Security, Encryption, and Certificates FAQ

Overview

In Security Center 5.4, several new capabilities will be added that further strengthen the security of the platform itself, as well as the privacy of data. The aim is to prevent unauthorized access to stored and transmitted messages and data, as well as prevent attacks through the use of stronger encryption and authentication mechanisms. With growing demand for privacy, new capabilities in Security Center 5.4 will strengthen Genetec's position and overall value proposition.

This FAQ addresses some of the most common questions in relation to the new capabilities of Security Center: Encryption, Authentication, and Digital Certificates. These concepts are first described in generic terms; the FAQ then outlines how these new measures are used within Security Center 5.4.

Encryption vs. Authentication vs. Authorization

What is the difference between encryption, authentication, and authorization?

- **Encryption** is used to encrypt data so that only authorized users can see it.
- **Authentication** determines whether an entity is who they claim to be, eg. in the case of an individual, it is usually based on a username/password combination. Authentication does not actually say anything about what someone is authorized to do or has the right to do.
    - Client-side authentication uses username/password combinations, tokens (dual authentication), and other techniques.
    - Server-side authentication uses certificates to identify trusted third parties.
- **Authorization** is the function of specifying the rights, eg. defining the access rights someone has over a set of recourses such as a private data, computing resources, or an application. When users log into a Security Center system, what they are allowed or authorized to do depends on the set of privileges assigned to them by administrators.

Security system users need to be first authenticated (eg. during login process) and then given specific access rights (authorization) to take action within the system.

Encryption

What is encryption?

Encryption is the process of encoding data that is at rest or in transit to prevent unauthorized access and that only the right entities can see it. Encryption hides data or the contents of a message in such a way that the original information is then recovered through a corresponding decryption process. This is achieved by taking the original data and mathematically encoding it using an encryption key.

Why is encryption needed?

Encryption helps protect private information, sensitive data, and enhance the security of outgoing data or data in transit. Encryption hides sensitive data from people who should not be able to see it, and ultimate purpose is to protect the confidentiality of digital data stored on a computer or communicated over a network.

How does encryption work?

An encryption algorithm is used to encrypt the data that is to be communicated or stored (also known as plaintext) into unreadable or encrypted data. Encrypted data (also known as ciphertext) can only be read once decrypted. For example, at the sender's end of the conversation an encryption key is used to encrypt the data to be communicated, while a decryption key is used at the receiving end to decode encrypted data.

What are the different types of encryption schemes or algorithms?

The are two different types of encryption algorithms: Symmetric and Asymmetric. Symmetric encryption uses the same private encryption key for encrypting (encoding) and decrypting (decoding), while asymmetric uses a private and public key pair.

What is Private Key Cryptography?

In a symmetric-key or private key encryption scheme, the encryption and decryption keys are the same. Communicating parties must have the same key to enable secure communication, or the same key is used to encrypt and decrypt stored data. An example of a symmetric-key algorithm is AES.

Pros: Fast, easily implemented by hardware, commonly used for bulk data encryption.

Cons: Complications with distribution and control of private keys, eg. anyone with your key can decrypt your messages even if it wasn't intended for them.

What is Public Key Cryptography?

In a public key or asymmetric encryption scheme, two separate but mathematically linked encryption keys are used; one key is a private key while the other is a public key. The public key is used to encrypt data and can be distributed, while the private key is used to decrypt the data and must be kept private. This process also works in the other direction, eg. a private key is used to encrypt data that will then be decrypted by a public key. Messages encrypted by a public key cannot be decrypted by a public key. An example of an asymmetric-key algorithm is RSA.

Pros: Eliminates the preliminary exchange of secret keys, public keys can be shared with anyone, provides the underlying architecture used in digital certificates, digital signatures, and Public Key Infrastructure or PKI (see below)

Cons: Much slower than private key encryption, requires greater computational power.

What is encryption strength?

Encryption strength measures the effectiveness or strength of an encryption algorithm based on the number of bits of the encryption key used by the algorithm. Also referred as key size or key length, the strength of an algorithm increases with the bit length of the key used. Common examples of specifying encryption strength will include the number of bits in the key, eg. 128-bit SSL encryption or AES-256.

How is encryption used in communications?

Protocols such as TLS (Transport Layer Security) and its predecessor SSL (Secure Sockets Layer) are cryptographic protocols used to provide secure communications over a network. Both use certificates and asymmetric cryptography to authenticate the counterpart in a conversation and then negotiate a symmetric session key. The symmetric session key is then used to encrypt data during the conversation.

