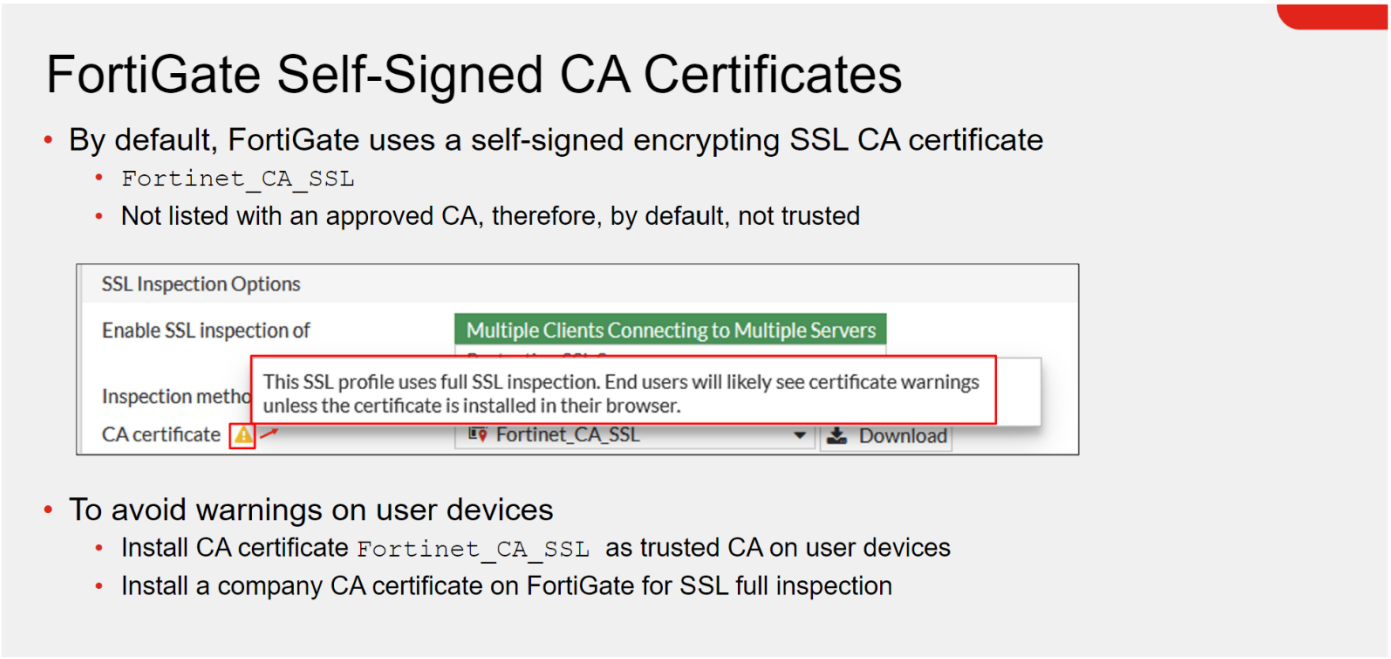
So:

If you know the address of the server you want to exempt, you can exempt that address. You can exempt specific address type including IP address, IP address range, IP subnet, FQDN, wildcard-FQDN, and geography.

If you want to exempt all bank web sites, an easy way is to exempt the ***Finance and Banking*** category, which includes all finance and bank web sites identified in FortiGuard.

f you want to exempt commonly trusted web sites, you can bypass the SSL allowlist in the SSL/SSH profile by enabling Reputable websites. The allowlist includes common web sites trusted by **FortiGuard**.

FortiGate Self-Signed CA Certificate



You can select the CA certificate **used for traffic re-encryption between FortiGate and the destination**. By default, FortiGate uses the preloaded **Fortinet_CA_SSL** certificate.

By default, FortiGate uses a self-signed CA certificate for the re-encryption required by the SSL full inspection. Because the corresponding CA is not prepopulated in client device certificate stores, users will likely see certificate warnings for traffic flows protected by the full SSL inspection.

To avoid the warning on User Devices:

- You can install the **Fortinet_CA_SSL** certificate as trusted CA on the user devices. You can install it as part of the deployment process for all your company computers.

To import *Fortinet_CA_SSL* into your browser:

1. On the FortiGate, go to *Security Profiles* > *SSL/SSH Inspection* and edit the *deep-inspection* profile.

   The default *CA Certificate* is *Fortinet_CA_SSL*.
2. Click *Download* and save the certificate to the management computer.
3. On the client PC, use the *Certificate Import Wizard* to install the certificate into the *Trusted Root Certificate Authorities* store.

   If a security warning appears, select *Yes* to install the certificate.

- Alternatively, you can install on FortiGate a CA certificate, used for traffic re-encryption, that is signed by your company CA. This certificate will already be recognized as valid by your company devices.

The certificate used to re-encrypt the traffic after the SSL full inspection must follow some specific requirements.

Full SSL Inspection – Certificate Requirements

Full SSL Inspection—Certificate Requirements

- Full SSL inspection requires that FortiGate acts as a CA to generate an SSL private key and certificate
  - The CA certificate requires these two extensions to issue certificates:
    - cA=True
    - keyUsage=keyCertSign
- FortiGate can use:
  - Preloaded, self-signed `Fortinet_CA_SSL` certificate
  - A certificate issued by the company CA
- The root CA certificate must be imported into the client machines

To perform Full SSL inspection, **FortiGate performs as a web proxy**, and must act as a **CA** in order to re-encrypt the traffic. The FortiGate internal CA must generate an **SSL private key** and **certificate** each time it needs to re-encrypt a new traffic flow. The key pair and certificate are generated immediately, so the user connection with the web server is not delayed.

Although, from the user point of view, it appears as though the user browser is connected to the web server, the browser is in fact connected to FortiGate. To perform this proxy role, and generate a certificate that correspond to the server visited, the CA certificate must allow the generation of new certificates. To achieve this, it must have the following extensions:

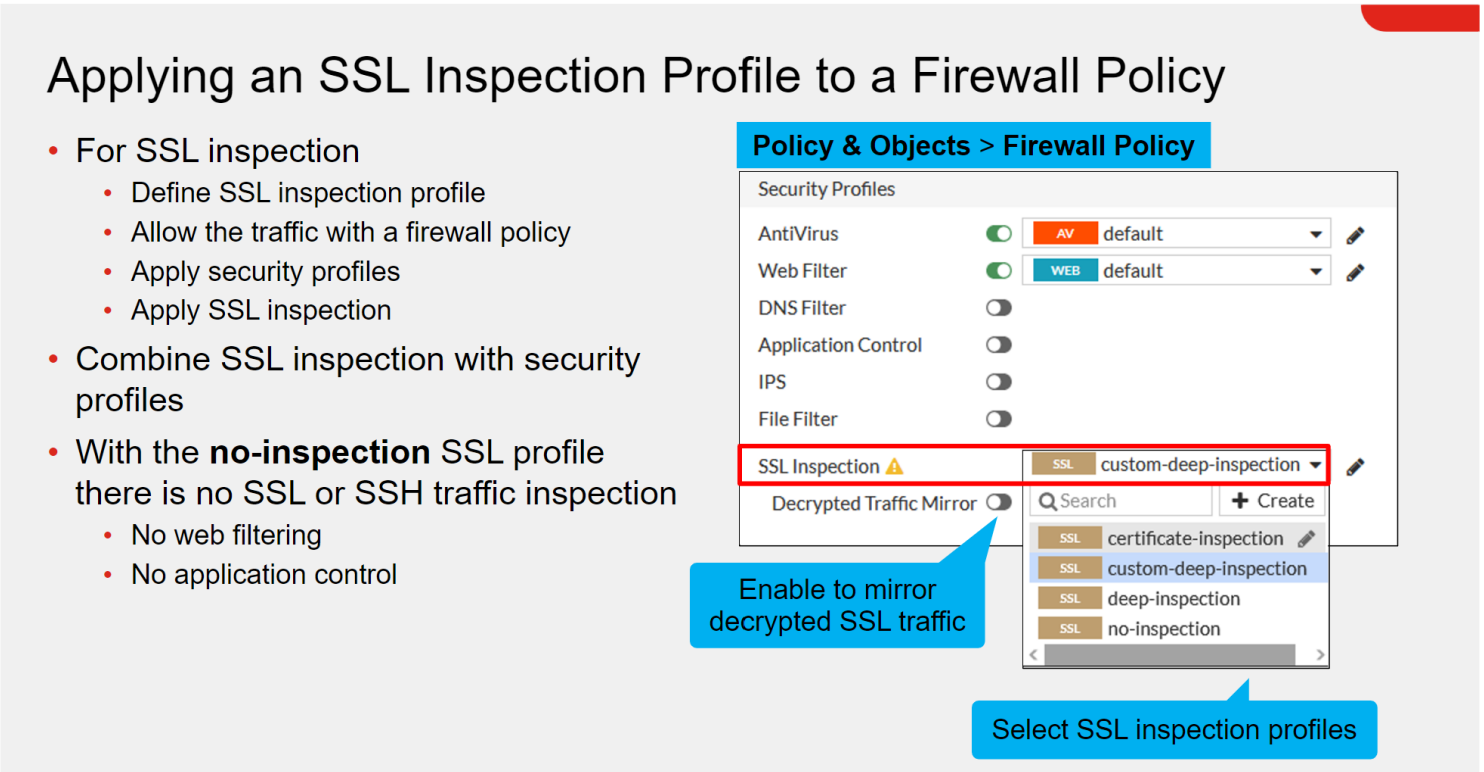
- **cA** set to **True**, ➔ The cA=True value identifies the certificate as a CA certificate.
- **keyUsage** extension set to **keyCertSign**. ➔ (**Key Usage** is used to define what a certificate is allowed to do. Usage bit **5** is KeyCertSign. **this allows a public key to verify signature of a certificate**). The keyUsage=keyCertSign value indicates that the certificate corresponding to the private key is permitted to sign certificates.

All FortiGate devices come with the self-signed **Fortinet_CA_SSL** certificate that you can use for Full SSL inspection. If your company has an internal CA, you can request the CA administrator to issue a certificate for your FortiGate device. The FortiGate device then acts as a subordinate CA.

Important:

If you use the **Fortinet_CA_SSL** certificate, or a certificate issued by your company CA, to trust FortiGate and accept re-encrypted SSL sessions without warning, you must import the root CA certificate used to your client devices.

Applying an SSL Inspection profile to a Firewall policy



To perform SSL inspection on traffic flowing through the FortiGate device:

1. **Define SSL Inspection Profile**
2. **Define Firewall Policy and Allow the traffic with this Firewall Policy**
3. **Apply Security Profiles within the Firewall policy**
4. **Apply corresponding SSL Insepction**

You must allow the traffic with a firewall policy and apply an SSL inspection profile to the policy.

Note that:

An SSL inspection profile alone will not trigger a security inspection. You must combine it with other security profiles like Antivirus, Web Filter, Application Control, or IPS.

By default, firewall policies are set with the **no-inspection** SSL profile. Therefore, any encrypted traffic flows through uninspected. For instance, with the **no-inspection** profile, FortiGate cannot perform any web filtering for HTTPS traffic.

- **To allow web filtering,**
- **To allow DNS filtering,**
- **To allow application control for HTTPS traffic,**

→ **You must select an SSL inspection profile with certificate inspection or a deep inspection enabled.**

- **For antivirus,**
- **For IPS control**

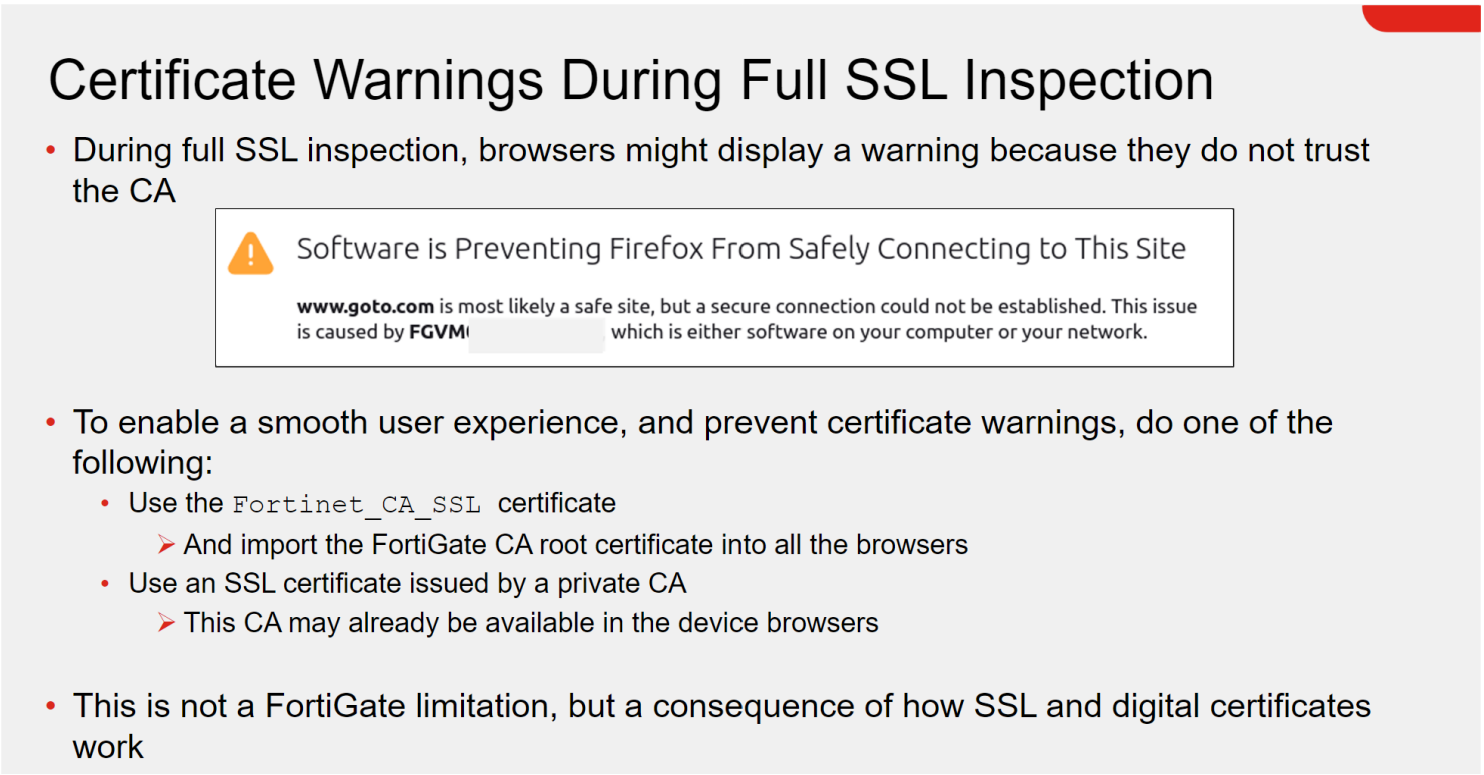
→ **you should use a deep-inspection profile.**

You can see a warning sign near the SSL inspection profile selection menu on the GUI. You will see this warning each time you select an SSL inspection profile with **deep inspection**. It is there to warn about the certificate warning that can appear on the user browser when traffic is allowed with this policy. If you hover over the warning sign you can read this message: “This SSL profile uses full SSL inspection. End users will likely see certificate warnings unless the certificate is installed in their browser.”

If you select a **profile** with **full SSL inspection** enabled, the option **Decrypted Traffic Mirror** appears. Enable this option if you want FortiGate to send a copy of the decrypted SSL traffic to an interface. It works only with **flow-based inspection**. When you enable **Decrypted Traffic Mirror**, FortiGate displays a window with the terms of use for this feature. The users must agree to the terms before they can use the feature.

You will apply an SSL profile to a firewall policy the same way for inbound or outbound traffic flow inspection. It is the SSL profile applied that specifies the certificate in use when the FortiGate device re-encrypts the traffic.

Certificate Warnings during Full SSL Inspection



When doing **full SSL inspection** using the FortiGate self-signed CA, your browser might display a certificate warning each time you connect to an HTTPS site. This is because the browser is receiving certificates signed by FortiGate, which is a CA it does not know and trust. This is not a limitation of FortiGate, but a consequence of how digital certificates are designed to work.

There are two ways to avoid those warnings:

- The first option is to download the default FortiGate certificate for SSL proxy inspection and install it on all the workstations as a trusted root authority.

To import *Fortinet_CA_SSL* (default FortiGate certificate) into your browser:

1. On the FortiGate, go to *Security Profiles* > *SSL/SSH Inspection* and edit the *deep-inspection* profile.

   The default *CA Certificate* is *Fortinet_CA_SSL*.
2. Click *Download* and save the certificate to the management computer.
3. On the client PC, use the *Certificate Import Wizard* to install the certificate into the *Trusted Root Certificate Authorities* store.

   If a security warning appears, select *Yes* to install the certificate.

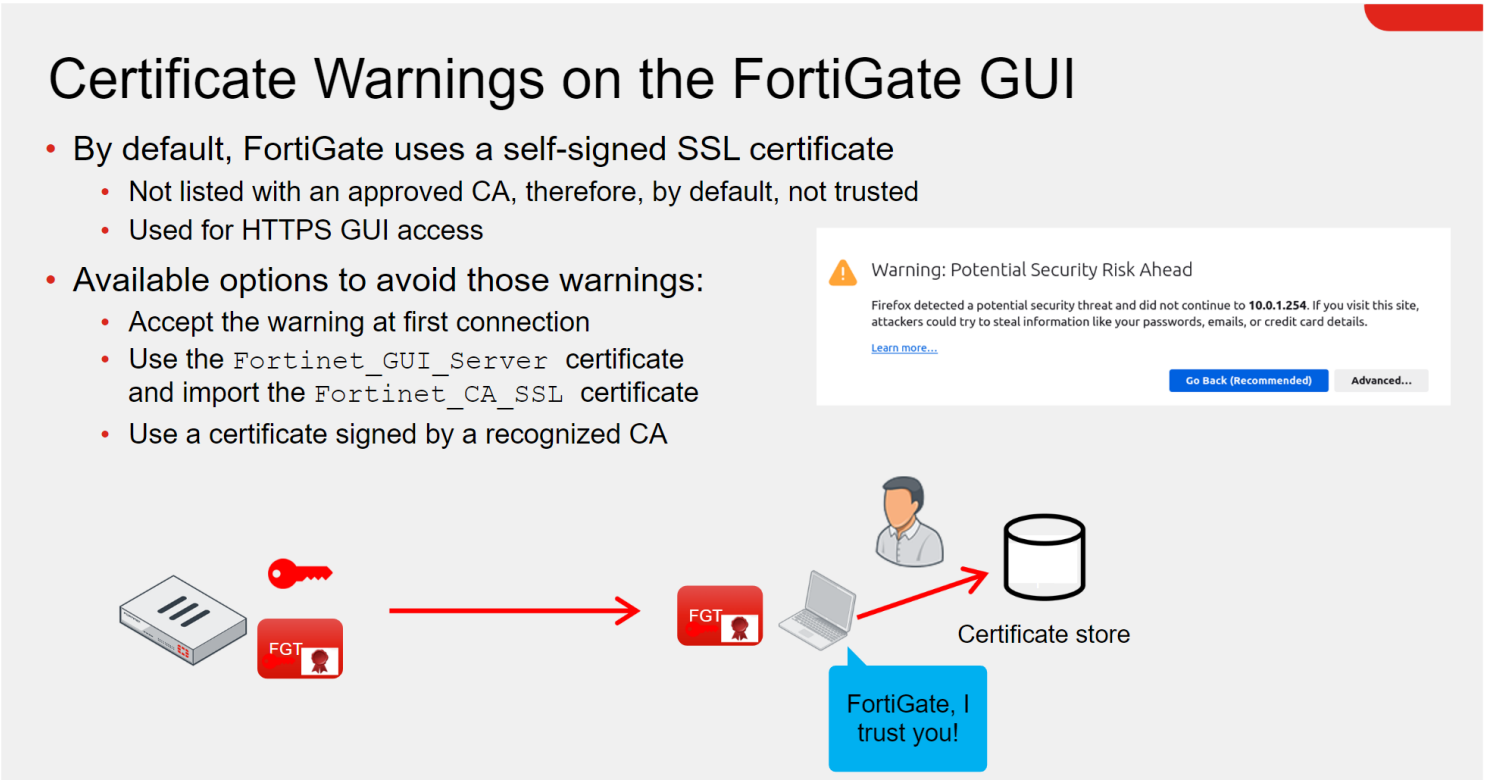
- The second option is to generate a new SSL proxy certificate from a private CA. In this case, the private CA certificate must still be imported into all the browsers.

If you use an SSL certificate signed by a subordinate CA:

- You must ensure that the entire chain of certificates—from the SSL certificate to the root CA certificate—is installed on FortiGate.
- Verify also that the **root CA** is installed on all client browsers.

This is required for trust purposes. Because FortiGate sends the chain of certificates to the browser during the SSL handshake, you do not have to import the intermediate CA certificates into the browsers.

Certificate Warnings on the FortiGate GUI

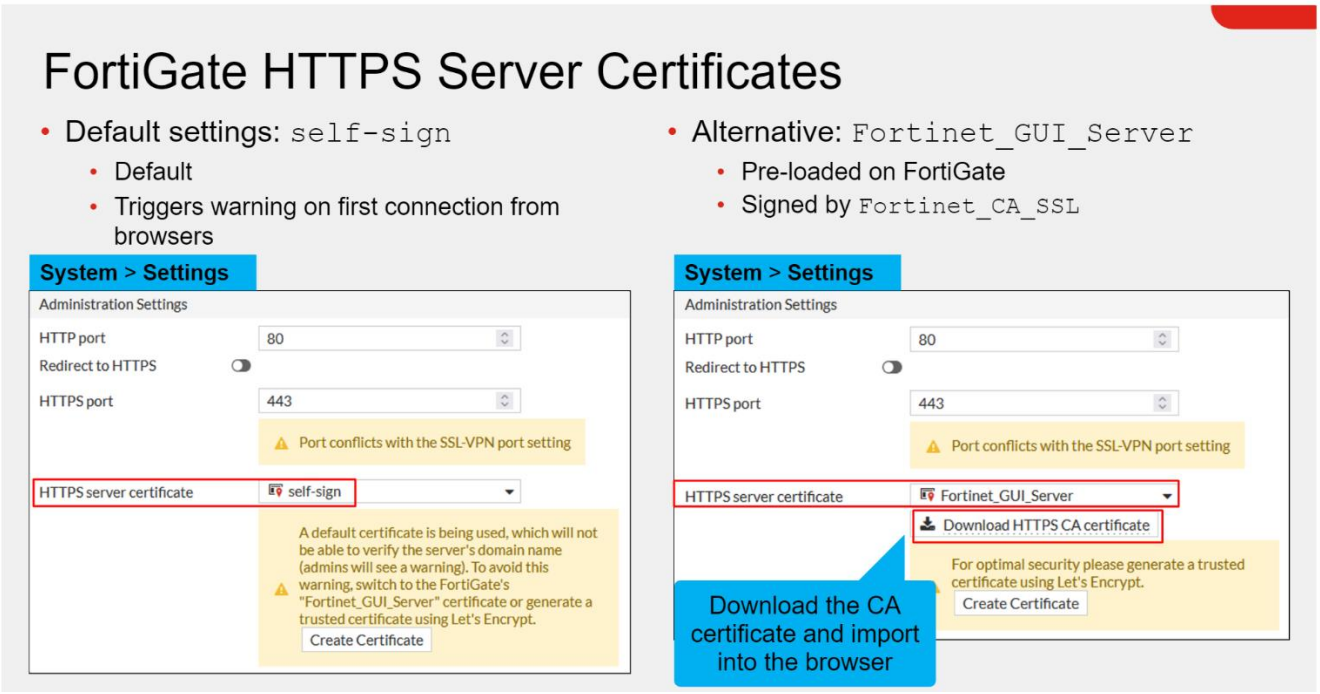


By default, FortiGate uses a self-signed certificate to authenticate itself to HTTPS clients. Because the corresponding CA certificate is not prepopulated in the certificate stores of client devices, the first HTTPS connection to a FortiGate device triggers a security warning.

Available options to avoid those warnings:

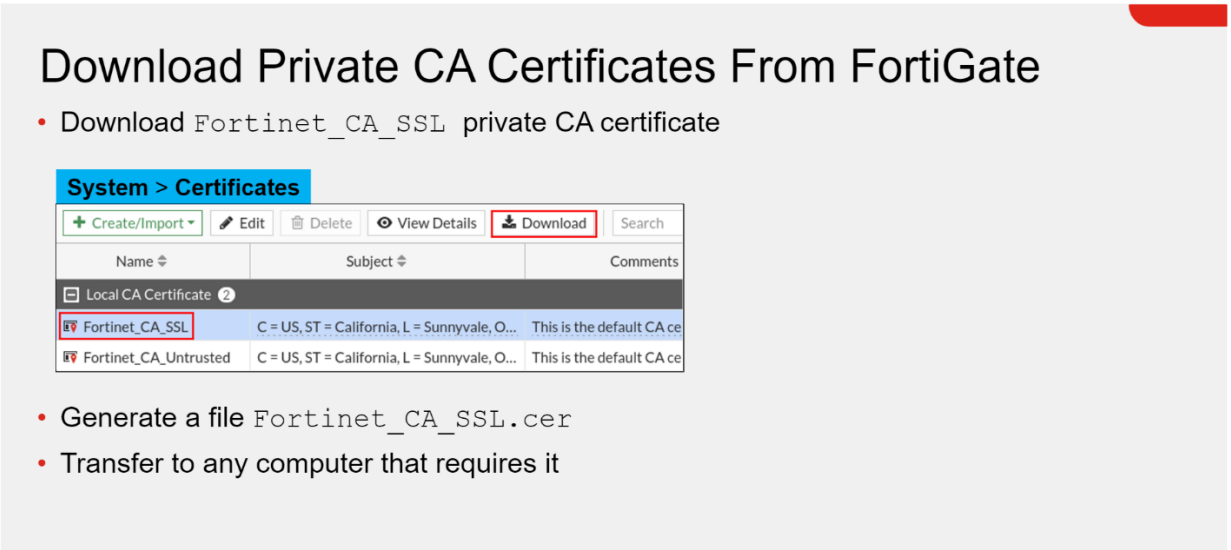
- If you trust the FortiGate device and want to keep the self-signed certificate to establish SSL sessions, you can accept the warning and establish the connection. When you accept the warning, your browser imports the FortiGate selfsigned certificate into its certificate store. So, the next time you connect to this FortiGate device, your browser already trusts the certificate presented.
- Alternatively, you can configure FortiGate to use the **Fortinet_GUI_Server** certificate and add the FortiGate self-signed CA certificate— **Fortinet_CA_SSL**—to the local certificate store of any computer that needs to connect to the FortiGate device. For subsequent connections to the FortiGate GUI interface, those devices trust the certificate and allow connections without warning.
- Another option for companies who manage their own CA is to generate a certificate for each of your FortiGate devices and use them to secure HTTPS connections.

