



Catch Me If You Can: Toward Automatic Exploit Generation of Known API Vulnerabilities

Tesi di Laurea Magistrale in
Informatica

Relatori

Prof. Fabio Palomba
Prof. Andrea De Lucia

Candidato

Emanuele Iannone
Matr. 0522500588

Anno Accademico 2019-2020



AGENDA



01

IL CONTESTO

Dove si colloca il lavoro?

02

LO SCOPE

Di cosa tratta il lavoro?

03

SIEGE

Cosa presenta il lavoro?

04

CONCLUSIONI

Cosa abbiamo appreso?

L'INCIDENTE EQUIFAX

The New York Times

Equifax Says Cyberattack May Have Affected 143 Million in the U.S.

By Tara Siegel Bernard, Tiffany Hsu, Nicole Perlroth and Ron Lieber

Sept. 7, 2017



[Equifax](#), one of the three major consumer credit reporting agencies, said on Thursday that [hackers](#) had gained access to company data that potentially compromised sensitive information for 143 million American consumers, including Social Security numbers and driver's license numbers.

The attack on the company represents one of the largest risks to personally sensitive information in recent years, and is the third major cybersecurity threat for the agency since 2015.

Equifax, based in Atlanta, is a particularly tempting target for hackers. If identity thieves wanted to hit

L'INCIDENTE EQUIFAX

140 MLN

Utenze colpite

The New York Times

Equifax Says Cyberattack May Have Affected 143 Million in the U.S.

By Tara Siegel Bernard, Tiffany Hsu, Nicole Perlroth and Ron Lieber

Sept. 7, 2017



[Equifax](#), one of the three major consumer credit reporting agencies, said on Thursday that [hackers](#) had gained access to company data that potentially compromised sensitive information for 143 million American consumers, including Social Security numbers and driver's license numbers.

The attack on the company represents one of the largest risks to personally sensitive information in recent years, and is the third major cybersecurity threat for the agency since 2015.

Equifax, based in Atlanta, is a particularly tempting target for hackers. If identity thieves wanted to hit

200.000

Carte di credito

L'INCIDENTE EQUIFAX



140 MLN

Utenze colpite

200.000

Carte di credito

\$1.400.000.000

Costi di rimedio



L'INCIDENTE EQUIFAX



14



IN DETTAGLIO

CVE-2017-5638 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

Current Description

The Jakarta Multipart parser in Apache Struts 2 2.3.x before 2.3.32 and 2.5.x before 2.5.10.1 has incorrect exception handling and error-message generation during file-upload attempts, which allows remote attackers to execute arbitrary commands via a crafted Content-Type, Content-Disposition, or Content-Length HTTP header, as exploited in the wild in March 2017 with a Content-Type header containing a `#cmd=` string.

10

L'INCIDENTE EQUIFAX



140 M

Utenze c

IN DETTAGLIO

OWASP Top 10 - 2017

A1:2017-Injection
A2:2017-Broken Authentication
A3:2017-Sensitive Data Exposure
A4:2017-XML External Entities (XXE)
A5:2017-Broken Access Control
A6:2017-Security Misconfiguration
A7:2017-Cross-Site Scripting (XSS)
A8:2017-Insecure Deserialization
A9:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring

.000

li credito



AGENDA



01

IL CONTESTO

Dove si colloca il
lavoro?

02

LO SCOPE

Di cosa tratta il
lavoro?

03

SIEGE

Cosa presenta il
lavoro?

04

CONCLUSIONI

Cosa abbiamo
appreso?

VULNERABILITÀ DI LIBRERIA

*“Un difetto di sicurezza di una **libreria di terze parti** il cui sfruttamento comporta un danno alla security policy definita **sul sistema client**”.*

VULNERABILITÀ DI LIBRERIA

*“Un difetto di sicurezza di una **libreria di terze parti** il cui sfruttamento comporta un danno alla security policy definita **sul sistema client**”.*



DETECTION

Come identificare le vulnerabilità di libreria?



ASSESSMENT

Come valutare le vulnerabilità di libreria?



MITIGATION

Come mitigare le vulnerabilità di libreria?

VULNERABILITÀ DI LIBRERIA

*“Un difetto di sicurezza di una **libreria di terze parti** il cui sfruttamento comporta un danno alla security policy definita **sul sistema client**”.*



DETECTION

Come identificare le vulnerabilità di libreria?

Analisi Inclusione

Analisi Statica

Analisi Dinamica

Analisi Ibrida



ASSESSMENT

Come valutare le vulnerabilità di libreria?



MITIGATION

Come mitigare le vulnerabilità di libreria?