TLS allows for data and message confidentiality as well as message authentication. Versions of TLS and SSL are used in applications such as email, instant messaging, and web browsing.

Digital Certificates and Digital Signatures

Is encryption enough to protect the confidentiality, integrity, and authenticity of a message?

Although encryption may hide the contents of a message or the confidentiality of a message, it may be possible to change an encrypted message without knowing its contents, thereby modifying the message's integrity. Additional techniques are needed to protect the integrity (message has not been altered) and authenticity (sender is who they say they are) of a message. One such technique is the use of digital signatures;

An example is a man-in-the-middle-attack where the attacker intercepts, actively eavesdrops, and possibly alters the communication between two parties who believe they are communicating with each other over a private connection. The conversation is in essence controlled by the attacker and the attacker has successfully convinced communicating parties that they are in a private conversation. In other words, the client application believes they are communicating with a legitimate server, when in fact they are not (or vice versa).

What is a digital signature?

A digital signature allows a recipient to establish whether a message was created by a known sender (authentication), that the message was not altered in any way during transit (integrity), and that the sender cannot deny having sent the message (non-repudiation). A digital signature is therefore an authentication mechanism that acts as the equivalent of a handwritten signature, and that is attached to the message sent by a sender.

A digital signature scheme typically consists of three algorithms:

- A key generation algorithm that generates a private key and a corresponding public key.
- A signing algorithm that produces a signature for a specific message and private key.
- A signature verifying algorithm that access or rejects the message's claim to authenticity based on the message, the public key, and the digital signature.

The authenticity of a digital signature can be verified using the public key for a signature generated from a message and a private key, AND it should be computationally infeasible to generate a valid signature for a party without knowing that party's private key. Digital signatures are used in some types of digital certificates, which are discussed next.

What is a Digital Certificate?

A public key certificate or digital certificate, is an electronic document that is used to prove the ownership of a public key. A certificate includes information about the public key, information about the owner's identity, and the digital signature of an entity that attests to the correctness of the certificate's contents. If a signature is deemed valid and the person or entity examining the certificate trusts the signer, then they know they can use that public key to communicate with its owner. By certifying the ownership of a public key by the named subject of the certificate, the digital certificate can help verify whether a sender is who they claim to be.

A certificate establishes authenticity by guaranteeing that the data it contains cannot be forged. Once trust is established, the information in the certificate will confirm that we are communicating with the right entity or person.

What is the role of the signer?

In a model of where a trust relationship is established, the entity that verifies the certificates contents is known as the signer. In a Public Key Infrastructure scheme, the signer is a Certificate Authority (CA), an entity or company that charges customers a fee to issue digital certificates for them. In this relationship of trust scheme, the signer is trusted by the entity examining the certificate, and they know they can use that specific key to communicate with its owner. The CA is known the trusted third party, trusted by both the owner of the certificate and the party relying on the certificate.

Examples of companies that issues digital certificates are Comodo and Symantec (formerly VeriSign). Users can also issue self-signed certificates, without the need of going through a CA.

How are digital certificates issued?

Digital certificates can be issued in a variety of ways:

- A Certificate Authority (CA) can issue certificates.
- Operating systems have embedded tools to create certificates. For example, Windows Server has the ability to create certificates through Active Directory Certificate Services. Certificates can then be issued to Windows users and Windows-based servers and computers.
- Utilities are also available for creating unmanaged certificates, eg. from Microsoft utilities such as SelfSSL.exe. With a self-signed certificate, the certificate is signed by the same entity whose identity it certifies.

How are certificates used?

One of the most common uses of certificates is for HTTPS-based web sites. A web browser will validate that a web server is authentic so that the user feels secure in that communications with the web sites are protected and the web site is who it claims to be. In this scenario, certificates are used by the TLS protocol to prevent attackers from impersonating a secure website.

Another use is during the encryption of email messages that rely on public key cryptography and authentication: each user can publish a public key that others can use to encrypt messages to them (digital signing and message encryption using certificates).

What is a Public Key Infrastructure (PKI) scheme?

PKI is a set of software, hardware, procedures and policies used to manage, distribute and store digital certificates, as well as manage public key encryption. PKI facilitates the secure transfer of information over public and private networks and addresses the issue of large-scale distributed authentication. It allows organizations to move away from inadequate authentication (eg. relying solely on passwords) to more rigorous authentication that requires confirmation of the identity of the parties involved in the communication and the validation of information being transferred.

Claims-Based Authentication

What is Claims-Based Identity or Authentication?

