



INFRASTRUCTURE DE PRODUCTION

SÉCURISER LES COMPOSANTS

00.00.2017

Photo by Diogène Morin on [Unsplash](#)

TABLE DES MATIÈRES

1 | SÉCURISER SES DÉVELOPPEMENTS

2 | SSL

3 | AUTHENTIFICATION

4 | SSO

Photo by Martha Dominguez de Gouveia on [Unsplash](#)



1

**SÉCURISER SES
DÉVELOPPEMENTS**

OWASP (OPEN WEB APPLICATION SECURITY PROJECT)

- guide de sécurisation des applications web
- référence des bonnes/mauvaises pratiques de développement
- Plusieurs projets :
 - **Top Ten** : Liste des failles les plus couramment utilisées par des utilisateurs malintentionnés sur internet
 - **Webscarab** : Un outil d'audit de sécurité.
 - **Webgoat** : Il s'agit cette fois d'une application web volontairement non sécurisée.

SURVEILLER LES LIBRAIRIES EMBARQUÉES

COMPANY	DATE	RESULTS	REFERENCE
Wordpress	2017-10	SQLi vulnerability in plugins - patched in v4.8.3	If your websites use WordPress, put down that coffee and upgrade to 4.8.3
GoDaddy	2017-10	Researcher showed site security tool is easily bypassed to allow sql	This bug let a researcher bypass GoDaddy's site security tool
Inmarsat	2017-10	AmosConnect satellite communications for ships is vulnerable.	mosConnect: Maritime Communications Security Has Its Flaws
Equifax	2017-10 (long term)	Personal data for over 145 million people compromised	Equifax was warned
Jigsaw Holding in South Africa	2017-10	75m database records available for download	Revealed the real source of SAs massive data breach
Catholic United Financial	2017-10	personal info for over 130k users.	Data breach at Arden Hills-based Catholic financial services provider affects nearly 130k accounts
BPC Banking - SmartVista software	2017-10	numerous vulnerable ecommerce sites	Vendor BCP Baking Silent on Patching SQL Injection in SmartVista Ecommerce Software
EMC	2017-07	multiple vulnerabilities found, some	EMC products hit by multiple vulnerabilities including SQL injection

Source: <http://codecurmudgeon.com/wp/sql-injection-hall-of-shame/>

Severe security vulnerability found in Apache Struts using lgtn.com (CVE-2017-9805)

September 05, 2017 Posted by Bas van Schaik



Originally published on 5 September 15:30 BST. Updated on 6 September: added a warning regarding multiple working exploits having been published by third parties. Included details of Struts version 2.3.34

Security researchers at lgtn.com have discovered a critical remote code execution vulnerability in Apache Struts — a popular open-source framework for developing web applications in the Java programming language. All versions of Struts since 2008 are affected; all web applications using the framework's popular REST plugin are vulnerable. Shortly after the patched versions of Struts were released on 5 September, multiple working exploits were observed on various internet sites. Users are strongly advised to upgrade their Apache Struts components as a matter of urgency. This vulnerability has been addressed in Struts versions 2.3.34 and 2.5.13.

lgtn provides free software engineering analytics for open-source projects; at the time this post is published, over 50,000 projects are continuously monitored. Anyone can write their own analyses; ranging from checks for enforcing good coding practices to advanced analyses to find security vulnerabilities. The lgtn security team actively helps the open-source community to uncover critical security vulnerabilities in OSS projects.

This particular vulnerability allows a remote attacker to execute arbitrary code on any server running an application built using the Struts framework and the popular REST communication plugin. The weakness is caused by the way Struts *deserializes* untrusted data. The lgtn security team have a simple working exploit for this vulnerability which will not be published at this stage. At the time of the announcement there is no suggestion that an exploit is publicly available, but it is likely that there will be one soon.

The Apache Struts development team have confirmed the severity of this issue and released a patch today:

“ This is critical, as all you have to do is use the REST plugin.

”

LA LISTE DES VULNÉRABILITÉS

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	⊗	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	⊗	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

