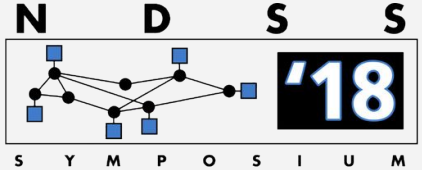


Mind Your Keys? A Security Evaluation of Java Keystores

Marco Squarcina (Università Ca' Foscari & Cryptosense)

Riccardo Focardi
Università Ca' Foscari
Cryptosense

Francesco Palmarini
Università Ca' Foscari
Yarix

Graham Steel
Cryptosense

Mauro Tempesta
Università Ca' Foscari

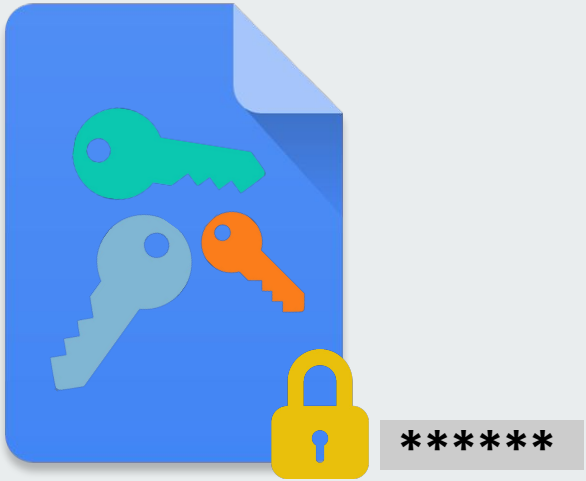
BACKGROUND MOTIVATIONS

Key Storage

HW Solutions

- HSM
- Smartcards

Key Storage



Keystore

- File containing **crypto keys** and certificates
- Content is secured by a **password**

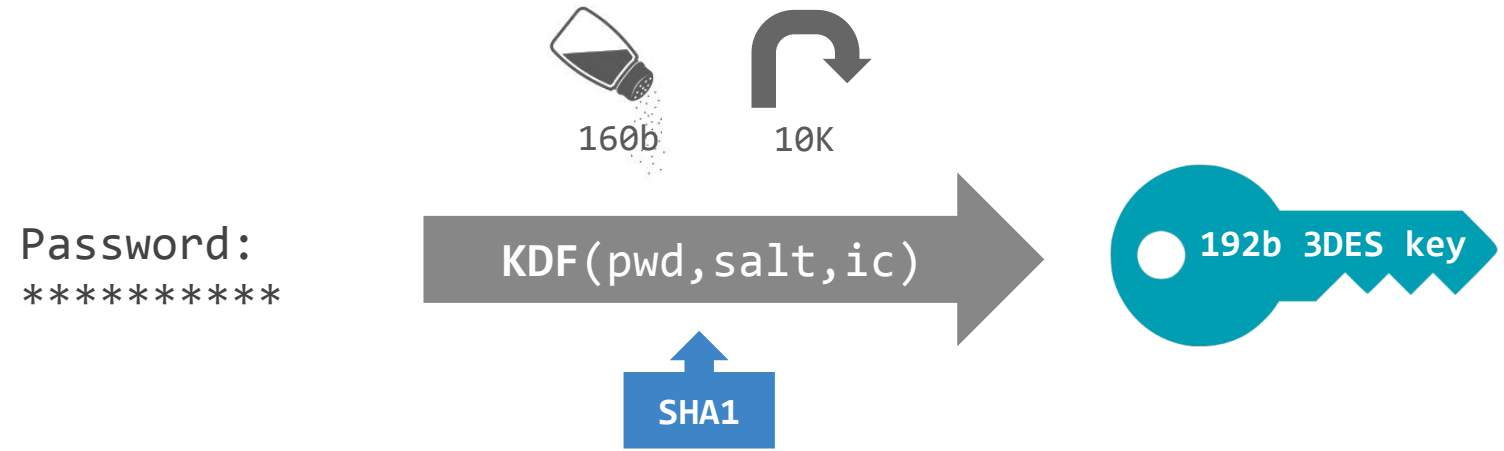
Key Storage

Keystore

- File containing **crypto keys** and certificates
- Content is secured by a **password**

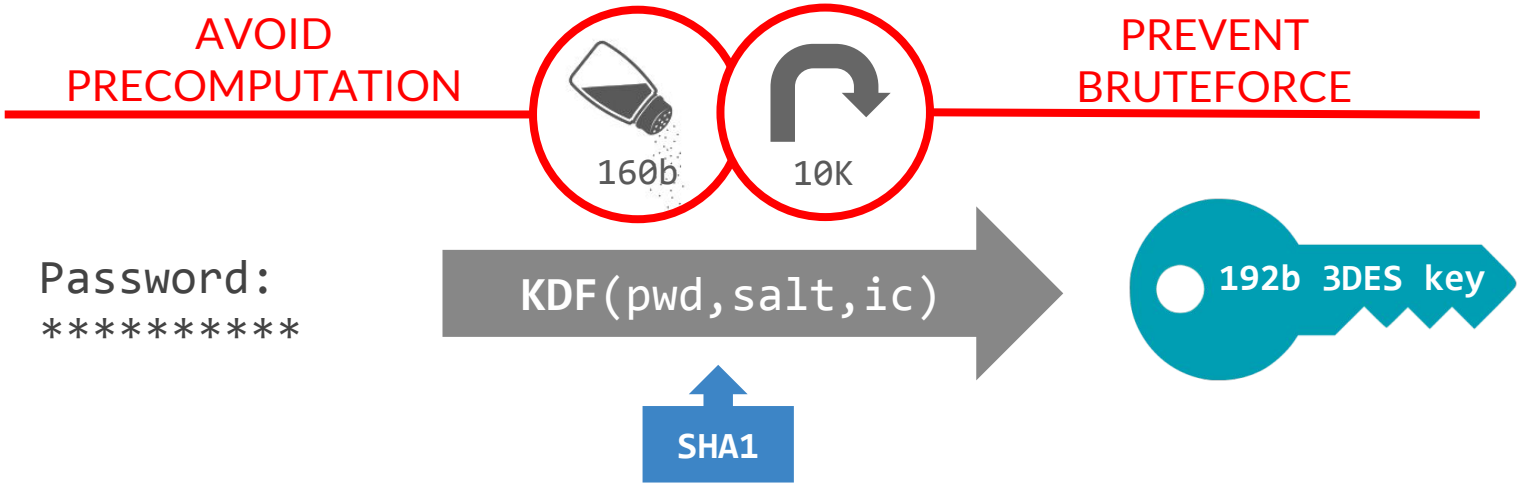
Password-based Key Derivation

- Ciphers require a key of a specific length
- Produce a key which can be used as a cryptographic key for a given cipher (e.g. 3DES)