Claims-Based Identity is used by applications to acquire the identify information they need about users inside or outside their organization. It simplifies authentication process for applications because it allows applications to know certain things about the user without having to interrogate the user. The facts (or claims) are then transported in an envelope called a secure token.

What is a Claim?

A claim is a statement that one subject makes about itself or another subject. The statement can be about a name, identity, key, group, privilege, or capability, for example. The subject making the claim is the provider (issuer). Claims are issued by a provider, are given one or more values, and are then packaged in security tokens that are created by an issuer, commonly known as a Security Token Service (STS). An STS is a trusted third party that can often better handle the identity claims than an individual application.

Claims are not what the subject can or cannot do (authorization). They are what the subject is and is not (authentication). They work like certificates.

What is an example of Claims-Based Authentication?

A good analogy for claims-based identity is the authentication that happens at an airport:

> *You can't simply walk up to the gate and present your passport or driver's license. Instead, you must first check in at the ticket counter. You present whatever credential makes sense: for overseas travel, you present your passport. For domestic travel, you present your driver's license. After verifying that your picture ID matches your face (authentication), the agent verifies that you have purchased a ticket for a specific flight (authorization). You receive a boarding pass to be used at the gate.*
>
> *With the boarding pass, the gate agent knows your name and frequent flyer number (authentication and personalization), your flight number and seating priority (authorization), and perhaps even more. The gate agent has everything that he or she needs to do their job efficiently. There is also special information on the boarding pass. It is encoded in the bar code and/or the magnetic strip on the back. This information (such as a boarding serial number) proves that the pass was issued by the airline and is not a forgery.*
>
> *The boarding pass is a signed set of claims made by the airline about you. It states that you are allowed to board a particular flight at a particular time and sit in a particular seat. Of course, agents don't need to think*

very deeply about this. They simply validate your boarding pass, read the claims on it, and let you board the plane.

It's also important to note that there may be more than one way of obtaining the signed set of claims that is your boarding pass. You might go to the ticket counter at the airport, or you might use the airline's web site and print your boarding pass at home. The gate agents boarding the flight don't care how the boarding pass was created; they don't care which issuer you used, as long as it is trusted by the airline. They only care that it is an authentic set of claims that give you permission to get on the plane.

How is Claims-Based Authentication used by Applications?

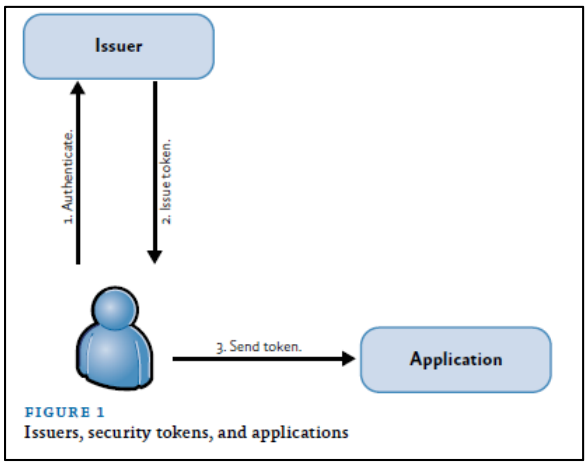


FIGURE 1
Issuers, security tokens, and applications

In software, a bundle of claims is called a security token. Each security token is signed by the issuer who created the token. A claims-based application considers users to be authenticated if they present a valid, signed security token from a trusted issues. In other words, instead of providing credentials, a client app can provide claims (token) for authentication purposes.

Figure 1 shows the basic pattern for using claims. The user presents a token to the application (provided by the issuer) and is then considered authenticated by the application. The user can now access the application but his or her access rights will be limited to what they are authorized to do.

Third Party Claims Providers

One of the benefits of claims-based authentication is that an application can use third party claims providers. For example, applications (incl. web applications and mobile apps) can use more internet-friendly authentication techniques with a common example being the use of your LinkedIn/Facebook/Windows Live account to log into an application. By supporting this, the application trusts that LinkedIn has confirmed you are who you say you are through trust.