LES ATTACKS

■ 2017-A3 – Sensitive Data Exposure

- Insecure Cryptographic Storage
- Insufficient Transport Layer Protection

■ 2017-A4 – XML eXternal Entity (XXE) Attack

■ 2017-A8 – Insecure Deserialization

- CVE-2017-5954
- CVE-2017-9424
- CVE-2017-9805
- CVE-2017-1000034

■ 2017-A9 – Using Known Vulnerable Components

■ 2017-A10 – Insufficient Logging & Monitoring

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE meh [<!ENTITY xxeFun SYSTEM "file:///etc/ passwd">
]>>
<someStuff>
<isHere> Hi! &xxeFun; </isHere>
</someStuff>
```

Dependencies										
Status	Group Id	Artifact Id	Current Version	Scope	Classifier	Type	Next Version	Next Incremental	Next Minor	Next Major
⚠	com.fasterxml.jackson.core	jackson-annotations	2.0.4	compile		jar		2.0.5	2.1.0	
⚠	com.fasterxml.jackson.core	jackson-core	2.0.4	compile		jar		2.0.5	2.1.0	
⚠	com.fasterxml.jackson.core	jackson-databind	2.0.4	compile		jar		2.0.5	2.1.0	
⚠	com.google.guava	guava	11.0	compile		jar		11.0.1	12.0-rc1	12.0
⚠	com.ibm.icu	icu4j	49.1	compile		jar				50.1
⚠	com.theoryinpractise	halbuilder	1.0.4	compile		jar		1.0.5		
⚠	commons-codec	commons-codec	1.3	compile		jar			1.4	
⚠	commons-logging	commons-logging	1.1.1	compile		jar				
⚠	joda-time	joda-time	2.0	compile		jar			2.1	
⚠	net.sf.ehcache	ehcache-core	2.5.1	compile		jar		2.5.2	2.6.0	
⚠	org.apache.httpcomponents	httpclient	4.1.2	compile		jar		4.1.3	4.2	
⚠	org.apache.httpcomponents	httpclient-cache	4.1.2	compile		jar		4.1.3	4.2	
⚠	org.apache.httpcomponents	httpcore	4.1.2	compile		jar		4.1.3	4.2	
⚠	org.jdom	jdom	1.1	compile		jar		1.1.2		2.0.0
✅	org.slf4j	slf4j-api	1.7.2	provided		jar				



2

SSL & TLS

SSL (Secure Sockets Layer)
TLS (Transport Layer Security)

SSL / TLS

- SSL/TLS : protocole pour créer un **canal** de communication **authentifié**, protégé en **confidentialité** et en **intégrité**.
- L'objectif : Sécurisation du protocole **HTTP**
- Son champ d'application s'est élargi depuis : protection d'autres services comme **SMTP** ou **LDAP**, création de réseaux privés virtuels (**VPN**), sécurisation de réseaux **sans-fil** (EAP-TLS).

HISTORIQUE

<https://www.ssi.gouv.fr/guide/recommandations-de-securite-relatives-a-tls/>

- SSL 1.0 -1994 – Netscape
- L'objectif de Netscape était de créer un canal sécurisé entre le client et le serveur pour faire transiter des données sensibles (Numéro de carte de crédit par exemple)