FortiGate HTTPS Server Certificates



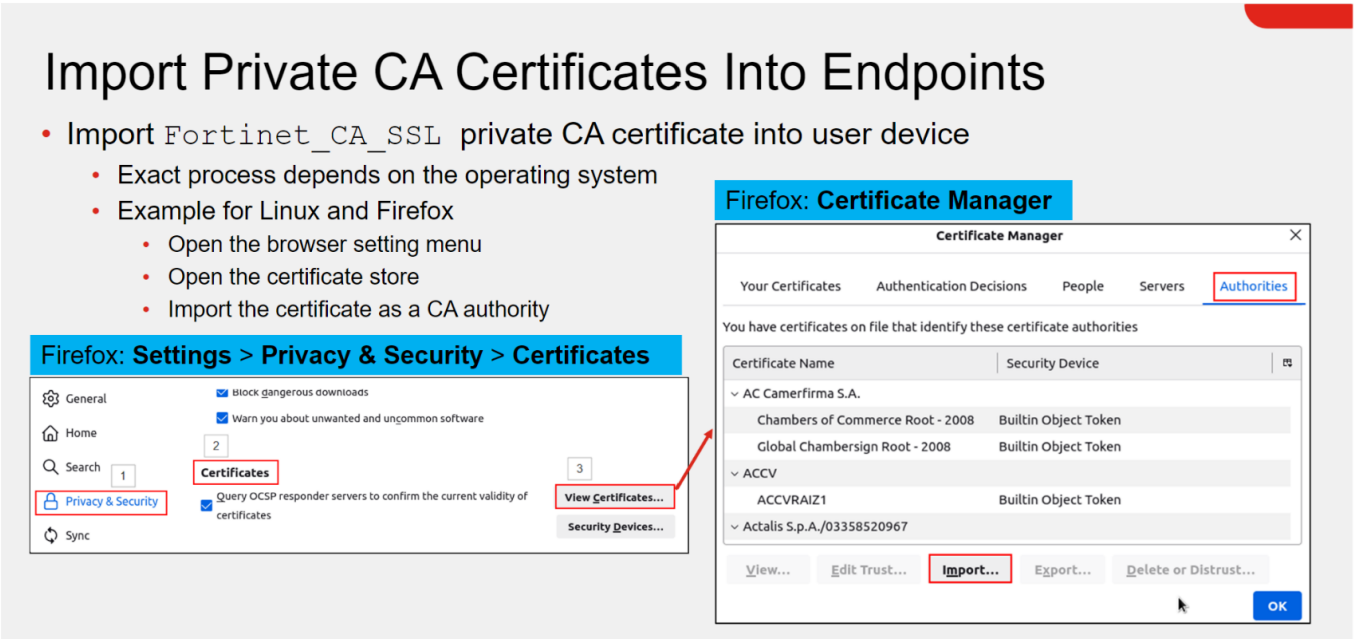
You can select the certificate that FortiGate presents for HTTPS GUI access from the **Settings** menu. By default, FortiGate uses the **self-sign** certificate, which is not recognized as a trusted certificate by the browsers. Alternatively, you can select the **Fortinet_GUI_Server** certificate, which is signed by the **Fortinet_CA_SSL**. With this certificate, to avoid the browser warning on HTTPS access to the FortiGate GUI, you must import the **Fortinet_CA_SSL** certificate into your management devices.

Download Private CA Certificate from FortiGate



Before you can import the default CA certificate— **`Fortinet_CA_SSL`**— into the user devices, you must download it from FortiGate. You can get it from the **FortiGate certificate store** available under the **System > Certificates** path. Upon download, FortiGate generates a **.cer** file that you can import into any device as required.

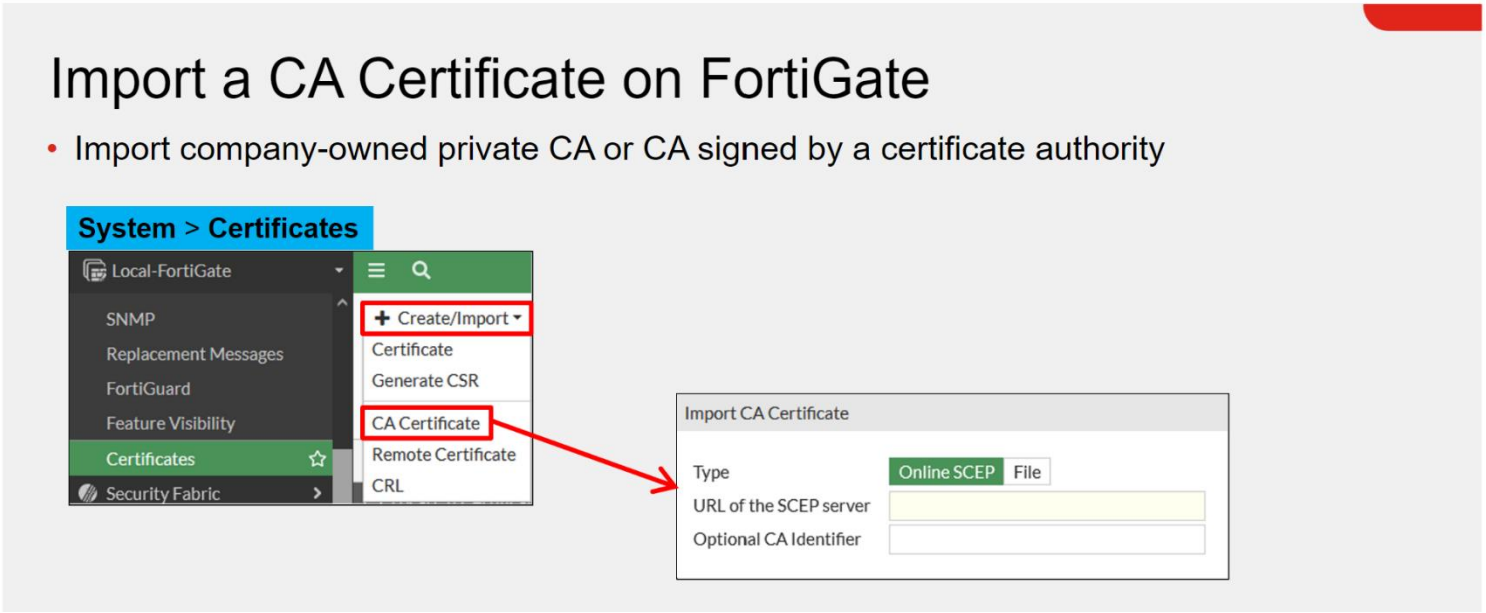
Import Private CA Certificate into Endpoints



After you download the CA certificate from FortiGate, you can import it into any web browser or operating system. Not all browsers use the same certificate repository. For example, Firefox uses its own repository, while Internet Explorer and Chrome store certificates in a system-wide repository. In order to prevent certificate warnings, **you must import the SSL certificate as a trusted root CA**.

When you import the certificate, make sure that you save it to the certificate store for root authorities. The example on this slide shows the menu you use to import a certificate into the Firefox browser.

Import a CA Certificate on FortiGate



If your company has a **private signing CA** or a **signing CA signed by a certificate authority**, you can import the corresponding certificate onto the FortiGate device as shown on this slide. Note that you can import the certificate by connecting to the SCEP server or as a file. SCEP stands for Simple Certificate Enrollment Protocol, and it is a popular and widely available certificate enrollment protocol.

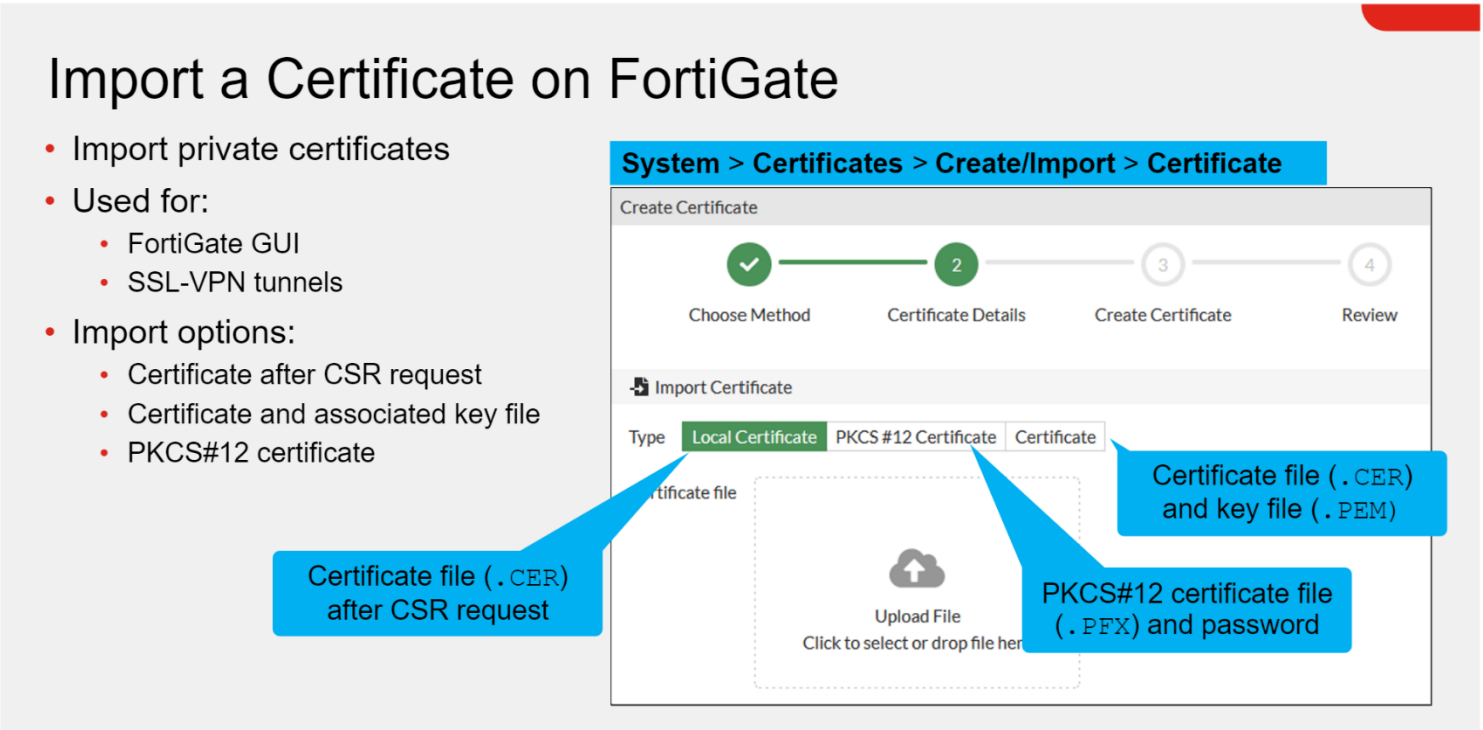
SCEP = Simple Certificate Enrollment Protocol is an open-source protocol that is widely used to make digital certificate issuance at large organizations easier, more secure, and scalable.

Note That:

The certificates feature is hidden by default in FortiOS. In the GUI, go to ***System*** **>** ***Feature Visibility*** and enable ***Certificates***.

Import a Private Certificate on FortiGate

(If your company manages its own certificate authority)



If your company manages its own certificate authority, you can generate certificates for:

- **FortiGate GUI**
- **SSL Access**
- You can also generate certificates that you will use for **SSL-VPN tunnels.**
- **IPsec VPN**

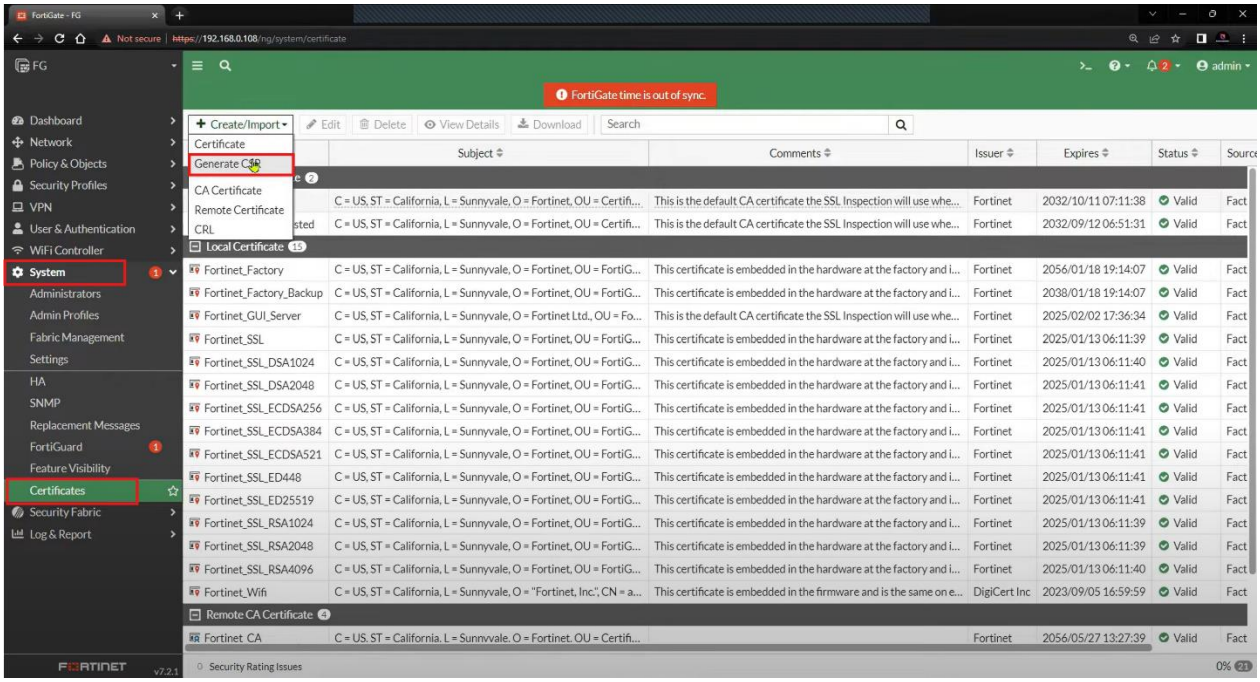
System > Certificates > Creat/Import > Certificate

FortiGate offers three options to import private certificates:

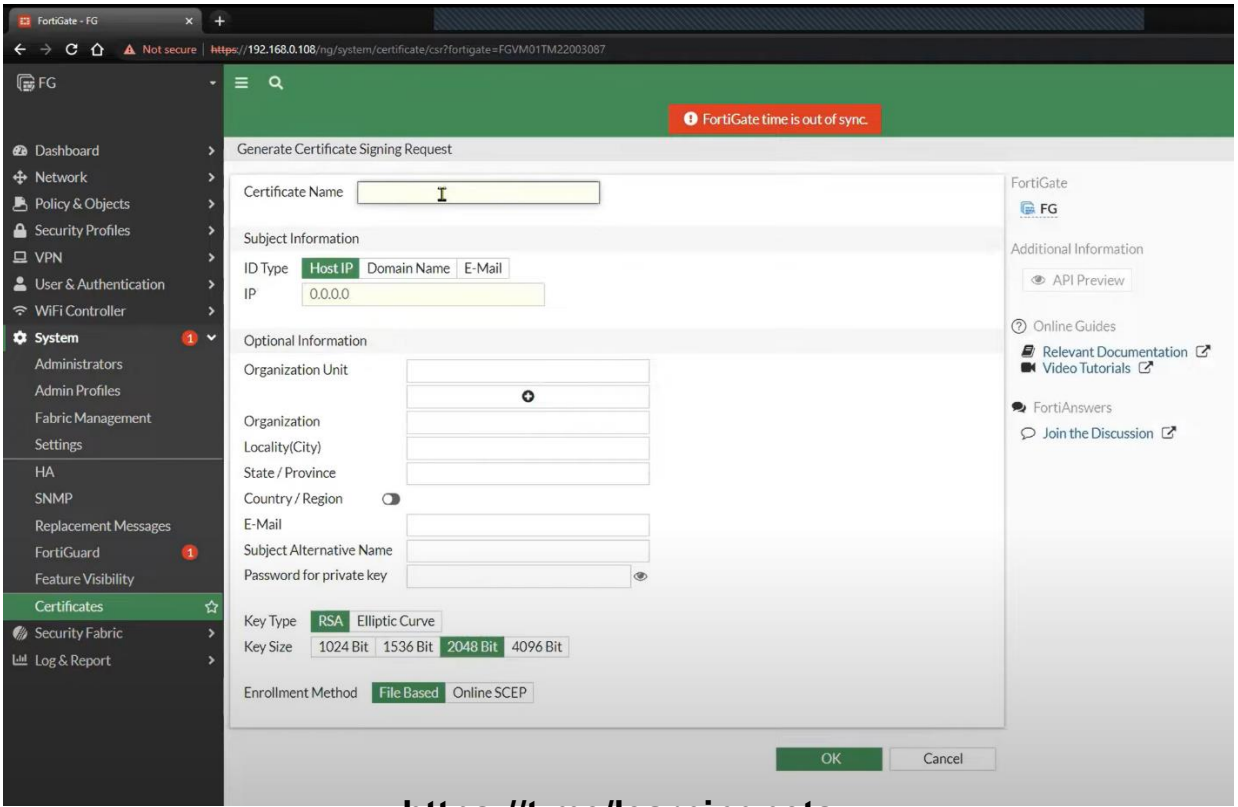
1. You can first generate a Certificate Signing Request (CSR) and submit it to the CA for certificate generation. With this process, the key file is automatically generated and stored on FortiGate when it generates the CSR. Later, you import only the certificate file (.CER) provided by the CA.
2. Another option is to import the certificate file and the associated key into the FortiGate certificate store.
3. Alternatively, you can load a PKCS#12 certificate file, which is identified as a .PFX file. It contains the certificate and associated private key.

Generating a CSR in the FortiGate

1. In the administrative web portal select "**System**" and then "**Certificates**." If "**Certificates**" is not displayed, you may have to enable the option within "**Feature Visibility**".
2. Click "**Generate**" and the "**Generate Certificate Signing Request**" page will open.



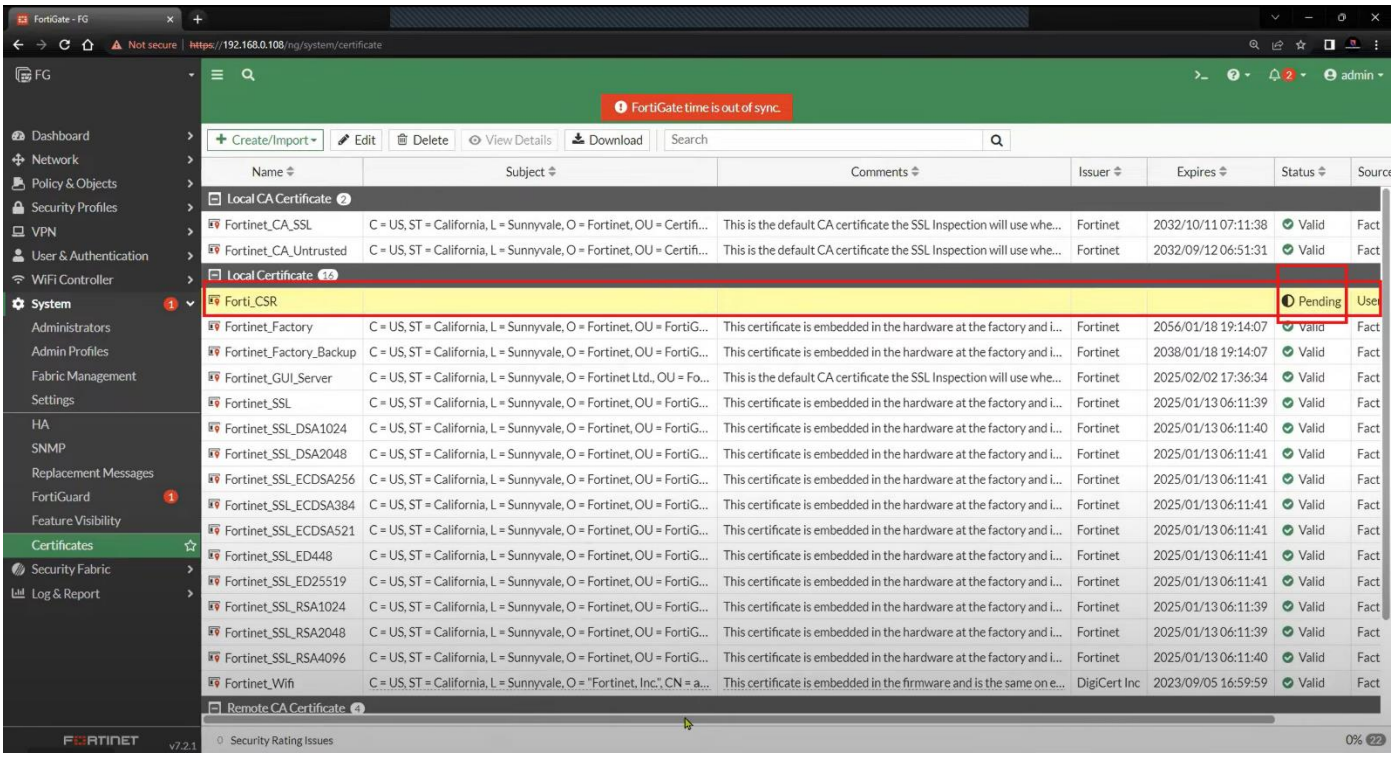
3. Configure the CSR request by filling in the following fields:



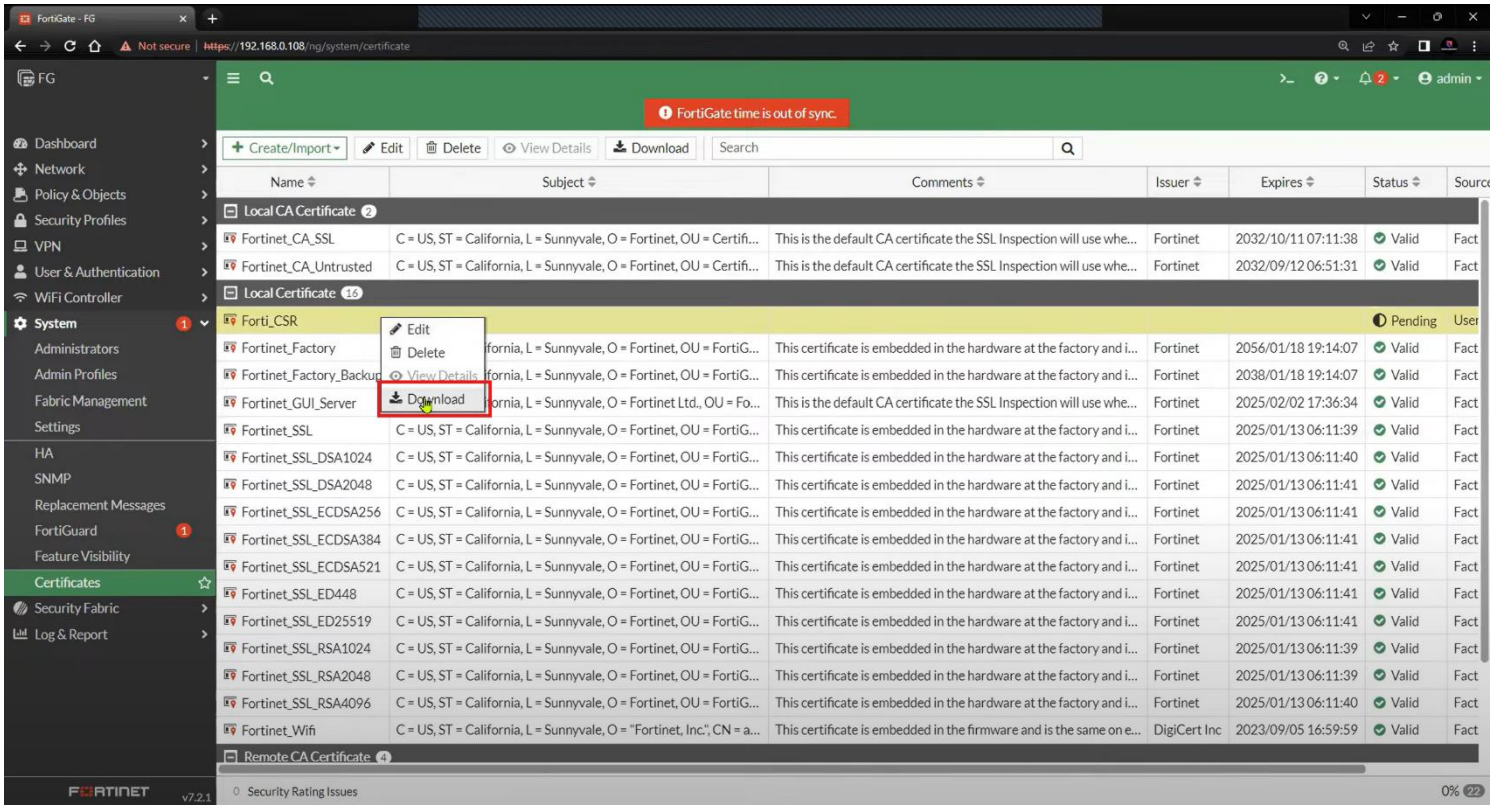
1. **Certificate Name** – The friendly name for the certificate used in the appliance.
2. **ID Type** – The type of ID, "**Domain Name**" is selected most often.
3. **Domain Name** – The common name used for the certificate.
4. **Organization** – The name of organization ordering the certificate.
5. **Locality (City)** – The name of the city the organization is located in.
6. **State / Province** – The name of the state or province the organization is located in.
7. **Country / Region** – The name of the country or region the organization is located in.
8. **E-Mail** – The organization's technical contact email address.
9. **Subject Alternative Name** – The additional names or SANs that will be used for the certificate.
10. **Password for private key** – The password used to access the encrypted private key associated with the CSR.
11. **Key Type** – The type of key used for the certificate; "**RSA**" is selected most often.
12. **Key Size** – The size of key used for the certificate; the minimum key size recommend is "**2048 Bit**."
13. **Enrollment Method** – The type of enrollment method used, "**File Based**" is recommended.

4. Click "**OK**."

5. The Certificate Name you entered for the CSR will appear in the certificate list with a status of "**PENDING**."

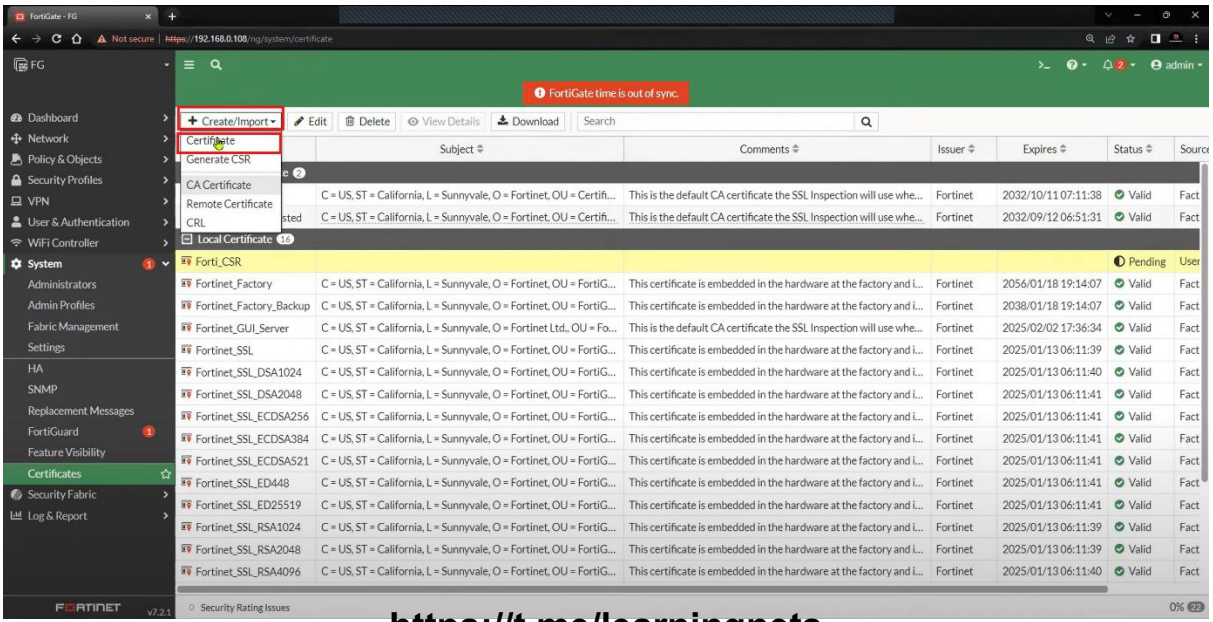


6. Select the newly created CSR in the certificate list with the "**PENDING**" status and click "**Download**" to save the CSR to your computer.

