Cisco Webex is a cloud collaboration platform that provides messaging, calling and meeting features. The Cisco Webex Teams application is a client application that connects to this platform and provides a comprehensive tool for teamwork. Users can send messages, share files, and meet with different teams, all in one place. This White Paper provides an overview of the security features of Cisco Webex Teams running on Windows, Mac, iOS, Android, and Web. *

The intended audience for this whitepaper includes collaboration specialists who wish to learn more about security for Webex Teams, and security specialists, such as InfoSec teams, who want specific details of how security is implemented in the Webex Teams app.

* Some of the Cisco products, services, and features described in this document are still under development or planned for future. After being described, a planned feature will be marked with a “🔨” icon. Cisco will have no liability for delay in the delivery or failure to deliver the products, services or features marked with this icon.
Contents
Introduction – Secure Cloud Collaboration ............................................................4
The Webex Teams Applications ..........................................................................5
Webex Teams Application Architecture ............................................................6
  User Interface Layer ....................................................................................6
  Services Layer ..........................................................................................6
  Network Layer ..........................................................................................7
Downloading the Webex Teams Application ......................................................7
Signed Software Images ..................................................................................8
Webex Teams Application – Software Upgrades ..............................................9
Webex Teams – Connecting and Authenticating with Webex Services ..........9
  TLS Overview and TLS Version History .....................................................10
    TLS 1.0 ..................................................................................................10
    TLS 1.1 ..................................................................................................10
    TLS 1.2 ..................................................................................................11
    TLS 1.3 ..................................................................................................11
  Webex Teams – TLS Version and Cipher Suite Negotiation ......................12
    Elliptical Curve Diffie Hellman Ephemeral (ECDHE) Key Generation ...13
    RSA Authentication ..............................................................................13
  Encryption Algorithms ..............................................................................14
    Advanced Encryption Standard (AES) ....................................................14
    GCM and CBC Encryption Modes .......................................................14
  Secure Hash Algorithms : SHA-256 and SHA-384 ....................................15
    SHA-1 ..................................................................................................15
    SHA-2 ..................................................................................................15
  Authenticating the Webex Cloud Connection (Certificate Validation) ....16
    Webex Teams Certificate Validation ......................................................16
    Certificate Chain – Digital Signature Verification ................................17
    Certificate Issuer Verification ................................................................17
    Certificate Validity Period ....................................................................18
    Certificate Revocation Status ...............................................................18
    Key Usage Certificate Extensions ........................................................18
    Server Hostname Validation .................................................................18
    Certificate Public Key Pinning ...............................................................18
Webex Teams Sign-in ........................................................................................................19
Overview .....................................................................................................................19
Webex Teams Authentication Methods ........................................................................19
Authentication using the Webex Identity Service ....................................................19
Authentication using SAML based SSO with an Enterprise, or Cloud-based IdP19
Webex Teams Authorization ......................................................................................20
OAuth Tokens ...........................................................................................................20
The Access Token ..................................................................................................20
The Refresh Token .................................................................................................21
OAuth Authorization Code Grant and combined SAML based SSO Authentication21
Webex Teams Media Encryption ..............................................................................25
Webex Teams Apps – Data at Rest Protection .........................................................28
Windows, Mac, iOS and Android Applications – Data Storage ............................28
Webex Teams App for Web – Data Storage .........................................................29
Webex Control Hub – Security features for Webex Teams Applications ...............29
Data Wipe and Token Revocation ..........................................................................30
Blocking External Communications .......................................................................31
Collaboration Restrictions – Controlling What Content Users Can Share ..........32
Web Client Inactivity Timeout .................................................................................32
Managing Webex Teams on Mobile Devices .........................................................33
Enterprise Network Security ...................................................................................34
Firewall and Proxy Traversal ...................................................................................34
HTTP Proxy Traffic Inspection and Certificate Pinning ..........................................35
Webex Teams Proximity and Device Pairing ............................................................36
Proximity for Cloud-Registered Webex Devices ....................................................37
Proximity for On-Premises Registered Webex Devices ........................................38
Other Webex Device Discovery Mechanisms .......................................................40
Webex Teams Application Behavior: Camera and Microphone Use and Control ......41
Penetration Vulnerability Tests ..............................................................................41
Cloud Collaboration Applications – Security Feature Checklist ..........................41
Introduction – Secure Cloud Collaboration

Many cloud collaboration providers refer to a plethora of security features when it comes to securing the cloud and protecting customer data. Vendors often refer to architectural features such as “encryption of data in–transit” and “encryption of data at rest” as underpinning security mechanisms for their service. But what do terms such as these really mean when it comes to securing cloud collaboration services and the devices that use them?

This white paper is part of a series that covers all aspects of security for cloud collaboration. The series of white papers is intended to:

- Look at each aspect of security for cloud collaboration, from the Enterprise network to the Cloud and the security of the devices and applications that use cloud collaboration services.
- Clarify how security standards and protocols are used to protect cloud communications and user data.
- Inform the reader of the best practices that Cisco uses to ensure that the Webex cloud, Webex Teams apps and Webex devices are secure.
- Provide a security benchmark by which other cloud collaboration products can be measured.

This white paper provides an in-depth description of how Cisco secures the Webex Teams desktop, mobile and web applications. Many aspects of security for Webex Teams apps are covered including:

- Secure application onboarding
- Secure upgrades
- Authenticating Webex cloud services
- User authentication
- Secure media
- Secure data storage
- Securely traversing the Enterprise network edge
- Media transmission and application behaviour
- Administrative security and compliance controls

This white paper does not cover the following topics in detail:

- End-to-end encryption
- Encryption keys and the Key Management Service
- Webex platform and service security

For information on the above topics, see the Webex Teams Security White Paper
The Webex Teams app is a downloadable software image that provides voice, video, and messaging services to its users. To secure Webex Teams, Cisco uses several best practices and methodologies, including:

- Authenticated image files
- Encryption of data at rest
- Encryption of data in transit
- Secure software development using Cisco’s Secure Development Lifecycle (CSDL) *
- Controlled security feature implementation using Cisco’s Product Security Baselines (PSB) *
- User authentication using Identity Providers (IdPs) that support Single Sign On (SSO) using version 2 of the Security Assertion Mark-up Language (SAML) protocol.
- User authorization using OAuth2
- Security and compliance features configured in Webex Control Hub, Webex Teams’ administrative portal

The following sections discuss these topics in-depth and provide details on how Webex Teams security features are implemented.