- SSLv2 → 1995
- SSLv3 → 1996 – RFC 6101

- La normalisation par l'IETF (Internet Engineering Task Force) : TLS
 - SSL a fait l'objet d'une normalisation par l'IETF appelée TLS (Transport Layer Security)
 - TLS 1.0 - Janvier 1999 – IETF (Internet Engineering Task Force) RFC 2246
 - TLS 1.1 – Avril 2006 –RFC 4346
 - TLS 1.2 – Août 2008 – RFC 5246
 - TLS 1.3 – a venir, plus de simplicité, de vitesse et de sécurité.



« Le protocole TLS fourni une sécurité au communication sur internet. Le protocole autorise les clients et serveurs a communiquer de manière à empêcher l'écoute, la falsification ou la fabrication de messages. »

The Transport Layer Security (TLS) Protocol Version 1.2

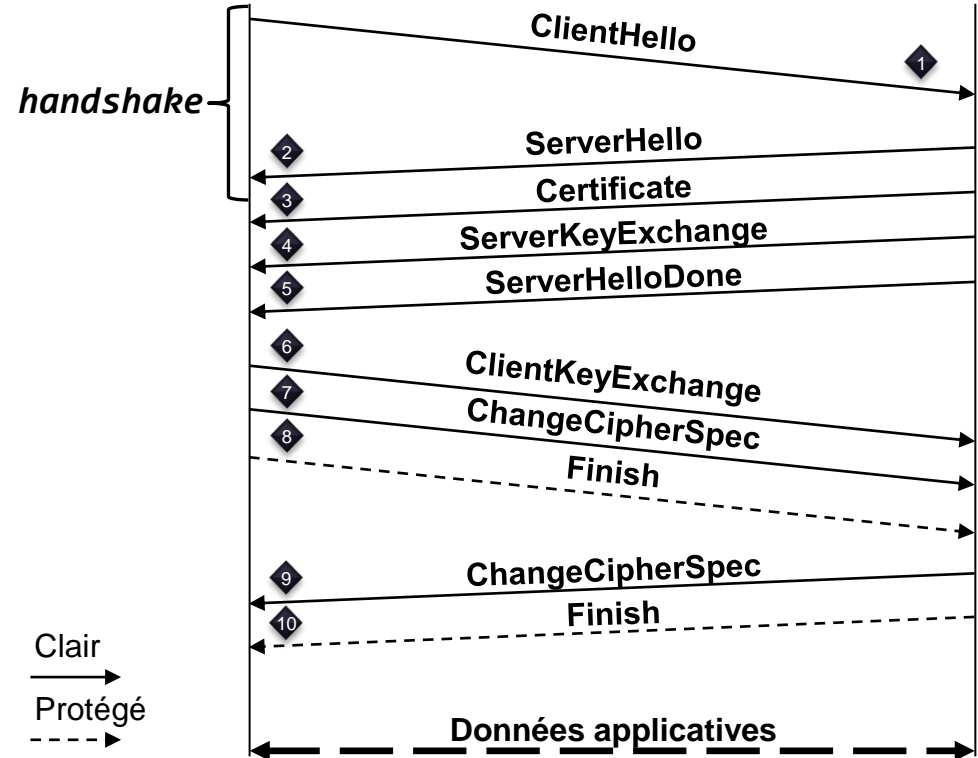
SSL / TLS

- **Protocole situé entre le niveau transport et le niveau application dans le modèle OSI et TCP**
- **SSL et TLS : nouvelle couche**
- **Objectif :**
 - Authentification
 - Confidentialité
 - Identification et intégrité
- **Mécanisme**
 - chiffrement asymétrique
 - chiffrement symétrique
- **SSL peut être découpé en deux parties :**
 - SSL et TLS Handshake Protocol
 - SSL et TLS Record Protocol

Modèle OSI	TCP/IP
Application	Application
Présentation	
Session	
Transport	Transport
Réseau	Réseau
Liaison	Liaison
Physique	Physique

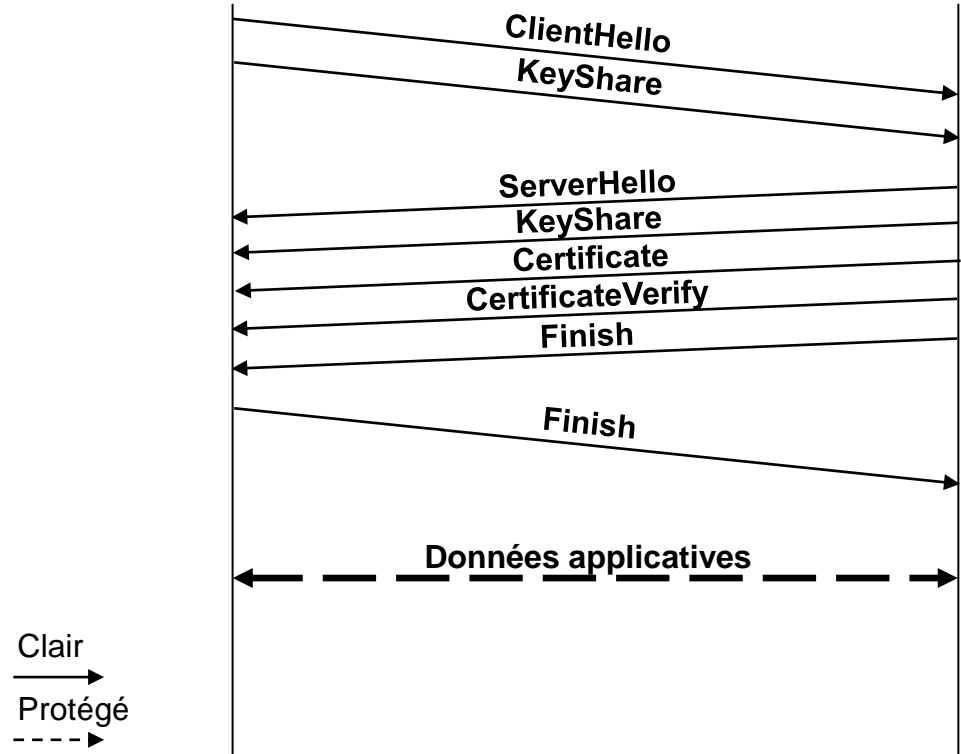
EXEMPLE DE NÉGOCIATION TLS 1.0 - 1.2

■ Avec la suite TLS



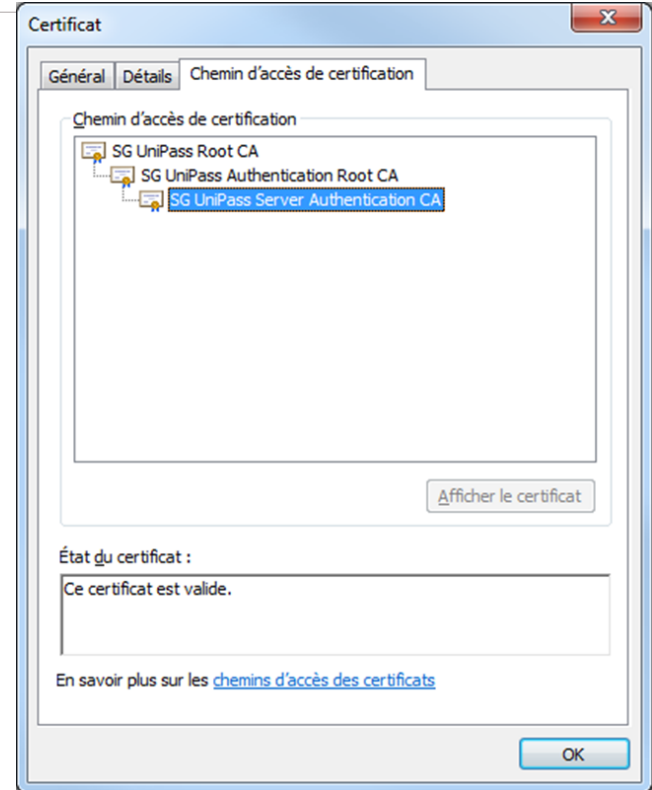
EXEMPLE DE NÉGOCIATION TLS 1.3

■ Avec la suite TLS 1.3



JAVA ET SSL

- Pour utiliser le SSL / TLS il faut d'abord avoir un certificat :
- Un certificat numérique signé est un standard de l'industrie permettant de vérifier l'authenticité d'une entité, comme un serveur, un client, une application. Pour assurer une sécurité maximal, un certificat est fourni par une autorité tierce (CA), dite de confiance. Exemple Verisign
 - Toute les autorités tierce de confiance ont leurs certificats enregistré sur les postes utilisateurs.
- L'ensemble des ces certificats sont dans un magasin de certificat, possédant plusieurs niveaux :
 - Premier niveau
 - Niveau intermédiaire
 - Niveau personnel



LE MAGASIN DE CERTIFICAT JAVA

- **Keytool : Client java, fourni en standard, pour gérer le magasin de certificats de java**

- Ce magasin est différent de celui de Windows

#Show Keys & Certs in Keystore