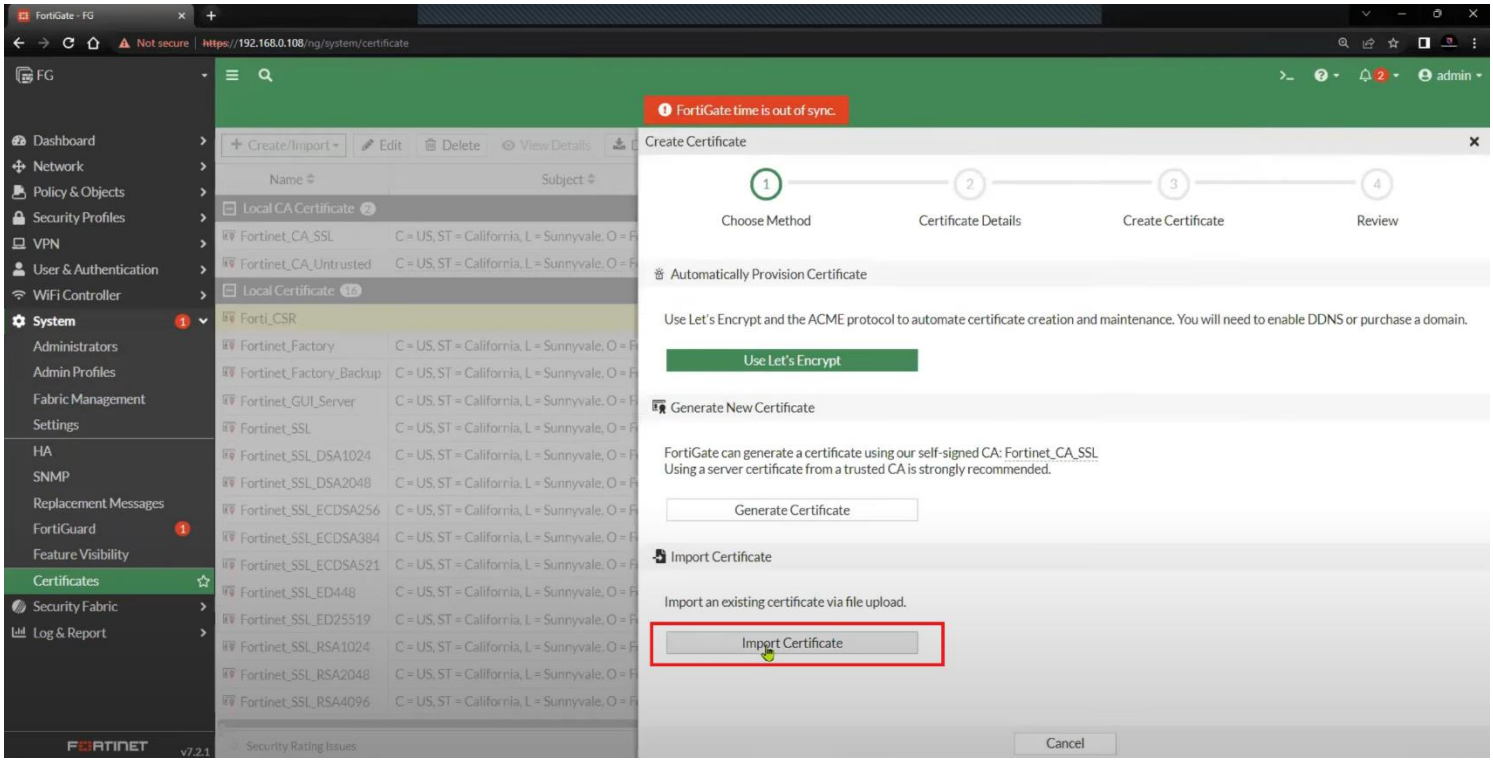


7. Use the newly downloaded ".**csr**" file to upload into CertCentral to request to issue your certificate. Alternatively, you can open the ".**csr**" file with a text editor to copy and paste the contents of the CSR into CertCentral.

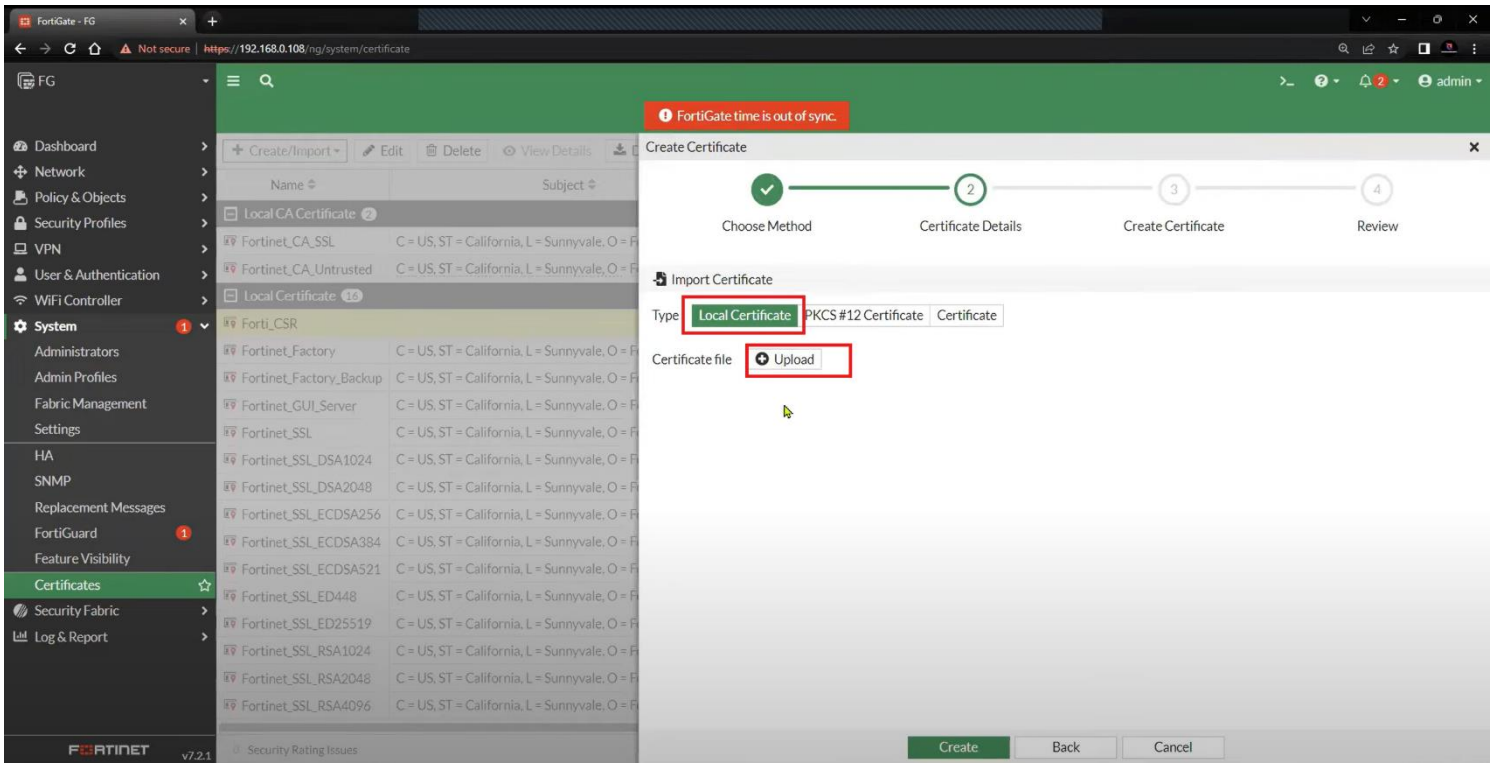
8. Go to **System** > **Certificates** > **Creat/Import** > **Certificate**



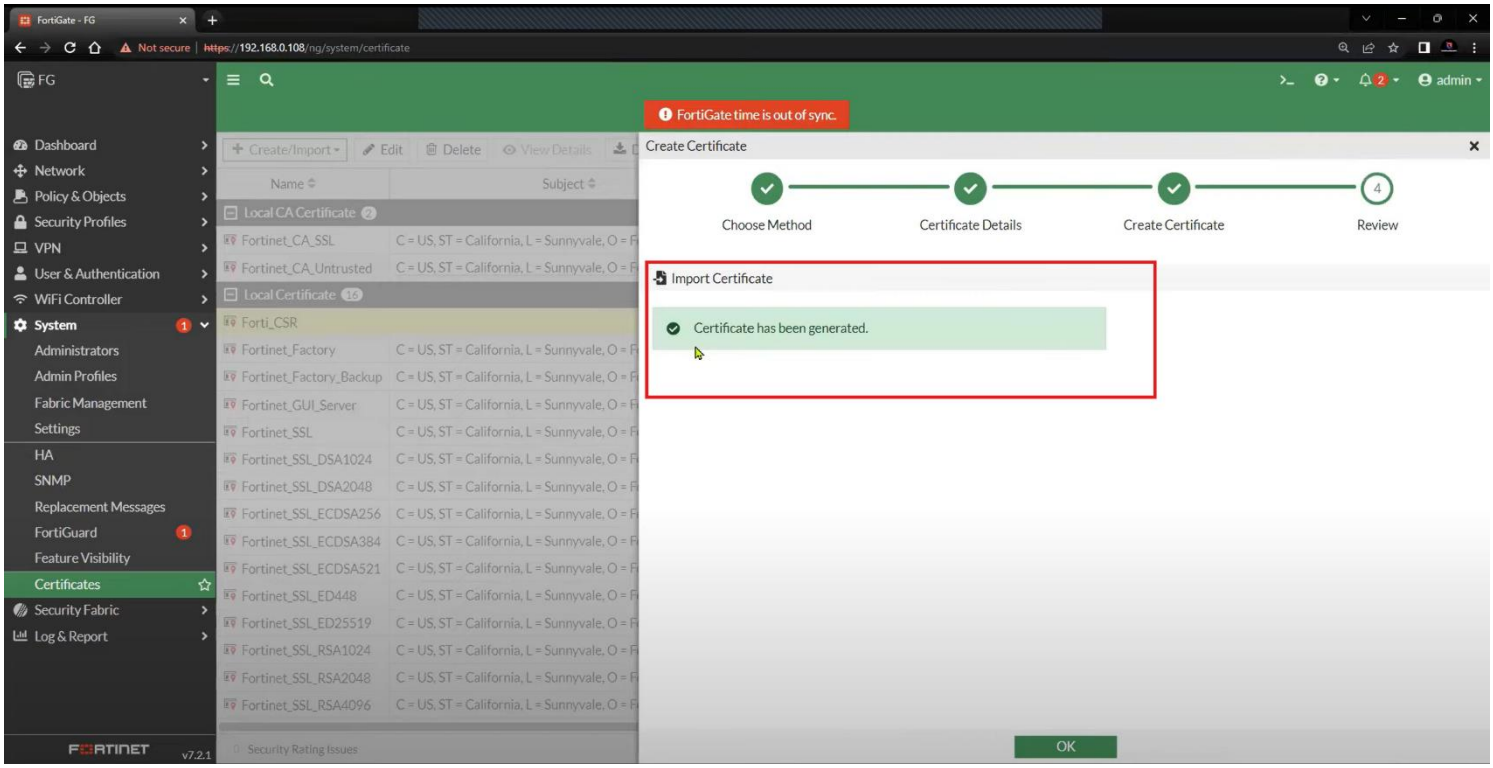
9. Click on the **Import Certificate**

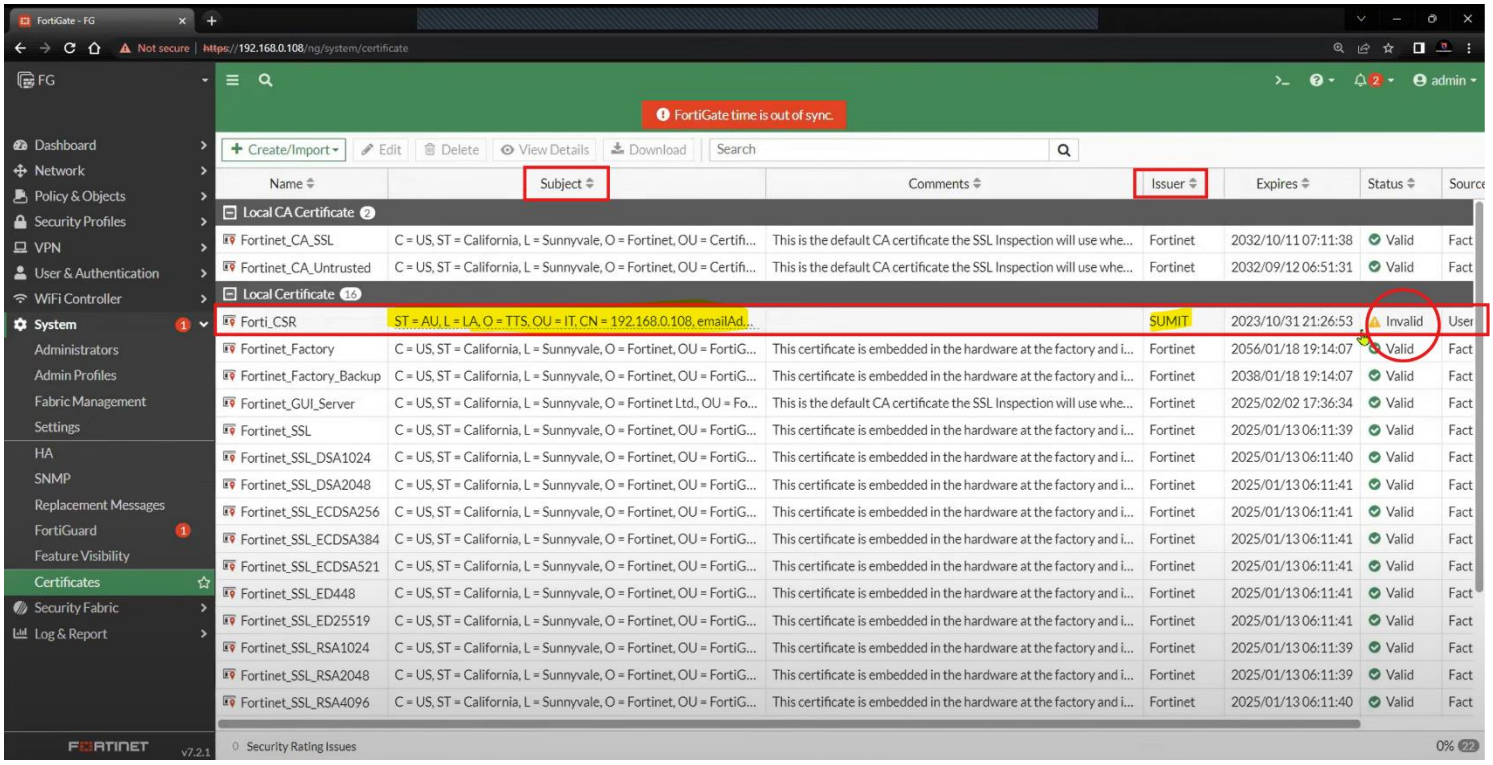


10. Click on the **Local Certificate** and then **Upload.** Choose the <u>CSR file</u>.



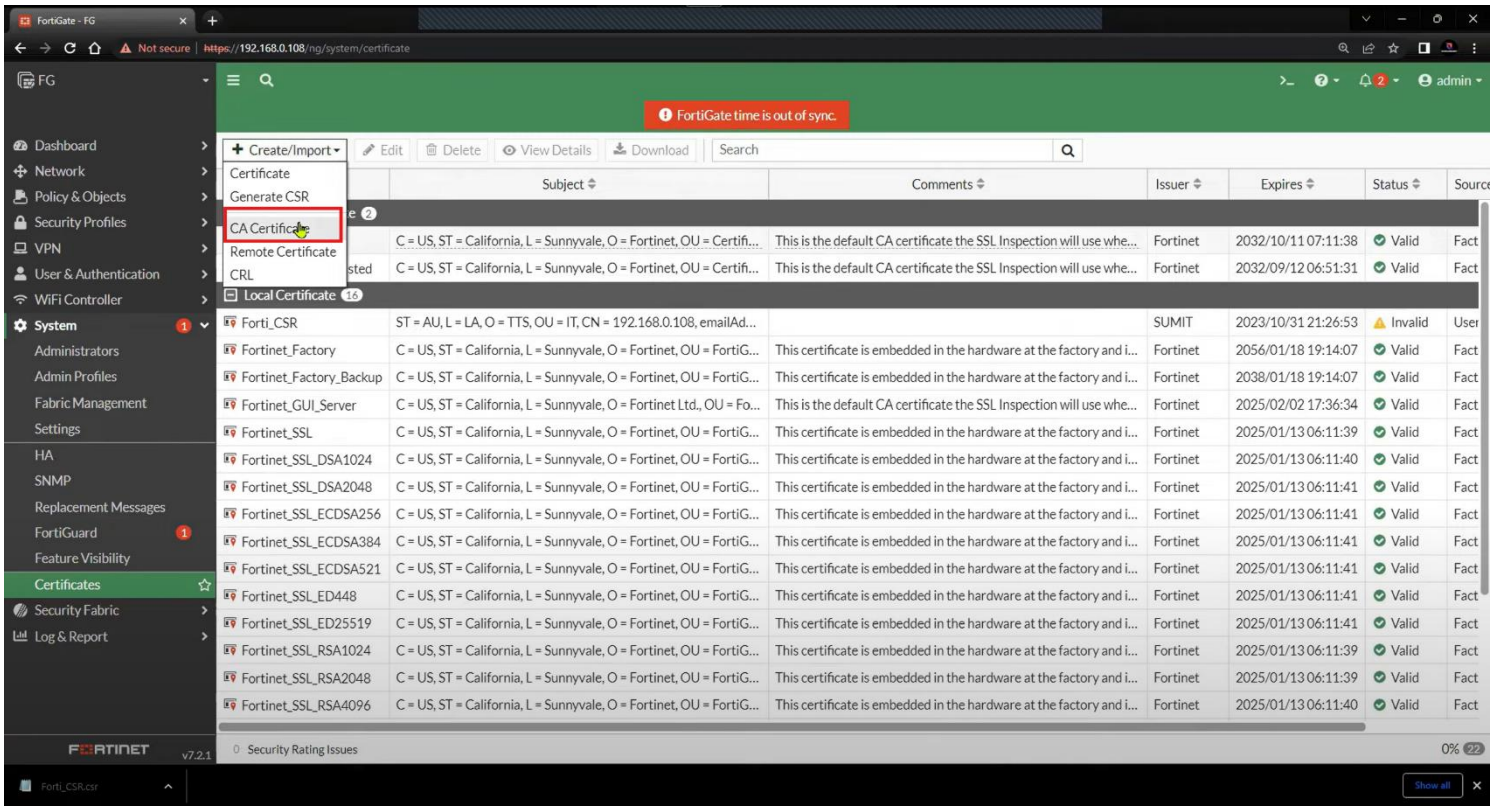
11. Click on the **Create.** You see: **"Certificate has been generated"**