VULNERABILITÀ DI LIBRERIA

*“Un difetto di sicurezza di una **libreria di terze parti** il cui sfruttamento comporta un danno alla security policy definita **sul sistema client**”.*



DETECTION

Come identificare le vulnerabilità di libreria?

Analisi Inclusione

Analisi Statica

Analisi Dinamica

Analisi Ibrida



Eclipse STEADY

SAP



ASSESSMENT

Come valutare le vulnerabilità di libreria?



MITIGATION

Come mitigare le vulnerabilità di libreria?

VULNERABILITÀ DI LIBRERIA

*“Un difetto di sicurezza di una **libreria di terze parti** il cui sfruttamento comporta un danno alla security policy definita **sul sistema client**”.*



DETECTION

Come identificare le vulnerabilità di libreria?



ASSESSMENT

Come valutare le vulnerabilità di libreria?

CVE Scoring System

Raggiungibilità 



MITIGATION

Come mitigare le vulnerabilità di libreria?

VULNERABILITÀ DI LIBRERIA

*“Un difetto di sicurezza di una **libreria di terze parti** il cui sfruttamento comporta un danno alla security policy definita **sul sistema client**”.*



DETECTION

Come identificare le vulnerabilità di libreria?



ASSESSMENT

Come valutare le vulnerabilità di libreria?



MITIGATION

Come mitigare le vulnerabilità di libreria?