**The Webex Teams Applications**

Webex Teams uses HTTPS and Secure Web Sockets (WSS) over TLS for REST based signalling, and SRTP (transported over UDP/TCP/TLS) for media. The Webex Teams app is available in several forms:

- Webex Teams for Windows, Mac, iOS and Android
- Webex Teams for Web, using HTML5 and WebRTC
- Webex Teams with calling using Unified CM, a hybrid application using Webex Teams messaging over HTTPS and SIP signalling for voice and video communications
- The Jabber UC hybrid application, using SIP signalling for voice and video communications, and Webex Teams messaging over HTTPS

This document focuses on the implementation of security in the Webex Teams app, although much of the discussion on security can also be applied to the Webex Teams components running in the hybrid applications.
Webex Teams Application Architecture

Webex Teams application software, known as the Unified Collaboration Framework (UCF), is a modular software library that supports multiple platform operating systems (Windows/Mac/iOS and Android). Figure 1 shows the UCF architecture that includes layers of abstraction, such as the user interface, services, and network layers. These isolate core functional roles allowing features and functions to be independently developed without impacting features in other layers.

Figure 1 Webex Teams Unified Client Framework Architecture

User Interface Layer
The user interface layer provides the visual front end for the Webex Teams app. This layer is responsible for rendering dialogues and orchestrating end user workflows. For example, creating new spaces, switching between spaces, pop out video windows, images, hover over effects etc. The user interface layer also implements accessibility requirements, localization, dark or light mode, and operating system or platform-specific functionality. For example, setting the client to automatically start when the computer starts.

The implementation of the user interface layer varies depending on the platform for Webex Teams, for example, Windows, Mac, iOS, or Android.

Services Layer
The UCF services layer provides a core set of services to the user interface layer, for example:

- Conversation service (messaging)
- Encryption service (encrypting and decrypting user generated content)
• Avatar service (user photos/ images)
• Calendar service
• Telephony service (voice and video)
• Proximity service

The UCF services layer also manages and tracks stateful services and provides an interface to the network layer.

Network Layer
The network layer is accessible only by the UCF services layer and provides a range of functions including:

• HTTPS, Secure WebSocket (WSS) and TLS session establishment
• Management of OAuth tokens and functions, for example, token refresh
• Certificate validation, for example, server, intermediate and root certificates
• Media engine – UDP/TCP/TLS media transport, media encryption/decryption
• TLS/HTTP proxy services, for example, proxy server address acquisition, proxy authentication

The Webex Teams app also makes use of the platform’s Operating System (OS) services, for example:

• Certificates stored in the OS trust store, for example, Public or Enterprise Root CA certificates
• OS Secure storage – master encryption keys used to encrypt stored user data

Downloading the Webex Teams Application

The Webex Teams app can be downloaded from the Webex website, Apple App store, or Google Play App store:

• [https://www.webex.com/downloads.html](https://www.webex.com/downloads.html)
Signed Software Images

Prior to uploading the Webex Teams image to a download repository or app store, Cisco uses a CA-signed software publishing certificate to digitally sign the software image. Using the code-signing infrastructure of each platform vendor (Microsoft/Apple/Google) to co-sign a PKCS #7-signed data object file containing the signed Webex Teams image, digital signature, and software publishing certificate.

Figure 2 Uploading a Digitally Signed Webex Teams Software Image

Figure 3 shows when a user downloads the Webex Teams software image, the platform operating system verifies the digital signature PKCS #7-signed data object file. Then the platform operating system creates a hash of the Webex Teams software image and uses the CA-signed software publisher certificate’s public key to decrypt the digital signature received with the downloaded software image. It compares the hash in the received digital signature to the hash that was generated locally. If the two hashes match, then the integrity of the received file is verified and can be installed by the platform OS. If the two hashes do not match, the application installation fails.
Webex Teams Application – Software Upgrades

The Webex Teams applications and Webex services use a continuous development model of iterative software development to deliver new features and improve existing ones. To keep the applications and services in sync, digitally signed software images for Webex Teams are made available as and when they are required. Software updates are automatically pushed to the Windows and Mac operating system and the user is prompted or given the option to upgrade their application. Software upgrades for iOS and Android can be downloaded from their respective App stores.

Webex Teams – Connecting and Authenticating with Webex Services

The Webex Teams app makes multiple TLS/HTTPS connections to the Webex cloud, these connections are outbound only and some connections are upgraded from HTTPS to bi-directional Secure WebSocket (WSS) connections.

The use of Transport Layer Security (TLS) to provide encryption of data in transit is a common industry practice, but not all TLS implementations are the same. For any cloud service, the TLS version and cipher suites offered by clients and servers should be inspected to determine if vulnerable TLS versions and cipher suites can be negotiated. The Transport Layer Security (TLS) protocol implementation used by Webex Teams are discussed in detail below.
TLS Overview and TLS Version History

Transport Layer Security (TLS) was created to provide authentication, confidentiality (data encryption), and data integrity between client and server applications. Determining the identity of the server is achieved by the server sending its CA signed certificate, chain of intermediate certificates and optionally the CA root certificate to the client for verification, as shown in Figure 4. The session is secured using symmetric encryption, the encryption cipher suite, and encryption key generation method for the session being negotiated before any data is sent. Key generation involves an exchange of values between the client and server, that allows both to generate a shared secret that is not transmitted and therefore not available to eavesdroppers. Once the cipher suite has been negotiated and the shared symmetric encryption key generated, each encrypted message is sent with a message authentication code to detect if the data in transit has been modified.

![Figure 4 Webex Teams TLS Handshake](image)

TLS 1.0

TLS 1.0 was first defined in 1999 and is based on the Secure Sockets Layer Protocol Version 3.0 (SSL 3.0). Evolving regulatory requirements and security vulnerabilities discovered in TLS 1.0 and SSL, have led to recommendations that they be disabled, in favour of the newer TLS versions of 1.1 and 1.2. More recently, the initiative to deprecate SSL and TLS 1.0 has be driven by the Data Security Standards defined by the Payment Card Industry (PCI). As of June 30, 2018, in order to comply with the PCI Data Security Standard (DSS), all websites must use TLS 1.1 or higher.

TLS 1.1

In April 2006, TLS 1.1 was published as a minor update to TLS 1.0. For more information on TLS 1.1, see [RFC 4346](https://tools.ietf.org/html/rfc4346). The primary changes in TLS 1.1 provide protections against Cipher Block Chaining (CBC) attacks, by adopting explicit initialization vector selection...
and changing the way that padding errors are processed. Most web browser vendors plan to deprecate TLS 1.1 in 2020.

**TLS 1.2**
In 2008, TLS 1.2 was released and published as [RFC 5246](https://tools.ietf.org/html/rfc5246). TLS 1.2 is currently the most widely used version of TLS and has several improvements in security when compared to TLS 1.1, particularly for negotiation of cryptographic algorithms.