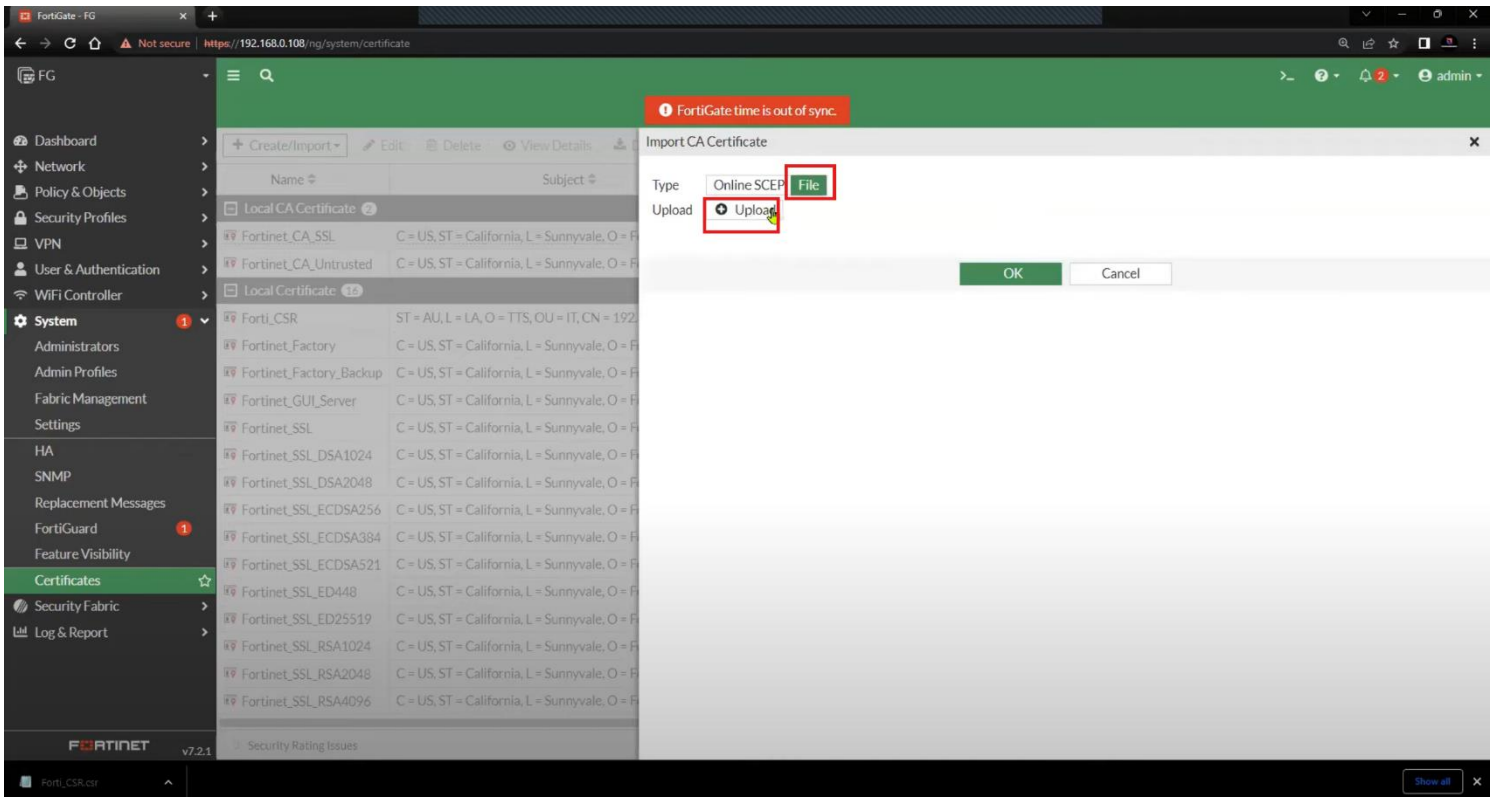


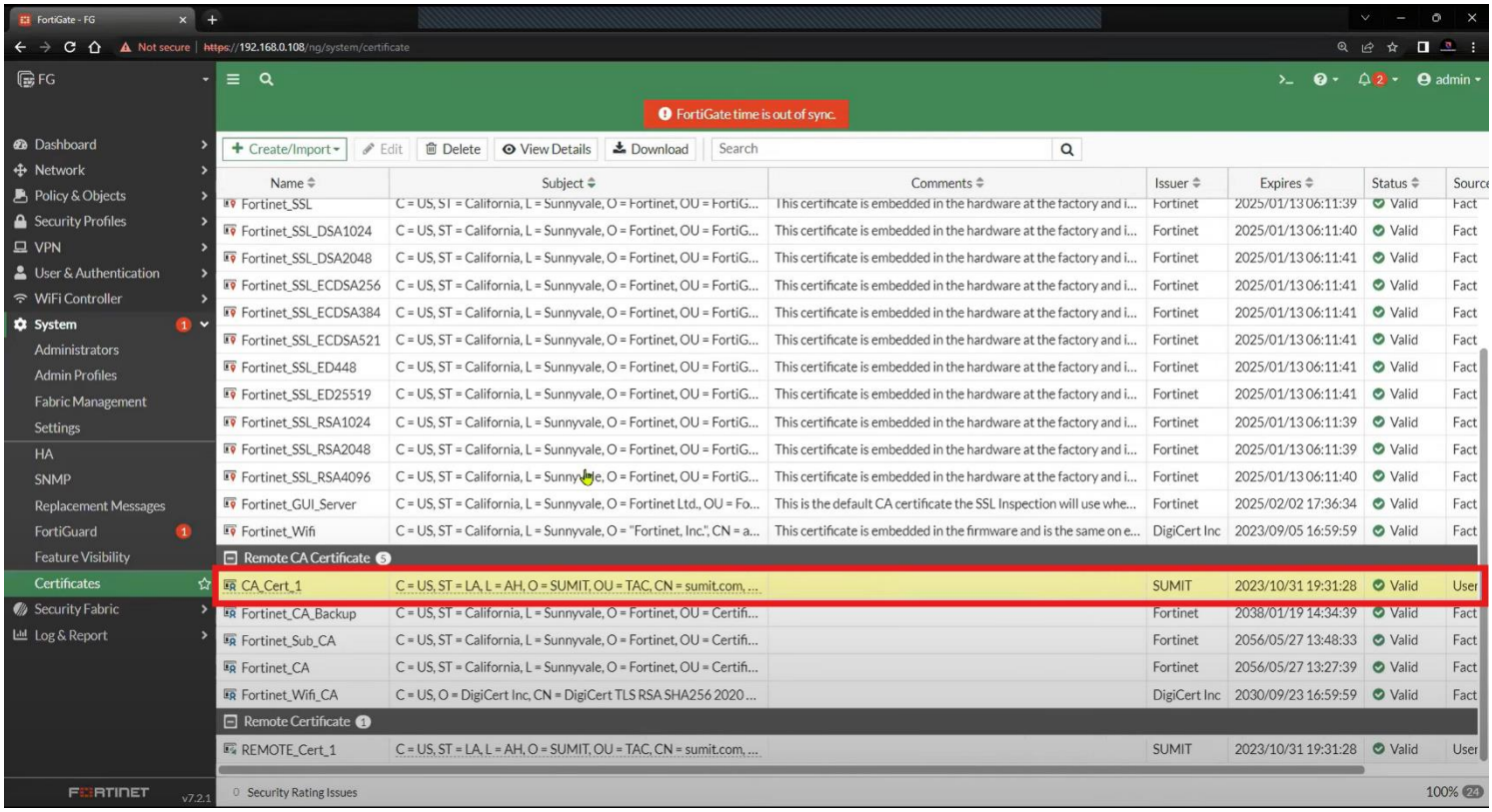
To import CA Certificate

1. Go to System > Certificates > Creat/Import > CA Certificate

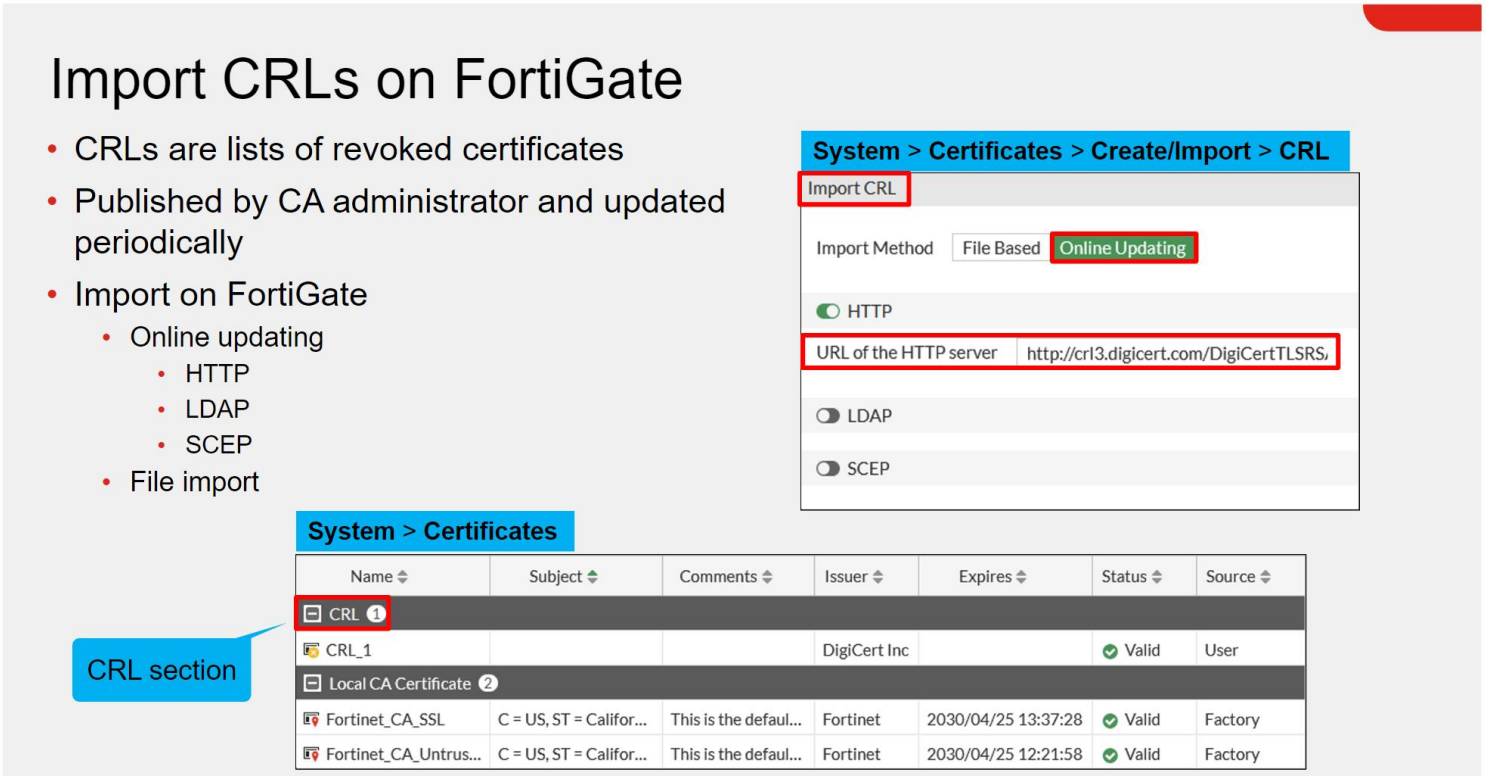


2. Go to File > Upload > Choose the CA Certificate > OK





Import CRLs on FortiGate



Because it is not possible to recall a certificate, the Certificate Revocation List (CRL) details certificates signed by valid CAs that should no longer be trusted. Certificates may be revoked for many reasons, such as if the certificate was issued erroneously, or if the private key of a valid certificate has been compromised.

CA administrators publish CRLs and periodically update them.

Import on FortiGate Options:

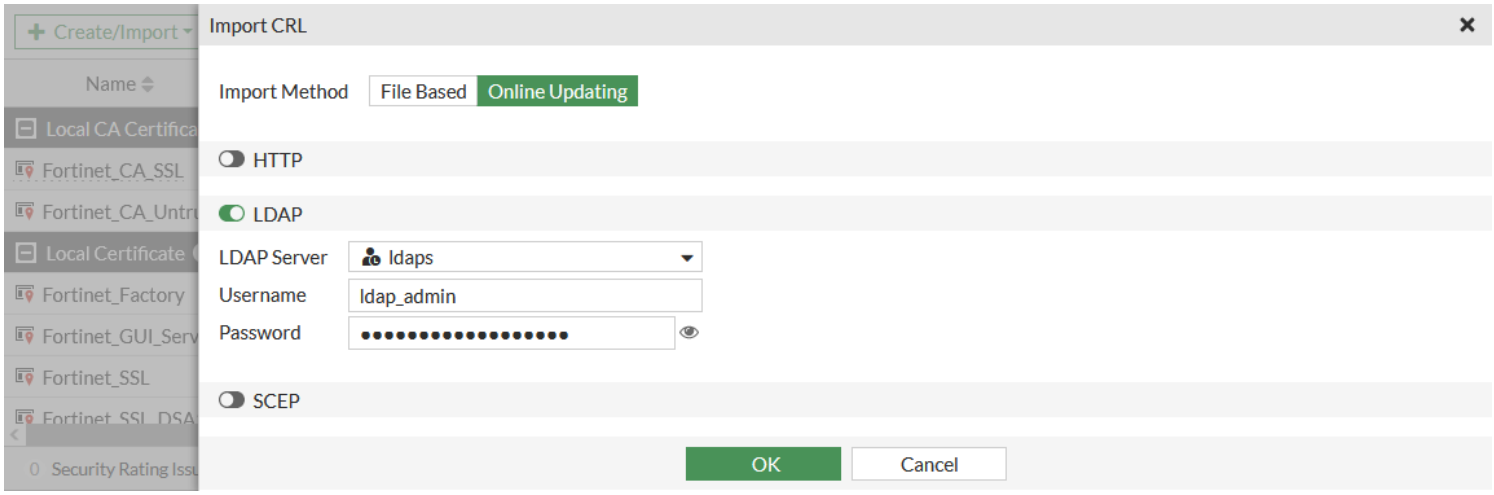
- You can load CRLs into the FortiGate device as **files** provided by CA administrators, into the FortiGate certificate store by importing CRL files.
- or direct FortiGate to connect to the CRL repositories and load the corresponding list. The recommended method to keep the list of revoked certificates up to date is to load them through one of the following available protocols: HTTP, LDAP, or SCEP.

You can get the CRL distribution point associated with a certificate by editing it and navigating to the CRL endpoints information part.

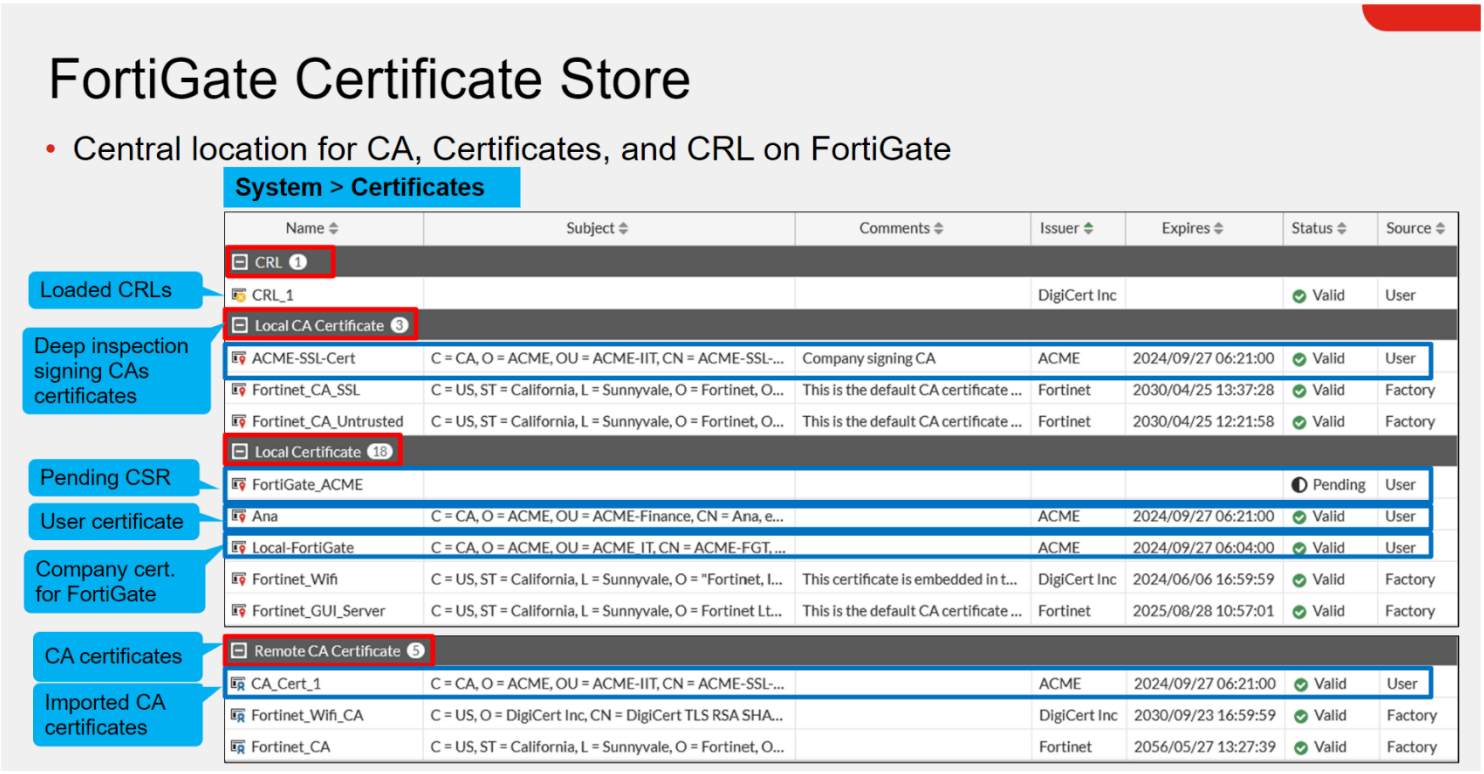
Note that the CRL section on the FortiGate GUI **Certificates** menu is visible only after you have loaded at least one CRL.

To import a CRL in the GUI:

1. Go to ***System*** **>** ***Certificates*** and select ***Create/Import*** **>** ***CRL***.
2. Set the ***Import Method*** to ***File Based*** or ***Online Updating***.
    - ***File Based***: Upload the CRL file directly from the management computer. CAs publish files containing the list of certificates that should no longer be trusted.
    - ***Online Updating***: This is the preferred method to keep the list of revoked certificates up to date. Configure the protocols as required.
        - ***HTTP*****:** Enter the *URL of the HTTP server*.
        - ***LDAP*****:** Select the *LDAP Server* and enter the *Username* and *Password*.
        - ***SCEP*****:** Select the *Certificate* and enter the *URL of the SCEP server*.



FortiGate Certificate Store



The central location to review the certificates imported into a FortiGate device is the certificate list available in the **Certificates** section of the System menu. **System > Certificates**

In this table you can view:

• The **CRL** section, which contains all loaded CRLs.

• The **Local CA Certificate** section, which contains the FortiGate Deep Inspection signing CA certificate. By default, it contains the Fortinet_CA_SSL and Fortinet_CA_untrusted certificates. If you import a signing CA certificate from your company, it will appear in this section.

• The **Local Certificate** section, which contains device and user certificates. In the example shown on this slide you can see a user certificate, Ana, and a device certificate, Local-FortiGate. For both, the issuer is ACME, which is the company private CA in this example.

• The **Remote CA Certificate** section, which is section where FortiGate displays all imported CA certificates that are not signing CA certificates.

Note that:

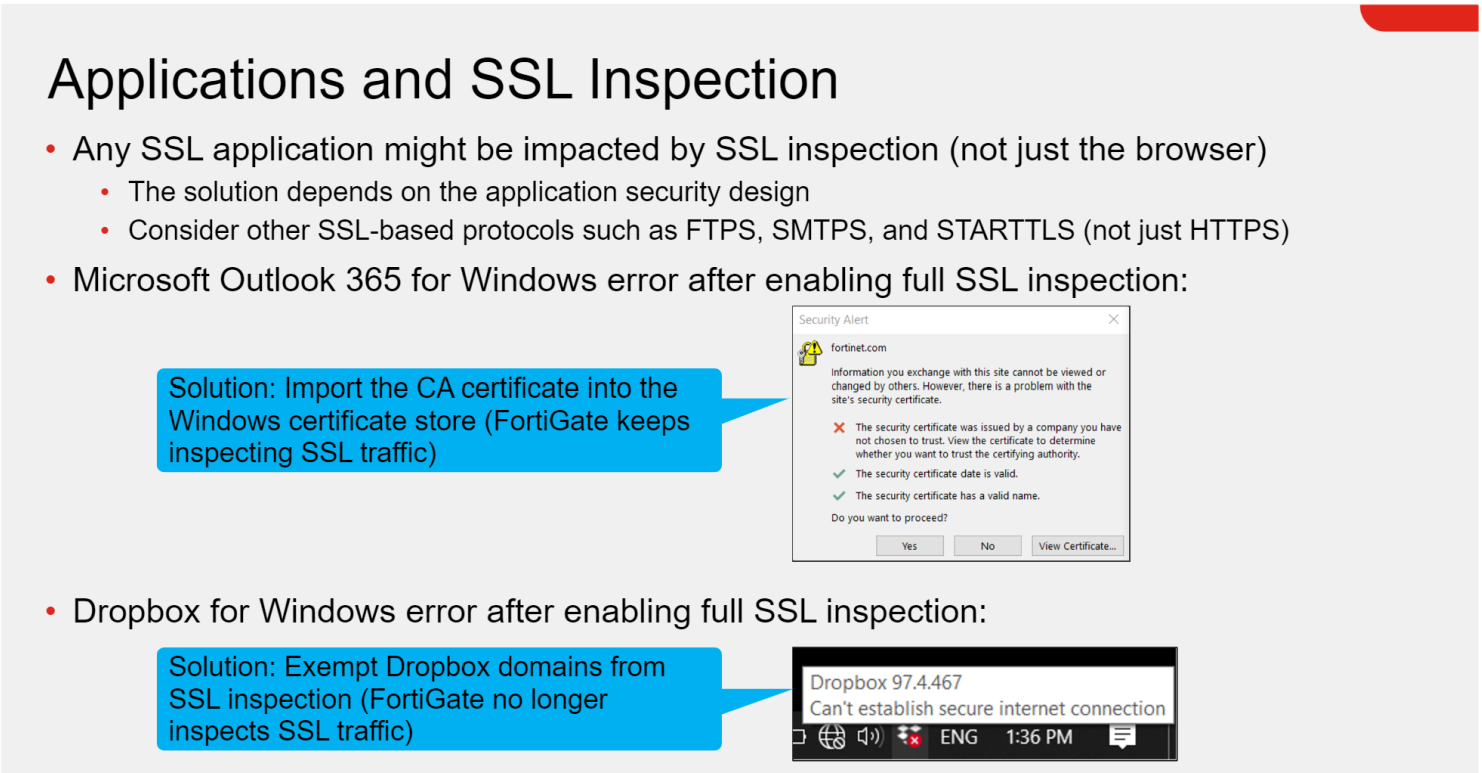
• The CRL section is visible only after you have loaded at least one CRL.

• FortiGate displays CRLs only if the corresponding CA certificate is imported into the certificate store.

• FortiGate shows the certificate signing requests (CSR) in the Local Certificate section with the status **Pending**.

• The Source column indicates the origin of the certificate, either Factory for certificates always present or User for certificates imported by an administrator user.

Applications and SSL Inspection



More and more **applications** are using SSL to securely exchange data over the internet. While most of the content in this lesson centers around the operation and impact of SSL inspection on browsers, the same applies to other applications using SSL as well. After all, the browser is just another application using SSL on your device.

For this reason, when you enable SSL inspection on FortiGate, you need to consider the potential impact on your SSL-based applications. For example, **Microsoft Outlook 365 for Windows** reports a certificate error when you enable full SSL inspection because the CA certificate used by FortiGate is not trusted. To solve this issue:

- You can import the CA certificate into your Windows certificate store as a trusted root certificate authority. Because Microsoft Outlook 365 trusts the certificates in the Windows certificate store, then the application won't report the certificate error anymore.
- Another option is to **exempt** your Microsoft Exchange server addresses from SSL inspection. While this prevents the certificate error, you are no longer performing SSL inspection on email traffic.

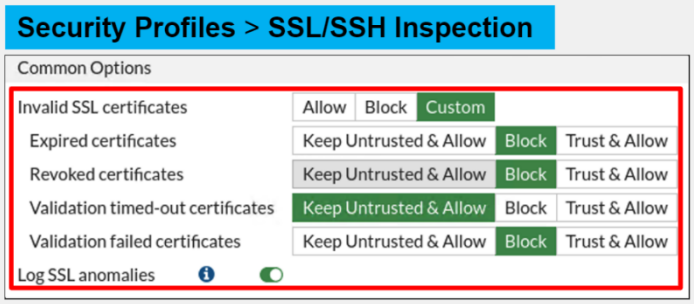
There are other applications that have built-in extra security checks that prevent MITM attacks, such as HSTS. For example, **Dropbox** uses certificate pinning to ensure that no SSL inspection is possible on user traffic. As a result, when you enable full SSL inspection on FortiGate, your Dropbox client stops working and reports that it can't establish a secure connection. In the case of Dropbox, the only way to solve the connection error is by **exempting** the domains Dropbox connects to from SSL inspection. (HTTP Strict Transport Security (HSTS) is a simple and widely supported standard to protect visitors by ensuring that their browsers always connect to a website over HTTPS)

In addition, remember that SSL is leveraged by different protocols, not just HTTP. For example, there are other SSL-based protocols such as FTPS, POP3S, SMTPS, STARTTLS, LDAPS, and SIP TLS. If you have an application using any of these SSL-based protocols, and you have turned on SSL inspection along with a security profile that inspects those protocols, then the applications may report an SSL or certificate error. The solution depends on the security measures adopted by the application.

Invalid SSL Certificates

Invalid Certificates

- FortiGate can detect invalid certificates for a variety of reasons
  - Invalid certificates produce security warnings due to problems with the certificate details
- FortiGate can **Keep Untrusted & Allow**, **Block,** or **Trust & Allow** invalid certificates
- Selecting **Custom** allows the user to select the action for each reason



FortiGate can detect certificates that are **invalid** for the following reasons:

- **Expired:** The certificate is expired.
  - ***Expired certificates*:** Action to take when the certificate is expired. The default action is block.
- **Revoked:** The certificate has been revoked based on CRL or OCSP information.
  - **Revoked certificates:** Action to take when server certificate is revoked. The default action is block.
- **Validation timeout:** The certificate could not be validated because of a communication timeout.
  - ***Validation timed-out certificates*:** Action to take when the certificate validation times out. For certificate inspection, the default action is Allow. For deep inspection, the default action is Keep Untrusted & Allow.
- **Validation failed:** FortiGate could not validate the certificate, or it is not yet valid.
  - ***Validation failed certificates*:** Action to take when the certificate validation fails. The default action is block.

When a certificate fails for any of the reasons above, you can configure any of the following **actions**:

- **Keep untrusted & Allow:** FortiGate allows the website and lets the browser decide the action to take. FortiGate takes the certificate as untrusted.
- **Block:** FortiGate blocks the content of the site.
- **Trust & Allow:** FortiGate allows the website and takes the certificate as trusted.

The certificate check feature can be broken down into two major checks, which are done in parallel:

- FortiGate checks if the certificate is invalid because of the four reasons described above.
- FortiGate performs certificate chain validation based on the CA certificates installed locally and the certificates presented by the SSL server.

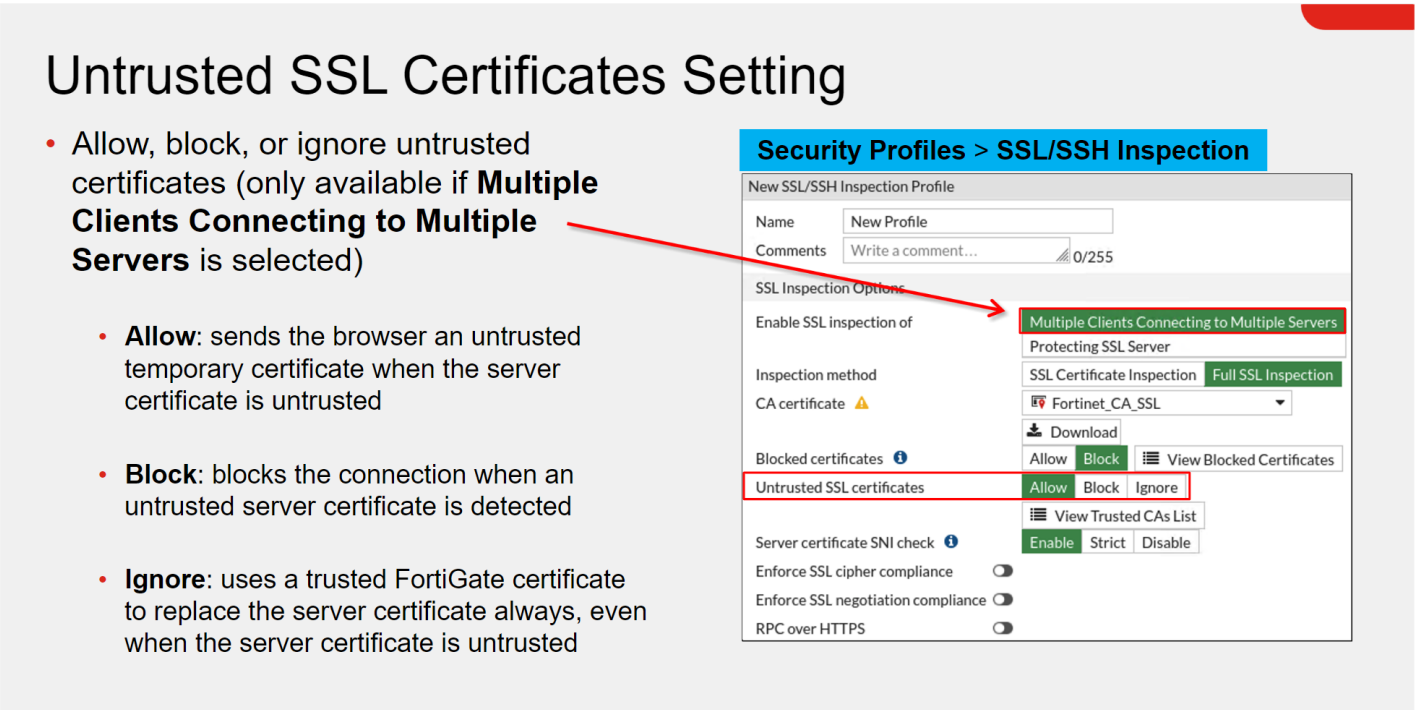
Based on the actions configured, and the check results, FortiGate presents the certificate as either **trusted** (signed by **Fortinet_CA_SSL**) or **untrusted** (signed by **Fortinet_CA_Untrusted**), and either **allows** the content or **blocks** it. You can also track certificate anomalies by enabling the Log SSL anomalies option.

Log SSL anomalies

Enable this feature to record and log traffic sessions containing invalid certificates.

By default, SSL anomalies logging is enabled. Logs are generated in the UTM log type under the SSL subtype when invalid certificates are detected.

Untrusted SSL Certificates Setting



This option is only available if **Multiple Clients Connecting to Multiple Servers** is selected.

The browser presents a certificate warning when you attempt to access an HTTPS site that uses an **untrusted certificate**. Untrusted certificates include self-signed SSL certificates, unless the certificate is imported into the browser-trusted certificate store. FortiGate has its own configuration setting on the **SSL/SSH Inspection** profile, which includes options to **Allow**, **Block**, or **Ignore** untrusted SSL certificates.

Allow:

When you set the Untrusted SSL certificates setting to **Allow**:

- If FortiGate detects an **untrusted SSL certificate**, FortiGate generates a temporary certificate signed by the built-in **Fortinet_CA_Untrusted** certificate. FortiGate then sends the temporary certificate to the browser, which presents a warning to the user indicating that the site is untrusted.
- If FortiGate receives a **trusted SSL certificate**, then it generates a temporary certificate signed by the built-in **Fortinet_CA_SSL** certificate and sends it to the browser.

If the browser trusts the **Fortinet_CA_SSL** certificate, the browser completes the SSL handshake. Otherwise, the browser also presents a warning message informing the user that the site is untrusted. In other words, for this function to work as intended, you must import the **Fortinet_CA_SSL** certificate into the trusted root CA certificate store of your browser. The **Fortinet_CA_Untrusted** certificate must not be imported.

Block:

When the setting is set to Block, and FortiGate receives an **untrusted SSL certificate**, FortiGate blocks the connection outright, and the user cannot proceed.

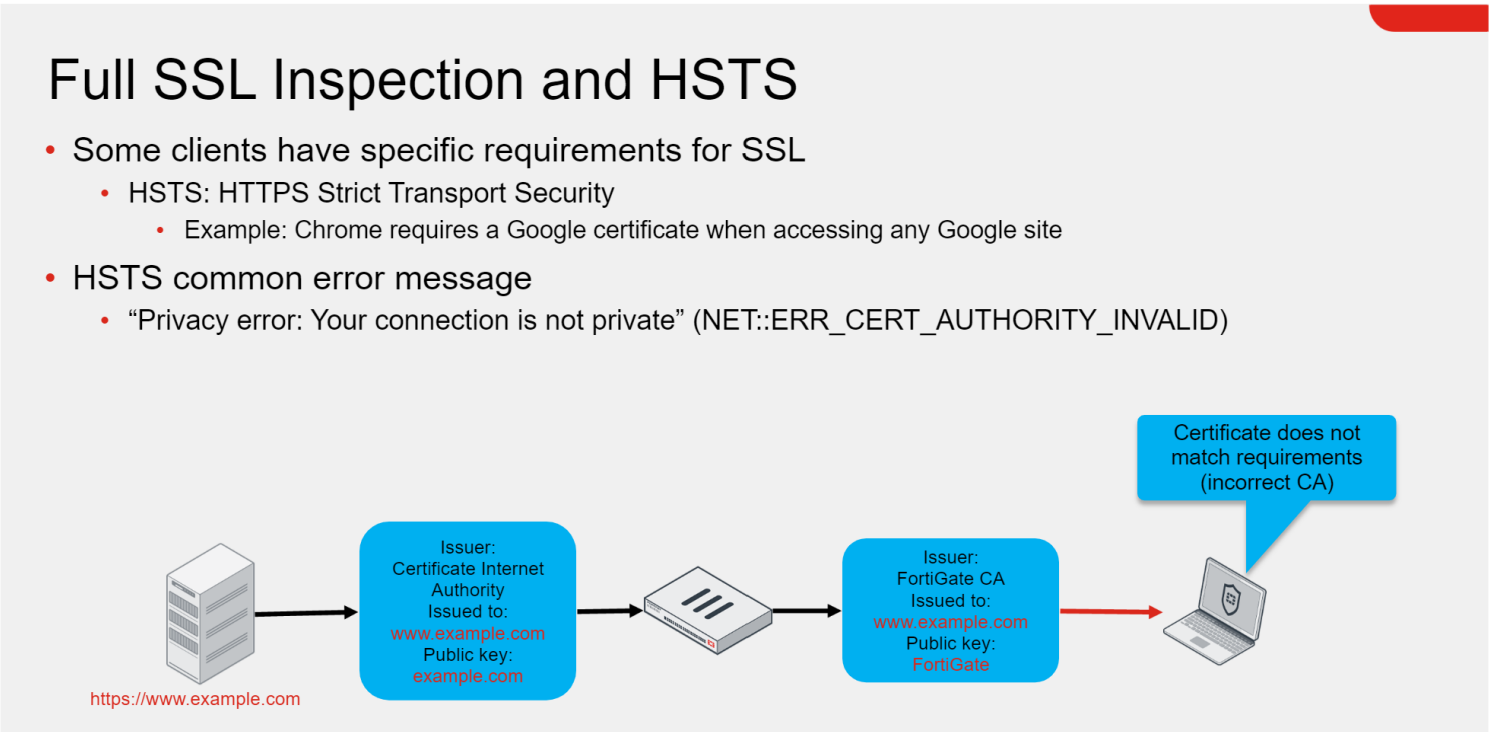
Ignore:

When the setting is set to Ignore, FortiGate sends the browser a temporary certificate signed by the **Fortinet_CA_SSL** certificate, regardless of the SSL certificate status—trusted or untrusted. FortiGate then proceeds to establish SSL sessions.