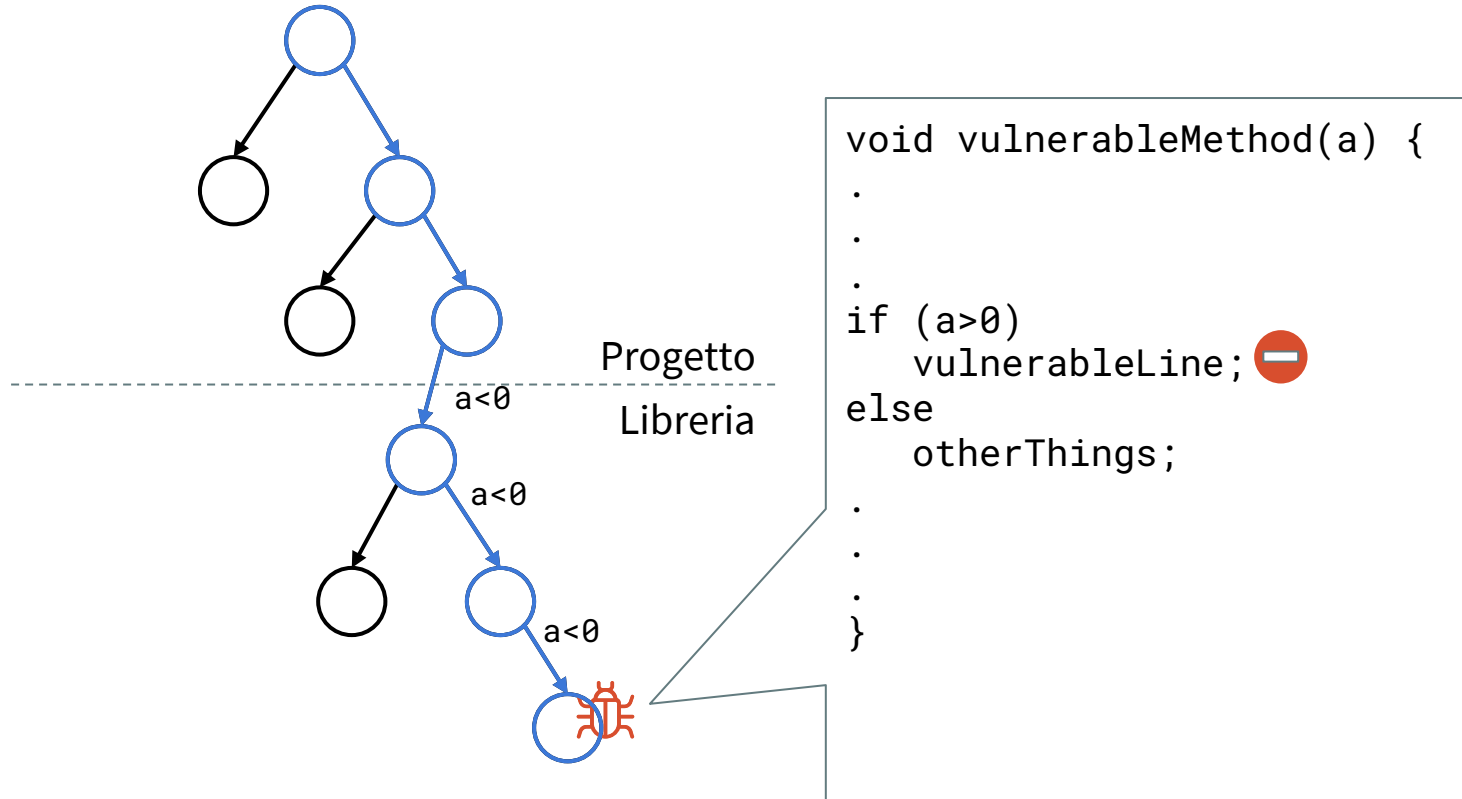
Stabilità API 


Sforzo sviluppo 

Popolarità




STEADY: UN ESEMPIO





*“Data una libreria inclusa nel mio progetto software i cui costrutti vulnerabili siano...
raggiungibili staticamente, sono davvero sicuro che la vulnerabilità sia effettivamente **exploitabile**?”*



*“Data una libreria inclusa nel mio progetto software i cui costrutti vulnerabili siano...
raggiungibili staticamente, sono davvero sicuro che la vulnerabilità sia effettivamente **exploitabile**?”*

*“Data una libreria inclusa nel mio progetto software i cui costrutti vulnerabili siano...
irraggiungibili dinamicamente, sono davvero sicuro che la vulnerabilità sia effettivamente **non-exploitable**?”*



AGENDA

01

IL CONTESTO

Dove si colloca il lavoro?

02

LO SCOPE

Di cosa tratta il lavoro?