```
keytool -list -v -keystore keystore -storepass changeit
```

#Show Certs in the Truststore

```
keytool -list -v -keystore cacerts -storepass changeit
```

- **Le magasin par défaut de java est situer dans :**

- **Le mot de passe : changeit**

- **Definir ses propres magasins**

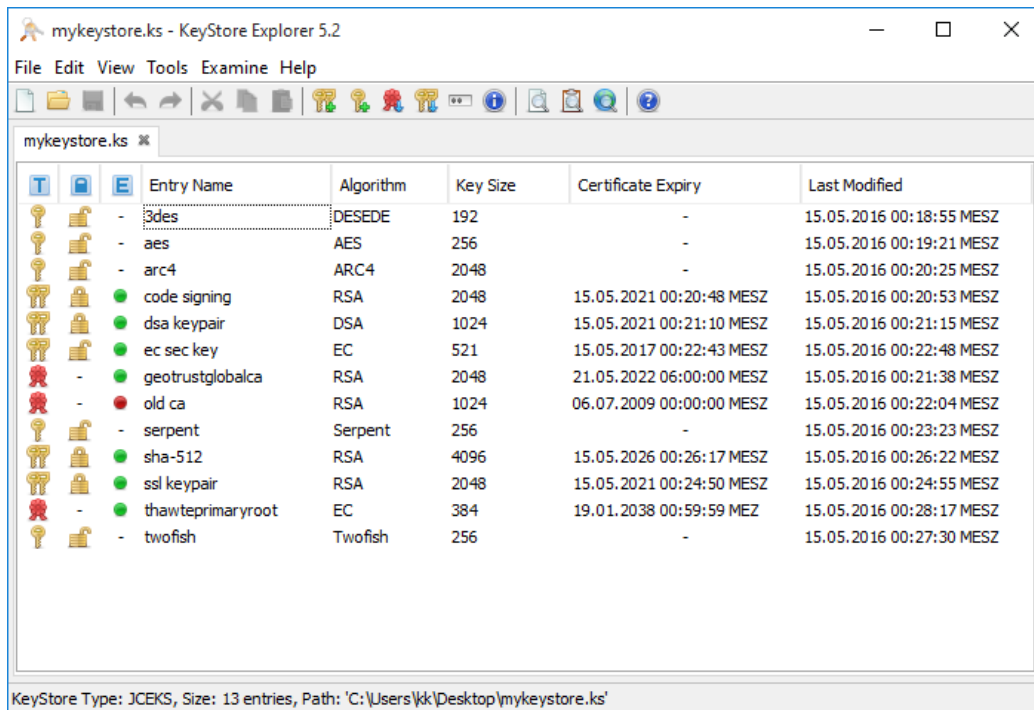
JRE_HOME/lib/securty/cacerts

```
-Djavax.net.ssl.keyStore=keystore.jks
```

```
-Djavax.net.ssl.trustStore=cacerts.jks
```


LE MAGASIN DE CERTIFICAT JAVA

- Un outil plus pratique que keytool : KeyStore explorer



SSL / TLS – TOMCAT ET SPRING BOOT

- Sécurisation de tomcat
- Ajout de SSL dans la configuration de tomcat

```
# Configuration SSL
security.require-ssl=true
server.ssl.enabled=true
server.ssl.key-store=classpath:server-keystore.jks
server.ssl.key-store-type=JKS
server.ssl.key-store-password=*****
server.ssl.key-alias=server
server.ssl.key-password=123456
```

JAVA JSSE

- JSSE = Java Secure Socket Extension : package par défaut pour la sécurité
- Optionnel avant le JDK 1.4. Intégré en standard depuis
- API difficile a utiliser
- Préférer les API permettant de s'affranchir de toute la complexité de la sécurité

```
Request.Get("http://localhost:8443/")  
    .connectTimeout(1000)  
    .socketTimeout(1000)  
    .execute().returnContent().asString();
```

JAVA JSSE

■ Le code précédent ne fonctionne pas