A summary of the major differences between TLS 1.1 and TLS 1.2 described in RFC 5246 include:

- Stronger hashing algorithms (SHA-2) for MAC and pseudorandom function (PRF) computations
- Client and server ability to specify the accepted hash and signature algorithms.
- Support for cipher suites using Authenticated Encryption with Associated Data (AEAD) e.g. AES-128-GCM and AES-256 GCM, which are computationally more efficient and not only encrypt, but also authenticate data.
- AES cipher suites that support elliptical curve cryptography

**TLS 1.3**
TLS 1.3 was defined in [RFC 8446](https://tools.ietf.org/html/rfc8446) in August 2018.

A summary of the major differences between TLS 1.2 and TLS 1.3 described in RFC 8446 include:

- The list of supported symmetric encryption algorithms has been pruned of all legacy algorithms. The remaining algorithms are all Authenticated Encryption with Associated Data (AEAD) algorithms.
- A zero-roundtrip time (0-RTT) mode was added to improve connection setup times.
- Static RSA and Diffie–Hellman cipher suites have been removed. All public key based key exchange mechanisms now provide forward secrecy.
- Elliptic curve algorithms are now in the base specification and new digital signature algorithms, such as EdDSA, are included.

TLS 1.3 has several significant improvements over TLS 1.2, but has yet to be widely adopted.
Webex Teams - TLS Version and Cipher Suite Negotiation

Webex Teams apps use TLS version 1.2 for signalling. Webex cloud services (messaging, encryption, file storage and so on) only support TLS version 1.2.

TLS 1.2 supports many cipher suites, greater than 100 ciphers, for more information, see https://www.openssl.org/docs/manmaster/man1/ciphers.html. Many of the cipher suites listed in the above link have known vulnerabilities and should be avoided, for example, cipher suites that use DES or RC4 encryption. Webex Teams apps can negotiate only a small subset of cipher suites with Webex Services.

Figure 5 shows that during TLS session establishment, the app sends a list of its supported cipher suites in order of preference in a Client Hello message to the TLS server. The server selects one cipher suite, based on the subset of cipher suites that both the client and server support and the server preference, and returns this selected cipher suite to the app in a Server Hello message.

Figure 5 Webex Teams Application – Client to Server TLS Negotiation

TLS signalling connections from Webex Teams apps to Webex services use TLS version 1.2 only and negotiate the following strong cipher suites with Webex services, in order of preference:

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
These cipher suites meet the guidelines defined in the US National Institute of Standards and Technology (NIST) Special Publication 800-52 Revision 2. For more information, see [Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations](https://www.nist.gov/itl/csd/ct/觱w/environment/security-guidelines).

These ECDHE based cipher suites are a subset of those listed in Section 3.3.1.1.2 – “Cipher Suites for RSA Certificates”, the other ciphers defined in Section 3.3.1.1.2 use Diffie Hellman Ephemeral (DHE) for key generation which is less CPU efficient than ECDHE.

All the negotiable cipher suites have the following features in common:

**Elliptical Curve Diffie Hellman Ephemeral (ECDHE) Key Generation**

Elliptical Curve Diffie Hellman (ECDH) is an anonymous key agreement protocol that allows two parties, each having an elliptic curve public and private key pair, to establish a shared secret over an insecure channel, through the exchange of their elliptic curve public keys. The shared secret is used to derive the symmetric encryption key used by the TLS session.

With Elliptical Curve Diffie Hellman Ephemeral (ECDHE) key generation, for each TLS session: the client and the server exchange new elliptic curve public keys to generate a new shared secret and derive a new symmetric encryption key. Using ECDHE provides “forward secrecy”, whereby if the elliptic curve private key, shared secret, or symmetric key for a TLS session is compromised, previous and future TLS sessions cannot be decrypted using these compromised values.

**RSA Authentication**

ECDHE-RSA key generation uses the Rivest–Shamir–Adleman (RSA) public key associated with the server certificate to sign and authenticate the ephemeral elliptic curve public keys exchanged between the client and server.

- The RSA public key in the server’s RSA certificate is sent from the server to the client prior to key negotiation and is used to sign the ephemeral elliptic curve public key sent by the client to the server.
- The server’s RSA private key is used to sign the ephemeral elliptic curve public key sent by the server to the client.
- The server’s signed ephemeral elliptic curve public key has its integrity authenticated by the client, verifying its digital signature using the server’s RSA certificate public key.
- The client’s signed ephemeral elliptic curve public key has its integrity authenticated by the server using its private key.
Encryption Algorithms

Advanced Encryption Standard (AES)
The negotiable symmetric key encryption algorithms used between Webex Teams apps and Webex services use a subset of the Advanced Encryption Standard (AES) ciphers defined by the US National Institute of Standards and Technology (NIST):

- AES-256-GCM
- AES-128-GCM
- AES-256-CBC
- AES-128-CBC

Standardized in 2001 the AES algorithm features a cipher block size of 128 bits and three key length options: 128, 192 or 256 bits.

AES symmetric encryption is widely used today, mostly with 128-bit or 256-bit key lengths. The US National Security Agency classifies AES-256 as strong enough to protect TOP SECRET data. For more information, see https://apps.nsa.gov/iaarchive/programs/iad-initiatives/cnsa-suite.cfm.

GCM and CBC Encryption Modes
The AES encryption algorithms used by Webex Teams support two encryption modes for fixed size blocks of plaintext:

- Galois Counter Mode (GCM)
- Cipher Block Chaining (CBC) mode

Galois Counter Mode (GCM) encryption was introduced with TLS1.2. GCM encryption operates on 128-bit blocks of data and can be performed in parallel, making the algorithm more efficient than other algorithms such as CBC. GCM is an Authenticated Encryption with Associated Data (AEAD) algorithm. The output of an encryption cycle for a block of plaintext is the encrypted cipher text and an authentication tag. Only AEAD based encryption algorithms are allowed with TLS 1.3. AES encryption using GCM is faster and more secure than AES CBC.

Cipher Block Chaining (CBC) mode is an older encryption algorithm and is available for use in TLS versions 1.0, 1.1 and 1.2. CBC mode encryption also uses a 128-bit data block size, but cannot take advantage of parallel processing as encryption is sequential and each block of plaintext XORed with the previous ciphertext block before being encrypted. CBC is useful for interoperability with older TLS implementations and can be faster when used on older CPU hardware.
TLS 1.2 supports over 100 negotiable cipher suites, some of which use cipher suites with known weaknesses or performance inefficiencies.

Secure Hash Algorithms: SHA-256 and SHA-384
SHA-256 and SHA-384 are part of the SHA-2 family of secure hash algorithms. Secure Hash Algorithms (SHA) are used in TLS to authenticate the data’s origin and/or to validate the integrity of data. Hash algorithms such as the SHA family can be used in TLS as follows:

- To provide message authenticity when the encryption algorithm doesn’t use authenticated encryption, for example: AES-CBC
- As part of a Pseudo Random Function (PRF) used to compute the shared master secret during client-server key exchange. For more information, see https://tools.ietf.org/html/rfc5246#section-5
- To authenticate the encrypted Finished message of the TLS Handshake protocol

The SHA-2 algorithms are part of a family of secure hash algorithms defined in NIST FIPS 180-4. This family of secure hash algorithms can be grouped into three sets:

- SHA-1: 160-bit hash output
- SHA-2: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256
- SHA-3: SHA3-224, SHA3-256, SHA3-384, SHA3-512

SHA-1

The use of SHA-1 with certificates, should not be confused with its use in a TLS cipher suite. For example, although not preferred, SHA-1 for message authentication is still supported in TLS cipher suites such as TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA. In this context, SHA (SHA-1) is used to generate a keyed-hash message authentication code (HMAC) for each encrypted message. HMAC uses a symmetric cryptographic key, specifically generated for this MAC function during the TLS handshake, in conjunction with a hash function (SHA-1) to provide a strong form of message authentication.

