AGID | Agenzia per l'Italia Digitale

# Raccomandazioni Agid in merito allo standard Transport Layer Security (TLS)

Abbreviazione: AgID-RACCSECTLS-01

Versione: 2020-01

Data: 03/11/2020

# Acronimi

- AES - Advanced Encryption Standard
- AEAD - Authenticated Encryption with Associated Data
- DEA - Data Encryption Algorithm
- DHE - Diffie-Hellman in ephemeral mode
- DSA - Digital Signature Algorithm
- ECC - Elliptic Curve Cryptography
- ECDSA - Elliptic Curve Digital Signature Algorithm
- HKDF - Expand Key Derivation Function
- HMAC - Keyed-hash Message Authentication Code
- IETF - Internet Engineering Task Force
- KDF - Key Derivation Function
- MD5 – Message Digest Algorithm
- PFS - Perfect Forward Secrecy
- PSK - Pre-Shared Key
- RFC – Request For Comments
- RSA – Rivest, Shamir e Adleman
- SHA – Secure Hash Algorithm
- SSL – Secure Socket Layer
- TDEA -Triple Data Encryption Algorithm
- TLS – Transport Layer Security
- 3DES - Triple Data Encryption Standard

## Terminologia

Le parole chiave «DEVE», «DEVONO», «NON DEVE», «NON DEVONO», «E' RICHIESTO», «DOVREBBE», «NON DOVREBBE», «RACCOMANDATO», «NON RACCOMANDATO» «PUO'» e «OPZIONALE» nel testo del documento debbono essere interpretate come descritto nel seguito, in conformità alle corrispondenti traduzioni contenute nel documento IETF RFC 2119.

# 1 Introduzione

Transport Layer Security (TLS) è un protocollo che permette di stabilire un canale con le proprietà di integrità e riservatezza in senso crittografico tra un client e un server. Dopo aver stabilito una connessione sicura tramite il protocollo TLS, le applicazioni possono utilizzarla per scambiare dati. TLS viene utilizzato in molteplici contesti applicativi (HTTPS, SMTPS, etc..).

Questo documento fornisce un insieme di raccomandazioni in merito ai protocolli di sicurezza e alle Cipher Suite rappresentanti lo stato dell'arte **al momento della sua stesura**.

Data la continua evoluzione tecnologica e la possibile scoperta di nuove vulnerabilità, il presente documento sarà aggiornato periodicamente ed eventualmente verranno emanati specifici avvisi di sicurezza.

Il documento è suddiviso nei seguenti paragrafi:

1. Versioni TLS
2. Cipher Suites
3. Lunghezza chiavi
4. Raccomandazioni generali
5. TLS 1.2 versus 1.3

## 1.1 Versioni TLS disponibili

Ad oggi sono disponibili le seguenti versioni TLS:

- TLS 1.3 (pubblicato nel 2018)
- TLS 1.2 (pubblicato nel 2008)
- TLS 1.1 (pubblicato nel 2006)
- TLS 1.0 (pubblicato nel 1999)

TLS 1.0 e 1.1 sono protocolli obsoleti che non supportano i moderni algoritmi crittografici e risultano vulnerabili ad attacchi. Questi due protocolli sono da ritenersi **deprecati** come da comunicazioni di Google, Microsoft, Cisco, Apple e Mozilla.

## 1.2 Suite di cifratura (cipher suite)

Il supporto crittografico in TLS è fornito attraverso l'uso di varie suite di crittografia. Una suite di cifratura definisce una combinazione di algoritmi per lo scambio di chiavi e per fornire riservatezza e integrità della sessione durante lo scambio di messaggi. Al momento della negoziazione di una sessione TLS il client presenta una serie di cipher suite supportate che propone al server il quale ne seleziona una.

### 1.2.1 Suite di cifratura versione 1.2

Le suite di crittografia TLS 1.2 contengono quattro singoli algoritmi che rendono sicuro il canale in fase di handshake.