```
javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: PKIX
path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable
to find valid certification path to requested target
    at sun.security.ssl.Alerts.getSSLException(Alerts.java:192)
    at sun.security.ssl.SSLSocketImpl.fatal(SSLSocketImpl.java:1627)
    at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:204)
    at sun.security.ssl.Handshaker.fatalSE(Handshaker.java:198)
    at
sun.security.ssl.ClientHandshaker.serverCertificate(ClientHandshaker.java:994)
    at sun.security.ssl.ClientHandshaker.processMessage(ClientHandshaker.java:142)
    at sun.security.ssl.Handshaker.processLoop(Handshaker.java:533)
    at sun.security.ssl.Handshaker.process_record(Handshaker.java:471)
    at sun.security.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:904)
    ...
```

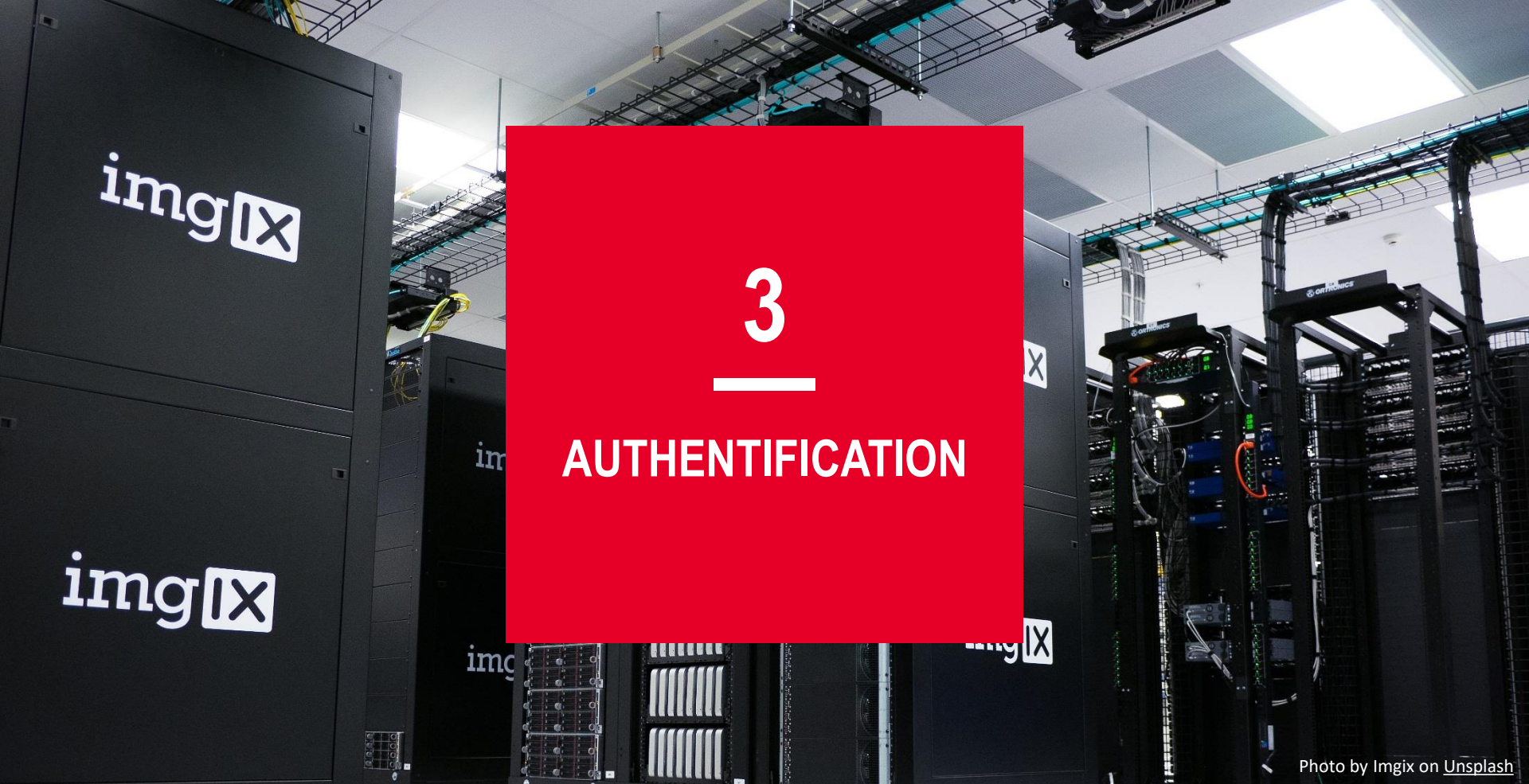


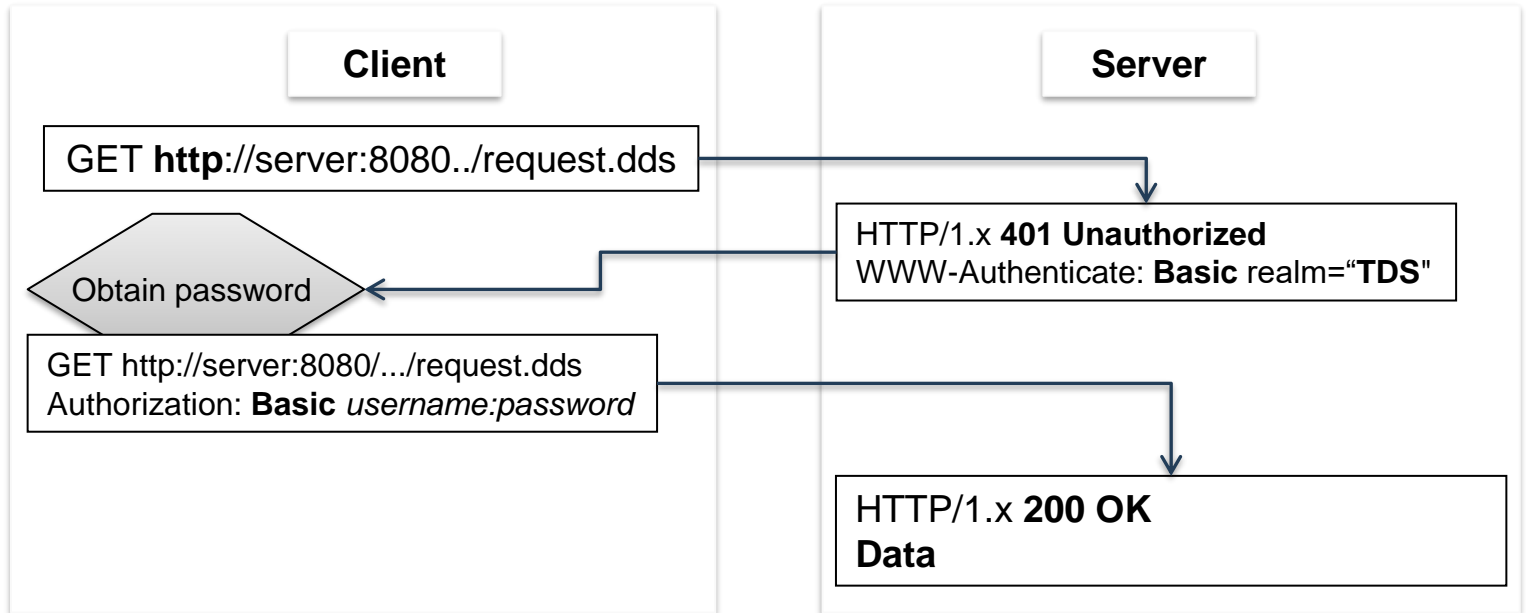
Photo by [Imgix on Unsplash](#)

AUTHENTIFICATION

- **Protéger les données d'une application et s'assurer que la personne qui consulte ces données est bien la personne qui peut y avoir accès**
- **Il existe plusieurs niveaux et plusieurs mécanismes d'authentification**
 - Des niveaux de dureté (facilité à casser le code d'accès)
 - Protocole et fonctionnement de l'authentification
- **Ceux que vont être présentés :**
 - Basic authentication
 - Form-login authentication
 - SSL-Mutuel (présentation d'un certificat)

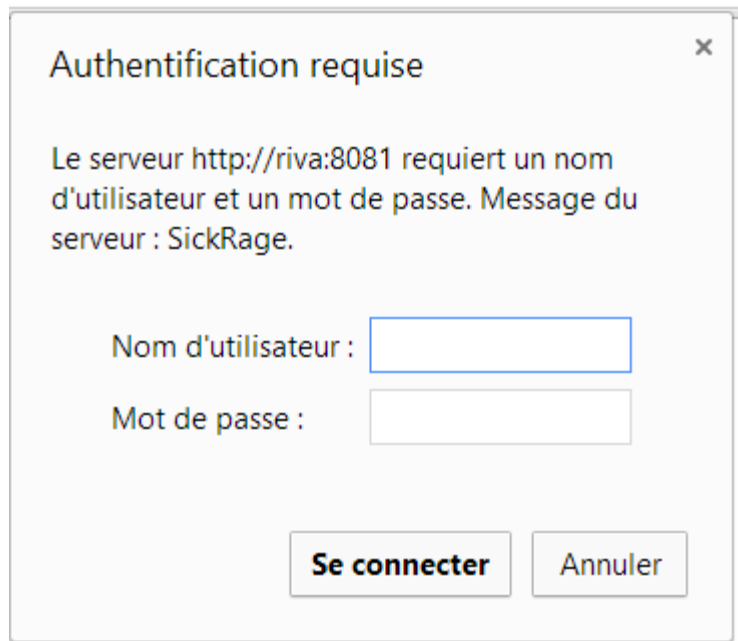
BASIC AUTHENTICATION

- Mécanisme standard spécifiée par la RFC 2617
- la L'authentification HTTP ou identification HTTP est un mécanisme à base de username et de mot de passe qui sont transmis dans l'entête http



BASIC AUTHENTICATION

- Mécanisme le plus simple à mettre en œuvre
- Ouverture d'une popup navigateur (non configurable)
- Ne permet pas des ajouts comme un clavier virtuel



A screenshot of a browser authentication dialog box. The title bar says 'Authentication requise' with a close button (X) in the top right corner. The main text reads: 'Le serveur http://riva:8081 requiert un nom d'utilisateur et un mot de passe. Message du serveur : SickRage.' Below this, there are two input fields. The first is labeled 'Nom d'utilisateur :' and the second is labeled 'Mot de passe :'. At the bottom right, there are two buttons: 'Se connecter' and 'Annuler'.