03

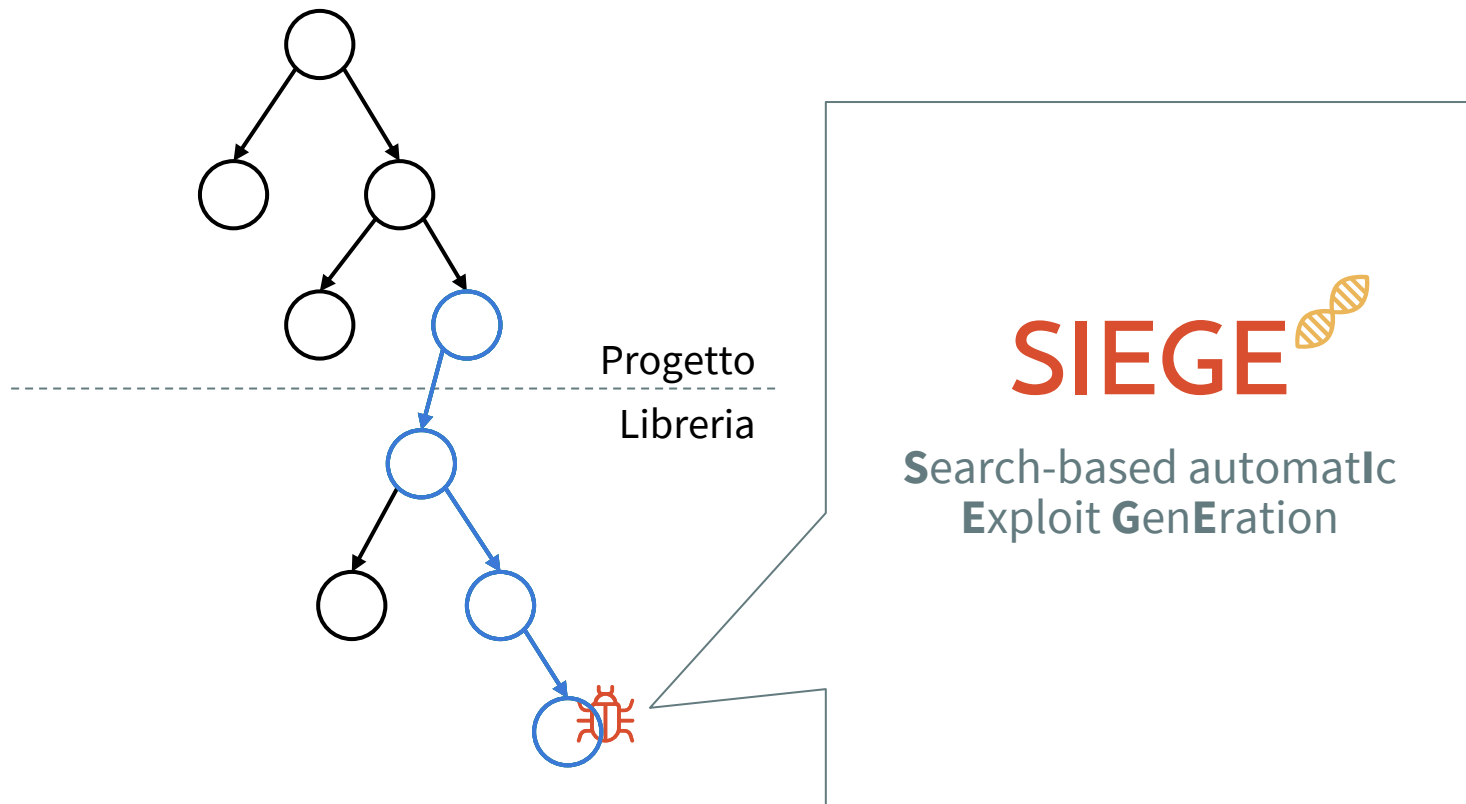
SIEGE

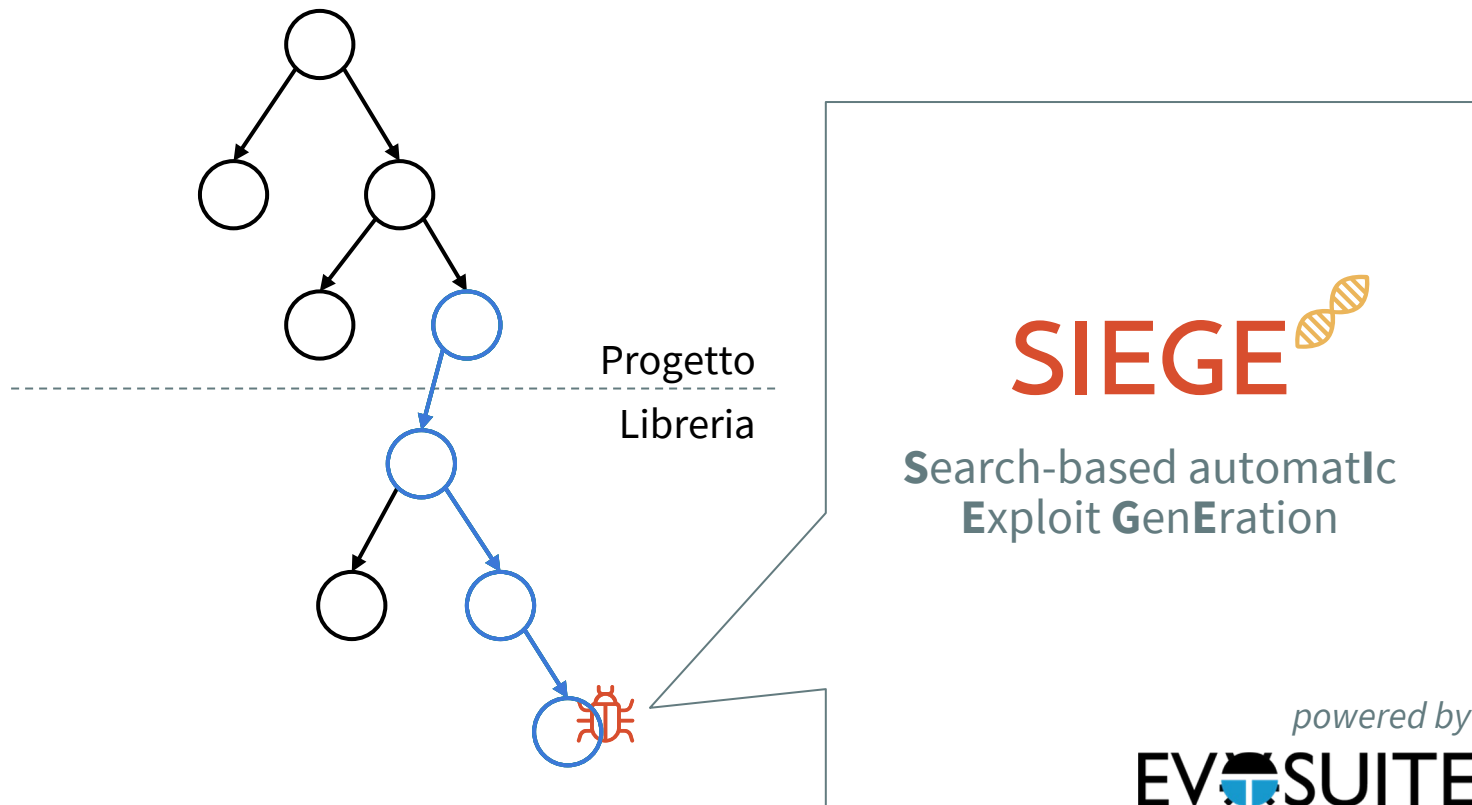
Cosa presenta il lavoro?

04

CONCLUSIONI

Cosa abbiamo appreso?