Password-based Key Derivation

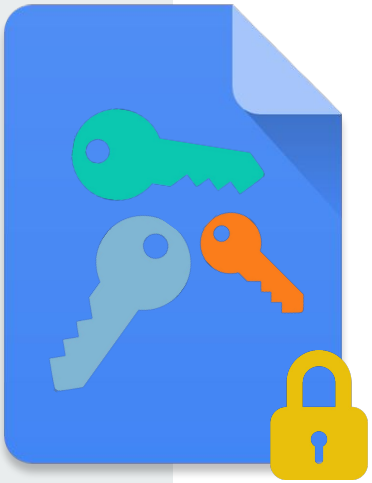
- Ciphers require a key of a specific length
- Produce a key which can be used as a cryptographic key for a given cipher (e.g. 3DES)



Keystore Types

Oracle JRE/JDK

- JKS
- JCEKS
- PKCS#12



Bouncy Castle

- BKS
- UBER
- BCPKCS#12
- BCFKS

ORACLE®

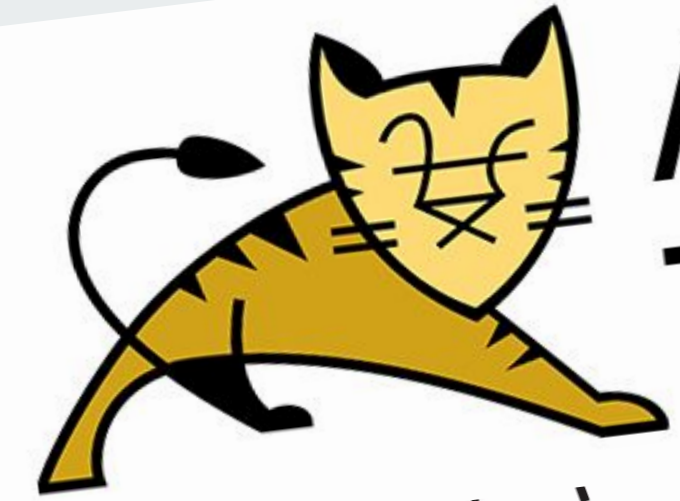
Keystore Ty

EJBCA
PKI by PrimeKey

y Castle

- BKS
- UBER
- BCPKCS#12
- BCFKS

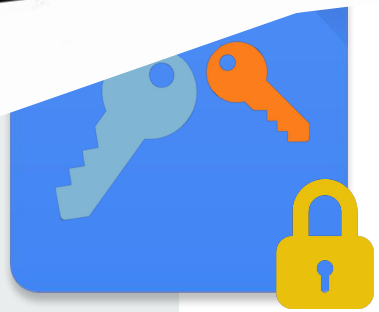
ORACLE



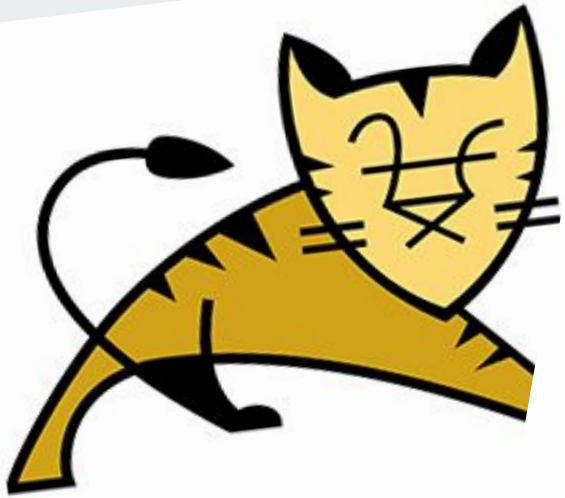
Apache Tomcat

tle

- BKS
- UBER
- BCPKCS#12