- HTTP+HTML form-based authentication
- Permet de proposer n'importe qu'elle type de forme d'authentification
- Toujours associé à un transport SSL



The image shows a login form for a system named 'Connexion'. At the top, there is a logo consisting of a stylized elephant head in grey, with the word 'Connexion' in green text below it. The form itself is a white box with a thin grey border. It contains two input fields: the first is labeled 'Adresse e-mail ou nom d'utilisateur' and the second is labeled 'Mot de passe'. Both fields have a small icon of a person with a plus sign in the bottom right corner. Below the password field is a checkbox labeled 'Mémoriser une semaine'. At the bottom of the form is a large green button with the text 'Connexion' in white. Below the button is a link that says 'Mot de passe oublié ?' in blue text.



Java SE Security





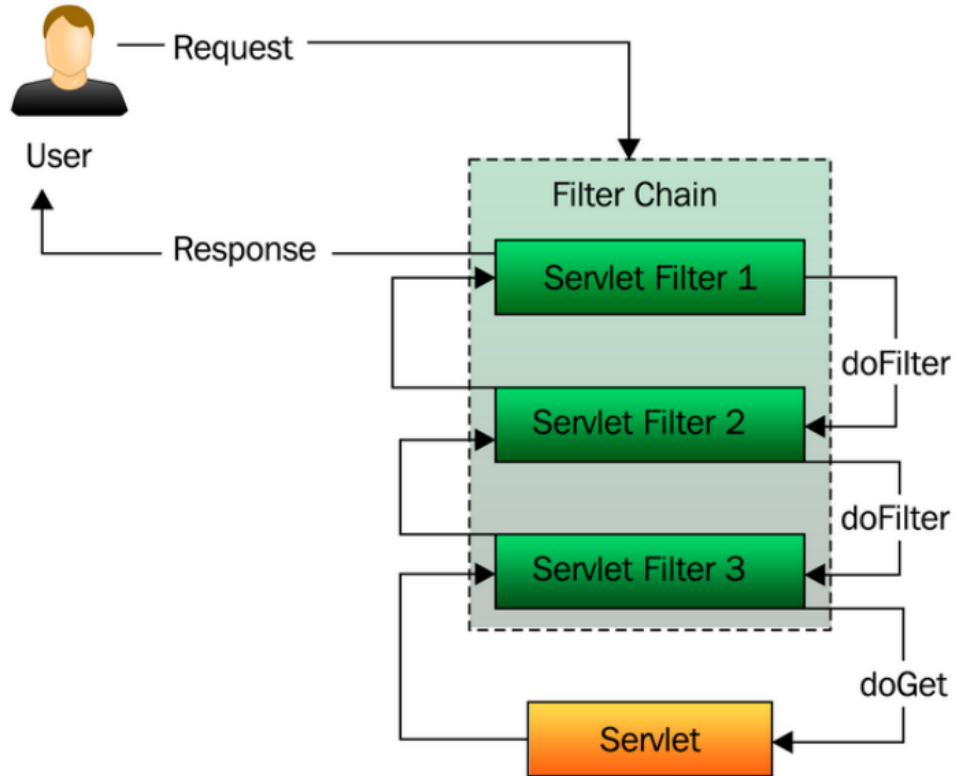
- Fourni une Authentication et des services Authorization
- Authorization est base sur des Access Control List
- Multiple provider
 - Oauth
 - CAS
 - Properties
 - ...
- Extensible avec la chaine de filtre
- Configuration des restrictions sur les urls



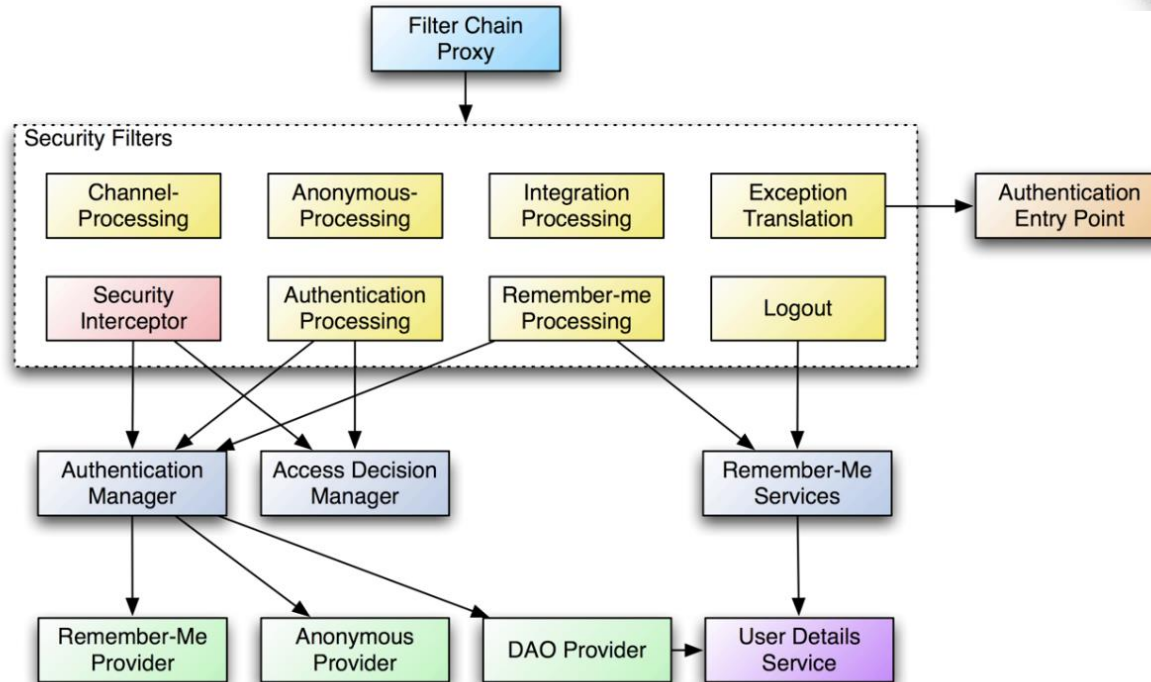
- **A l'origine : ACEGI project**
 - Créé par Ben Alex en 2003
- **Configuration par XML**
- **Renommé en “spring security” au passage de la version 2.0**
- **Ajout du namespace Security**

- **Déclaration de toute la sécurité en dehors du code par déclaration**

SPRING SECURITY



SPRING SECURITY



Filtre de sécurité proposé

- Form
- Basic
- LDAP
- Kerberos
- Container (eg. Tomcat)
- JAAS
- JA-SIG CAS
- OpenID
- SiteMinder
- Atlassian Crowd
- OpenID
- X.509
- Digest

Ordre des filtres

- CHANNEL_FILTER
- SECURITY_CONTEXT_FILTER
- CONCURRENT_SESSION_FILTER
- HEADERS_FILTER
- CSRF_FILTER
- LOGOUT_FILTER
- X509_FILTER
- PRE_AUTH_FILTER
- CAS_FILTER
- FORM_LOGIN_FILTER
- BASIC_AUTH_FILTER
- SERVLET_API_SUPPORT_FILTER
- JAAS_API_SUPPORT_FILTER
- REMEMBER_ME_FILTER
- ANONYMOUS_FILTER