SIEGE

Come Funziona

CVE-2018-1324 Apache Commons Compress 1.15

```
302 public void parseCentralDirectoryFormat(final byte[] data, final int
    ↪offset, final int length) {
303     this.format = ZipShort.getValue(data, offset);
304     this.algId = EncryptionAlgorithm.getAlgorithmByCode(ZipShort.getValue
    ↪(data, offset + 2));
305     this.bitlen = ZipShort.getValue(data, offset + 4);
306     this.flags = ZipShort.getValue(data, offset + 6);
307     this.rcount = ZipLong.getValue(data, offset + 8);
308
309     if (rcount > 0) {
310         this.hashAlg = HashAlgorithm.getAlgorithmByCode(ZipShort.getValue(
    ↪data, offset + 12));
311         this.hashSize = ZipShort.getValue(data, offset + 14);
312         // srlist... hashed public keys
313         for (int i = 0; i < this.rcount; i++){
314             for (int j = 0; j < this.hashSize; j++) {
315                 // ZipUtil.signedByteToUnsignedInt(data[offset + 16 + (i * this.
    ↪hashSize) + j]));
316             }
317         }
318     }
319 }
```



SIEGE

Come Funziona

Client di CVE-2018-1324

```
1 public class CompressCaller1 {  
2     private X0017_StrongEncryptionHeader seh;  
3  
4     public CompressCaller1() {  
5         this.seh = new X0017_StrongEncryptionHeader();  
6     }  
7     public void call(byte[] data, int offset, int length) {  
8         this.seh.parseCentralDirectoryFormat(data, offset, length);  
9     }  
10 }
```

CVE-2018-1324 Apache Commons Compress 1.15

```
302 public void parseCentralDirectoryFormat(final byte[] data, final int  
303     ↪offset, final int length) {  
304     this.format = ZipShort.getValue(data, offset);  
305     this.algId = EncryptionAlgorithm.getAlgorithmByCode(ZipShort.getValue  
306     ↪(data, offset + 2));  
307     this.bitlen = ZipShort.getValue(data, offset + 4);  
308     this.flags = ZipShort.getValue(data, offset + 6);  
309     this.rcount = ZipLong.getValue(data, offset + 8);  
310  
311     if (rcount > 0) {  
312         this.hashAlg = HashAlgorithm.getAlgorithmByCode(ZipShort.getValue(  
313         ↪data, offset + 12));  
314         this.hashSize = ZipShort.getValue(data, offset + 14);  
315         // srlist... hashed public keys  
316         for (int i = 0; i < this.rcount; i++){  
317             for (int j = 0; j < this.hashSize; j++) {  
318                 // ZipUtil.signedByteToUnsignedInt(data[offset + 16 + (i * this.  
319                 ↪hashSize) + j]));  
320             }  
321         }  
322     }  
323 }
```



SIEGE

Come Funziona

Client di CVE-2018-1324

```
1 public class CompressCaller1 {  
2     private X0017_StrongEncryptionHeader seh;  
3  
4     public CompressCaller1() {  
5         this.seh = new X0017_StrongEncryptionHeader();  
6     }  
7     public void call(byte[] data, int offset, int length) {  
8         this.seh.parseCentralDirectoryFormat(data, offset, length);  
9     }  
10 }
```

Coverage Goal di SIEGE

1. Contesto di chiamata

- call() -> parseCentralDirectoryFormat()

2. Nodi di controllo

- true di if (rcount > 0)

3. Linea Vulnerabile

- 313

CVE-2018-1324 Apache Commons Compress 1.15

```
302 public void parseCentralDirectoryFormat(final byte[] data, final int  
303     ↪offset, final int length) {  
304     this.format = ZipShort.getValue(data, offset);  
305     this.algId = EncryptionAlgorithm.getAlgorithmByCode(ZipShort.getValue  
306     ↪(data, offset + 2));  
307     this.bitlen = ZipShort.getValue(data, offset + 4);  
308     this.flags = ZipShort.getValue(data, offset + 6);  
309     this.rcount = ZipLong.getValue(data, offset + 8);  
310  
311     if (rcount > 0) {  
312         this.hashAlg = HashAlgorithm.getAlgorithmByCode(ZipShort.getValue(  
313         ↪data, offset + 12));  
314         this.hashSize = ZipShort.getValue(data, offset + 14);  
315         // srlist... hashed public keys  
316         for (int i = 0; i < this.rcount; i++){  
317             for (int j = 0; j < this.hashSize; j++) {  
318                 // ZipUtil.signedByteToUnsignedInt(data[offset + 16 + (i * this.  
319                 ↪hashSize) + j]));  
320             }  
321         }  
322     }  
323 }
```