SHA-2
The SHA-2 family of hashing algorithms produce hash outputs of either 224, 256, 384, or 512 bits. SHA-2 is the most commonly used set of secure hash algorithms with TLS 1.2 cipher suites and is used by Certificate Authorities for digital signatures in all new
certificates. TLS 1.3, published as RFC 8446 in August 2018, specifies the use of SHA-2 hashing algorithms only, specifically SHA-256 and SHA-384.

**Authenticating the Webex Cloud Connection (Certificate Validation)**

When a Webex Teams app establishes a TLS connection to the Webex cloud, the Webex server sends its CA signed server certificate, CA Root certificate, and any intermediate certificates to the Webex Teams app, as shown in Figure 6. Before proceeding with the TLS handshake, the Webex Teams app validates the chain of certificates to verify the authenticity of the Webex service. This certificate verification process is discussed in detail below.

**Figure 6 Verifying the Authenticity of a Webex Service**

As part of the TLS handshake protocol, once the Client Hello and Server Hello messages have been exchanged and a cipher suite negotiated, the server sends the app a Certificate message which includes the CA signed server certificate, intermediate certificate and CA root certificate. The Webex Teams app performs a series of checks, as depicted in Figure 7, to validate the received certificates and authenticate the Webex service:

**Webex Teams Certificate Validation**

As part of the TLS handshake protocol, once the Client Hello and Server Hello messages have been exchanged and a cipher suite negotiated. The server sends the app a Certificate message which includes the CA signed server certificate, intermediate certificate and CA root certificate. The Webex Teams app performs a series of checks, as depicted in Figure 7, to validate the received certificates and authenticate the Webex service:
Figure 7 Certificate and Certificate Chain Validation

The primary processes used to validate the integrity of the chain of certificates sent to the app are as follows:

**Certificate Chain – Digital Signature Verification**

Starting with the CA Root certificate, the public key of the certificate is used to decrypt the digital signature of the first sub-root intermediate certificate and produce its hash value. A hash of the intermediate certificate is then calculated and compared with the decrypted hash value, if they match the integrity of intermediate certificate is validated. The same process is then repeated with any other intermediate certificates until the server certificate is reached. As a final step, the public key of the intermediate certificate that signed the server certificate is used to decrypt the digital signature of the server certificate and produce its hash value. A hash of the server certificate is then calculated and compared with the decrypted hash value, if they match the integrity of server certificate is validated.

**Certificate Issuer Verification**

The name of the issuer in the server certificate is compared with the name in the subject field of its signing intermediate certificate for matching values. This process continues up the certificate chain, until the name of the issuer in the intermediate certificate signed by the Root CA certificate is compared with the name in the subject field.
field of the CA Root certificate. Finally, the name of the issuer and subject name should match in the self-signed CA Root certificate.

Certificate Validity Period
The “Valid from” and “Valid until” values in each certificate are checked to make sure that each certificate has not expired.

Certificate Revocation Status
The Webex Teams app uses the Online Certificate Status Protocol (OCSP), defined in RFC 6960, to determine the revocation state of each certificate. After receiving the chain of certificates from the server, the OCSP client (Webex Teams app) sends a Certificate Status Request with each server certificate’s serial number and issuer details to the Certificate Authority’s designated OSCP responder. The OCSP responder checks the revocation status of each certificate and returns a digitally signed response indicating the status of each certificate as either good, revoked or unknown.

Key Usage Certificate Extensions
Key Usage extensions define how the use of public key in the certificate can be restricted. For example, for encipherment, signatures, certificate signing etc. For more information, see https://tools.ietf.org/html/rfc5280#section-4.2.1.3

For example, the TLS cipher suite TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 uses ECDHE for key generation and the certificate’s RSA key to sign the exchanged ECDHE ephemeral keys. In this case, the “Digital Signature” flag should be set in the key usage extension of the certificate. The weaker TLS cipher suite TLS_RSA_WITH_AES_128_CBC_SHA uses RSA for key generation and encryption, in this case the “Key Encipherment” flag should be set.

Server Hostname Validation
The Webex Teams app validates that the name of the service it wants to connect to matches a name in the server certificate’s Subject or Subject Alternative Names.

Certificate Public Key Pinning
Certificate pinning provides protection against compromised certificate authorities (CA). A chain of certificates can appear to be verified with the correct digital signatures and revocation status but are in fact provided by a rogue or compromised CA. Webex Teams apps implement certificate public key pinning by embedding a copy of the public keys of intermediate certs in the application software. These pinned public keys are then used by the Webex Teams app to verify that the certificates received in the TLS Server Certificate message are the ones expected. If the received certificates public keys match the embedded pins, or another certificate in the OS trust store the connection will proceed. If
no match is found, the TLS connection is rejected. For more information, see the HTTP Proxy Traffic Inspection and Certificate Pinning section in this document.

Webex Teams Sign-in

Overview
On initial startup, once the Webex Teams TLS sessions have been established and the authenticity of the Webex services verified, the user is prompted for their email address by the Webex Identity service. The user’s email address is used to determine which Webex Teams organization the user belongs to. If the organization is using the Webex Identity service to authenticate the user, typically a consumer user, for example user@gmail.com, the user is prompted for their password. If the organization is using Single Sign On (SSO) with an Identity Provider (IdP), the SSO sign in page is presented and the user is prompted for their username and password for authentication.

Once authenticated, the Webex Authorization service uses the OAuth2 protocol to grant applications with controlled access to Webex services. The Authorization service delivers access and refresh tokens to the Webex Teams app. The Access token contains scopes that define which Webex services the user is authorized to use. Access and refresh tokens are securely cached by the Webex Teams app and used as a proof of authentication, so that the user does not have to sign in every time the application launches.

Webex Teams Authentication Methods

Authentication using the Webex Identity Service
Primarily used by consumer users or small to medium businesses without an IdP, the Webex Identity service offers a secure username and password-based authentication service over TLS.