View Trusted CAs List

Click ***View Trusted CAs List*** to see a list of the factory bundled and user imported CAs that are trusted by the FortiGate.

Full SSL Inspection and HSTS



Replacing the certificate for the traffic can cause problems. Some software and servers have specific limitations and requirements on the certificates that are allowed to be used.

HSTS?

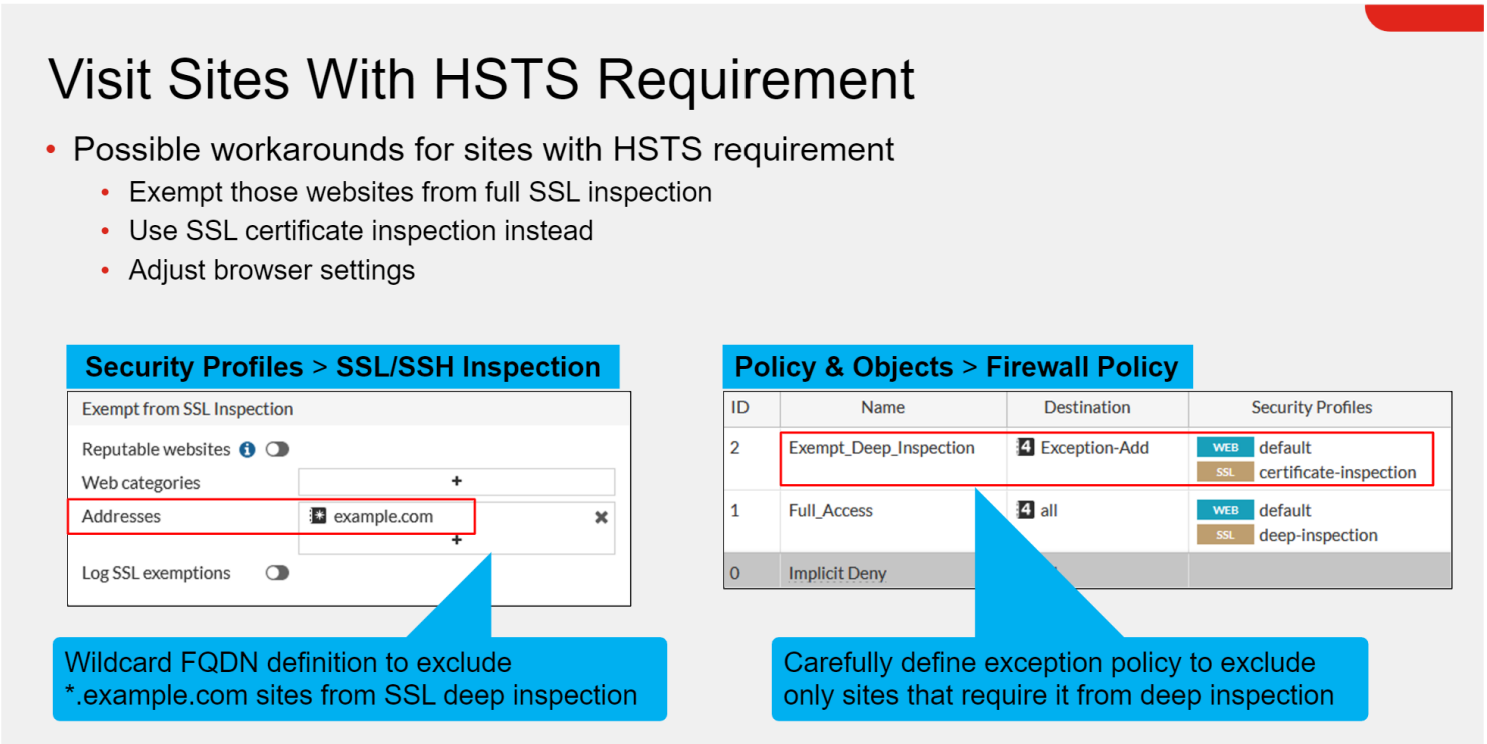
HSTS is a security features designed to detect man-in-the-middle SSL attacks by making sure that any certificate presented when accessing a server resource is signed by a specific CA.

Problem:

If the browser detects any other CA, it simply refuses to continue the SSL handshake, and prevents access to the website. If you are using a Chrome browser, for such sites, you will get the privacy error message "Your connection is not private" this slide shows.

Solution:

Visit Sites with HSTS Requirement



When replacing the certificate for the traffic causes problems and prevents users from accessing some websites, the solutions available are limited.

You can select one of the following workarounds according to the level at which you can act:

- At the FortiGate level, you can **exempt** the affected websites from full SSL inspection and use **certificate inspection** instead.
- If you can act at the browser level, you can disable HSTS validation per website or globally (refer to the browser manual for the process).

If you want to use certificate inspection instead of deep inspection only for a few sites, you must be careful when defining the policy. It must be restrictive enough to match exclusively the sites that you want to allow, and which do not support deep inspection. Otherwise, you might get sites allowed to pass through with only certificate inspection instead of deep inspection.

LAB

Certificate Operations

In this lab, you will configure full SSL inspection using a self-signed SSL certificate on FortiGate to inspect outbound traffic. Next, you will review some situations that prevent full SSL inspection, and implement workarounds. Finally, you will learn how to deal with some certificate anomalies.

Objectives

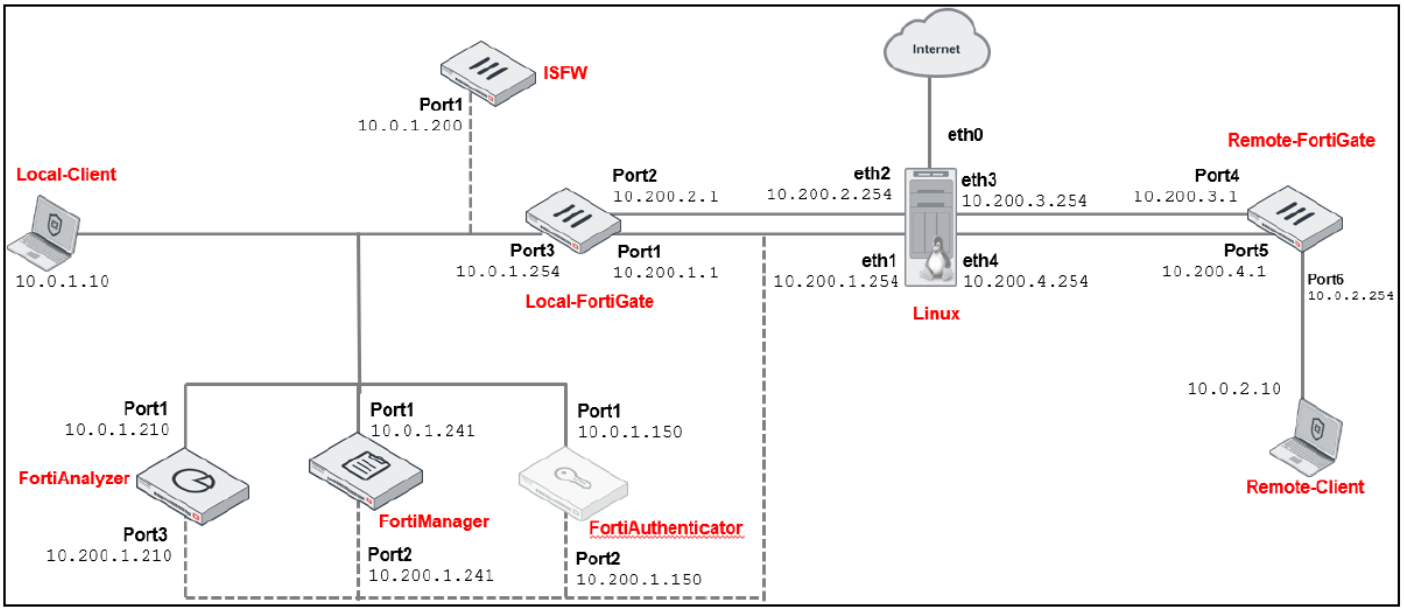
- Configure and enable full SSL inspection on outbound traffic
- Deal with certificate anomalies

We have two exercises in this LAB:

Exercise 1: Configuring Full SSL Inspection on Outbound Traffic

Exercise 2: Dealing with Anomalies

LAB Topology:



Exercise 1:

Configuring Full SSL Inspection on Outbound Traffic

Full SSL inspection on outbound traffic allows FortiGate to inspect encrypted internet traffic and apply security profiles to that traffic. It protects your network and end users from potential malware that could come from secure websites, like HTTPS websites, that internal users visit. FortiGate employs a man-in-the-middle (MITM) technique to inspect the traffic and apply security profiles, such as antivirus, web filter, and application control.

In this exercise, you will configure and enable full SSL inspection on all outbound traffic.

Configure SSL Inspection

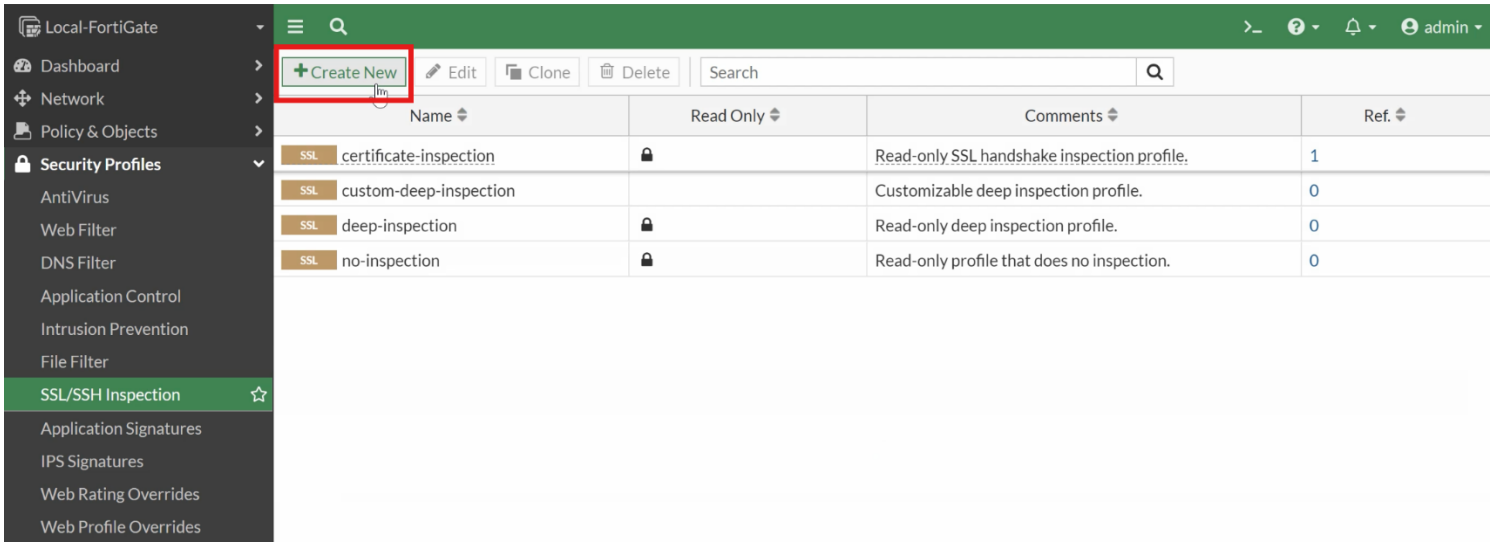
By default, FortiGate includes four security profiles for SSL/SSH inspection:

- **certificate-inspection**,
- **custom-deep-inspection**,
- **deep-inspection**,
- **no-inspection**.

You can modify the settings for the **custom-deep-inspection** profile only or create a personalized profile. The other profiles are read-only. Because this exercise involves configuring full SSL inspection on FortiGate, you will configure a new SSL/SSH inspection profile for this purpose.

To configure SSL inspection

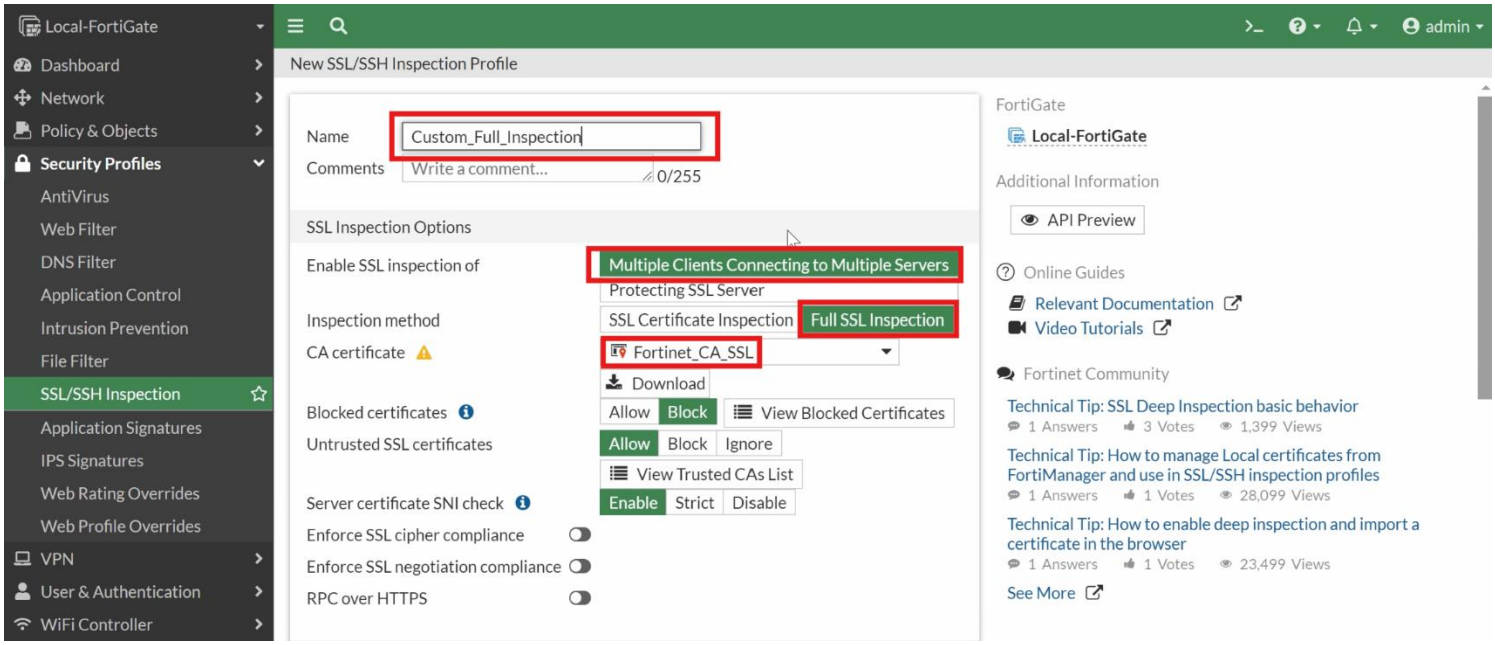
1. Connect to the Local-FortiGate GUI, and then log in with the username `admin` and password `password`.
2. Click **Security Profiles** > **SSL/SSH Inspection**.
3. Click **Create New** to create a new profile.



4. In the **Name** field, type `Custom_Full_Inspection`.

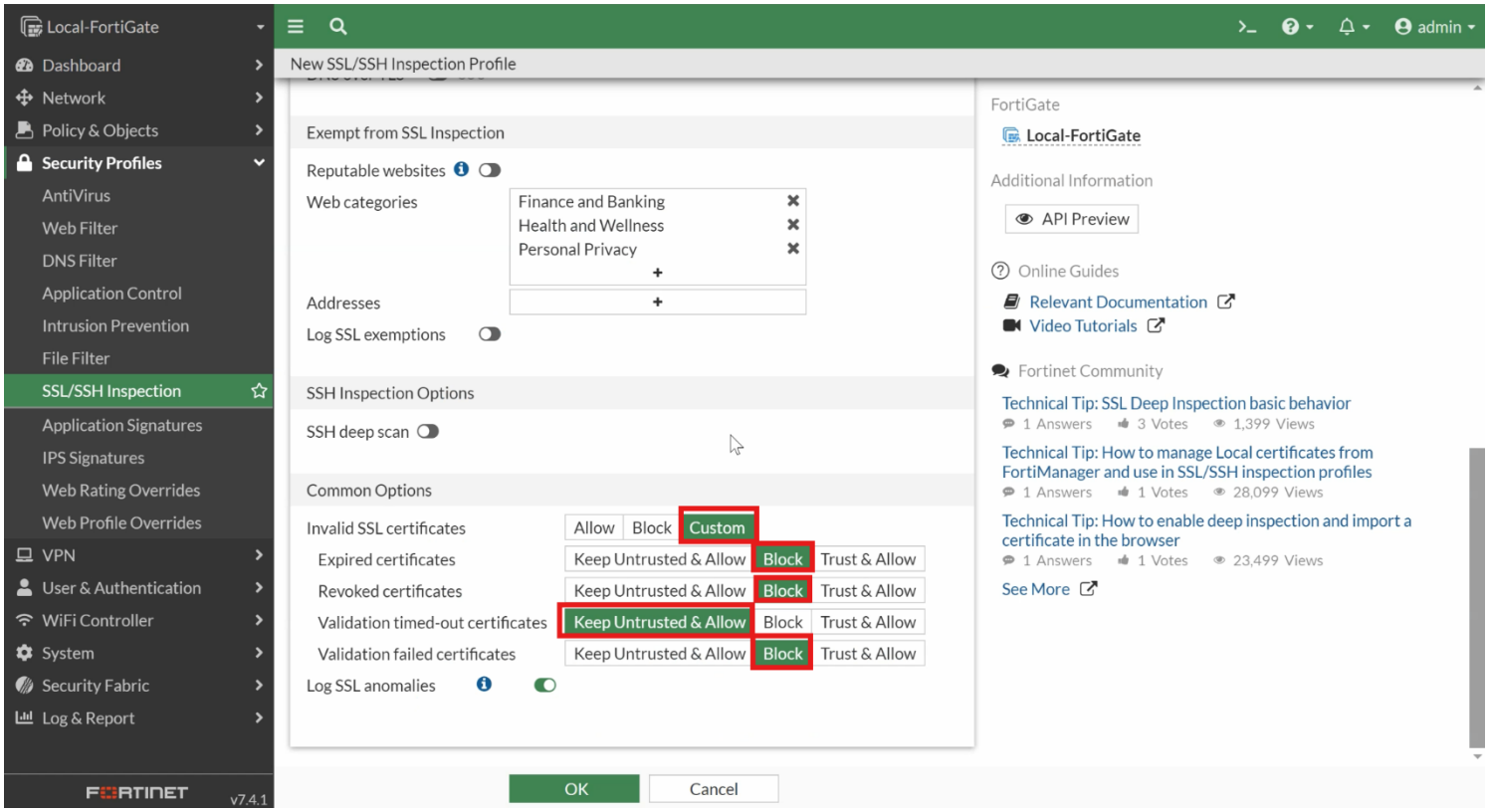
5. In the **SSL Inspection Options** section, verify that the following settings are configured (default values):

Field	Value
Enable SSL inspection of	Multiple Clients Connecting to Multiple Servers
Inspection method	Full SSL Inspection
CA certificate	Fortinet_CA_SSL



6. Scroll down to the bottom of the page, and then in the Common Options section, do the following:

- In the **Invalid SSL certificates** field, select Custom.
- Confirm that the other settings are configured as shown in the following image (default values):



7. Click OK.

Enable SSL Inspection in a Firewall Policy

You must enable SSL inspection in a firewall policy to start inspecting SSL traffic. In this policy, you will use **SSL inspection associated with web filtering**. For the purposes of this lab, you will enable the default web filter security profile.

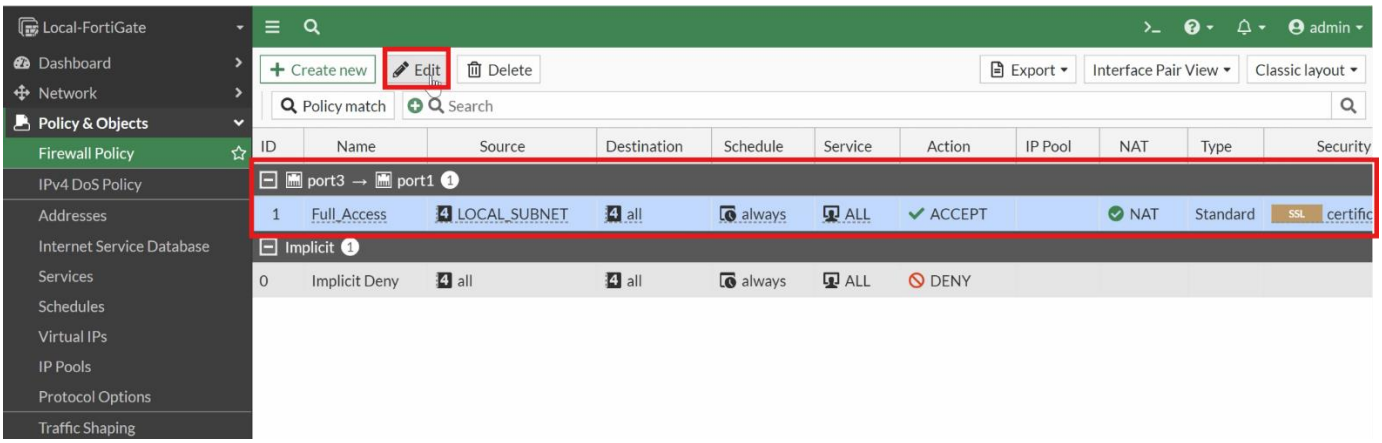
To enable SSL inspection in a firewall policy

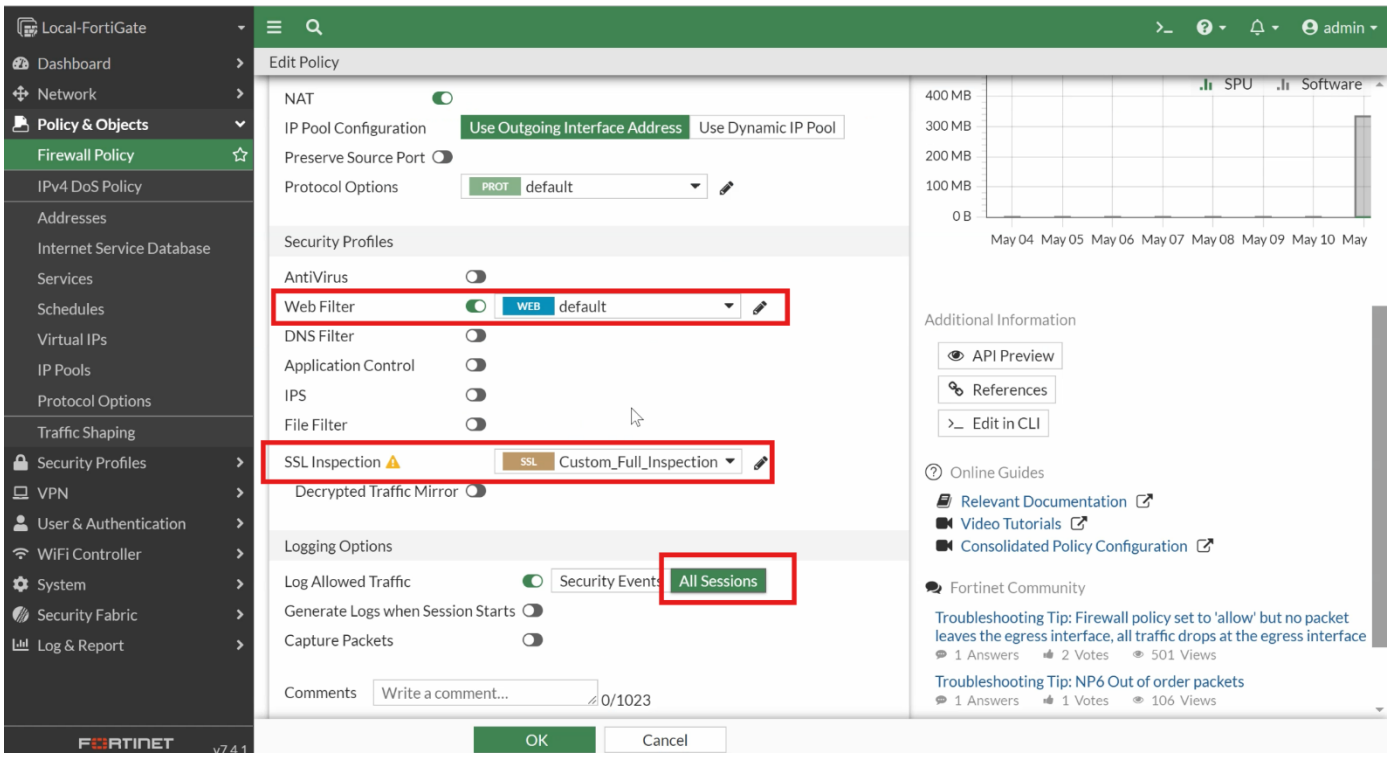
1. Continuing on the Local-FortiGate GUI, click **Policy & Objects** > **Firewall Policy**.
2. Edit the **Full_Access** firewall policy.
3. In the **Security Profiles** section, enable the following security profiles:

Security Profile	Value
Web Filter	default
SSL Inspection	Custom_Full_Inspection (This is the profile you created previously)

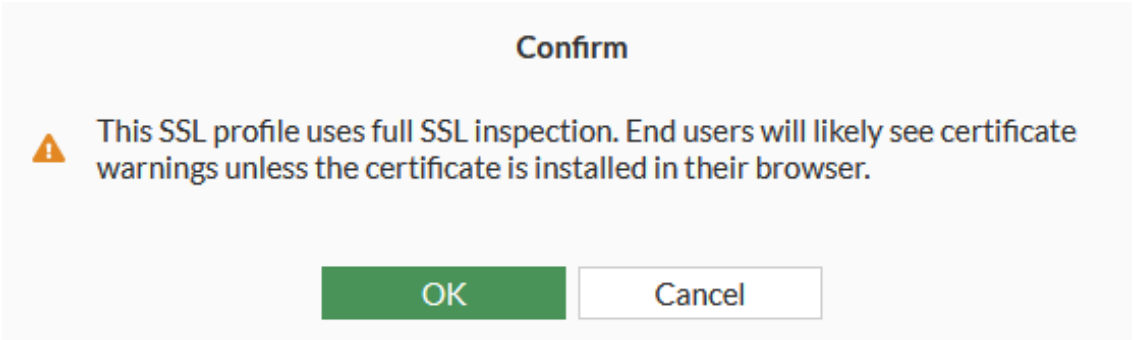
4. In the **Logging Options** section, enable **Log Allowed Traffic**, and then select **All Sessions**.