SIEGE

Come Funziona

Client di CVE-2018-1324

```
1 public class CompressCaller1 {
2     private X0017_StrongEncryptionHeader seh;
3
4     public CompressCaller1() {
5         this.seh = new X0017_StrongEncryptionHeader();
6     }
7     public void call(byte[] data, int offset, int length) {
8         this.seh.parseCentralDirectoryFormat(data, offset, length);
9     }
10 }
```

I Generazione - Individuo Migliore

```
1 CompressCaller1 compressCaller1_0 = new CompressCaller1();
2 byte[] byteArray0 = new byte[2];
3 byte byte0 = (byte)0;
4 byteArray0[0] = byte0;
5 byte byte0 = (byte) (-8);
6 byteArray0[1] = byte0;
7 int int0 = (-2552);
8 compressCaller1_0.call(byteArray0, byte0, int0);
```

CVE-2018-1324 Apache Commons Compress 1.15

```
302 public void parseCentralDirectoryFormat(final byte[] data, final int
    ↪offset, final int length) {
303     this.format = ZipShort.getValue(data, offset);
304     this.algId = EncryptionAlgorithm.getAlgorithmByCode(ZipShort.getValue
    ↪(data, offset + 2));
305     this.bitlen = ZipShort.getValue(data, offset + 4);
306     this.flags = ZipShort.getValue(data, offset + 6);
307     this.rcount = ZipLong.getValue(data, offset + 8);
308
309     if (rcount > 0) {
310         this.hashAlg = HashAlgorithm.getAlgorithmByCode(ZipShort.getValue(
    ↪data, offset + 12));
311         this.hashSize = ZipShort.getValue(data, offset + 14);
312         // srlist... hashed public keys
313         for (int i = 0; i < this.rcount; i++){
314             for (int j = 0; j < this.hashSize; j++) {
315                 // ZipUtil.signedByteToUnsignedInt(data[offset + 16 + (i * this.
    ↪hashSize) + j]));
316             }
317         }
318     }
319 }
```



SIEGE

Come Funziona

Client di CVE-2018-1324

```
1 public class CompressCaller1 {
2     private X0017_StrongEncryptionHeader seh;
3
4     public CompressCaller1() {
5         this.seh = new X0017_StrongEncryptionHeader();
6     }
7     public void call(byte[] data, int offset, int length) {
8         this.seh.parseCentralDirectoryFormat(data, offset, length);
9     }
10 }
```

LXXII Generazione - Individuo Imperfetto

```
1 CompressCaller1 compressCaller1_0 = new CompressCaller1();
2 byte[] byteArray0 = new byte[12];
3 int int0 = 1220;
4 compressCaller1_0.call(byteArray0, byteArray0[1], int0);
```

CVE-2018-1324 Apache Commons Compress 1.15

```
302 public void parseCentralDirectoryFormat(final byte[] data, final int
    ↪offset, final int length) {
303     this.format = ZipShort.getValue(data, offset);
304     this.algId = EncryptionAlgorithm.getAlgorithmByCode(ZipShort.getValue
    ↪(data, offset + 2));
305     this.bitlen = ZipShort.getValue(data, offset + 4);
306     this.flags = ZipShort.getValue(data, offset + 6);
307     this.rcount = ZipLong.getValue(data, offset + 8);
308
309     if (rcount > 0) {
310         this.hashAlg = HashAlgorithm.getAlgorithmByCode(ZipShort.getValue(
    ↪data, offset + 12));
311         this.hashSize = ZipShort.getValue(data, offset + 14);
312         // srlist... hashed public keys
313         for (int i = 0; i < this.rcount; i++){
314             for (int j = 0; j < this.hashSize; j++) {
315                 // ZipUtil.signedByteToUnsignedInt(data[offset + 16 + (i * this.
    ↪hashSize) + j]));
316             }
317         }
318     }
319 }
```



SIEGE

Come Funziona

Client di CVE-2018-1324

```
1 public class CompressCaller1 {
2     private X0017_StrongEncryptionHeader seh;
3
4     public CompressCaller1() {
5         this.seh = new X0017_StrongEncryptionHeader();
6     }
7     public void call(byte[] data, int offset, int length) {
8         this.seh.parseCentralDirectoryFormat(data, offset, length);
9     }
10 }
```

LXXII Generazione - Individuo Perfetto

```
1 CompressCaller1 compressCaller1_0 = new CompressCaller1();
2 byte[] byteArray0 = new byte[19];
3 byteArray0[0] = (byte)3;
4 byteArray0[14] = (byte)2;
5 byteArray0[2] = (byte)2;
6 byteArray0[3] = (byte) (-1);
7 byteArray0[4] = (byte) (-1);
8 compressCaller1_0.call(byteArray0, (byte)3, 2);
```

EXPLOIT!