Authentication using SAML based SSO with an Enterprise, or Cloud-based IdP
Webex Teams supports authentication with Enterprise (on-premises, or cloud) IdPs that support Single Sign on (SSO) using Security Assertion Markup Language version 2.0 (SAML 2.0). For more information on tested SSO IdPs and SSO set up details, see: https://help.webex.com/article/lfu88u

Using SSO with an IdP provides a number of benefits for users and administrators of Webex Teams:

- Users can use the same credentials to sign in to multiple applications, avoiding the requirement to remember multiple sets of credentials for multiple applications.
• If users are authenticated by their enterprise or cloud directory, for example, Active Directory (AD), Azure AD, or LDAP directories, then the enterprise can decide on the degree of password complexity, password lifetime and password re-use policies.

• SSO can be combined with Multi-Factor Authentication, where the user is required to provide more than one form of proof of authenticity in addition to a username and password, for example:
  
  o Something the user has, for example a security token, a smartphone app that generates a onetime passcode, a code contained in SMS text message.
  o A physical characteristic of the user for biometric security, for example, a fingerprint scan, voice or face recognition.
  o The location of the user, for example, are they on the enterprise network either physically or using a VPN connection?

**Webex Teams Authorization**

Webex Teams uses the Open Authorization protocol version 2.0 (OAuth 2.0) to provide an authenticated user’s application with access to authorized Webex services without the need to share the user’s credentials with the service. Webex Teams also uses OAuth to grant third party application integrations with authorized access to Webex Services.

**OAuth Tokens**

Once the user has authenticated, the Webex Teams app is provided with an access token and refresh token by the Webex authorization service.

Access tokens use the JSON Web Token (JWT) format with JSON Web Signature (JWS) and JSON Web Encryption (JWE). For more information, see the following:


**The Access Token**

The encrypted and signed access token uses the JSON Web Token (JWT) format to encapsulate information about the organization. For example, organization ID (Org ID), User (UUID), Device (Client ID) and the scope of Webex services that the user has been granted access to. Webex service access tokens have a lifetime of six hours.
**OAuth Access Token Scopes**

The scopes in a Webex Teams access token define what the application is authorized to do on behalf of the user, for example:

- Read messages
- Write messages
- Read space memberships
- Write space memberships

The combination of user scopes, the user’s organization, user id, and device id in the access token are used to authorize a user’s access to resources in the Webex cloud.

**The Refresh Token**

The refresh token is an opaque string that encapsulates the information required to request a new access token from the Webex authorization service, without prompting the user to re-authenticate. Access tokens are typically refreshed when they reach 75% of their 6-hour lifetime at 4.5 hours. A new refresh token is delivered with the access token. Webex Teams refresh tokens have a lifetime of 60 days. If Webex Teams doesn’t refresh its access token within this 60 day period, for example, when Webex Teams is offline for a long period, the user must re-authenticate to regain access to the Webex service.

Note – By default, the Webex Teams apps refresh token is renewed every time its access is renewed, maintaining user sign-in. If you want to disable the renewal of the refresh token when the access token is renewed, or to adjust refresh token lifetime, open a service request with Cisco TAC.

**OAuth Authorization Code Grant and combined SAML based SSO Authentication**

Before using any Webex service, the Webex Teams app must present its access token to the service, which decrypts the token and verifies that the scopes that it contains match some or all of the scopes provided by the service. If the access token’s scopes do not match that of the service, access is denied. If a Webex Teams app attempts to communicate with a Webex service without an access token, the service redirects the application to Webex authorization service, where the application can request an access token with the scopes that it requires and a refresh token.

OAuth 2.0 defines several methods by which an application can request and obtain tokens. The primary and considered to be most secure method used by Webex Teams apps and Webex services is the Authorization Code Grant Flow defined in detail in RFC 6749 and summarized below. In Webex Teams the user is requesting an access token so that their Webex Teams app is granted access to Webex services, unlike other
commonly described OAuth grant flows where the user is granting a third-party application with access to a resource owned by the user.

OAuth 2.0 is an industry standard for authorization; prior to granting access and refresh tokens to an application, the application user must be authenticated. Webex Teams supports authenticating users using either the Webex Identity Service or using an Enterprise IdP and Identity Store, for example, AD, Azure AD, LDAP and so on.

The flows in the diagrams below show the OAuth Authorization Code grant flow combined with SSO-based authentication using SAML to an enterprise IdP.

In the initial request shown in Figure 8, the user has not authenticated and does not have an access token to present to the Webex service. The service redirects the Webex Teams app to the authorization service. The Webex Teams app makes an authorization code grant request to the authorization service. This authorization code grant request contains the user’s email address, client ID and a list of scopes that the access token will need to access Webex services. The request from the application does not contain a refresh token, and the authorization service then redirects the application to the Webex Identity sign in service.

Figure 8 Webex Teams – Webex Service Request Prior to Authorization and Authentication

In Figure 9 the Webex Teams app sends a sign in request with the user’s e-mail address to the Webex Identity service. The Webex Identity service determines that this user is using an external IdP and redirects the application to the IdP for SAML-based authentication. The user enters their username and password, which are passed to the
IdP for authentication with enterprise identity store, for example, AD, Azure AD, LDAP and so on. When the user has been authenticated the IdP returns a SAML assertion to the Webex Teams app.

Figure 9 SSO-based User Authentication using SAML to an Enterprise IdP

In Figure 10 the Webex Teams app returns the SAML assertion for the authenticated user to Webex Identity service. The Webex Identity service validates the user’s SAML assertion and redirects the application to the authorization service. The authorization service validates the SAML assertion and returns an authorization code to Webex Teams app to complete the OAuth authorization code grant flow.
Figure 10 User Authenticated – SAML Assertion Validated by Webex Identity Service with Redirect to the Authorization Service for the Authorization Code Grant Flow

Figure 11 shows the final step in the OAuth Authorization Code Grant flow. The authorization code, client ID and client secret are sent by the Webex Teams app to the authorization service which validates these values and then returns an encrypted and signed access token and optionally a refresh token. The Webex Teams app can then present the access token as proof of authorization (and implicit proof of authentication) to use a Webex service.
Figure 11 Authorization Code Grant Flow Completion – OAuth Tokens sent to the Webex Teams app for Webex Service Access

When the access token’s lifetime reaches 75% of its validity period (6 hours), the Webex Teams app sends its refresh token to Webex authorization service to request a new access token and refresh token. The OAuth access and refresh tokens are securely stored in the application platform’s OS and can be revoked by a Webex Teams administrator through the Reset Access option in Webex Control Hub.

Webex Teams Media Encryption

Webex Teams apps use real-time media for audio, video, and content sharing streams. Typically, media from any Webex Teams app transits from the user’s location to media nodes in the Webex cloud, where the streams are mixed and distributed. This is true for all call types for example, calls to one other person, and multiparty calls. Optionally, instead of media nodes in the Webex cloud, on-premises Video Mesh Nodes (VMNs) may also be deployed to mix and distribute media locally.

Cisco secures all Webex Teams media streams using the Secure Real-Time Transport Protocol (SRTP), described in RFC 3711. Voice and Video codecs, the media encryption cipher and encryption keys are securely negotiated over HTTPS, using SDES (RFC 4568).
Figure 12 shows Webex Teams encrypted media using the AES_CM_128_HMAC_SHA1_80 cipher suite.

