Sécurisation TLS 1.2 - Mise en œuvre à l'UT3

Capitoul - 11/02/2021

UT3
DSI - SSH
Philippe Balse

UNIVERSITÉ TOULOUSE III PAUL SABATIER
Université de Toulouse

Sécurisation TLS 1.2 - Mise en œuvre à l'UT3

Sommaire

- **Origines de la démarche**
- **Recommandations**
- **Mise en œuvre (configurations)**
- **Outils**
- **Bilan/limites**

Origines de la démarche

- Formations SIARS v2 CNRS (automne 2018)
  Présentation «Le chiffrement sur les services SSL/TLS»

- Recommandations ANSSI de sécurité relatives à TLS (mars 2020)

- Fin du support de TLS 1.0/1.1 dans les navigateurs
  Annoncé pour mars à juin 2020 (initialement)

TLS
Recommandations ANSSI

TLS (*Transport Layer Security*) est un protocole de sécurisation des flux réseau.

Modèle client-serveur, avec données applicatives encapsulées de manière à assurer:

- la confidentialité
- l'intégrité

et empêcher leur rejeu.

Le serveur est authentifié.

Un déploiement de TLS de qualité repose sur:

- l'utilisation de logiciels à jour
- l'ajustement des paramètres du protocole en fonction du contexte

➔ investissement

Condidentialité persistante (PFS)
Recommandations

Les suites cryptographiques (‘ciphersuites’) utilisées doivent assurer la propriété de **confidentialité persistante**, ou **PFS** (‘*Perfect Forward Secrecy*’)

- empêcher le déchiffrement de communications passées si la clé privée d’un correspondant est compromise

Négociation d’un ***secret éphémère***, à l’aide d’un échange de clé Diffie-Hellman, de type :

- ECDHE (courbes elliptiques)
- à défaut, DHE (groupes multiplicatifs)

Recommandations ANSSI:

- **ECDHE doit être privilégié**, avec courbes *secp521r1, secp384r1, secp256r1*
- DHE: au minimum groupes 2048 bits (>= 3072 bits préconisé par RGS)

Recommandations ANSSI (suite)

Authentification du serveur:

- mécanisme asymétrique
- ECDSA ou EdDSA à privilégier, RSA toléré

Taille des clés:

- ECDSA: 256 bits minimum
- RSA: 2048 bits minimum (avec exposant de la clé publique >= 65537)
  (OK avec Digicert ou Sectigo)

Chiffrement symétrique:
AES (256 ou 128 bits) à privilégier
ChaCha20: alternative acceptable

Fonction de hachage:
Famille SHA-2 doit être utilisée: SHA384, SHA256 (dans la pratique)

Suites cryptographiques recommandées
(pour TLS 1.2)

Clé publique ECDSA

Code TLS	Suite cryptographique
0xC02C	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
0xC02B	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
0xC0AD	TLS_ECDHE_ECDSA_WITH_AES_256_CCM
0xC0AC	TLS_ECDHE_ECDSA_WITH_AES_128_CCM
0xCCA9	TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256

Clé publique RSA

Code TLS	Suite cryptographique
0xC030	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
0xC02F	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
0xCCA8	TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256

En fournir plusieurs parmi celles proposées.
Ordre des suites: celui du serveur doit prévaloir sur celui du client.
Où sur serveur: dans configuration logicielle, ou au niveau «système»

Ordre des suites cryptographiques
Serveur, niveau «système»

Par ex., pour Linux + OpenSSL 1.1.1, cf */etc/pki/tls/openssl.cnf* ou */etc/ssl/openssl.cnf* (en fin de fichier) - *man config(5) :*

```
...
[default_conf]
ssl_conf = ssl_sect

[ssl_sect]
system_default = system_default_sect

# Ajout de Options = ServerPreference
# Nov. 2020 - DSI/SSH
[system_default_sect]
MinProtocol = TLSv1.2
Options = ServerPreference
CipherString = DEFAULT@SECLEVEL=2
```

Suites dégradées adoptables
(pour TLS 1.2 - ANSSI)

Code TLS	Suite cryptographique
0x009F	TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
0x009E	TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
0xCCAA	TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256
0xC09E	TLS_DHE_RSA_WITH_AES_128_CCM
0x0067	TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
0x006B	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
0xC024	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
0xC023	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
0xC028	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
0xC027	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256

A utiliser si on ne peut pas faire autrement.