CVE-2018-1324 Apache Commons Compress 1.15

```
302 public void parseCentralDirectoryFormat(final byte[] data, final int
    ↪offset, final int length) {
303     this.format = ZipShort.getValue(data, offset);
304     this.algId = EncryptionAlgorithm.getAlgorithmByCode(ZipShort.getValue
    ↪(data, offset + 2));
305     this.bitlen = ZipShort.getValue(data, offset + 4);
306     this.flags = ZipShort.getValue(data, offset + 6);
307     this.rcount = ZipLong.getValue(data, offset + 8);
308
309     if (rcount > 0) {
310         this.hashAlg = HashAlgorithm.getAlgorithmByCode(ZipShort.getValue(
    ↪data, offset + 12));
311         this.hashSize = ZipShort.getValue(data, offset + 14);
312         // srlist... hashed public keys
313         for (int i = 0; i < this.rcount; i++){
314             for (int j = 0; j < this.hashSize; j++) {
315                 // ZipUtil.signedByteToUnsignedInt(data[offset + 16 + (i * this.
    ↪hashSize) + j]));
316             }
317         }
318     }
319 }
```



RQ1. SIEGE riesce a generare exploit per un insieme di vulnerabilità di libreria?

CVE	URL	Commit
CVE-2017-4971	https://github.com/spring-projects/spring-webflow	57f2ccb66946943fbf3b3f2165eac1c8eb6b1523
CVE-2018-1000134	https://github.com/pingidentity/ldapsdk	8471904a02438c03965d21367890276bc25fa5a6
CVE-2016-8749	https://github.com/apache/camel	57d01e2fc8923263df896e9810329ee5b7f9b69
CVE-2017-1000393	https://github.com/jenkinsci/jenkins	d7ea3f40efedd50541a57b943d5f7bbed046d091
CVE-2018-8034	https://github.com/apache/tomcat	2835bb4e030c1c741ed0847bb3b9c3822e4fbc8a

S. E. Ponta, H. Plate, A. Sabetta, M. Bezzi, C. Dangremont, A Manually-Curated Dataset of Fixes to Vulnerabilities of Open-Source Software, MSR, 2019



RQ1. SIEGE riesce a generare exploit per un insieme di vulnerabilità di libreria?

CVE	URL	Commit
CVE-2017-4971	https://github.com/spring-projects/spring-webflow	57f2ccb66946943fbf3b3f2165eac1c8eb6b1523
CVE-2018-1000134	https://github.com/pingidentity/ldapsdk	8471904a02438c03965d21367890276bc25fa5a6
CVE-2016-8749	https://github.com/apache/camel	57d01e2fc8923263df896e9810329ee5b7f9b69
CVE-2017-1000393	https://github.com/jenkinsci/jenkins	d7ea3f40efedd50541a57b943d5f7bbed046d091
CVE-2018-8034	https://github.com/apache/tomcat	2835bb4e030c1c741ed0847bb3b9c3822e4fbc8a

S. E. Ponta, H. Plate, A. Sabetta, M. Bezzi, C. Dangremont, A Manually-Curated Dataset of Fixes to Vulnerabilities of Open-Source Software, MSR, 2019

IMPREVISTI

1. SIEGE lavora solo con **linee vulnerabili**
2. **Problema instrumentazione** di EvoSuite



RQ1. SIEGE riesce a generare exploit per un insieme di vulnerabilità di libreria?

Vulnerability	Library	Version	Description
CVE-2018-1324	APACHE COMMONS COMPRESS	1.15	DoS when a certain input (ZIP file) is provided
CVE-2011-1582	APACHE TOMCAT	7.0.12	Using a wrong classloader to bypass access restrictions
CVE-2014-9970	JASYPT	1.9.1	Exposition to Timing Attacks by using a linear time hashes comparison function
CVE-2018-1000067	JENKINS	2.89.3	Improper access restriction for a restricted functionality
CVE-2017-1000390	TIKAL MULTIJOB PLUGIN	1.26	Improper access restriction for a restricted functionality
CVE-2016-3092	APACHE COMMONS FILE UPLOAD	1.3.1	DoS due to certain sufficiently large buffers
CVE-2011-1498	APACHE HTTPCLIENT	4.1	Exposing sensitive information in logs
ZEPPELIN-2769	APACHE ZEPPELIN	0.6.0	Exposure to SQL Injection
CVE-2018-17194	APACHE NIFI	1.7.1	Time waste with a certain DELETE HTTP requests with non-empty body
CVE-2018-8718	MAILER PLUGIN	1.20	Exposition to CSRF for a certain functionality
PRIMEFACES-1194	PRIMEFACES	6.1	Exposition to XSS due to a lack of escaping of a user input

RQ1. SIEGE riesce a generare exploit per un insieme di vulnerabilità di libreria?