In line with RFC 3550 RTP – A Transport Protocol for Real-Time Applications, Cisco uses UDP with destination port 5004 as the preferred transport protocol for Webex Teams voice and video media streams.

The Webex Teams app also supports TCP with destination port 5004 as a fallback media transport protocol. However, we do not recommend TCP as a transport protocol for voice and video media streams, as TCP is connection oriented and designed to reliably deliver correctly ordered data to upper layer protocols. Using TCP, the sender retransmits lost packets until they are acknowledged, and the receiver buffers the packet stream until the lost packets are recovered. For media streams, this behavior manifests itself as increased latency or jitter and in turn this affects the media quality experienced by the call’s participants.

The Webex Teams app also supports TLS (secured TCP) as a tertiary option for media transport. Using TLS with destination port 443, can also mean that this Webex Teams media traffic will need to pass through an enterprise’s proxy server to reach media servers in the Webex cloud. Since proxy servers are primarily designed to intercept and forward HTTP-based web traffic, media quality can be impacted if the proxy server reaches its performance threshold and delays or drops packets when processing large numbers of high bandwidth media streams.

Webex Teams media flows in both directions using a symmetric inside-initiated, 5-tuple (source IP address, destination IP address, source port, destination port, protocol) stream outbound to the Webex cloud.

Webex Teams also uses STUN (RFC 5389) for firewall traversal and media node reachability testing.
Webex Teams apps use a media node discovery process at start up, when network connections change, and periodically to determine the reachability of media node clusters available to their organization (Cloud media nodes and on-premises Video Mesh Nodes). During call establishment, the Webex Teams app sends its reachability report to the Webex cloud. Based on the available transport protocol (UDP/TCP/TLS), Round Trip Times (RTT) and resource availability, the Webex cloud determines the media node that will be used by each app. As shown in Figure 13, media cascade connections are established if multiple media servers are used in a call.
Webex Teams Apps – Data at Rest Protection

Windows, Mac, iOS and Android Applications – Data Storage

Encryption of data at rest applies not only to content stored in the Webex cloud, but also to content stored by Webex Teams apps. The following content is securely stored by Webex Teams for Windows, Mac, iOS and Android:

- Messages
- Preview files, files converted to Portable Network Graphics (PNG) file format
- Space encryption keys
- Profile Pictures
- Space details
- Meeting details
- Whiteboard files
- OAuth tokens

Webex Teams apps on desktop and mobile devices store this content in an SQLite database that is encrypted using the AES-256-OFB algorithm. The master key for the SQLite database is encrypted by and stored in the platform OS secure store. For example, Windows Data Protect API, Mac/iOS Secure Enclave and Keychain, and Android Keystore.
Files downloaded by the Webex Teams app are decrypted prior to storage. The storage location of downloaded files is determined by the user, for example, the Windows Downloads folder.

**Webex Teams App for Web – Data Storage**
Webex Teams for web (https://teams.webex.com) does not permanently store content. Messages, files, encryption keys and tokens are deleted when the browser or browser tab is closed. One exception to this case is when the “Remember Me” option is selected by the user to bypass user authentication. In this case, the access and refresh tokens are stored and reused when Webex Teams is relaunched in the browser.

**Webex Control Hub – Security features for Webex Teams Applications**
This document has discussed the protocols and ciphers that are used to secure Webex Teams apps. Administrators of an organization of Webex Teams users also need to have tools and administrative features that provide them with control over how the Webex Teams app functions. For example, administrators may wish to implement file sharing controls, or to restrict communication with users outside of their organization.

The following section highlights some of the security features available in Webex Control Hub that provide security related controls. The list of features discussed below is not exhaustive, and security can also be implemented in adjunct applications that are integrated with Webex Teams (such as, Data Loss Prevention applications and Enterprise Content Management applications) for more details about Webex Control Hub management and security features see: https://www.cisco.com/c/en/us/products/collateral/collaboration-endpoints/webex-room-series/datasheet-c78-740770.html and https://www.cisco.com/c/en/us/products/collateral/collaboration-endpoints/webex-room-series/datasheet-c78-740771.html

Data Wipe and Token Revocation
Figure 15 shows the Reset Access option in Webex Control Hub, which allows an administrator to revoke all OAuth access and refresh tokens owned by the user. This option signs the user out of all Webex Teams applications and deletes all Webex Teams content stored in the SQLite database on iOS and Android mobile devices.

Figure 15 Reset Access: Delete Content, Revoke User Access and Refresh Tokens
Blocking External Communications

By default, Webex Teams allows application users in your organization to participate and communicate in spaces owned by any other Webex Teams organization. A key benefit of this capability is that you don’t need to create guest accounts in your organization for users to communicate with people outside your organization.

Figure 16 shows the two options to restrict communication for your organization’s users:

1. Block External Communication
2. Block External Communications and whitelist domains used by external Webex Teams organizations

Figure 16 Webex Control Hub – Controlling External Communications

More granular control over which users and organizations are allowed to communicate with one another can also be provided by a Data Loss Prevention (DLP) application by using the Webex Teams Events API to track space memberships and space events. For example, using the Cisco Cloudlock DLP exposure policies: https://docs.umbrella.com/cloudlock-documentation/docs/webex-teams-1#section-shared-with-external-user
Collaboration Restrictions – Controlling What Content Users Can Share

Figure 17 shows how an administrator can control how content is shared in spaces, using the following options:

- Control by device type:
  - Preview file downloads (documents converted into .png image files)
  - File downloads (documents such as Word docs, spreadsheets, and so on)
  - Whiteboards and Annotation downloads
- Control by device type:
  - File uploads
  - Whiteboards and Annotation uploads (where applicable)
- Control the use of animated GIF files in spaces
- Control previews of security-vetted website links in spaces

Figure 17 Webex Control Hub – Content Control

Web Client Inactivity Timeout

Webex Teams for web can be automatically signed out after a configurable inactivity time period, based on whether the application is on the network or off the network, as shown in Figure 18.
Managing Webex Teams on Mobile Devices

Webex Control Hub has several options that allow an administrator to control how Webex Teams is used on mobile devices. For example, file upload and download controls, reset access with data wipe function for mobile devices, and PIN lock screen enforcement as shown in Figure 19.

Webex Teams mobile apps are certified with Cisco Meraki® Systems Manager and VMware AirWatch MDM applications, but not limited to these providers. An organization using an MDM can include Webex Teams in their Enterprise app store and use the MDM application to provide basic controls such as prevent copy, screen capture, back-ups, remote wipe and pin lock requirements on devices. MDM applications are typically used on corporately owned mobile devices.