- BCFKS



ORACLE®



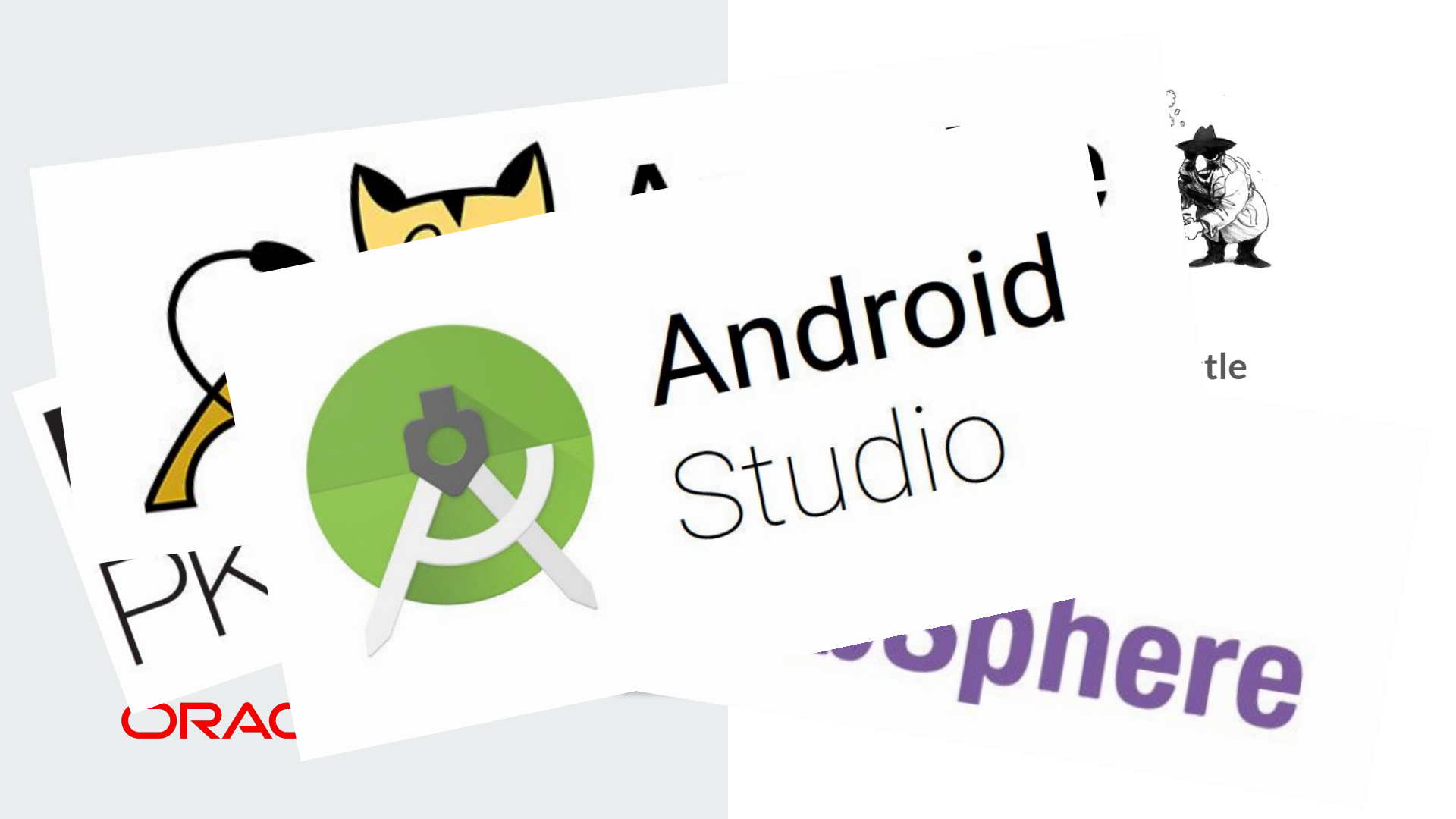
Apache

tle

IBM

WebSphere

ORACLE®

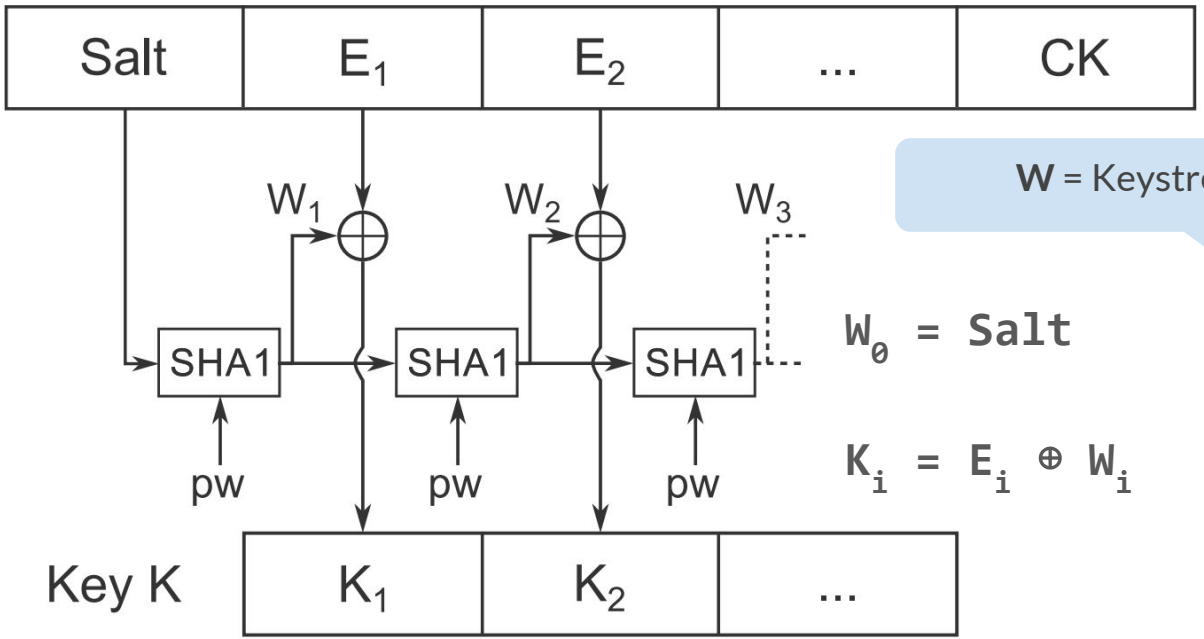

Android
Studio
tle
sphere

ATTACKS
FLAWS

Oracle JKS Password Cracking

E = Encrypted Key

Key Decryption in **JKS**



W = Keystream

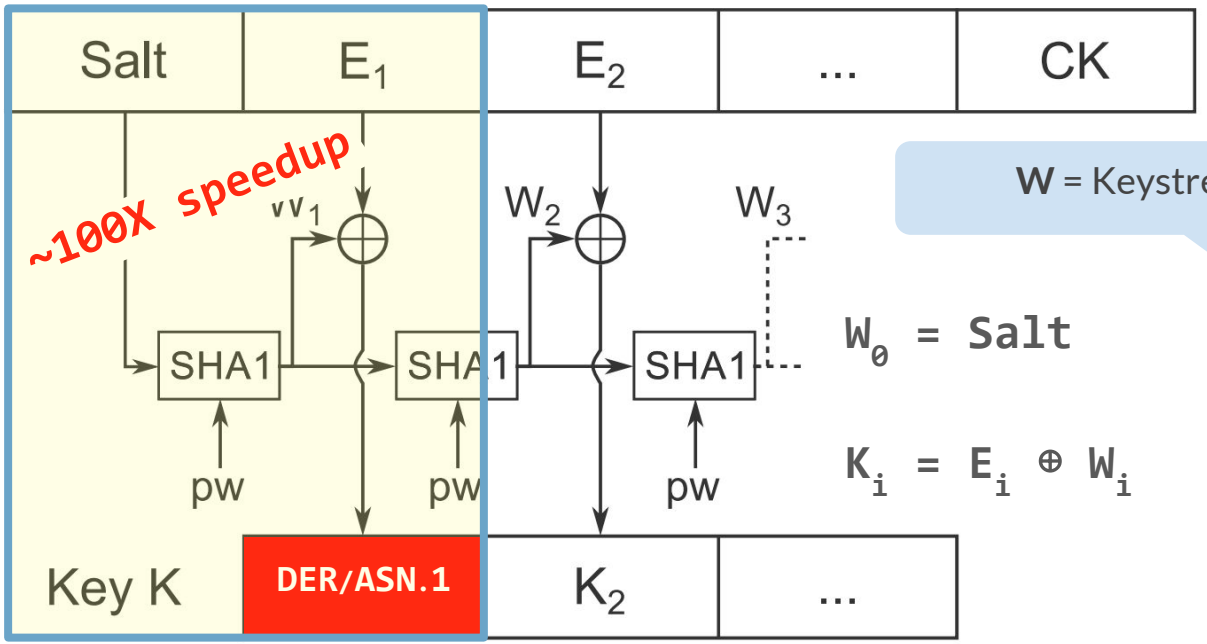
$W_0 = Salt$ $W_i = SHA1(pw||W_{i-1})$

$K_i = E_i \oplus W_i$ $CK = SHA1(pw||K)$

Oracle JKS Password Cracking

E = Encrypted Key

Key Decryption in **JKS**

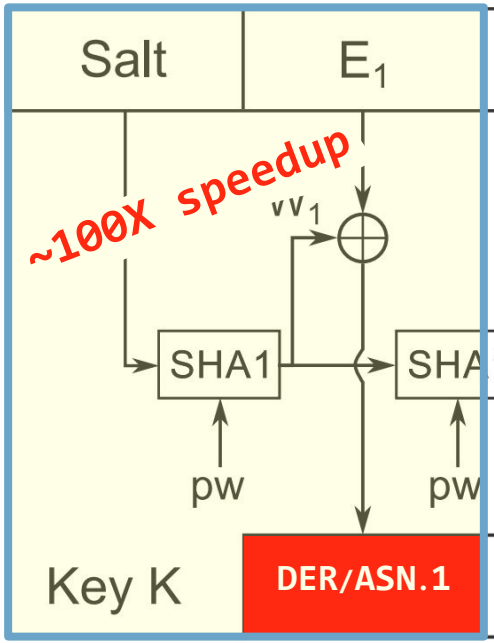


W = Keystream

$W_0 = Salt$ $W_i = SHA1(pw||W_{i-1})$