5. Click **OK**.

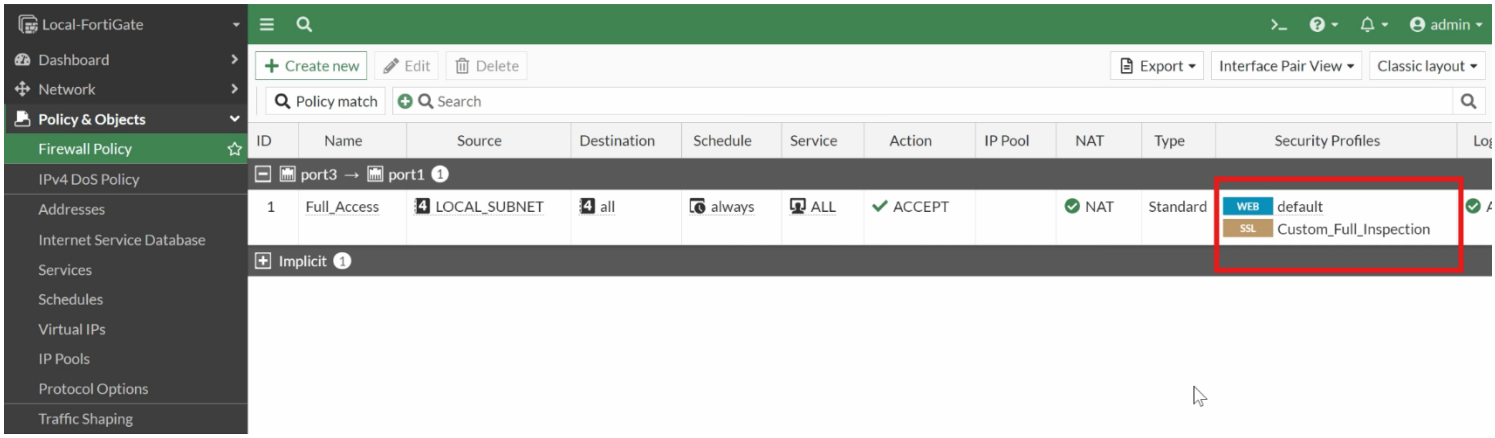




FortiGate displays a warning message to highlight that full SSL inspection is activated and might trigger warnings in users' browsers.



6. Read the warning message, and then click **OK**.



Install the Fortinet_CA_SSL Certificate

FortiGate includes an SSL certificate, named Fortinet_CA_SSL, that you can use for **full SSL inspection**. The SSL inspection profile you created in the previous step uses it. This certificate is signed by a certificate authority (CA) named **FortiGate CA**, which is not public. Because the CA is not public, each time a user connects to a secure website, the browser displays a certificate warning. This is because the browser receives traffic encrypted by certificates signed by FortiGate, using a CA it does not know and trust.

You can avoid this warning by downloading the **Fortinet_CA_SSL** certificate and installing it on all workstations as a public authority.

You will first test access to a secure website ***without*** the **Fortinet_CA_SSL** certificate installed in the browser. Then, you will install the Fortinet_CA_SSL certificate in the browser and test access to the secure website again.

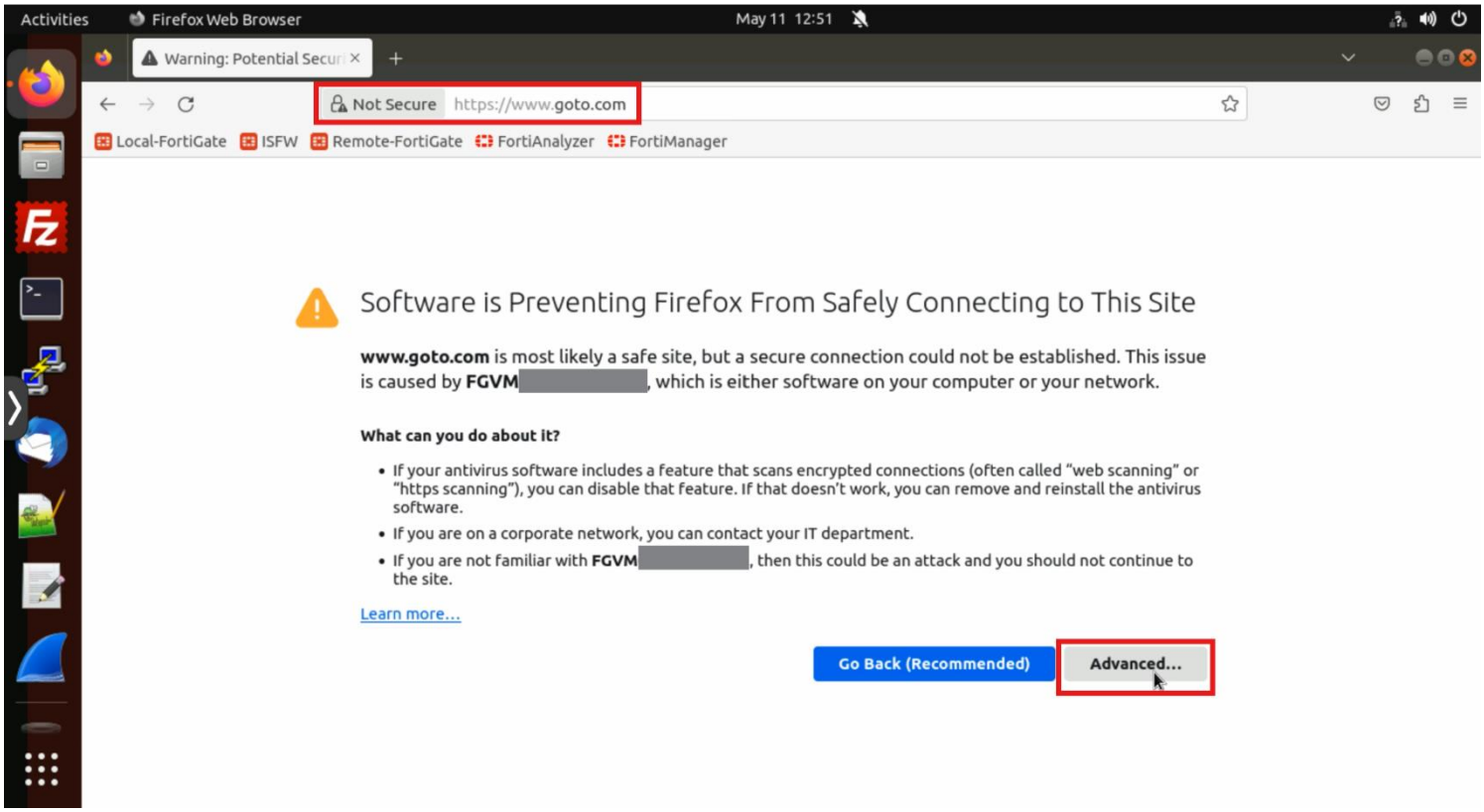
To test full SSL inspection without a trusted CA

1. Connect to the Local-Client VM, and then log in with the username `Administrator` and password `password`.

2. Open a browser, and then go to an HTTPS site, such as:

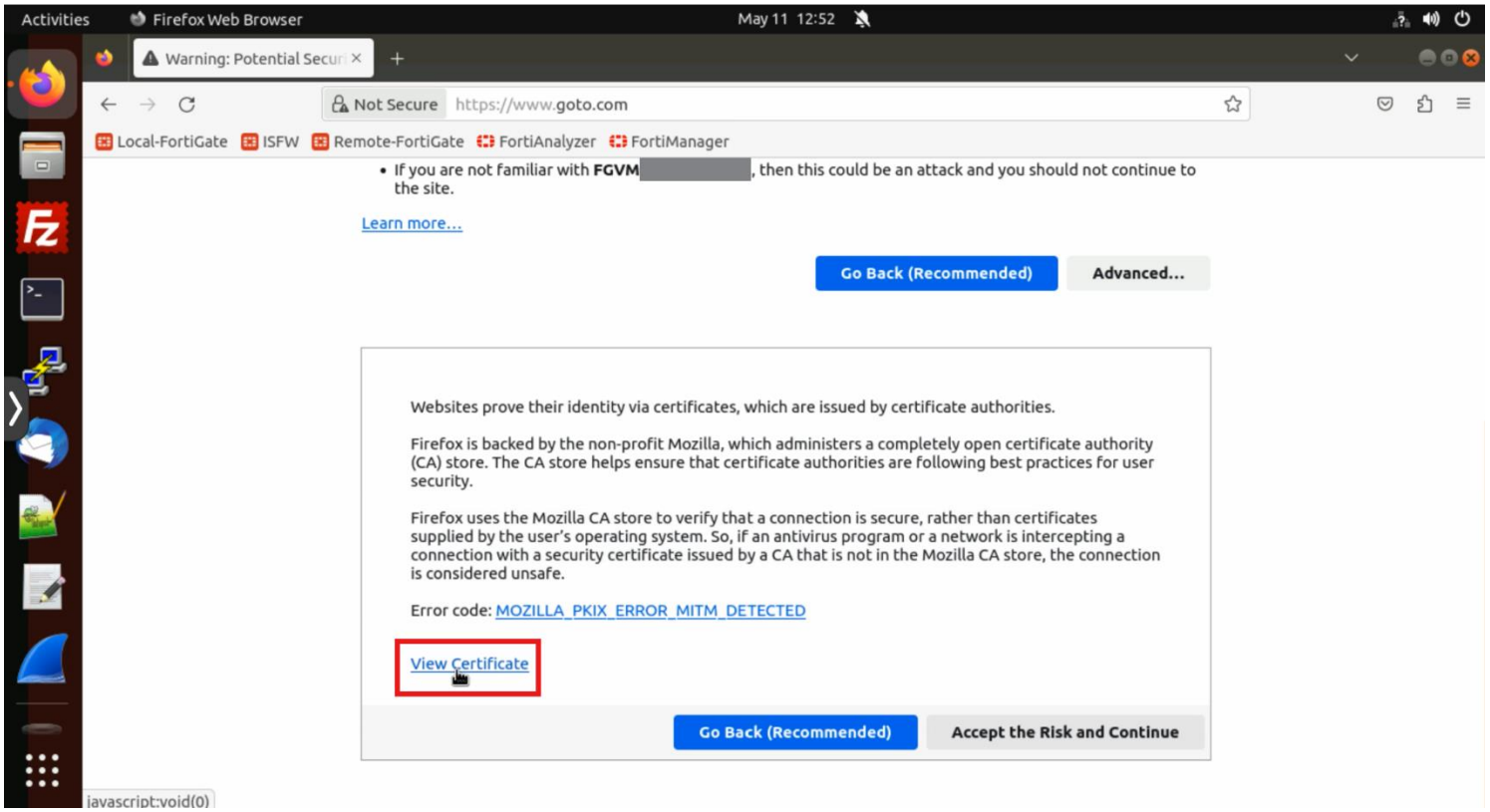
   https://www.goto.com

3. Notice the certificate warning.

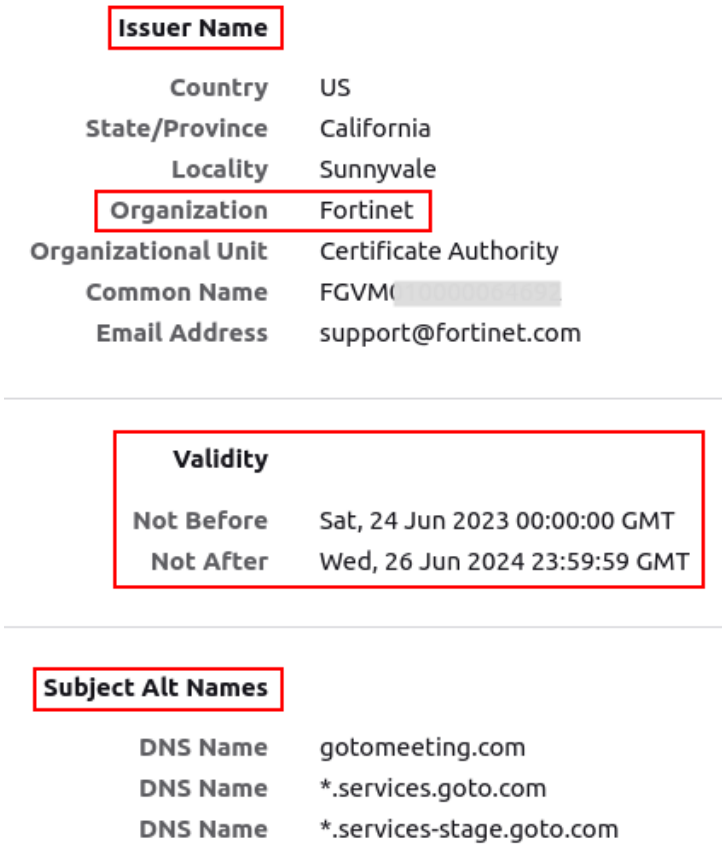


This warning appears because the browser receives certificates signed by the FortiGate CA private key, and the corresponding CA certificate is not in the certificate store of the Local-Client VM.

4. Click **Advanced**, and then click **View Certificate**.



You can see that the certificate is issued by Fortinet (Issuer Name), and that it is valid. The subject alternative names list includes a reference to the website you visited (goto.com, in our example).



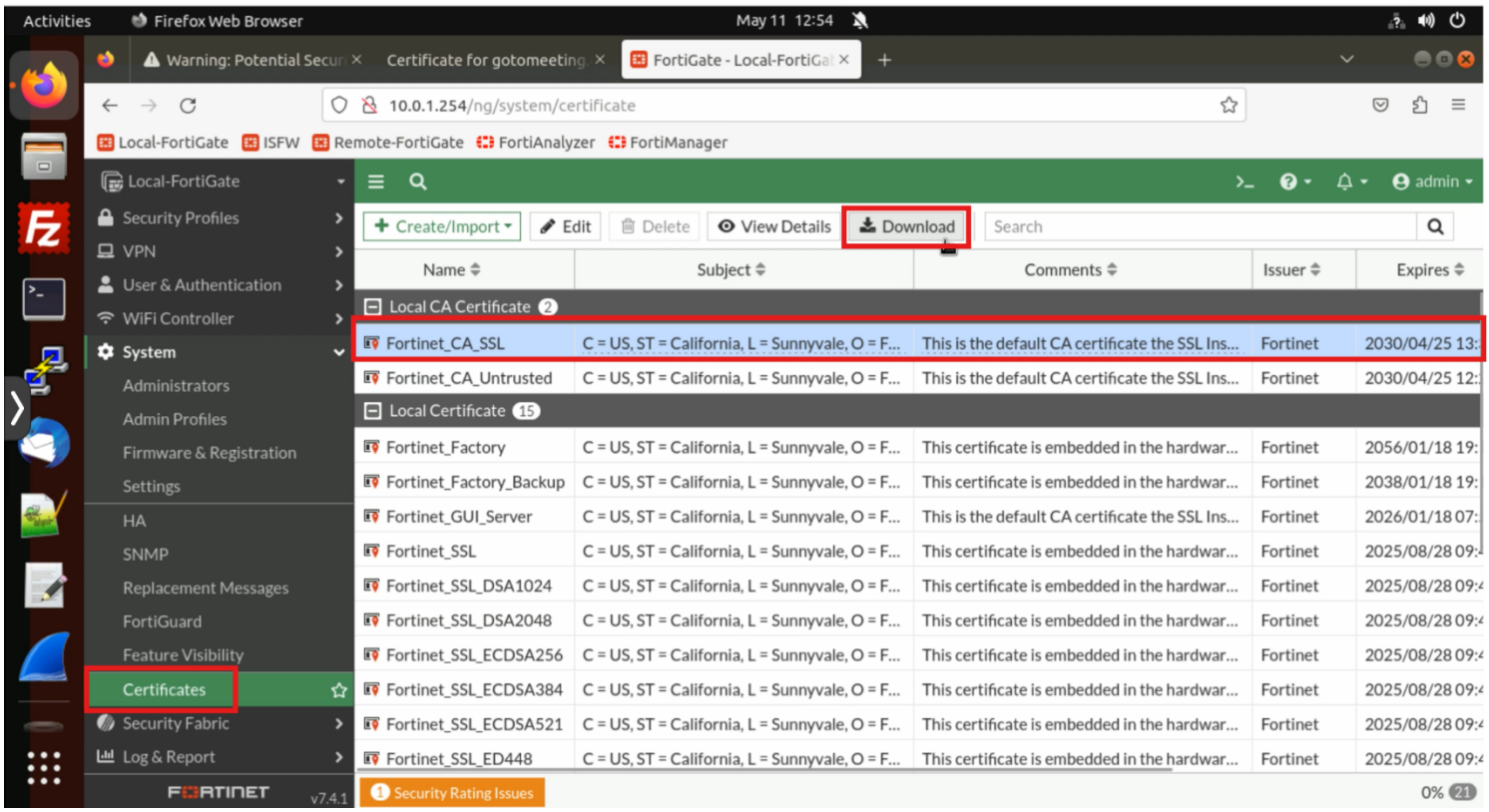
5. Do *not* click **Accept the Risk and Continue**.
6. Leave the browser tab open, and then continue to the next procedure.

To install the Fortinet_CA_SSL certificate in the browser

1. On the Local-Client, open a new browser tab, and then log in to the Local-FortiGate GUI at `10.0.1.254` with the username `admin` and password `password`.

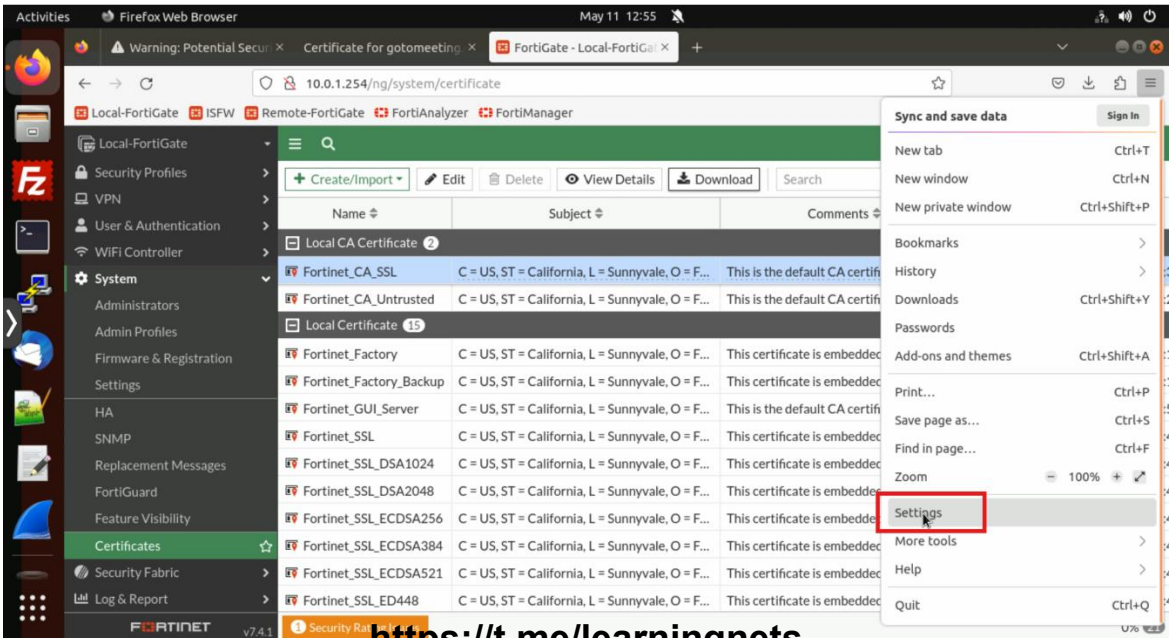
This time, you might see a warning because the FortiGate GUI presented a certificate signed by a CA that your browser doesn't trust.

2. If you get the warning message, click **Advanced**, and then click **Accept the Risk and Continue**.
3. Click **System** > **Certificates**.
4. In the **Local CA Certificate** section, click **Fortinet_CA_SSL**, and then click **Download**.



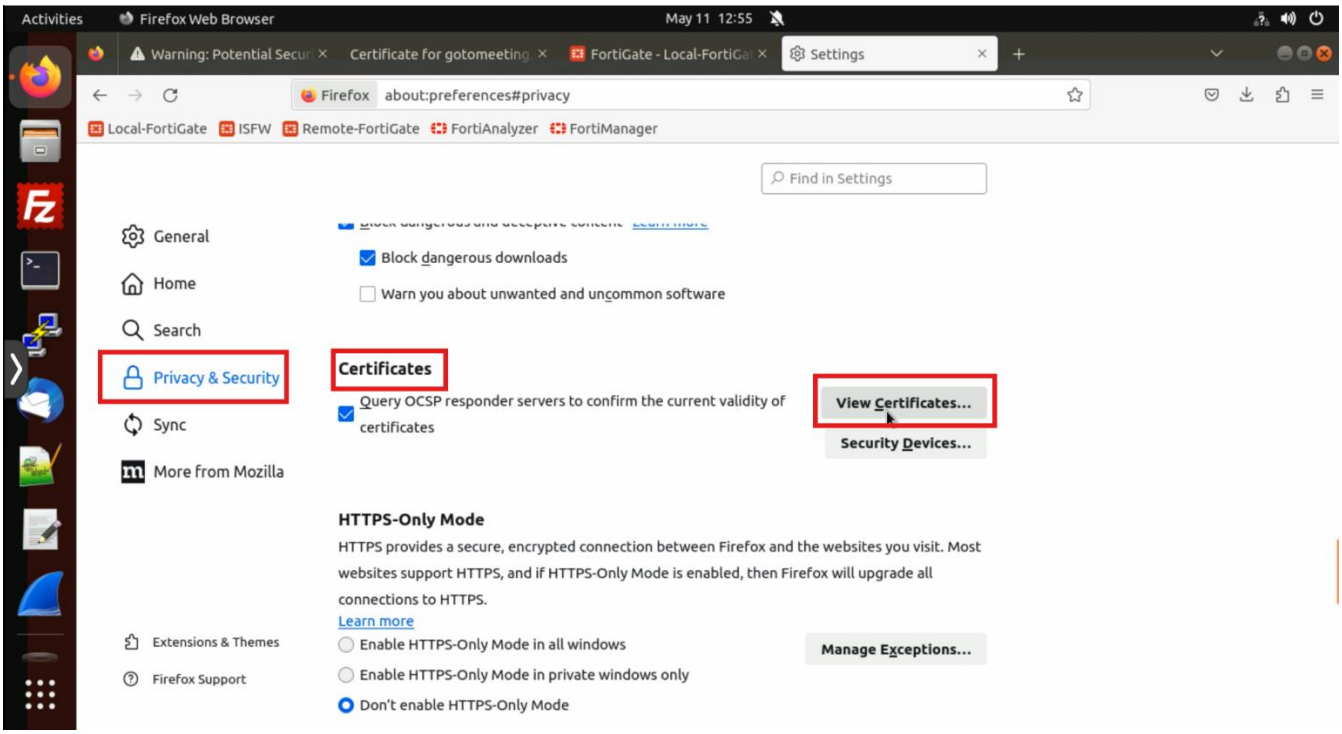
The browser downloads the certificate to the **Downloads** folder of your Local-Client VM.

5. Continuing in Firefox, in the upper-right corner, click the **Open menu** icon, and then click **Settings**.



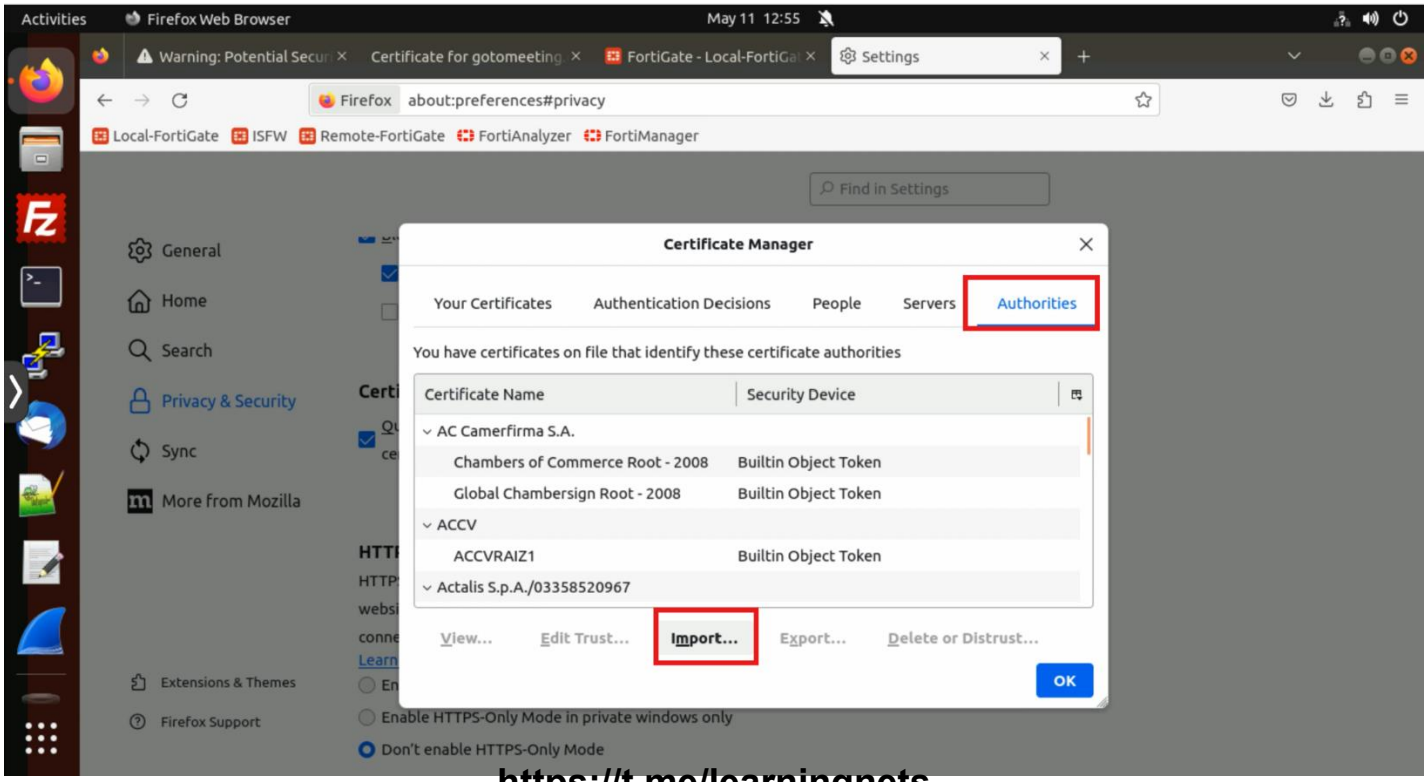
6. Click **Privacy & Security**.