Recommandations applicables pour …

Java

Version 8 (à confirmer par la pratique) et supérieures

OS

- Debian: versions 7 à 10
- CentOS/RedHat: versions 7 et 8 (v6 probablement OK)
- Windows Server: à partir des versions 2016

✔ Modifier la 'Group Policy', par *gpedit.msc* (ligne de commande)
Appliquer les suites et courbes recommandées

✔ désactiver TLS 1.0 et TLS 1.1 dans la base de registres:

Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols

Cf: https://saputra.org/threads/disable-tls-1-1-and-1-0-on-windows-server-2019-with-iis-10-0.148

Partiellement:

- Windows Server 2012 R2 (*ECDHE-RSA-AES*-GCM-SHA** pas supportés)

Configurations/cas pratiques
Apache

Debian 10

```
SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
SSLCipherSuite EECDH+AESGCM:EEDH+AES
SSLCipherSuite TLSv1.3 TLS_AES_256_GCM_SHA384:TLS_AES_128_GCM_SHA256:TLS_AES_128_CCM_SHA256:TLS_CHACHA20_POLY1305_SHA256
SSLHonorCipherOrder  on
SSLSessionTickets   off
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
SSLOpenSSLConfCmd Curves secp521r1:secp384r1:prime256v1
SSLOpenSSLConfCmd ECDHParameters Automatic
```

Debian 9

```
SSLProtocol all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
SSLCipherSuite EECDH+AESGCM:EEDH+AES
SSLHonorCipherOrder on
SSLSessionTickets off
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
SSLOpenSSLConfCmd Curves secp521r1:secp384r1:prime256v1
SSLOpenSSLConfCmd ECDHParameters Automatic
```

Debian 8 et 7

```
SSLProtocol all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
SSLCipherSuite EECDH+AESGCM:EEDH+AES
SSLHonorCipherOrder on
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
```

CentOS 7/Redhat 7: idem que Debian 9, sauf directive *SSLOpenSSLConfCmd* non supportée.

Configurations/cas pratiques
HAProxy/Nginx

HAProxy (v1.8/Debian 9)

```
global
...
        ssl-default-bind-ciphers EECDH+AESGCM:EEDH+AES
        ssl-default-bind-options no-sslv3 no-tlsv10 no-tlsv11
        #
        tune.ssl.default-dh-param 2048
...
frontend xxx_front
   bind *:443 ssl crt /etc/ssl/private/xxx.univ-tlse3.fr.pem ecdhe secp384r1
   ...
   rspadd Strict-Transport-Security:\ max-age=31536000;\ includeSubDomains
```

Nginx

```
ssl_protocols TLSv1.2;
ssl_ciphers "EECDH+AESGCM:EEDH+AES";
ssl_prefer_server_ciphers on;
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains";

# Si versions Nginx >= 1.11 et OpenSSL >= 1.0.2
# Cf http://nginx.org/en/docs/http/ngx_http_ssl_module.html#ssl_ecdh_curve
# Sinon, se contenter de la valeur par defaut
ssl_ecdh_curve secp521r1:secp384r1:prime256v1;
```

Configurations/cas pratiques
vsftpd - OpenLDAP

vsftpd (Debian 9) **:**

ssl_ciphers=EECDH+AESGCM:EEDH+AES
ssl_tlsv1=YES

- TLS 1.2 uniquement disponible

OpenLDAP (Debian 10 – v2.4.56, compilée avec OpenSSL)

TLSCipherSuite EECDH+AESGCM:DHE+AESGCM:!DSS
 # TLSECName possible uniquement si OpenSSL
TLSECName secp521r1:secp384r1:prime256v1
 # Paramètre nécessaire si on spécifie des ciphers DHE- dans TLSCipherSuite*
 # Fichier à créer au préalable avec 'openssl dhparam -out dhparam.pem 3072'
TLSDHParamFile /usr/local/openldap/etc/openldap/certs/dhparam.pem
 # Pour TLS >= v1.2
 # TLS 1.2 minimum également possible au niveau «système» dans Debian 10
 # Cf /etc/ssl/openssl.cnf, pour choix de ciphersuite par serveur
TLSProtocolMin 3.3

Configurations/cas pratiques
Postfix (1/2)

Postfix 2.11 (Debian 8)

```
# smtpd_tls_security_level remplace smtpd_use_tls depuis Postfix 2.3
smtpd_tls_security_level = may
smtpd_tls_protocols = TLSv1.2, !TLSv1.1, !TLSv1, !SSLv2, !SSLv3
smtpd_tls_ciphers = high
smtpd_tls_exclude_ciphers = aNULL, eNULL, MD5, LOW, MEDIUM, RC4, ADH, SHA
tls_preempt_cipherlist = yes
# Utiles si smtpd_tls_security_level = encrypt (i.e. 'mandatory TLS encryption')
smtpd_tls_mandatory_ciphers = high
smtpd_tls_mandatory_exclude_ciphers = aNULL, eNULL, MD5, LOW, MEDIUM, RC4, ADH, SHA
```