Mobile Application Management (MAM) applications can be used to control applications on employee-owned mobile devices. Using MAM you can manage the application rather than control the device. Wrapped versions of the Webex Teams app that allow them to be managed by a MAM, for example, IPA files for iOS and APK files for Android, are available as a Customer Connection Program offering. Using these wrapped apps a
Mobile Application Management (MAM) administrator can apply additional security policies to Webex Teams.

**Enterprise Network Security**

Most enterprises implement multiple products and features to protect their internal networks and data, and to control external access. Webex Teams app support the following enterprise network security features, protocols, and products:

- VLANs
- Network Access Control (802.1X)
- Firewall Traversal
- Proxy Server – Authentication, Traffic filtering and Inspection

Some of these security features are discussed at a high level in this document, for more details on the Enterprise Network Requirements for Webex Teams services, including Webex Teams IP subnets for media and URLs for signalling to services see (and subscribe to) the Network Requirements for Webex Teams Services article at: https://help.webex.com/article/WBX000028782

**Firewall and Proxy Traversal**

Most security conscious customers deploy both a firewall and proxy server to control access from applications and devices in their enterprise networks to the Internet and associated cloud services, such as Webex Teams. Specific implementations may vary, but a common deployment forces all HTTP-based traffic through a proxy server allowing only HTTP traffic originating from the proxy server to traverse the firewall and reach the Internet. Other traffic types such as UDP or TCP-based media from Webex Teams apps are not directed to the proxy server, but traverse the firewall only. Note: All on-premises Webex Teams apps, devices, and hybrid services initiate outbound connections only to cloud-based Webex Services.
HTTP Proxy Traffic Inspection and Certificate Pinning
Traffic inspection by a proxy server, where the proxy decrypts, inspects and re-encrypts the TLS/HTTPS traffic traversing the server is commonly used by security conscious customers. Traffic inspection by a proxy server has limited value for Webex Teams traffic, as decrypting and inspecting traffic only reveals signalling information, as user generated messages and files are end-to-end encrypted by the Webex Teams app. If you want to inspect and moderate user messages and files shared in Webex Teams spaces, use a DLP or CASB application, such as Cisco Cloudlock.

As shown in Figure 21, with traffic inspection, the proxy server presents its Enterprise CA signed certificate, instead of the Webex service certificate, to the Webex Teams app. This allows the proxy server to establish a TLS connection to the Webex Teams app and to encrypt/decrypt and inspect its signalling. Similarly, signalling traffic from the proxy server to the Webex service can also be encrypted/decrypted and inspected.
Webex Teams apps implement certificate public key pinning, by embedding a copy of the public keys of intermediate certs in the application software. These pinned public keys are then used by the Webex Teams app to verify that the certificates received in the TLS server certificate message are the ones expected. With traffic inspection by the proxy server, the enterprise CA signed certificate will not match the certificate pins embedded in the application software. In this case, the Webex Teams app searches the OS trust store for a certificate that matches the one received from the proxy server. If a matching certificate is found, the TLS connection from the Webex Teams app to the proxy server is allowed to be established.

**Webex Teams Proximity and Device Pairing**

Webex Teams desktop and mobile apps can use proximity to pair with Webex cloud-registered devices and on-premises Cisco video devices registered to Cisco Unified CM and Cisco TelePresence Video Communication Server (VCS). The device discovery and pairing mechanisms are similar for cloud-registered devices and Unified CM/VCS-registered devices; the content sharing and device control mechanisms both use TLS/HTTPS connections, but differ in the paths that they use between the Webex Teams app and the device.

Note: Webex Teams for web supports manual pairing only.
**Proximity for Cloud-Registered Webex Devices**

Cloud-registered Webex devices use ultrasonic signalling and tokens to pair with Webex Teams apps. Figure 22 shows that unique tokens are generated by the Webex cloud every 30 seconds and securely sent over TLS to the Webex device, which emits these tokens using ultrasound from the device speakers. A Webex Teams app within range of the ultrasound signal* can use the received token to pair with Webex device, by sending the token to the Webex cloud service. Once the device and app are paired, newly emitted tokens must be received by the Webex Teams app and sent to the Webex cloud service to maintain the paired connection.

* One reason for using ultrasound for proximity detection is its limited range; ultrasound signals typically do not pass through walls, limiting the pairing token’s range to the enclosed room that the endpoint is placed within.

Figure 22 Ultrasound Pairing for Webex Teams and Webex Devices

Figure 23 shows that once the paired connection between the device and app has been established using the Webex cloud, the Webex Teams app can control the Webex device, for example to make calls, mute and so on, and also share content on the Webex device. Both the Webex Teams app and the Webex device use their existing TLS connections to the Webex cloud, to exchange call control signalling and media for content sharing.
Figure 23 Content Sharing for Webex Teams and Webex Devices

Proximity for On-Premises Registered Webex Devices

Figure 24 shows that Unified CM and VCS registered Cisco video devices use ultrasonic signalling and tokens for proximity pairing with Webex Teams apps. Unique tokens are generated by the device every 180 seconds and emitted using ultrasound from the device speakers. A Webex Teams app within range of the ultrasonic signal can use the received token to pair with a Unified CM or VCS registered Cisco Video device, by sending the token to the device over a HTTPS connection. Once paired, newly emitted tokens must be received by the Webex Teams app and returned to the device to maintain the paired connection.

For more information, see the Configure On-Premises Devices for Cisco Webex Teams Users article at https://help.webex.com/article/pogihk
When the paired connection between the Cisco video device and Webex Teams app has been established, the Webex Teams app can control the device, for example to make calls, mute and so on, and also share content (see figure 26). The Webex Teams app and Unified CM or VCS registered video device use the directly established HTTPS connection to exchange call control signalling and media for content sharing.

Note: There are differences in how the Webex Teams app connects to cloud-registered and on-premises devices. When connecting to an on-premises device, the content that is shared between the Webex Teams app and the video device is always encrypted. However, we don’t enforce certificate verification when an HTTPS session is established with an on-premises device. Verifying certificates would prevent pairing with guest devices and would be complex to deploy and maintain.

Figure 25 shows the Webex Teams discovery of on-premises devices option in Webex Control Hub, this option is disabled by default.
Other Webex Device Discovery Mechanisms
Webex Teams apps can also use Wi-Fi to discover Webex devices and manually connect using a PIN. For more information, see the following:

- The Manage Wi-Fi Discovery of Webex Devices article at https://help.webex.com/article/nz9iowf

- The Find and Connect to Nearby Cisco Webex Devices from Cisco Webex Teams article at https://help.webex.com/article/qtsdfy

- The Manually Connect to Cisco Webex Devices from Cisco Webex Teams article at https://help.webex.com/article/nf29igm
Webex Teams Application Behavior: Camera and Microphone Use and Control

With cloud collaboration, users not only expect common security features, such as encryption of data in-transit and at rest, but in addition to this they also want to know that their cloud-connected devices can provide privacy when devices are both in use and idle. Device indications of camera and microphone usage are arguably more important for cloud connected devices. Users want to know that no media is being sent to the cloud when a camera or microphone is muted or not being used in a call.