7. In the **Certificates** section, click **View Certificates**.

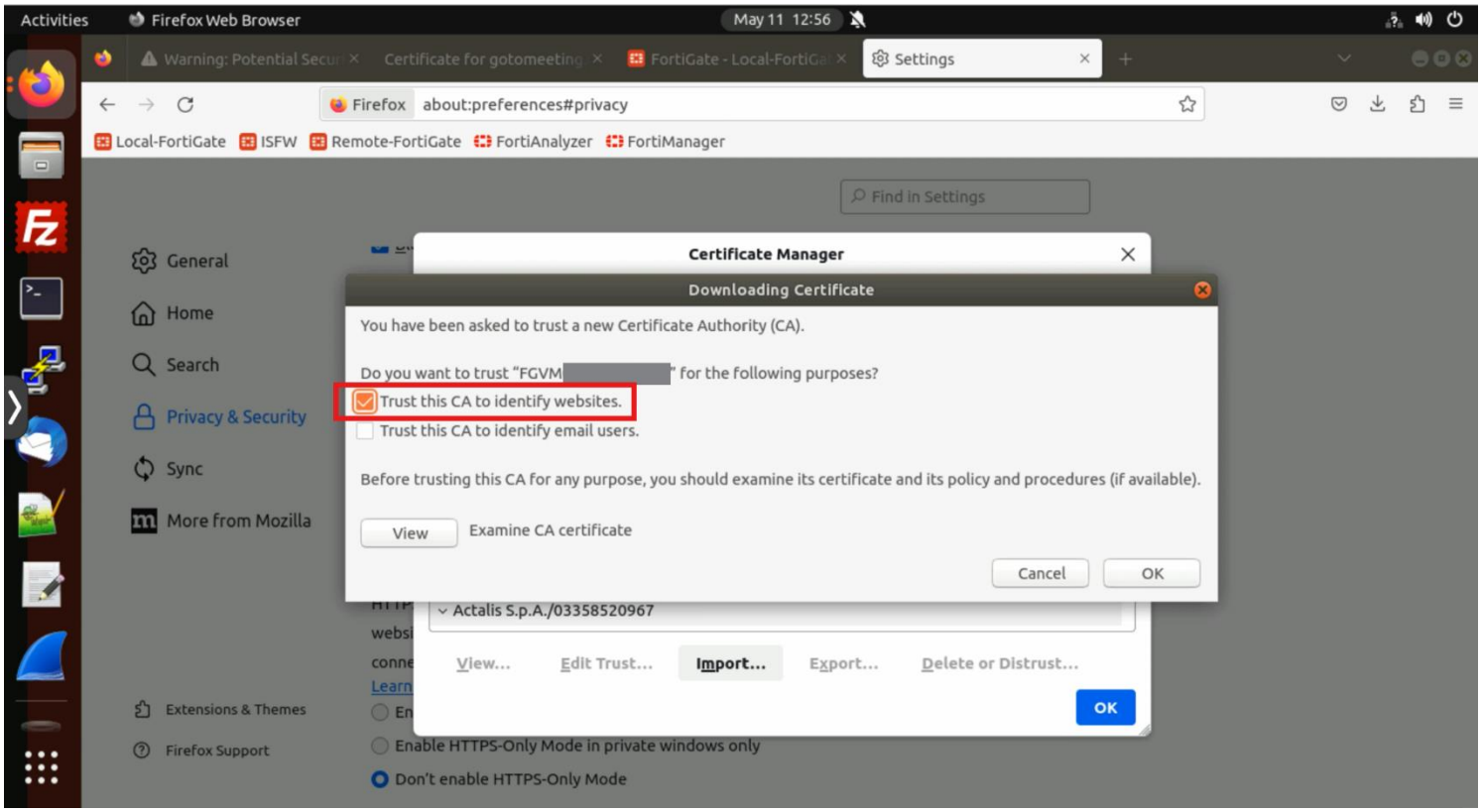


8. In the **Certificate Manager** window, click the **Authorities** tab.

You can see a list of certificates from the public CA. They are loaded by default in your browser.



9. Click **Import**.
10. In the **Downloads** folder, click `Fortinet_CA_SSL.cer`, and then click **Select**.
11. In the **Downloading Certificate** window, select **Trust this CA to identify websites**, and then click **OK**.



The **Fortinet_CA_SSL** certificate is added to the Firefox **Authorities** certificate store.

You can scroll down to see it in the list of authority certificates.

12. Click **OK** to exit the **Certificate Manager**.
13. Restart Firefox.

Test Full SSL Inspection

Now that you have imported the **Fortinet_CA_SSL** certificate into your browser, you will not receive certificate warnings when you access a secure website.

The CA that signed this certificate is not public, but your browser trusts it, because you added it as a trusted authority in the previous procedure.

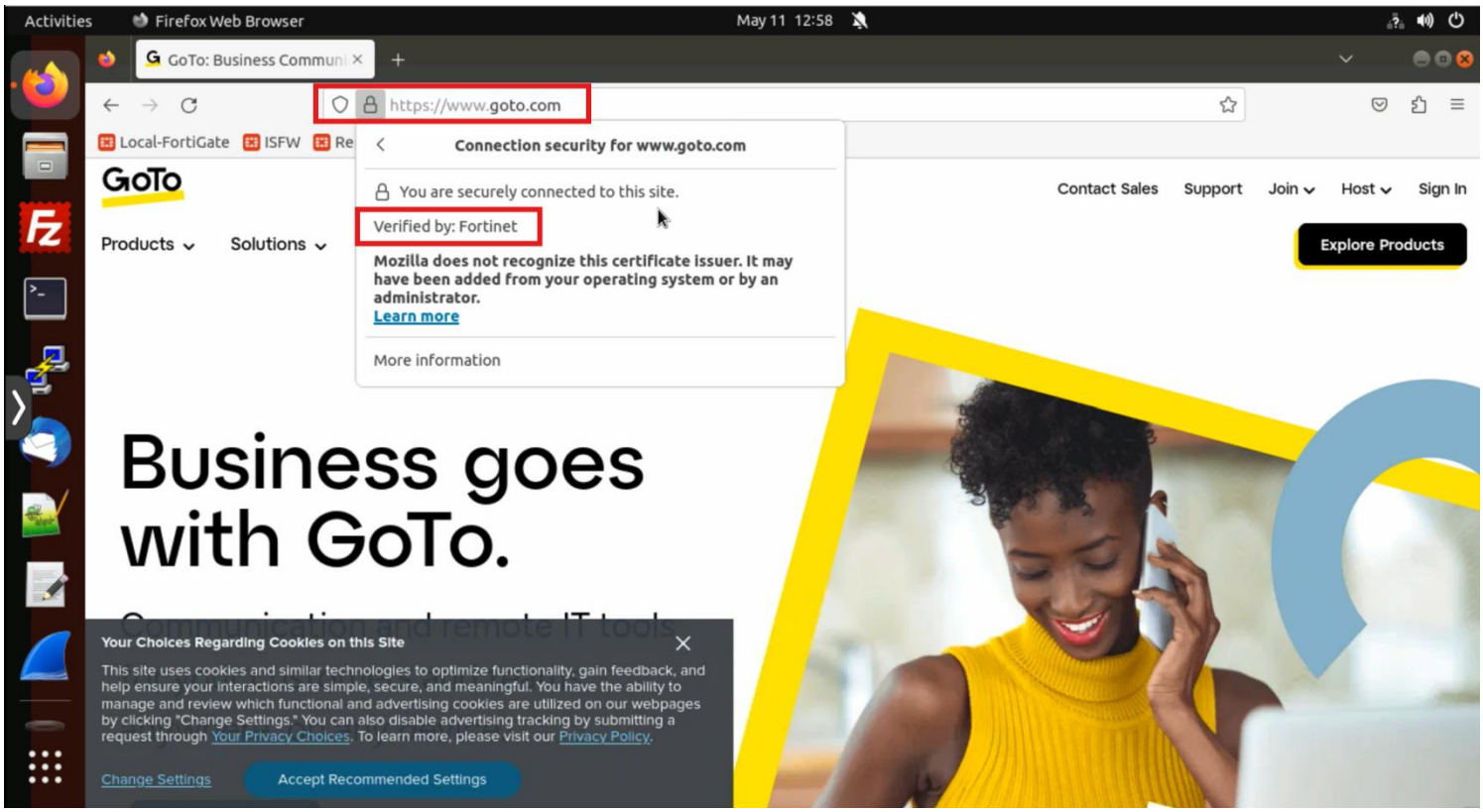
To test SSL full inspection

1. Continuing on the Local-Client VM, open a new browser session, and then go to a secure website, such as:

   https://www.goto.com

   This time, your browser opens the website without certificate warnings.

2. In the browser navigation bar, hover over the lock icon to see details.



You can see that the certificate is signed by Fortinet CA, and that the browser considers it valid.

3. Close the browser.

Exercise 2: Dealing with Anomalies

When you work with certificates, you might face some issues due to invalid or revoked certificates. You might also have to deal with restrictions that prevent the use of full SSL inspection.

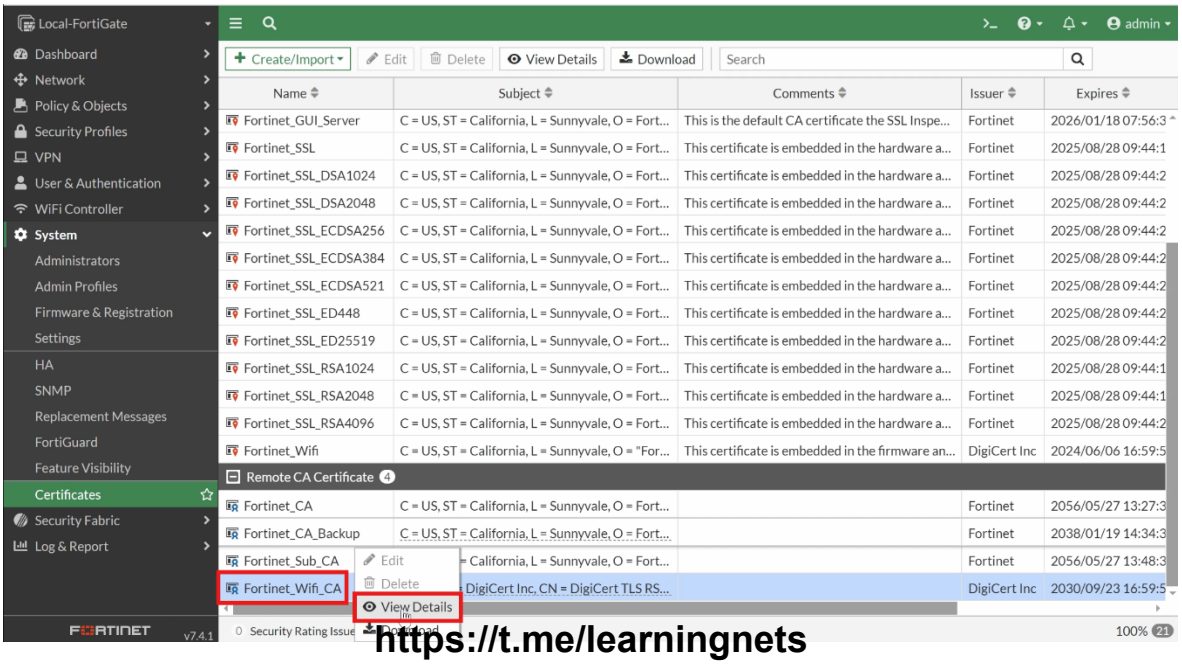
In this exercise, you will learn how to import a certificate revocation list (CRL) on the FortiGate GUI. Next, you will explore how FortiGate responds when it receives invalid certificates for traffic that match a deep inspection SSL profile. Finally, you will configure an exception to exclude a website from SSL full inspection.

Manage Invalid Certificates

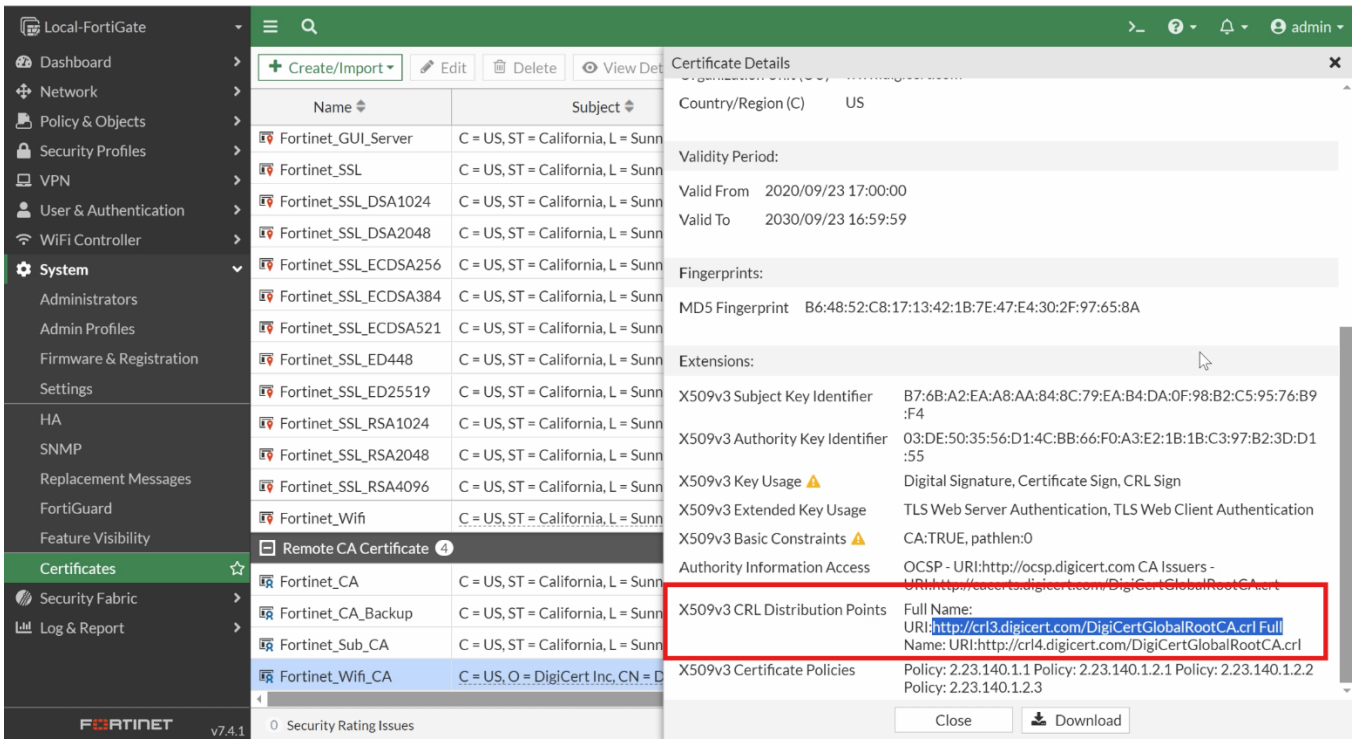
A certificate can be **invalid** because it expired or because the CA that issued it revoked it. A company might want to revoke a certificate because it was compromised, the key was lost, or, for example, because it was assigned to a user who left the company. To inform others of revoked certificates, CA administrators periodically publish CRLs. You will import a CRL.

To import a CRL

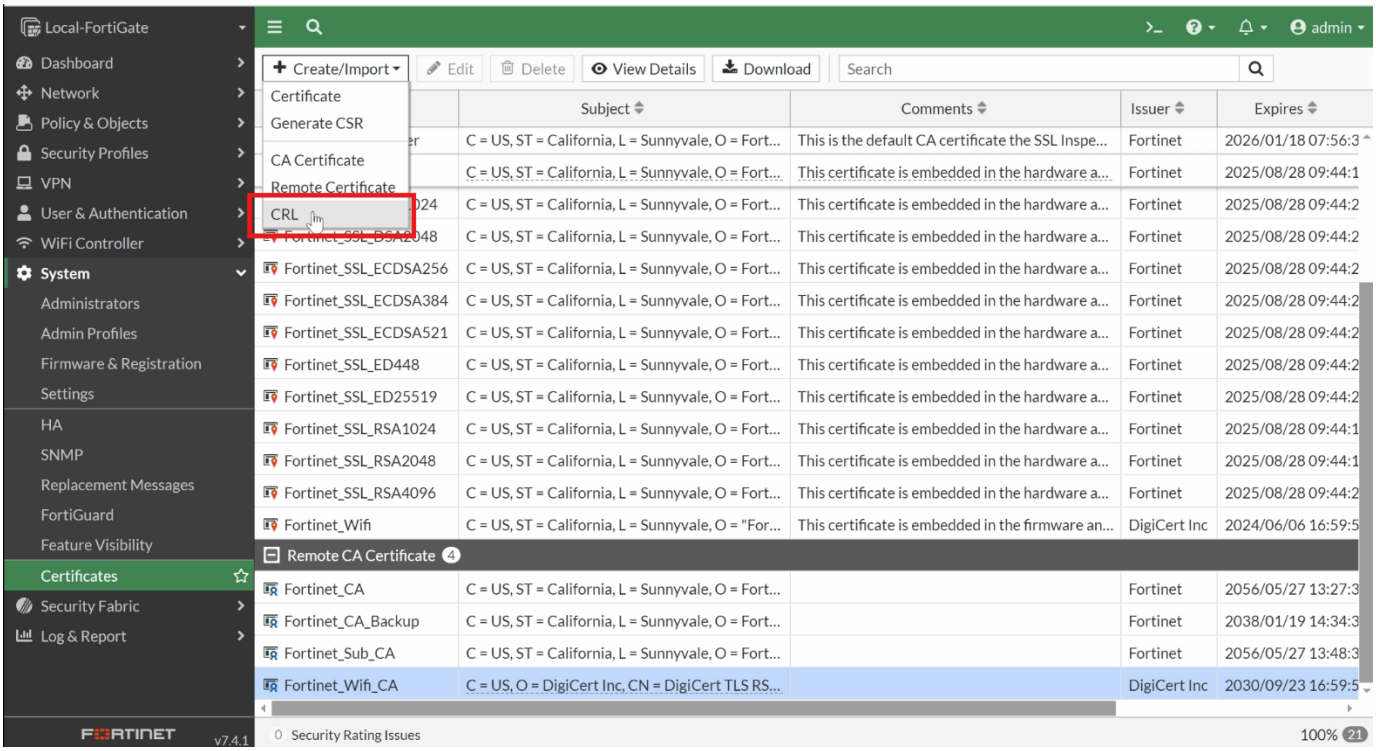
1. Connect to the Local-FortiGate GUI, and then log in with the username `admin` and password `password`.
2. Click **System** > **Certificates**.
3. In the **Remote CA Certificate** section, right-click the **Fortinet_Wifi_CA** certificate, and then select **View Details**.



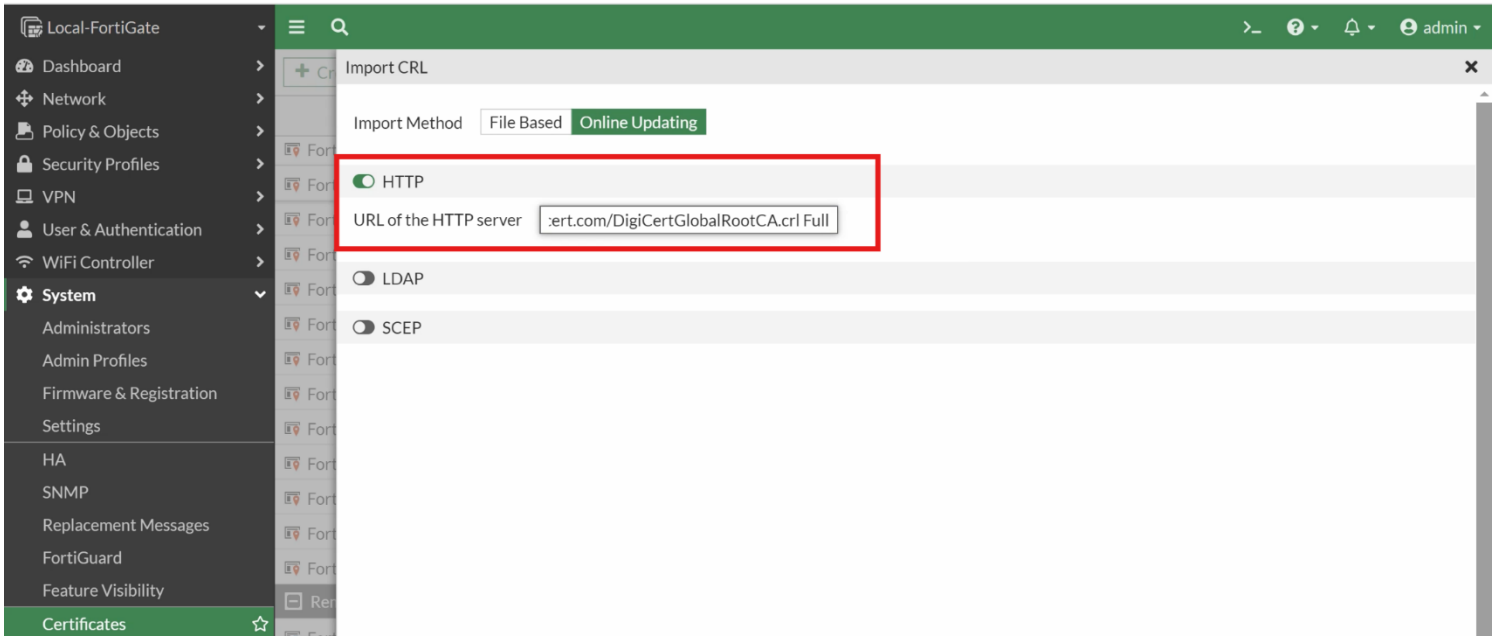
4. Scroll down to the **Extensions** section, and look for **X509v3 CRL Distribution Points**.
5. Highlight one of the URIs, and then press `Ctrl`+`C` to copy it for the distribution point.



6. Click **Close** to exit the **Certificate Details** window.
7. Click **Create/Import** > **CRL**.

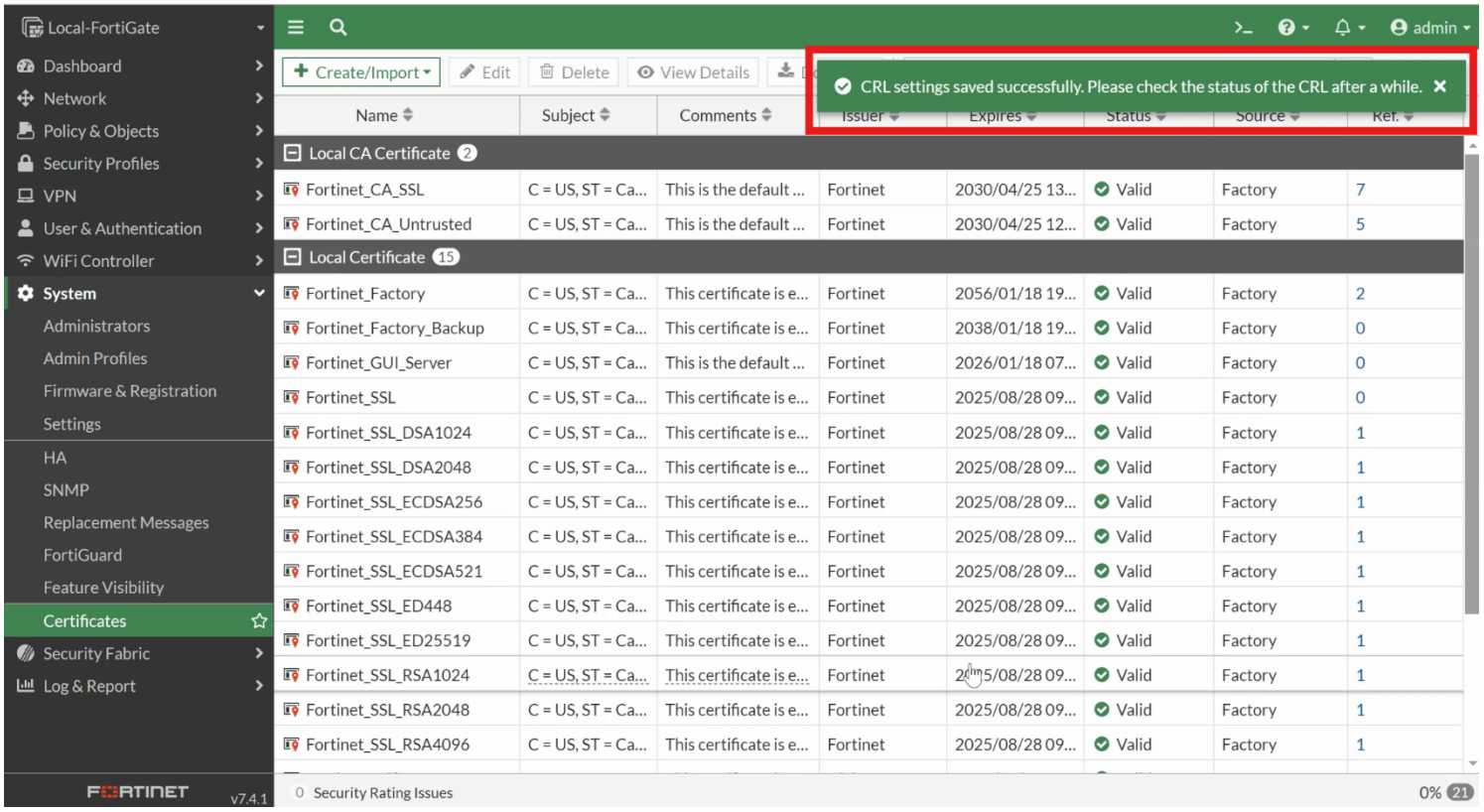


8. Enable **HTTP**, and then paste the URI of the CRL HTTP server that you just copied.



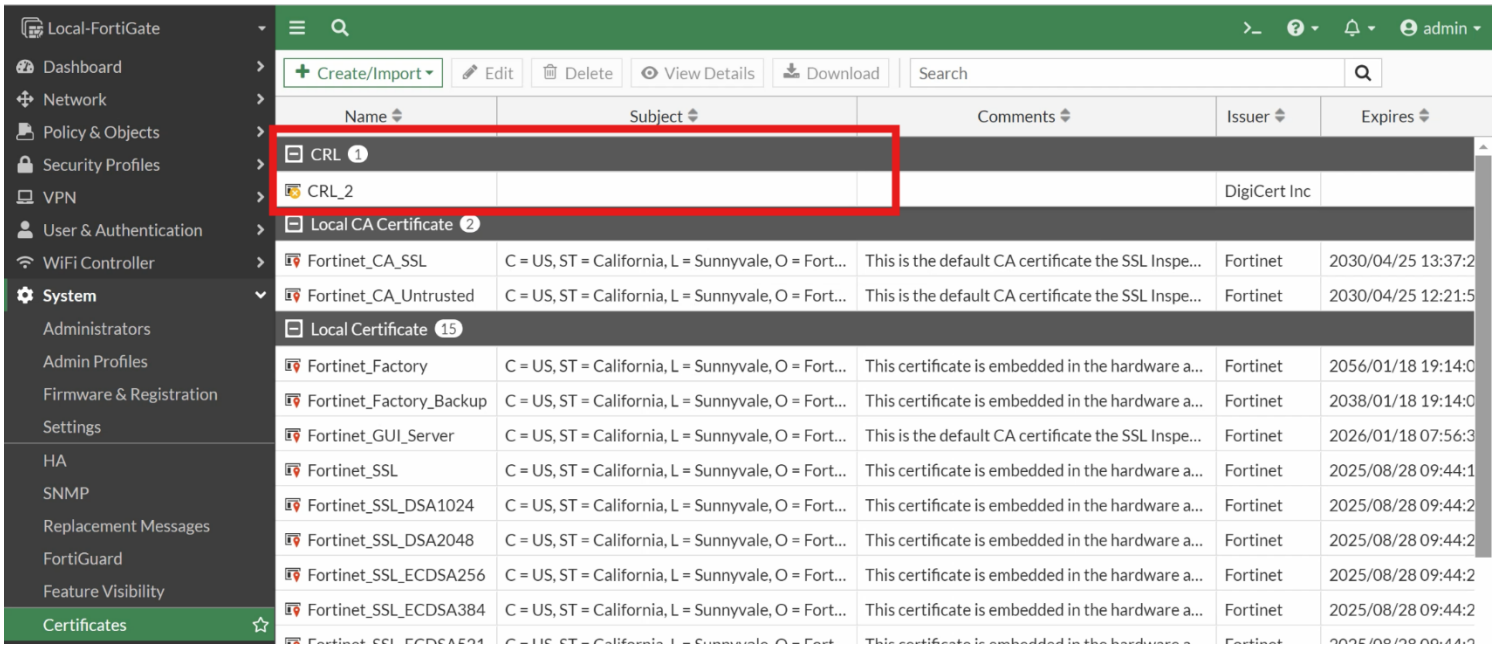
9. Click **OK**.