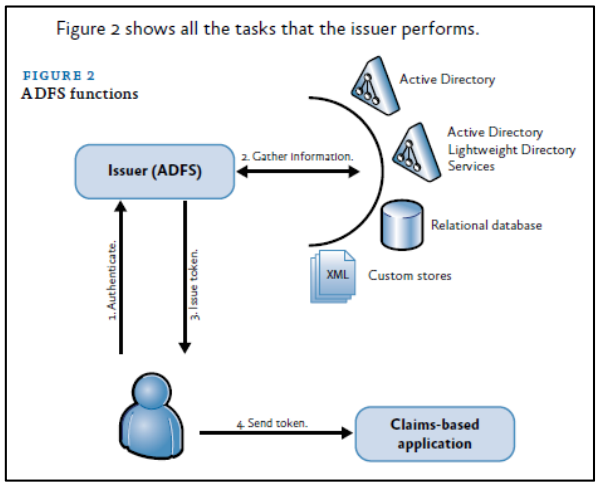
As these large providers offer well-established systems for authenticating users, it removes the burden for applications and services to develop and maintain their own identity system. Vendors such as Microsoft and Google that operate many different web services have moved to consolidate their identity services, and have invested significant resources, well beyond the means of most application developers, to ensure their security and integrity. Smaller vendors can take advantage of these systems to vet and identify users and claims, without the need to build and maintain their own repository of user information, or continue to invest in keeping their systems up-to-date to counter the latest security threats. Users of these systems benefit by only having a single sign-on required to access multiple applications and/or services.

Another example of claims-based authentication through third party claims providers is the use of Active Directory Federation Services or ADFS.

Active Directory Federation Services (ADFS) as a claims issuer

Microsoft ADFS can be installed on a Windows Server and act as a claims issuer. ADFS provides the logic to authenticate users, including authentication through certificates. ADFS can also be configured to accept security tokens from an issuer in another realm as proof of authentication. This is known as identity federation and is how you can achieve single sign-on across realms. In identity terms, a realm is the set of applications, URLs, domains, or sites for which a token is valid. A realm is sometimes described as a security domain because it encompasses all applications within a specified security boundary.

In Figure 2, ADFS authenticates a user, then creates claims about the user by extracting attributes from external Active Directory servers, and finally issues a security token to be used by a claims-based application.



The Need for Enhancements in Security

Why is there is a for security enhancements?

There are several reasons why security enhancements are required in day-to-day security operations:

- In many instances, **passwords are inadequate** as they can be easily deduced or cracked. There are shareware and freeware utilities commonly available whose sole purpose is to derive passwords and given the fact that most users tend to choose easy to remember passwords, the need for security beyond username/password combinations is a must. Although users can rely on more complex passwords, they are easier to forget which can result in downtime and additional IT support.
- Although encryption may hide the contents of a message, **encryption alone does not necessarily preserve the integrity of a message.** It may be possible to change an encrypted message without knowing its contents, thereby modifying the message's integrity. Encrypted communications and encrypting data at rest is a good step forward, but additional techniques such as stronger authentication and/or claims-based authentication.
- The **growing interconnectivity** between security systems over the Internet as in the case of distributed facilities, as well as multi-organization access to security systems as in the case of public-private entity cooperation and **connectivity both inside and outside an organization**, means that companies are more exposed to eavesdropping or man-in-the-middle attacks than ever before. Claims-based authentication or claims-based identity are just some of the technologies used to prevent this type of attack.
- Security systems such as video and access control have traditionally focused on securing people and assets. Given the **critical nature of a security system**, end users must now take into account potential risk of attack on the security platform itself, not just their assets.

New Security Capabilities in Security Center 5.4

What new security capabilities are supported in Security Center 5.4?

- **Advanced Communications Encryption and Certificates**.
  - With enhanced and more secure communications, Security Center 5.4 will now allow users to load and select certificates on the Directory server, or use a default self-signed certificate. Users can use the default certificate created on installation or select their own certificate, either a self-signed certificate or a certificate generated by a trusted third party in accordance to their IT security policy, eg. Comodo or Symantec (formerly VeriSign).
  - TLS is now used to open secure communications channels between various components of Security Center. In previous versions of Security Center, a version of SSL (the predecessor of TLS) was used for encrypted communications.
  - TLS leverages both encryption to ensure confidentiality and certificates to authenticate entities.
    - Client to Directory communications will now use TLS. A client app (eg. Security Desk) can now confirm they are really talking to the Directory through the use of certificates, after which they will exchange symmetric keys for encrypted communications. Encryption keys will change dynamically on a periodic basis.
    - During the initiation of a connection, the Directory will send its certificate to the client for validation.
    - Extension Server to Directory communications will also use TLS.
- **Advanced Stream Encryption (Fusion Stream Encryption)**
  - Streams (video, audio, and metadata) can now be optionally encrypted by the Archiver. To configure encryption, the user will need to provide a certificate containing a public key.
  - Security Center supports two (2) levels of encryption
    - Streams (video, audio, overlays) are encrypted with a symmetric key (AES-128) that changes periodically.
    - The key stream (sequence of symmetric keys used to encrypt the data streams) is encrypted with the public key of the certificate(s) used when configuring encryption for the camera (RSA). The key stream is therefore encrypted in such a way that it can only be decrypted by the client workstation. Security Center is actually sending an encrypted stream of encryption keys.
  - When the camera is configured, the certificate is sent from the workstation where the configuration is made to the Directory. The Directory forwards the certificate to the Archiver so that it can encrypt the key streams. The private key never leaves the workstation.