$K_i = E_i \oplus W_i$ $CK = SHA1(pw||K)$

Oracle JKS Pas cking



Key Decryption in **JKS**

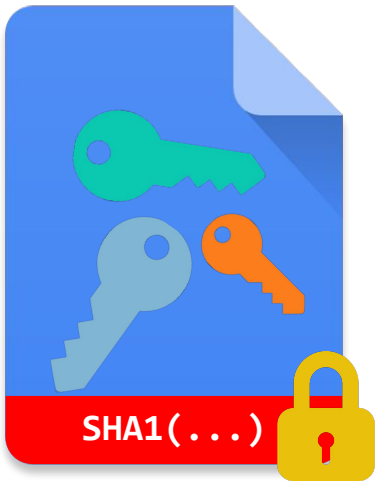
8 billions pw/s with one NVIDIA GTX 1080

HASHCAT

$||W_{i-1})$

$\oplus W_i$ CK = SHA(pw||K)

Oracle JKS/JCEKS Integrity Password Cracking



Oracle JKS/JCEKS Integrity Password Cracking

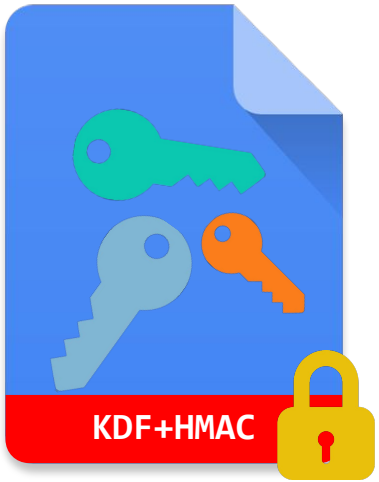
Oracle JKS/JCEKS Integrity Password Cracking

- Efficient **integrity-password bruteforce** (better w. rainbow-tables 🌈)
- Length extension attacks?
- Watch out when integrity password = confidentiality password!