- SESSION_MANAGEMENT_FILTER
- EXCEPTION_TRANSLATION_FILTER
- FILTER_SECURITY_INTERCEPTOR
- SWITCH_USER_FILTER

4

SSO

SSO – LA PROBLÉMATIQUE

- **Mot de passe pour tous les systèmes**

- Windows
- Mail
- Application
- Site
- ...

- **Mécanisme pour retenir tous les mots passes peut, voire pas du tout, sécuriser**

- **La multiplication des systèmes de sécurité devient à terme contre-productive.**
- **Ce n'est pas à l'utilisateur de centraliser les informations de sécurité mais au système informatique !**

SSO – PRÉSENTATION

- **Single Sign On : Authentification unique**
- **Permet à un utilisateur d'accéder à plusieurs services en ayant à effectuer qu'une opération d'authentification**
- **L'information d'authentification se propage sur chacun des services réseau.**

- **Les avantages de ce mécanismes sont multiples :**
 - Simplification de l'authentification avec moins de mots de passe à retenir
 - Sécurité améliorée avec centralisation des données de sécurités stockées et gérées par un serveur spécialisé
 - Facilité de maintenance des habilitations.

- **Le corolaire des avantages est l'unicité du mot de passe. S'il est dérobé le pirate aura accès à tous les système**

SSO – PRINCIPE

■ Concepts inspirés de Kerberos :

- Authentification assurée par un serveur dédié et transparente pour l'application (pas de recueil du couple identifiant+mot de passe)
- **Tickets** délivrés au client (maintien de la session d'authentification) et aux applications (transmission de l'identité de l'utilisateur)
- Relation de **confiance** entre les **applications** et le **serveur** d'authentification (cryptographie symétrique ou asymétriques, certificats X509)

■ Le type de SSO dont on va développer les éléments est le Web SSO qui s'adresse principalement aux applications Web

SSO – TERME ASSOCIÉ

- **Identification / Authentification / Autorisation**

- **Identification : vérification de l'identité de la personne se connectant via un login**
 - **Authentification : Vérification que la personne se connectant est bien la bonne personne grâce à un login et un mot de passe**
 - **Autorisation : Vérification des droits de l'utilisateur connecté**