Una suite di cifratura può, ad esempio, essere composta dai seguenti 4 gruppi di algoritmi:

| **Funzione** | **Algoritmo** |
|---|---|
| Scambio di chiavi | RSA, Diffie-Hellman, ECDH, SRP, PSK |
| Autenticazione | RSA, DSA, ECDSA |
| Cifratura dati | RC4, 3DES, AES |
| Hashing | HMAC-SHA256, HMAC-SHA1, HMAC-MD5 |

Un esempio di cipher suite TLS 1.2 con i relativi algoritmi utilizzati delle varie fasi è la seguente:

**TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256**

| Scambio di chiavi | Autenticazione | Cifratura a blocchi | Hashing |
|---|---|---|---|

### 1.2.2 Suite di cifratura versione 1.3

La nuova versione di TLS ha portato con sé molti miglioramenti. Tra i più importanti è stata la dismissione di algoritmi e cifrari vulnerabili o non adatti a garantire PFS, tra i quali:

- RC4 Stream Cipher
- RSA Key Exchange
- SHA-1 Hash Function
- CBC (Block) Mode Ciphers
- MD5 Algorithm
- Vari gruppi di Diffie-Hellman non-ephemeral
- EXPORT-strength ciphers
- DES
- 3DES

Inoltre, è stata migliorata la fase di handshake, dimezzando la latenza della crittografia e riducendo il tempo di handshake.

Le suite di crittografia TLS 1.3 non includono più gli algoritmi di scambio chiave e firma e l'autenticazione è stata unita alla crittografia in un unico algoritmo di tipo AEAD (Authenticated Encryption with Additional Data). Ciò ha semplificato le possibili combinazioni di cipher suite.

Attualmente sono solo cinque le cipher disponibili per TLS 1.3:

- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256
- TLS_AES_128_GCM_SHA256
- TLS_AES_128_CCM_8_SHA256
- TLS_AES_128_CCM_SHA256

| Cifratura dati AEAD | Hashing |
|---|---|

| *Componente* | *Contenuto* |
|---|---|
| TLS | La stringa "TLS" |
| AEAD | Algoritmo AEAD usato per la protezione dei dati |
| HASH | Algoritmo di hash usato con HKDF |

# 2 Requisiti minimi lato server

Questa sezione fornisce una serie minima di requisiti che un server deve implementare per soddisfare queste linee guida, ma non ha l'intenzione di fornire alcuna indicazione implementativa.
Si ribadisce che il presente documento verrà aggiornato all'eventuale insorgere di problemi di sicurezza legati ai protocolli e/o algoritmi.

## 2.1 Versioni TLS raccomandate

I servizi esposti DEVONO utilizzare la versione TLS 1.2 o superiori e DOVREBBERO rifiutare versioni del protocollo inferiori. Versioni precedenti del protocollo sono insicure o contengono vulnerabilità note. Periodicamente bisogna controllare tutte le versioni e rimanere aggiornati per evitare configurazioni errate e nuove vulnerabilità.

## 2.2 Suite di cifratura raccomandate

Per agevolare la configurazione del protocollo TLS, Mozilla fornisce un tool[1] che genera configurazioni sicure dei suoi principali software. È RACCOMANDATO l'utilizzo delle configurazioni "**Modern**" (rivolte ai client con TLS 1.3 e che non necessitano di retro-compatibilità) e "**Intermediate**" (compatibili con la maggior parte dei client). Viceversa, è NON RACCOMANDATO l'utilizzo di configurazioni di tipo "**Old**" perché potrebbero includere cipher suite vulnerabili.

[1] https://wiki.mozilla.org/Security/Server_Side_TLS

Di seguito sono elencate le versioni minime dei client per ciascuna categoria di configurazione:

| Configuration | Firefox | Android | Chrome | Edge | Internet Explorer | Java | OpenSSL | Opera | Safari |
|---|---|---|---|---|---|---|---|---|---|
| **Modern** | 63 | 10.0 | 70 | 75 | -- | 11 | 1.1.1 | 57 | 12.1 |
| **Intermediate** | 27 | 4.4.2 | 31 | 12 | 11 (Win7) | 8u31 | 1.0.1 | 20 | 9 |
| **Old** | 1 | 2.3 | 1 | 12 | 8 (WinXP) | 6 | 0.9.8 | 5 | 1 |

## 2.1 Cipher suite Modern

Sono accettate le suite di cifratura con le seguenti caratteristiche:

- **Versione TLS**: **1.3 (la 1.2 non è accettata)**
- **Tipo di certificato**: ECDSA (P-256)
- **Curva TLS**: X25519, prime256v1, secp384r1
- **Durata del certificato:** 90 giorni

**Suite di cifratura (TLS 1.3):**

- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256

**Note**: Le suite di cifratura moderne sono più sicure, ma potrebbero non essere compatibili con client obsoleti, rendendo inutilizzabile l'applicazione.