Oracle JKS/JCEKS Integrity Password Cracking

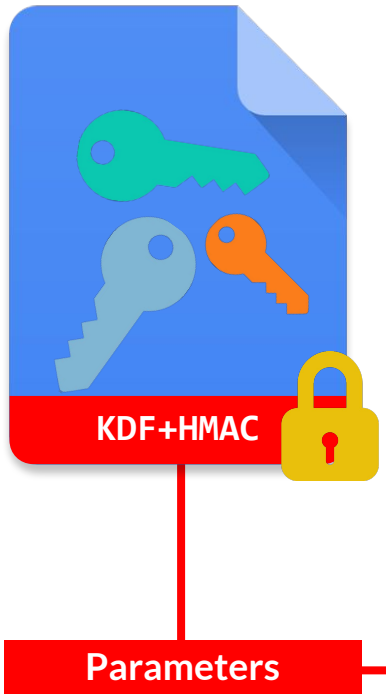
Release	Year
JDK Beta	1995
JDK 1.0	1996
JDK 1.1	1997
J2SE 1.2	1998
J2SE 1.3	2000
J2SE 1.4	2002
J2SE 5.0	2004
Java SE 6	2006
Java SE 7	2011
Java SE 8	2014
Java SE 9	2017
Java SE 10 (18.3)	2018

DoS by Integrity Parameters Abuse



- Oracle PKCS12
- Bouncy Castle BKS
- Bouncy Castle PKCS12

DoS by Integrity Parameters Abuse

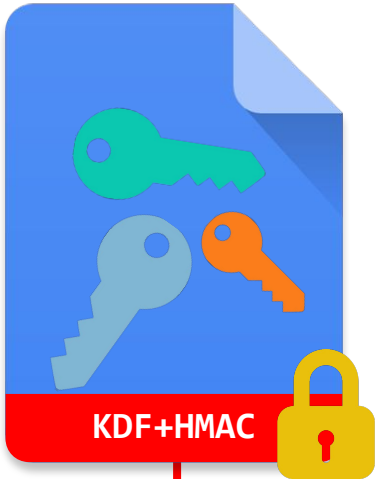


- Oracle PKCS12
- Bouncy Castle BKS
- Bouncy Castle PKCS12

ASN.1 Structure

```
...
SEQUENCE (3 elem)
  SEQUENCE (2 elem)
    SEQUENCE (2 elem)
      OBJECT IDENTIFIER 1.3.14.3.2.26 sha1 (OIW)
      NULL
    OCTET STRING (20 byte) C9C2AF5A...
  OCTET STRING (20 byte) 7B223BBC...
  INTEGER 1024
```

DoS by Integrity Parameters Abuse



- Oracle PKCS12
- Bouncy Castle B
- Bouncy Castle P

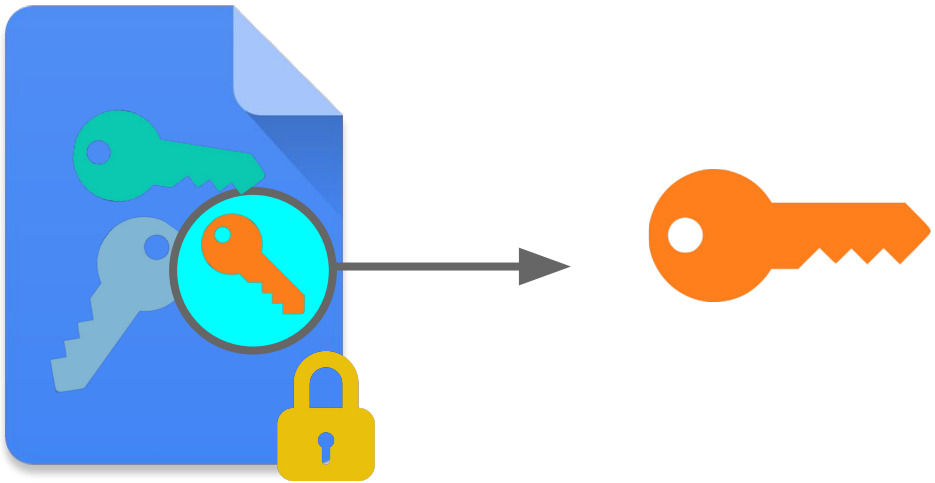
Iteration Count = $2^{31}-1$

DoS the application loading the keystore!

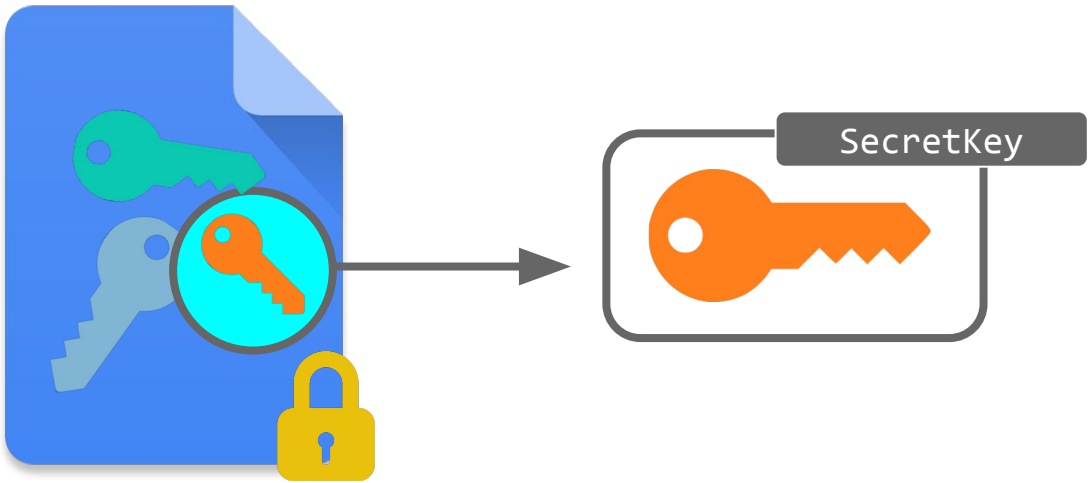
```
...
SEQUENCE (3 elem)
  SEQUENCE (2 elem)
    SEQUENCE (2 elem)
      OBJECT IDENTIFIER 1.3.14.3.2.26 sha1 (OIW)
      NULL
    OCTET STRING (20 byte) C9C2AF5A...
  OCTET STRING (20 byte) 7B223BBC...
  INTEGER 1024
```