- Encryption can be turned on/off on a per Archiver or per camera basis. Video streams sent to client applications or between Security Center servers will be encrypted. Servers will not be able to decrypt the data, adding a further layer of security.
- Client applications must have at least one valid certificate installed locally on the workstation, including both the private and public keys, to view encrypted video. When configuring encryption, the public key contained in the certificate is sent to the Directory; the server never needs to know or have access to the private key.
- Although the Auxiliary Archiver does not encrypt video, it does support the archiving of encrypted streams along with the associated key streams. One way to visualize this is to see the Auxiliary Archiver as a media player that requests all possible key streams.
- Video export:
  - Export of G64x which will include the key streams so that it can be played back on a workstation with the certificate(s) installed.
  - Decrypted video transcoded in ASF. A new privilege will be required to allow the removal of encryption upon export; requires the user to be logged into the Directory since this new privilege cannot be validated when offline.
- Limitations: Multicast from the unit, playback on Security Center 5.3 clients, software motion detection, and thumbnails are not supported when streams are encrypted. Security Center will not give access to a decoded frame via the SDK, so Mobile, RTSP Media Router, Analytics plugins using the client stream and Stratocast connectivity will not work.

- **Claims-Based Authentication**.
  - Security Center 5.4 adds support for claims-based authentication using a Secure Token Service (STS). When enabled, the client apps will now communicate with the claims issuer (STS) during the login process as opposed to communicating with the Directory or the Active Directory Role.
  - Security Center now includes an Internal STS which transforms externally generated tokens (eg. by ADFS) into a standard format. This will allow Security Center to support multiple different STS' simultaneously in the future.
  - With claims-based authentication, the Directory no longer validates username/password combinations, but rather validates tokens generated by an STS.
  - As described earlier in this FAQ, the Security Center client apps will authenticate to an external STS (eg. ADFS), receive a token from the external STS, then forward the token to Security Center's Internal STS within Security Center which will transform the token into a standard format that is then forwarded to the Directory to login. Once the Directory validates the token with the STS and accepts the login session.
  - **Active Directory Federation Service (ADFS)**
    - Security Center 5.4 now supports claims-based authentication through third party Security Token Service (STS), like ADFS (Active Directory Federation Services).

- ADFS also provides users with single sign-on access to applications located across organizational boundaries (across domains). The benefit here is that in a Security Center Federated architecture, ADFS will allow users to authenticate across domains. ADFS will issue tokens to external users (from external domains) to connect to a Security Center Federation head end server located in another domain. This avoids the need for more complicated and harder to maintain configurations that leverage two-way trust relationships between domains.
- Security Center 5.4 provides new configuration options that allow administrators to associate claims they get from ADFS to privileges in Security Center.

o **Claims-Based Authentication Using an Internal Genetec STS**.

- End users who don't have access to ADFS or don't want to use ADFS can leverage Security Center's Internal STS to enable claims-based authentication. Using this capability, customers can benefit from a more secure, claims-based approach to authentication.
- This internal or Genetec-built STS replaces our traditional authentication mechanism based on username/password combinations. Client applications will authenticate with the Genetec STS, and the Directory will grant/deny access based on tokens issued by the Internal STS.

Sources of Information

The following documents are some of the sources used to develop this FAQ:

- A Guide to Claim-Based Identity and Access Control, 2nd Edition, Microsoft
- http://searchsecurity.techtarget.com/definition/digital-certificatehttps://support.microsoft.com/en-us/kb/195724
- https://www.comodo.com/resources/small-business/digital-certificates-intro.php
- http://searchsecurity.techtarget.com/definition/digital-certificate
- https://en.wikipedia.org/wiki/Public_key_certificate
- https://en.wikipedia.org/wiki/Cryptography
- https://en.wikipedia.org/wiki/Self-signed_certificate
- https://en.wikipedia.org/wiki/Certificate_authority
- https://en.wikipedia.org/wiki/Public_key_infrastructure
- https://en.wikipedia.org/wiki/Public-key_cryptography
- https://en.wikipedia.org/wiki/Digital_signature
- https://en.wikipedia.org/wiki/Public_key_certificate