The Webex Teams app allows its device’s camera and/or microphone to be muted during a call. While muted no media is sent from the microphone and/or camera to the Webex cloud.

Webex Teams apps also use their device microphone for Proximity detection to detect pairing tokens sent by Cisco Webex devices using the ultrasonic 20KHz to 22KHz frequency range. Device discovery using ultrasound and Wi-Fi can be disabled from the Devices menu of the Webex Teams app. When using device discovery for proximity and pairing, audio is not sent to the cloud; the microphone is used to detect pairing tokens sent in the ultrasonic frequency range. When a pairing token is detected, it is processed locally by the Webex Teams app and sent to the Webex cloud using a TLS encrypted signalling channel.

Penetration Vulnerability Tests

Cisco regularly undertakes penetration and vulnerability testing of Webex Teams apps. The results of these tests are available on request to customers who have signed a Cisco Non-Disclosure-Agreement (NDA). For more information, contact your Cisco account team.

Cloud Collaboration Applications – Security Feature Checklist

When evaluating any cloud collaboration solution, knowing the details of how a cloud collaboration client application implements security is a crucial part of your evaluation. Cisco Webex Teams rigorously implements and maintains security across all their applications, devices, and cloud services. The following table highlights the security features, ciphers, protocols and so on, that Cisco considers provide strong security and some of those that should be avoided.
### Table A Cloud Collaboration Security Feature Checklist

<table>
<thead>
<tr>
<th>Security Feature</th>
<th>Supported by Webex Teams</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Software Image</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Signed Install Images</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Signed Upgrade Images</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Encrypted Signalling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TLS Version Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TLS 1.0</td>
<td>No</td>
<td>Known vulnerabilities</td>
</tr>
<tr>
<td>TLS 1.2</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>TLS 1.3</td>
<td>Planned</td>
<td>Not widely deployed today</td>
</tr>
<tr>
<td>Server/Service Certificates</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSA Certificates</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>RSA key size =&gt; 2048 bits</td>
<td>Yes</td>
<td>NSA guidance</td>
</tr>
<tr>
<td>RSA key size &lt; 2048 bits</td>
<td>No</td>
<td>Not recommended</td>
</tr>
<tr>
<td>EC Certificates</td>
<td>No</td>
<td>Not widely deployed</td>
</tr>
<tr>
<td>TLS Cipher Suites</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key Generation / Key Exchange Protocol</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ECDHE</td>
<td>Yes</td>
<td>Forward Secrecy</td>
</tr>
<tr>
<td>DHE</td>
<td>Not Negotiable</td>
<td>Forward Secrecy (less CPU efficient than ECDHE)</td>
</tr>
<tr>
<td>RSA</td>
<td>Not Negotiable</td>
<td>No Forward Secrecy</td>
</tr>
<tr>
<td>DH</td>
<td>Not Negotiable</td>
<td>No Forward Secrecy</td>
</tr>
<tr>
<td>Authentication</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSA</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DSA</td>
<td>No</td>
<td>Signing only</td>
</tr>
<tr>
<td>ECDSA</td>
<td>No</td>
<td>Requires EC Certificate</td>
</tr>
<tr>
<td>Encryption Algorithms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AES-256-GCM</td>
<td>Yes</td>
<td>Preferred AEAD algorithm</td>
</tr>
<tr>
<td>AES-128-GCM</td>
<td>Yes</td>
<td>AEAD algorithm</td>
</tr>
<tr>
<td>AES-256-CBC</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>AES-128-CBC</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>RC4</td>
<td>Not Negotiable</td>
<td>Weak – not recommended</td>
</tr>
<tr>
<td>DES</td>
<td>Not Negotiable</td>
<td>Weak – not recommended</td>
</tr>
<tr>
<td>3DES</td>
<td>Not Negotiable</td>
<td>Weak – not recommended</td>
</tr>
<tr>
<td>Secure Hash Algorithms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHA-384</td>
<td>Yes</td>
<td>Preferred</td>
</tr>
<tr>
<td>SHA-256</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>SHA-1</td>
<td>Not Negotiable</td>
<td></td>
</tr>
<tr>
<td>Certificate Validation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Digital Signature Validation</td>
<td>Yes</td>
<td>All Certificates in the chain</td>
</tr>
<tr>
<td>Issuer Validation</td>
<td>Yes</td>
<td>All Certificates in the chain</td>
</tr>
<tr>
<td>Feature</td>
<td>Value</td>
<td>Notes</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Cert. Validity Period</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Cert. Revocation Status</td>
<td>Yes</td>
<td>OCSP</td>
</tr>
<tr>
<td>Server Hostname Validation</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Cert. Public Key Pinning</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>User Authentication</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAML based SSO</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Cloud and On Premises based IdP support</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Multi Factor Authentication support</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>User/Device Authorization</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OAuth 2</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Media Encryption</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RFC 3711 SRTP</td>
<td>Yes</td>
<td>All Media Always Encrypted</td>
</tr>
<tr>
<td>Media Encryption Cipher</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AES-CM-128-HMAC-SHA1-80</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Media Transport Protocol</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UDP</td>
<td>Yes</td>
<td>Preferred</td>
</tr>
<tr>
<td>TCP</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>TLS</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Application Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stored Application Content Encrypted on Device</td>
<td>Yes</td>
<td>Except Webex Teams for Web which does not store content</td>
</tr>
<tr>
<td>Content Encryption Algorithm</td>
<td>AES-256-OFB</td>
<td></td>
</tr>
<tr>
<td>Master Key Storage location</td>
<td>OS secure store</td>
<td></td>
</tr>
<tr>
<td>Administrative Security Features</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Wipe Capability</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>OAuth Token Revocation</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Default - Open Communication with external domains (no guest account requirement)</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Block External Communication (BEC)</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>BEC with communication allowed to whitelisted external domains</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>File Upload Restrictions</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>File and File Preview Download Restrictions</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>DLP/CASB support</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Enterprise Content Management support</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Web Client Inactivity Timeout</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Enterprise Network Features</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature</td>
<td>Yes</td>
<td>Manual, WPAD, PAC</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----</td>
<td>-------------------</td>
</tr>
<tr>
<td>Proxy Configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proxy Authentication</td>
<td></td>
<td>None, Basic, Digest, NTLM</td>
</tr>
<tr>
<td>Proxy TLS Traffic Inspection</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Application Media Control</td>
<td></td>
<td></td>
</tr>
<tr>
<td>In Call Video Mute</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>In Call Audio Mute</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Proximity Detection</td>
<td>Yes</td>
<td>Ultrasound /Wi-Fi /Manual Configurable : On/Off</td>
</tr>
<tr>
<td>Penetration Testing Attestation documentation</td>
<td>Yes</td>
<td>Available by request, under NDA</td>
</tr>
</tbody>
</table>