## 2.2 Cipher suite Intermediate

Sono accettate le suite di cifratura con le seguenti caratteristiche:

- **Versione TLS: 1.3, 1.2**
- **Curva TLS:** X25519, prime256v1, secp384r1
- **Tipo di certificato**: ECDSA (P-256) (raccomandato) o RSA (2048 bits)
- **Durata del certificato:** da 90 giorni (raccomandato) a 366 giorni
- **Dimensione del parametro DH**: 2048 (solo per Intermediate RFC7919)

**Suite di cifratura (TLS 1.3):**

- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_CHACHA20_POLY1305_SHA256

**Suite di cifratura (TLS 1.2):**

- ECDHE-ECDSA-AES128-GCM-SHA256

- ECDHE-RSA-AES128-GCM-SHA256
- ECDHE-ECDSA-AES256-GCM-SHA384
- ECDHE-RSA-AES256-GCM-SHA384
- ECDHE-ECDSA-CHACHA20-POLY1305
- ECDHE-RSA-CHACHA20-POLY1305
- DHE-RSA-AES128-GCM-SHA256
- DHE-RSA-AES256-GCM-SHA384

# 3 Ulteriori raccomandazioni

## 3.1 Rinegoziazione della sessione

La rinegoziazione di una sessione TLS è vulnerabile ad una serie di attacchi. Le implementazioni utilizzate DEVONO rispettare le indicazioni contenute in [RFC5746]. Un server DOVREBBE rifiutare una rinegoziazione della sessione iniziata dal client.
Analogamente a quanto indicato in [RFC5756], quando si utilizza HTTP/2 [RFC 7540] con TLS 1.3, il servizio NON DEVE permettere la post-handshake authentication, come indicato in [RFC8740].
Considerazioni simili valgono in tutti i contesti in cui richieste multiple HTTP vengono trasmesse con meccanismi di multiplexing su singola connessione.

## 3.2 Compressione TLS

La compressione TLS DOVREBBE essere disabilitata; essa è stata rimossa dalla versione 1.3 di TLS perché sfruttata in passato da diversi exploit, tra cui il noto CRIME.

## 3.3 Estensione Heartbeat

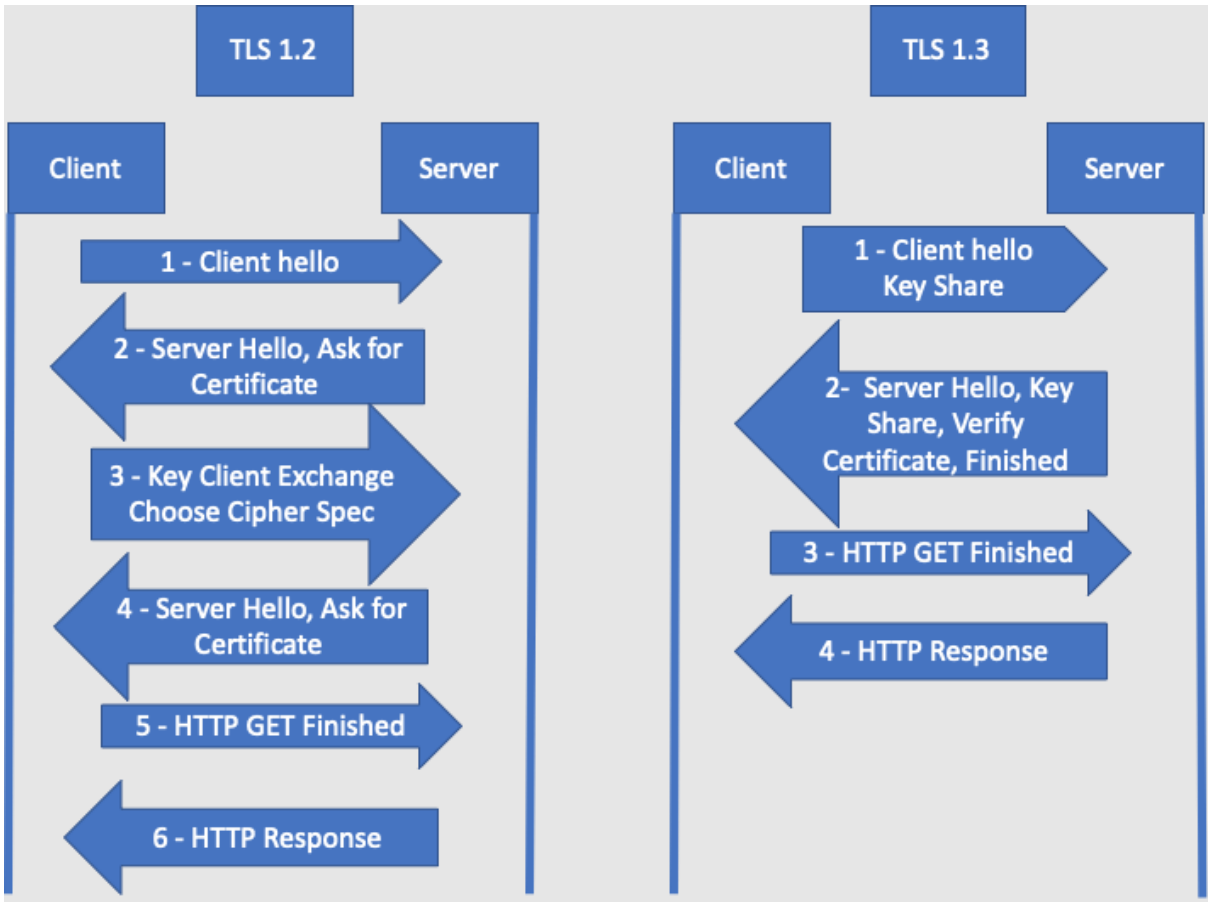
L'estensione Heartbeat specificata in [RFC6520] permette di prolungare la durata di una connessione TLS senza dover eseguire una rinegoziazione della sessione. Questa estensione è stata utilizzata in Heartbleed, con cui l'attaccante è in grado di accedere ad alcune aree di memoria del server che potrebbero contenere dati riservati. L'uso dell'estensione Heartbeat è NON RACCOMANDATO e nel caso fosse necessario il suo utilizzo, si raccomanda di verificare che non sia vulnerabile a *Heartbleed*.