Parameters

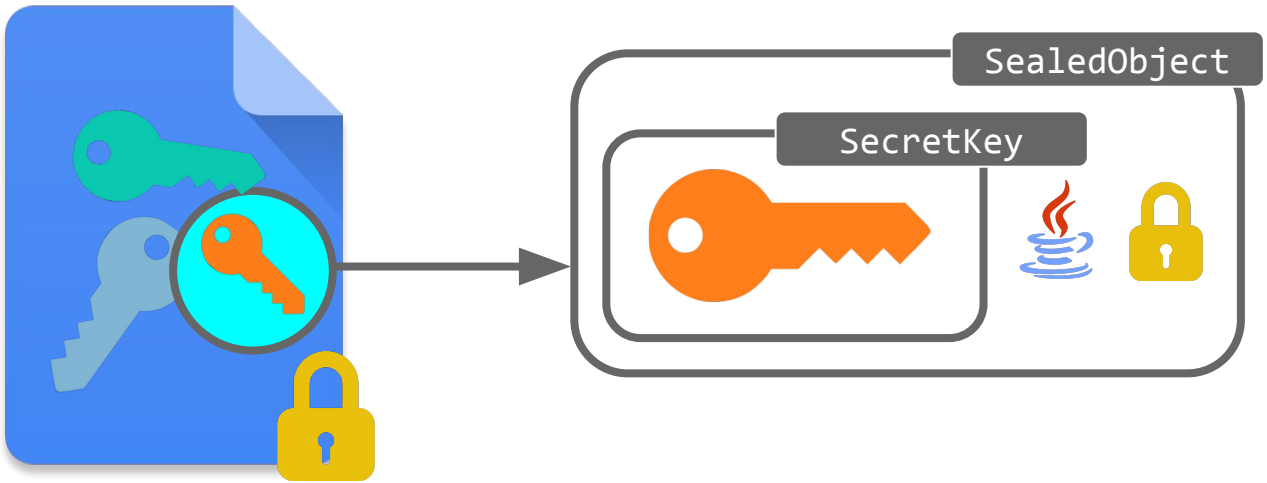
JCEKS Secret Keys Code Exec



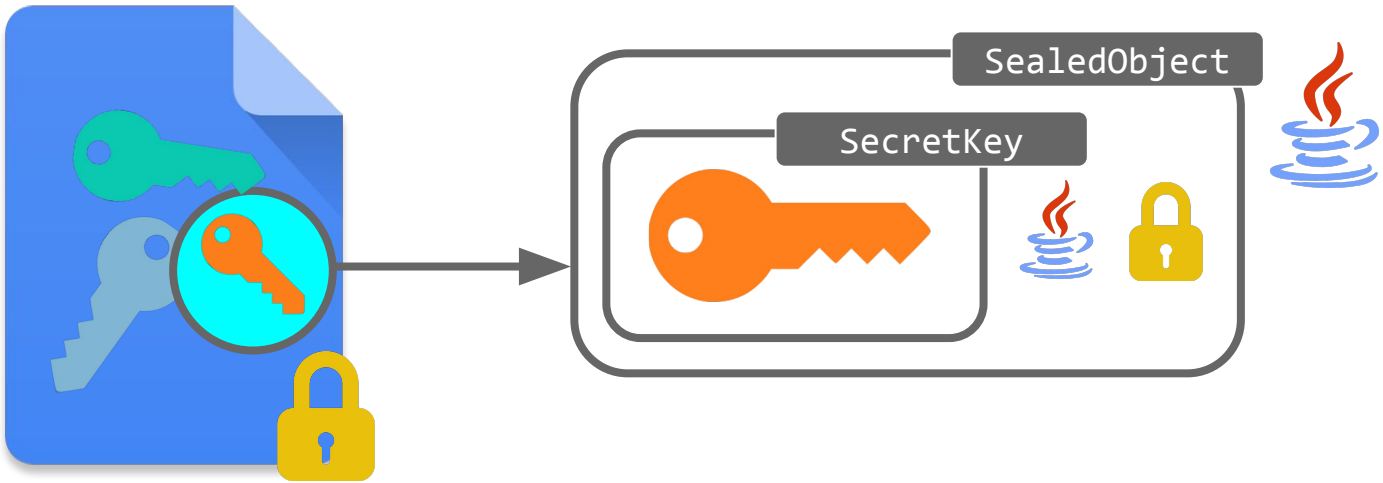
JCEKS Secret Keys Code Exec



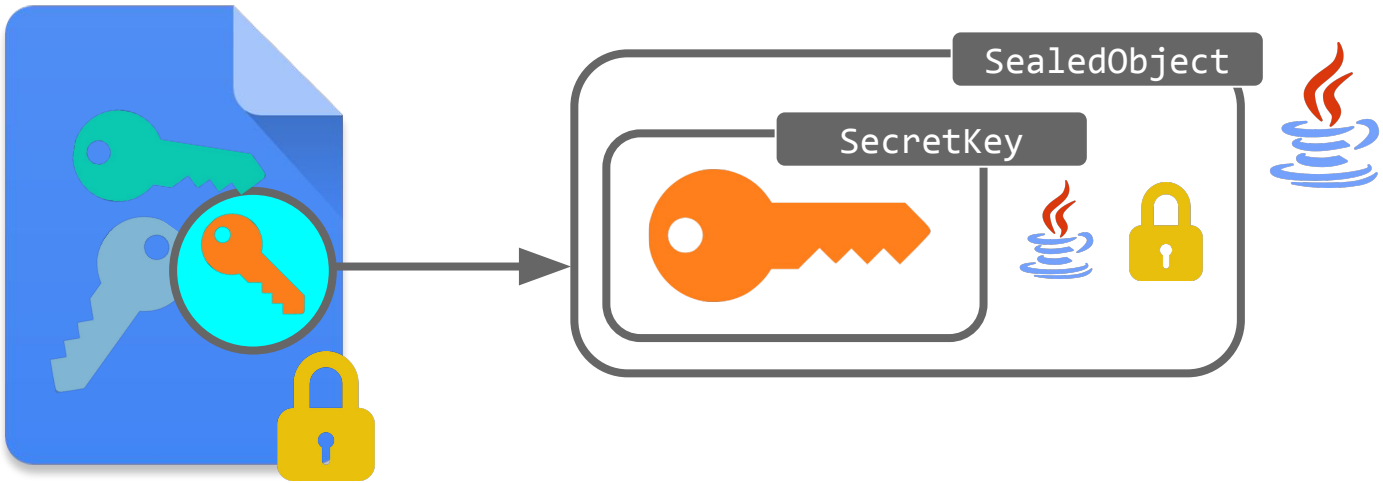
JCEKS Secret Keys Code Exec



JCEKS Secret Keys Code Exec



JCEKS Secret Keys Code Exec



KeyStore Load Mechanism

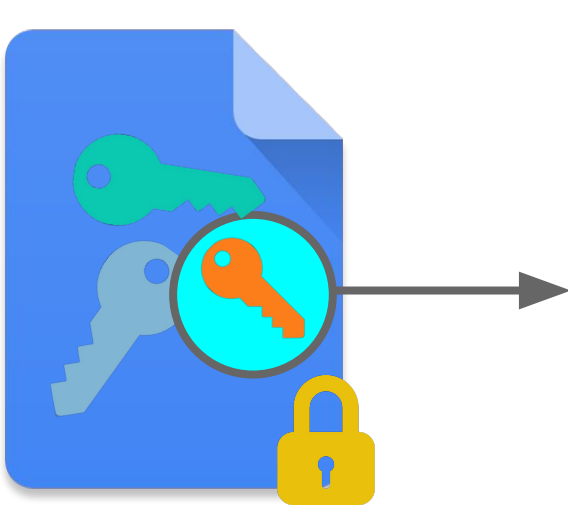
- **deserialize** each `SealedObject`