The FortiGate GUI briefly displays an acknowledgment message similar to the following example:



10. Wait a few seconds, and then click **System** > **Certificates** again to refresh the page.

The CRL section now includes the CRL you just added.



Note that you can load CRLs on the FortiGate only for a CA that FortiGate trusts. If you want to load the CRL that corresponds to your company CA, you must first load your company CA certificate on FortiGate.

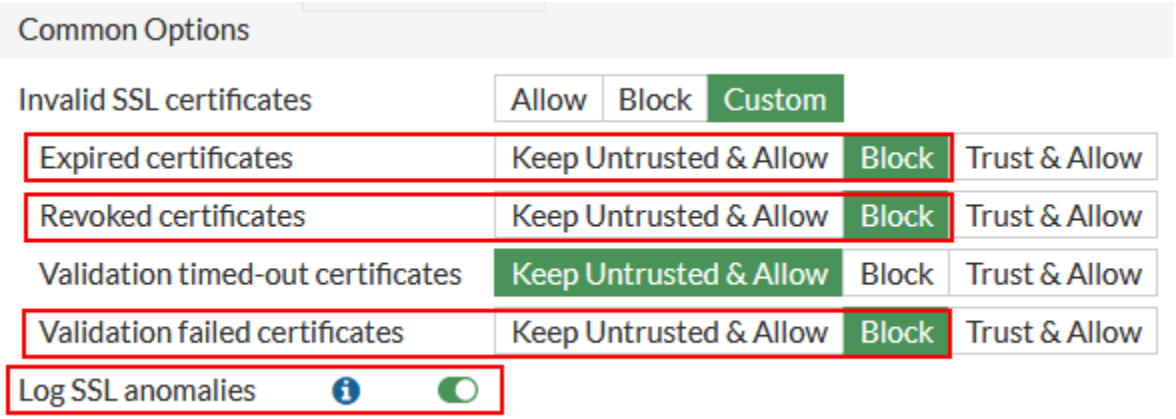
The Online Certificate Status Protocol (OCSP) is used for obtaining the revocation status of an X.509 digital certificate. It can be used as an alternative to CRLs. OCSP is disabled by default on FortiGate.

In this lab, we activated OCSP using the CLI commands shown below to receive certificate validation from well-known CAs that support OCSP.

```
config vpn certificate setting

set ocsp-option certificate

set ocsp-status enable

set strict-ocsp-check enable

end
```

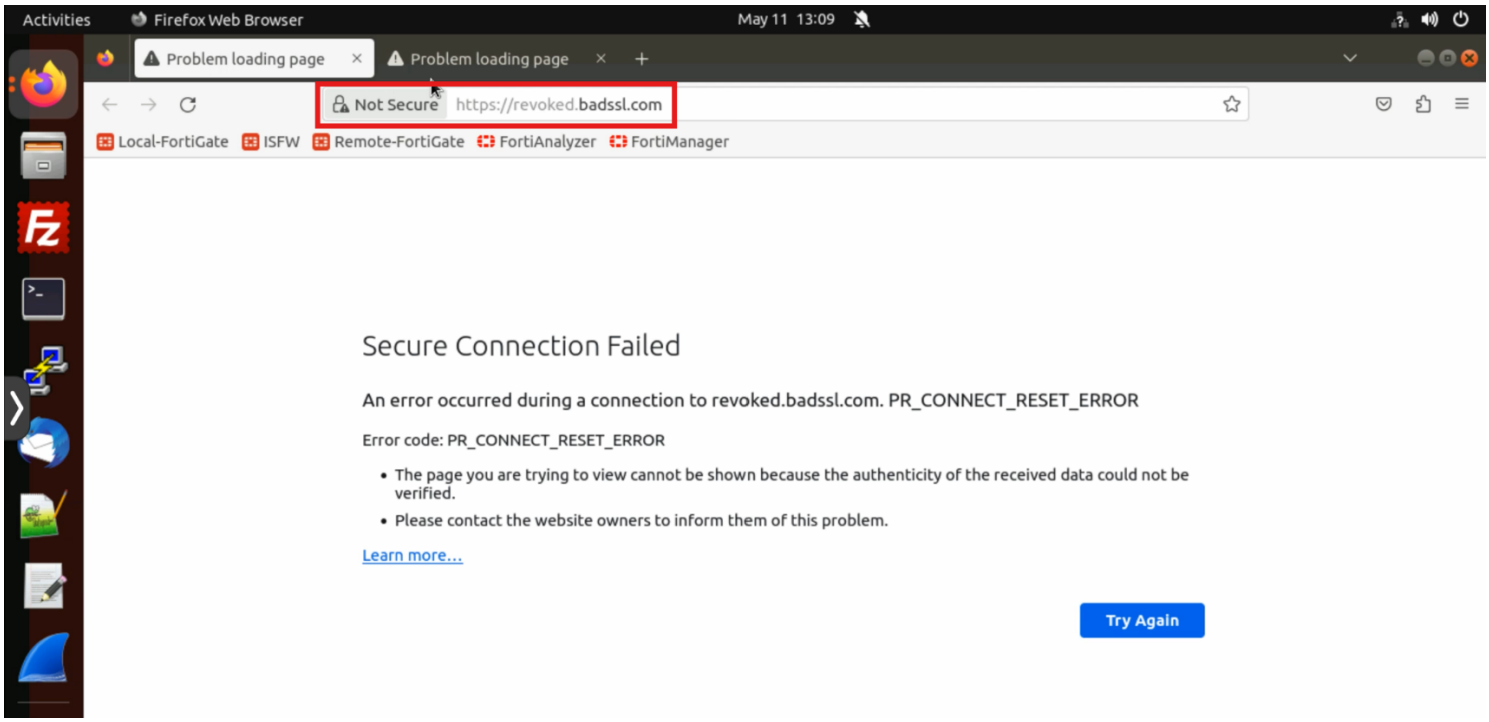
To block an invalid certificate with SSL full inspection

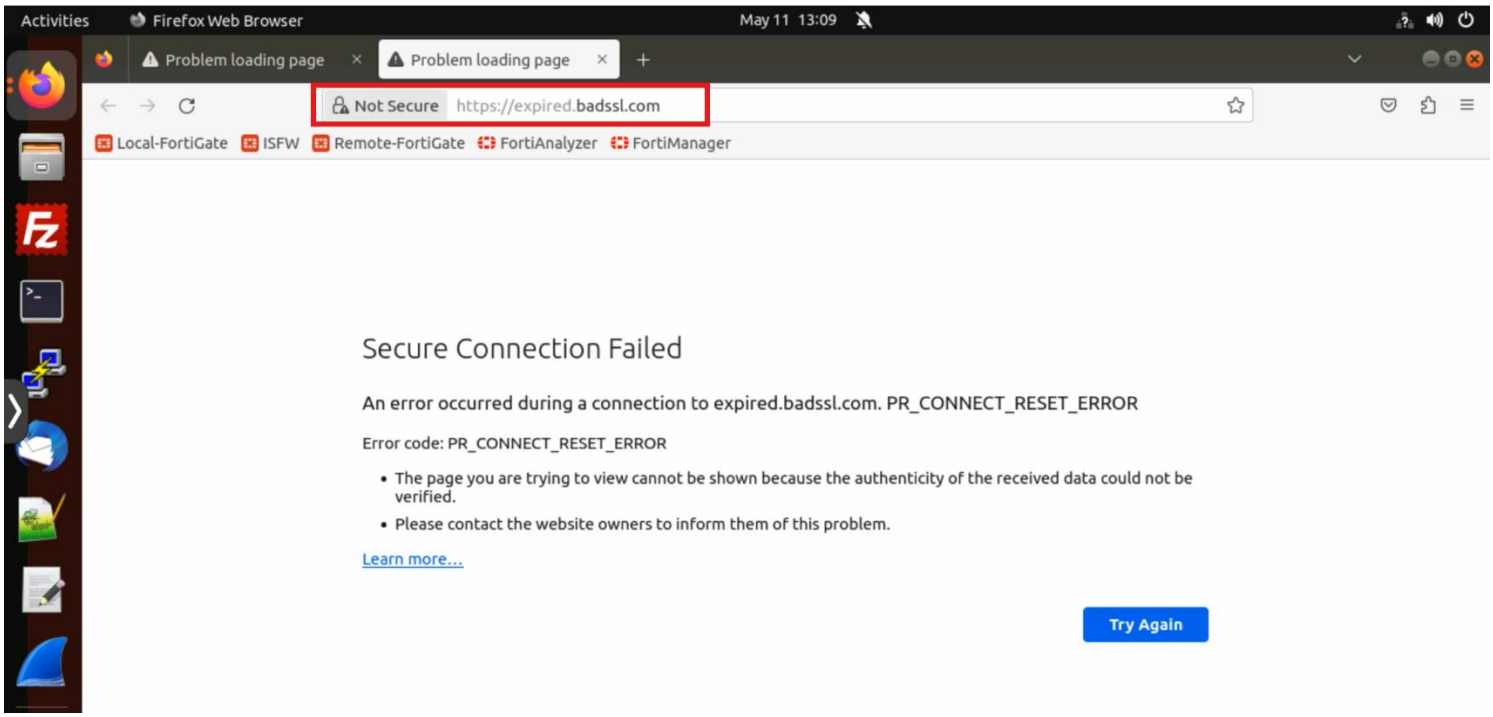
1. Continuing on the Local-FortiGate GUI, click **Policy & Objects** > **Firewall Policy**.
2. Select the **Full_Access** policy, and then click **Edit**.
3. Scroll down to the **Security Profiles** section, and then click the pen icon to edit the **SSL Inspection** profile **Custom_Full_Inspection**.
4. Confirm that the settings are the same as what is shown in the following image:



5. Do not make any changes, and then click **Cancel** to exit the SSL inspection profile menu.
6. Click **Cancel** to exit the policy configuration menu.
7. Connect to the Local-Client VM, and then log in with the username `Administrator` and password `password`.
8. Open a browser, and then visit `https://revoked.badssl.com/`.
9. In another browser tab, visit `https://expired.badssl.com/`.

FortiGate blocks access to the website and the browser displays a warning message similar to the following image:

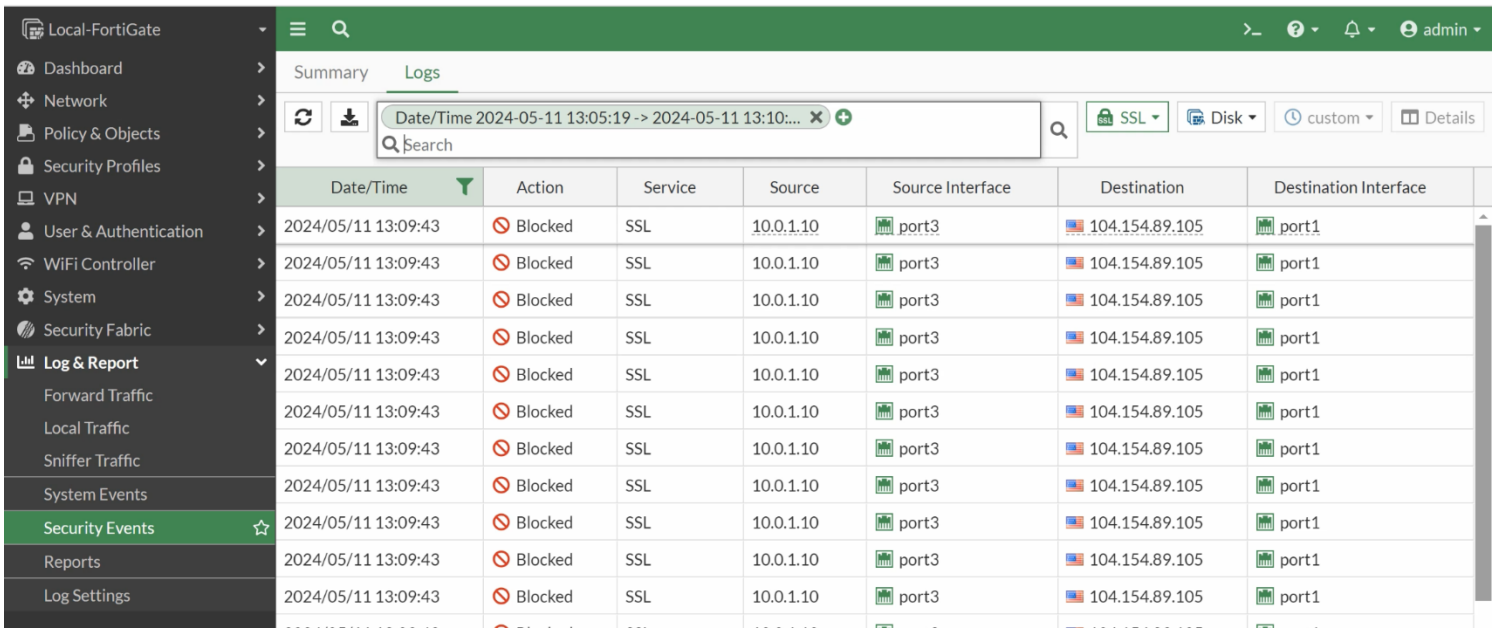




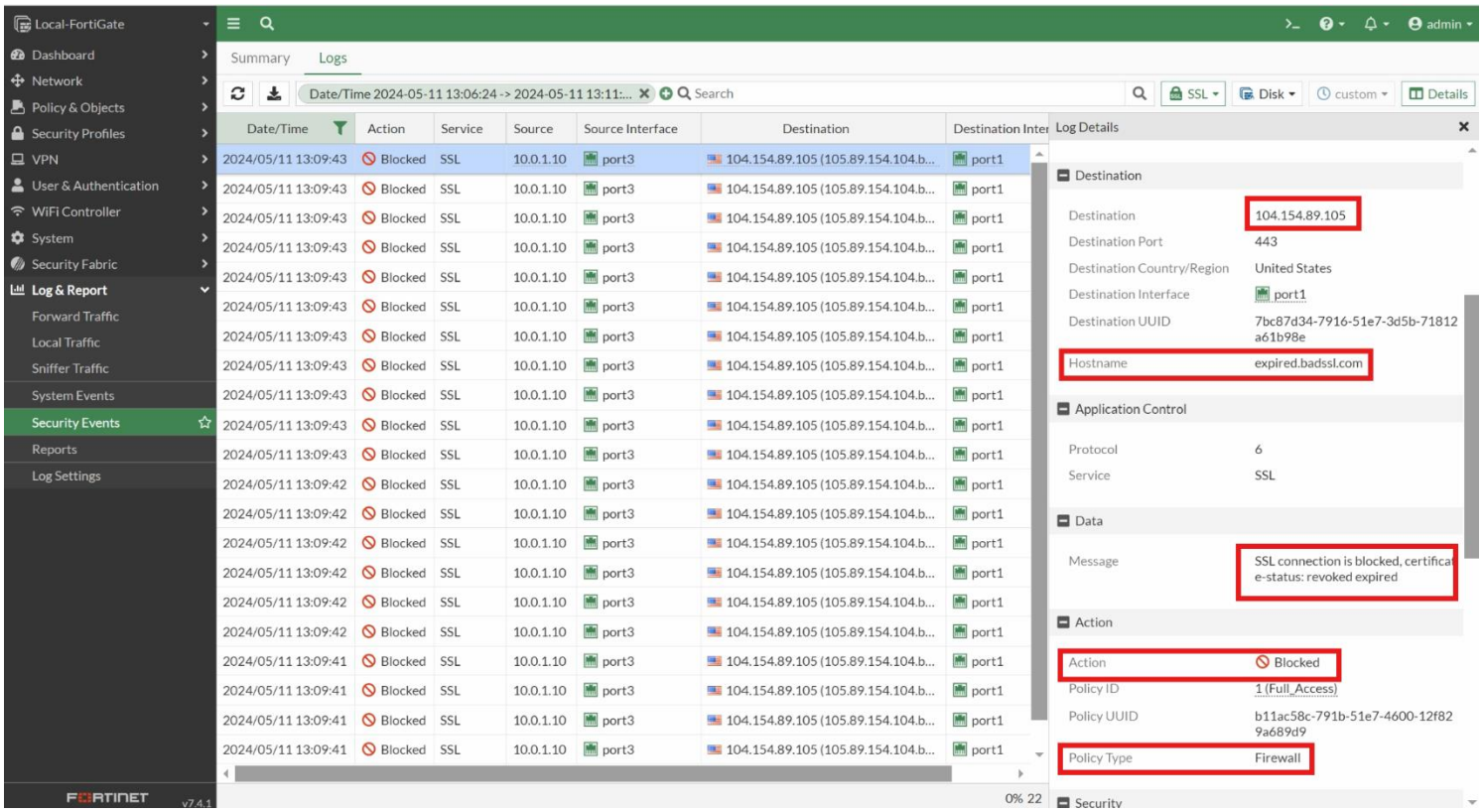
To review SSL log messages

1. Continuing on the Local-FortiGate GUI, click **Log & Report** > **Security Events**.
2. Expand the **SSL** widget to display the log list.

You can see that FortiGate blocked access to the website.



3. Double-click a log message to review the details.



You can see that the log message is similar for expired and revoked certificates.

Allow Exceptions to SSL Full Inspection

When replacing a certificate prevents users from accessing some websites, you can define exceptions and exclude some websites from full SSL inspection. You can also exclude some **websites** or **website categories** from full SSL inspection for legal reasons. For example, in some countries, it is forbidden to perform deep inspection on traffic between users and financial institution servers.

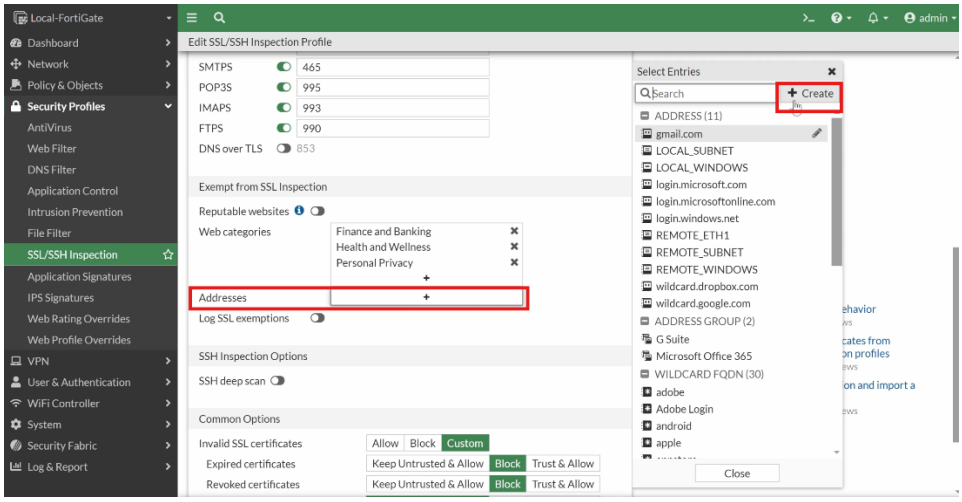
You will add an exception to the SSL/SSH deep inspection profile that you have already configured.

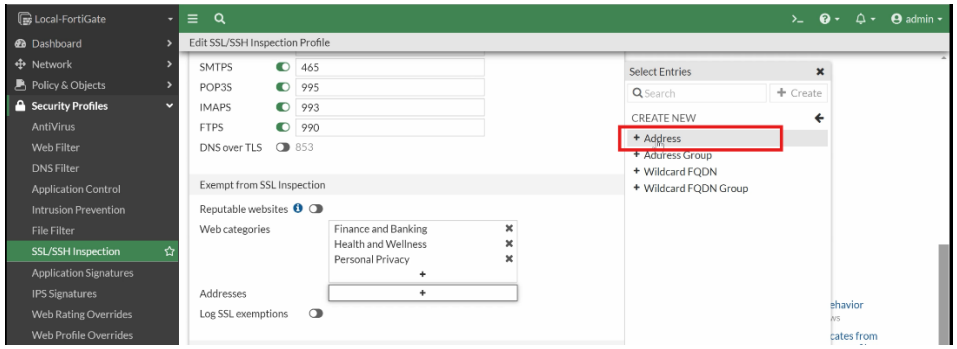
To configure a site exception to SSL full inspection

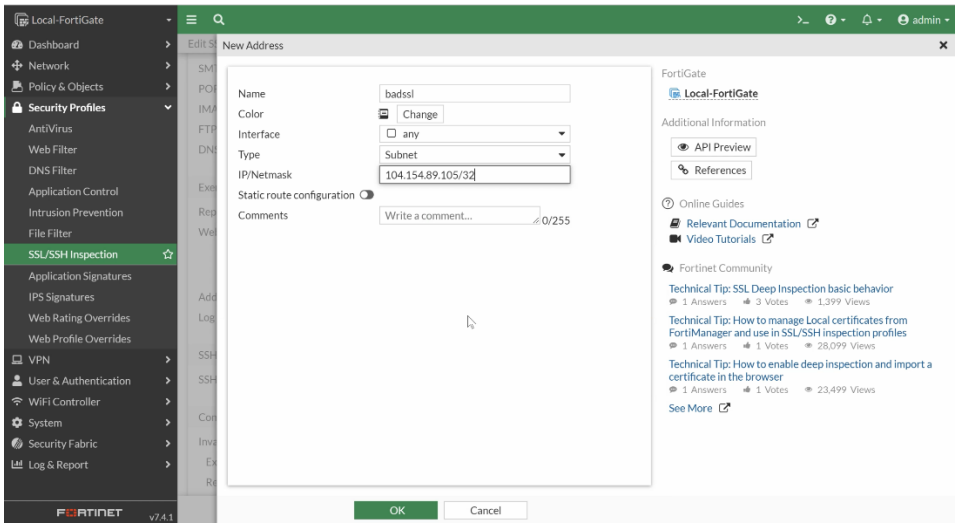
1. Connect to the Local-FortiGate GUI, and then log in with the username `admin` and password `password`.
2. Click **Security Profiles** > **SSL/SSH Inspection**.
3. Edit the **Custom_Full_Inspection** profile.
4. In the **Exempt from SSL Inspection** section, click **+** to create new **Addresses**.
5. Create a new address object with the following parameters:

New Address	Value
Name	badssl
Type	Subnet
IP/Netmask	104.154.89.105/32

6. Click **OK**.
7. In the SSL/SSH inspection profile, select the newly created address object **badssl**.
8. Click **OK** to save the configuration change of the SSL/SSH inspection profile.
9. Click **Policy & Objects** > **Firewall Policy**.
10. Edit the **Full_Access** policy and set **Custom_Full_Inspection** as the SSL inspection profile.
11. Click **OK**.







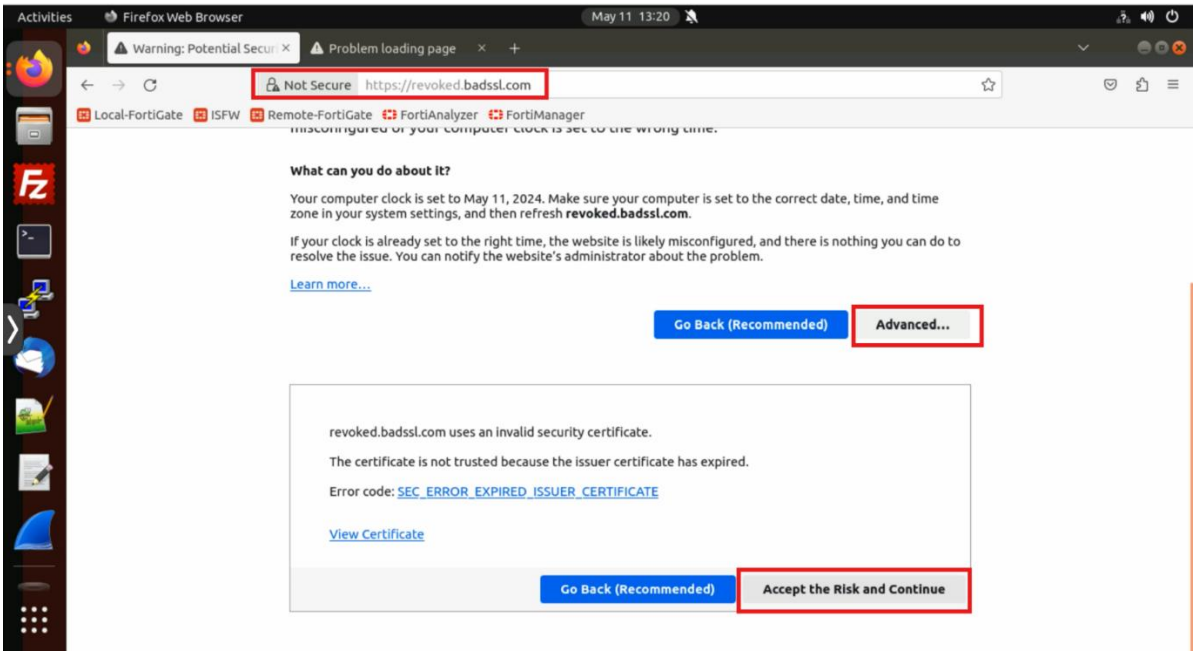
To check the SSL full inspection exception

1. On the Local-Client VM, navigate to one of the websites you tested previously:

- https://revoked.badssl.com/.
- https://expired.badssl.com/.

2. Click **Advanced**, and then click **Accept the Risk and Continue** to accept the browser warning.

Now, you can visit the website.



Usually, you will configure SSL full inspection exceptions only for websites that do not support MITM and that your company trusts. Those websites should have a valid certificate and therefore do not trigger a browser warning.