Postfix 3.4 (Debian 10)

```
smtpd_tls_security_level = may
smtpd_tls_protocols = TLSv1.3, TLSv1.2, !TLSv1.1, !TLSv1, !SSLv2, !SSLv3
smtpd_tls_ciphers = high
tls_high_cipherlist = EECDH+AESGCM:DHE+AESGCM:!DSS
smtpd_tls_exclude_ciphers = aNULL, eNULL, MD5, LOW, MEDIUM, RC4, ADH, SHA
tls_preempt_cipherlist = yes
# Utiles si smtpd_tls_security_level = encrypt (i.e. 'mandatory TLS encryption')
smtpd_tls_mandatory_ciphers = high
smtpd_tls_mandatory_exclude_ciphers = aNULL, eNULL, MD5, LOW, MEDIUM, RC4, ADH, SHA

# Paramétrage courbes ECDH (pas testé)
# smtpd_tls_eecdh_grade = auto
# Nom 'secp256r1' pas reconnu => prime256v1
# tls_eecdh_auto_curves = secp521r1 secp384r1 prime256v1
```

Configurations/cas pratiques
Postfix (2/2)

Les directives possibles évoluent selon les versions de Postfix

- consulter attentivement les documentations Postfix, qui parlent de ces variations:

- '*Postfix configuration parameters*'
- '*Postfix TLS Support*'
- '*TLS Forward Secrecy in Postfix*'

Configurations/cas pratiques
Sendmail sur CentOS 5

Contexte:

- Passerelles sortantes (pour utilisateurs authentifiés) ne supportant que TLS ≤ 1.0
- CentOS 5 final, OpenSSL 0.9.8 (natif), sendmail 8.14.5 (compilé)
- ***Juillet 2020: sortie de Thunderbird 78***, ne supportant que TLS ≥ 1.2
- Pas les ressources pour les remplacer rapidement => danger !

Solution:

- Compilation d'OpenSSL 1.0.2u (déc. 2019), pour support TLS 1.2
- Compilation de la dernière version sendmail 8.14.x, avec dépendance OpenSSL 1.0.2u
  + flags sendmail non documentés pour TLS et ECDH

➢ proche de la cible des recommandations ANSSI (delta: ancienneté de clients => 2 ciphersuites 'basses')

Configuration:

```
# Supporté à partir de sendmail 8.14.8
O DHParameters=2048

O CipherList=EECDH+AESGCM:EDH+AESGCM:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:!DSS
O ServerSSLOptions=+SSL_OP_NO_SSLv2 +SSL_OP_NO_SSLv3 +SSL_OP_NO_TLSv1_1
+SSL_OP_CIPHER_SERVER_PREFERENCE
O ClientSSLOptions=+SSL_OP_NO_SSLv2 +SSL_OP_NO_SSLv3 +SSL_OP_NO_TLSv1 +SSL_OP_NO_TLSv1_1
```

Outils

En ligne de commande:

- ***openssl s_client*** [*-tls1*] [*-tls1_2*] [*-tls1_3*] -connect serveur:port [*-starttls smtp|ftp*] ...
- ***nmap –script ssl-enum-ciphers*** -p port serveur
- ***ssldump***
- ***testssl.sh***
- ***openssl ciphers*** -v

Web:

- SSL Labs
- Cryptcheck

Bilan/limites

- Cible «TLS 1.2 + recommandations ANSSI» majoritairement atteinte
- Investissement important (auto-formation, tests, recherches logs, planifications pour maj configurations, …)
- Maîtrise du parc: serveurs, et clients (incluant postes utilisateurs)
- Applications métier: (im)possibilité de réglages TLS ?
- Windows: impact des réglages TLS sur les accès RDP difficile à anticiper
- GnuTLS: configurations présentées ici (pour OpenSSL) à refaire
- Suivre l'état de l'art si possible
- Bonus : prérequis HTTP/2 rempli (cf RFC7540 - '*TLS 1.2 ciphersuite blacklist*')

TLS 1.3 reste la cible finale à atteindre.

Bibliographie

- Dans l'enfer du SSL
- Mapping ciphersuite names: OpenSSL vs IANA
- TLS & Perfect Forward Secrecy
- RFC 8422: ECC Cipher Suites for TLS 1.2 and Earlier (blog S. Bortzmeyer)
- Taille des clés ECC vs RSA
- Windows versions and their cipher suites
- Analyses des configurations SSL/TLS de serveurs SMTP (article MISC mars 2018)
- Additional information on Oracle's JDK and JRE Cryptographic Algorithms
- Capturing specific SSL and TLS version packets using tcpdump

Péremption future d'informations dans ces pages (selon état de l'art du moment)