- then perform **Integrity Check**

JCEKS Secret Keys Code Exec

KeyStore Load Mechanism

- **deserialize** each `SealedObject`
- then perform **Integrity Check**

JCEKS Secret Keys Code Exec

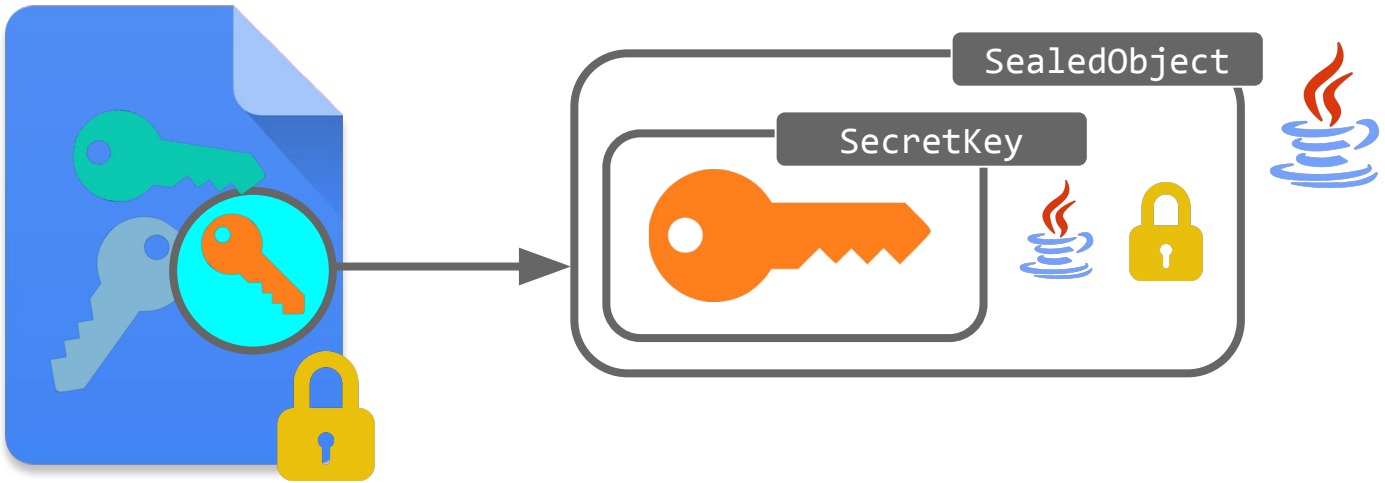


KeyStore Load Mechanism

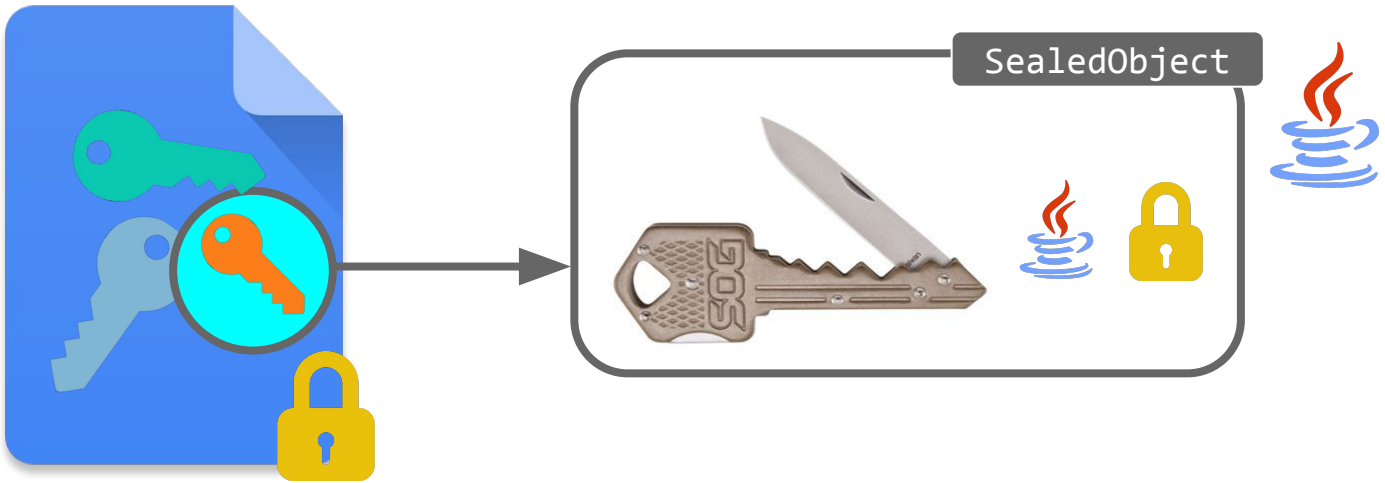
- **deserialize** each `SealedObject`
- <u>then</u> perform **Integrity Check**