# 4 TLS 1.2 vs TLS 1.3 versioni a confronto

Il protocollo TLS 1.3 è stato definito in RFC 8446 nell'agosto 2018 ed è più sicuro e veloce rispetto a TLS 1.2 RFC 5246. Le principali differenze includono:

- L'elenco degli algoritmi simmetrici supportati è stato epurato da tutti gli algoritmi legacy. Gli algoritmi rimanenti utilizzano algoritmi di crittografia autenticata con dati associati (AEAD) come ad esempio ChaCha20, Poly1305, Ed25519, x25519 e x448.
- È stata aggiunta una modalità zero-RTT (0-RTT) che elimina un round-trip durante la fase di configurazione della connessione.
- Le suite di cifratura statiche RSA e Diffie-Hellman sono state rimosse.
- Tutti i messaggi di handshake dopo "ServerHello" sono ora cifrati.
- Le funzioni di derivazione delle chiavi sono state ri-progettate, con la funzione di derivazione delle chiavi di estrazione ed espansione basata su HMAC (HKDF) utilizzata come primitiva.
- Gli stati dell'handshake sono stati ristrutturati per essere più coerenti e sono stati rimossi i messaggi superflui.
- ECC è ora nelle specifiche di base del TLS 1.3 e include nuovi algoritmi di firma.
- Sono stati inoltre apportati altri miglioramenti crittografici, tra cui:
    - la modifica del Padding RSA al fine di utilizzare lo schema RSA Probabilistic Signature Scheme (RSASSA-PSS) definito da RFC 8017;
    - la rimozione della compressione, dell'algoritmo di firma digitale (DSA) e dei gruppi Ephemeral Diffie Hellman (DHE) personalizzati.
- Il meccanismo di verifica della negoziazione della versione di TLS 1.2 è stato deprecato a favore di una lista di versioni gestite attraverso l'utilizzo di estensioni.
- La ripresa della sessione con e senza stato lato server e le ciphersuite basate su PSK delle versioni precedenti di TLS sono state sostituite da un unico nuovo scambio di chiavi PSK.
- In generale la direzione adottata da TLS 1.3 è verso algoritmi che garantiscano la PFS.

Come rappresentato nella seguente figura, per il TLS 1.2 sono necessari due round trip per completare l'handshake del TLS. La versione TLS 1.3 richiede un solo round-trip, che a sua volta diminuisce la latenza della crittografia. Si ottiene quindi un risparmio medio di tempo sull'handshake consentendo connessioni cifrate più veloci.




Inoltre, i meccanismi come TLS False Start e Zero Round Trip Time (0-RTT) del TLS 1.3 consentono di ridurre ulteriormente il tempo richiesto per l'handshake con gli host ai quali il client si è già collegato in precedenti sessioni.

Infine, TLS 1.3 offre una maggiore protezione contro i cosiddetti "downgrade attacks" ovvero i tentativi posti in essere da parte di un attaccante per indurre il server all'utilizzo di una vecchia versione del protocollo TLS soggetta a vulnerabilità note.