 - **Le SSO est principalement orienté Authentification**
-
- **Authentification forte**
 - **Fédération d'identité**

SSO – LES PROTOCOLES

■ OAuth2

Ce protocole permet à des applications tierces d'obtenir un accès limité à un service disponible via HTTP par le biais d'une autorisation préalable du détenteur des ressources.

■ SAML

Security assertion markup language (**SAML**) est un standard informatique définissant un protocole pour échanger des informations liées à la sécurité.

■ OpenId

C'est un site web, un service web... Le fournisseur d'identité (**OpenID** Provider, ou OP, ou IdP, ou IP ou juste provider) est un serveur d'authentification **OpenID** qui est contacté par le service pour obtenir une preuve de l'identité de l'utilisateur.

■ Kerberos

Protocole d'authentification réseau qui repose sur un mécanisme de clés secrètes (chiffrement symétrique) et l'utilisation de tickets,

SSO – LES APPLICATIONS

- **CAS :**

- Système de Web SSO réparti, avec un serveur SSO, un serveur d'authentification et éventuellement un proxy.

- **Shibboleth**

- Complément a cas en proposant une fédération d'identité et en utilisant le protocole SAML

- **Keycloak :**

- Système SSO de JBOSS qui gère l'authentification et les accès d'une personne.

- **OpenAM**

- **WSO2 Identity Server**

- **Gluu**

- **Crowd**

SSO - CROWD

- **Serveur de SSO (Single Sign-On)**

- **2 niveaux d'autorisation :**
 - Application autorisée à se connecter à crowd
 - Utilisateur autorisée sur l'application

- **Multiples sources :**
 - Base de données
 - Annuaire LDAP

- **Gestion des groupes**

- **Gestion des applications**

- **Pas de single sign-Out**

A person with dark hair and a beard, wearing a dark blue t-shirt and a black neck strap, is seen from behind, sitting at a dark wooden table in a cafe. They are using a silver laptop. To their left, another person's hands are visible typing on a laptop. To their right, another silver laptop is open, displaying a document. A small potted plant in a white lace pot sits on the table. In the background, other people and laptops are visible, suggesting a busy workspace. The text 'TP' is overlaid in the center of the image.

TP