- **Command execution JDK≤1.7.21 & JDK≤1.8.20**
- **DoS JDK>1.8.20**
- **Fixed Oct 2017 CPU**

JCEKS Secret Keys Code Exec after Decrypt



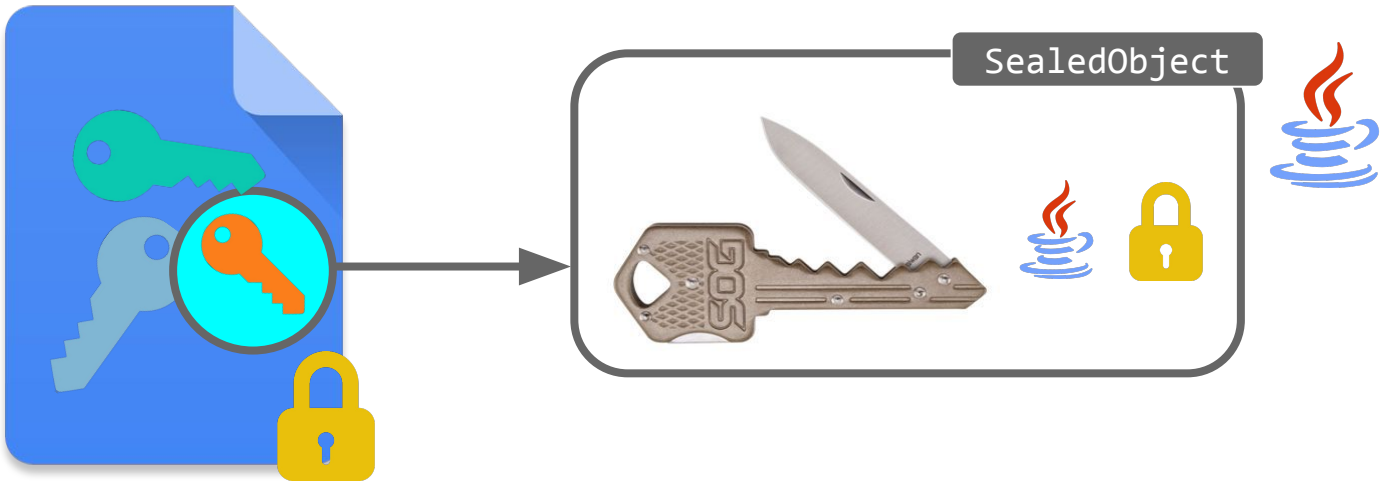
JCEKS Secret Keys Code Exec <u>after Decrypt</u>



Deserialize of `SecretKey`

- Extended **classpath**
- Use gadgets from any **3rd-party library**

JCEKS Secret Keys Code Exec after Decrypt

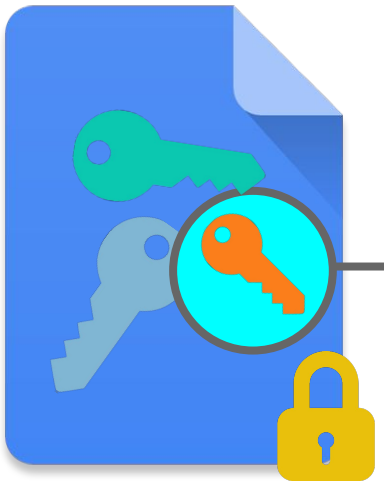


Deserialize of `SecretKey`

- Extended **classpath**
- Use gadgets from any **3rd-party library**

Command execution on latest JDK if integrity & key password are known!

JCEKS Secret Keys Code Exec after Decrypt



Deserialize of SecretKey

- Extended **classpath**
- Use gadgets from any **3rd-party library**

nd execution on
JDK if integrity &
y password are known!

DISCLOSURE CONTRIBUTIONS

Disclosure Timeline

Responses

- Oracle Keytool, **warning** on JKS/JCEKS
  - `The JCEKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format [...]`
- Oracle JCEKS KDF params for PBE
  - from 20 to **200K iterations** (max 5M)
- Oracle PKCS12
  - from 1024 to **50K** iterations for PBE (max 5M)
  - from 1024 to **100K** iterations for HMAC (max 5M)
- Partial fix to the Oracle JCEKS code execution
- Similar improvements in **Bouncy Castle**

Responses

CVE-2017-10356
CVSS 6.2

- Oracle Keytool, **warning** on JKS/JCEKS
  - `The JCEKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format [...]`
- Oracle JCEKS KDF params for PBE
  - from 20 to **200K iterations** (max 5M)
- Oracle PKCS12
  - from 1024 to **50K** iterations for PBE (max 5M)
  - from 1024 to **100K** iterations for HMAC (max 5M)

CVE-2017-10345
CVSS 3.1

- Partial fix to the Oracle JCEKS code execution
- Similar improvements in **Bouncy Castle**

Contributions

- Threat model for password-protected keystores, design rules for secure keystores
- Analysis of 7 keystores
  - Cryptographic implementation
  - Weaknesses & Attacks
- Brute force time comparison for key confidentiality and integrity passwords
- Concrete improvements to the security of Oracle JDK and Bouncy Castle keystores

THANK YOU!
(´▽`)ノ

???Q??????????U????????????E?????
??????S???T????????????????I??????
?O????????????????N???????????S???


squarcina@unive.it


@blueminimal


https://www.linkedin.com/in/squarcina/