Vulnerability	Library	Version	Description
CVE-2018-1324	APACHE COMMONS COMPRESS	1.15	DoS when a certain input (ZIP file) is provided
CVE-2011-1582	APACHE TOMCAT	7.0.12	Using a wrong classloader to bypass access restrictions
CVE-2014-9970	JASYPT	1.9.1	Exposition to Timing Attacks by using a linear time hashes comparison function
CVE-2018-1000067	JENKINS	2.89.3	Improper access restriction for a restricted functionality
CVE-2017-1000390	TIKAL MULTIJOB PLUGIN	1.26	Improper access restriction for a restricted functionality
CVE-2016-3092	APACHE COMMONS FILE UPLOAD	1.3.1	DoS due to certain sufficiently large buffers
CVE-2011-1498	APACHE HTTPCLIENT	4.1	Exposing sensitive information in logs
ZEPPELIN-2769	APACHE ZEPPELIN	0.6.0	Exposure to SQL Injection
CVE-2018-17194	APACHE NIFI	1.7.1	Time waste with a certain DELETE HTTP requests with non-empty body
CVE-2018-8718	MAILER PLUGIN	1.20	Exposition to CSRF for a certain functionality
PRIMEFACES-1194	PRIMEFACES	6.1	Exposition to XSS due to a lack of escaping of a user input

Per ciascuna vulnerabilità:

1. Definito 2 classi client

2. Lanciato SIEGE su

entrambi i client, con 4 diversi budget di ricerca (5, 15, 30, 60) per 3 volte.

RQ1. SIEGE riesce a generare exploit per un insieme di vulnerabilità di libreria?



Set #1	Budget (s)			
Vulnerable Library	5	15	30	60
COMPRESS	0.364 (37)	0.364 (164)	0.000 (336)	0.000 (285)
TOMCAT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
JASYPT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
JENKINS	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
MULTIJOB	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
FILE UPLOAD	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
HTTPCLIENT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
ZEPELIN	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
NIFI	3.000 (43)	3.000 (225)	3.000 (499)	3.000 (1079)
MAILER	3.000 (82)	3.000 (241)	3.000 (477)	3.000 (998)
PRIMEFACES	2.000 (34)	2.000 (111)	2.000 (226)	2.000 (454)

Set #2	Budget (s)			
Vulnerable Library	5	15	30	60
COMPRESS	2.000 (19)	0.364 (29)	0.000 (30)	0.000 (30)
TOMCAT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
JASYPT	0.000 (2)	0.000 (2)	0.000 (2)	0.000 (2)
JENKINS	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
MULTIJOB	2.500 (33)	2.500 (86)	2.500 (168)	2.500 (307)
HTTPCLIENT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
ZEPELIN	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
NIFI	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
MAILER	2.500 (18)	2.500 (61)	2.500 (129)	2.500 (259)
PRIMEFACES	2.000 (20)	2.000 (57)	2.000 (95)	2.000 (196)

RQ2. La generazione è influenzata dal modo in cui le classi client usano i costrutti vulnerabili?



Set #1	Budget (s)			
Vulnerable Library	5	15	30	60
COMPRESS	0.364 (37)	0.364 (164)	0.000 (336)	0.000 (285)
TOMCAT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
JASYPT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
JENKINS	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
MULTIJOB	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
FILE UPLOAD	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
HTTPCLIENT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
ZEPELIN	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
NIFI	3.000 (43)	3.000 (225)	3.000 (499)	3.000 (1079)
MAILER	3.000 (82)	3.000 (241)	3.000 (477)	3.000 (998)
PRIMEFACES	2.000 (34)	2.000 (111)	2.000 (226)	2.000 (454)

Set #2	Budget (s)			
Vulnerable Library	5	15	30	60
COMPRESS	2.000 (19)	0.364 (29)	0.000 (30)	0.000 (30)
TOMCAT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
JASYPT	0.000 (2)	0.000 (2)	0.000 (2)	0.000 (2)
JENKINS	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
MULTIJOB	2.500 (33)	2.500 (86)	2.500 (168)	2.500 (307)
HTTPCLIENT	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
ZEPELIN	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
NIFI	0.000 (1)	0.000 (1)	0.000 (1)	0.000 (1)
MAILER	2.500 (18)	2.500 (61)	2.500 (129)	2.500 (259)
PRIMEFACES	2.000 (20)	2.000 (57)	2.000 (95)	2.000 (196)



AGENDA

01

IL CONTESTO

Dove si colloca il lavoro?

02

LO SCOPE

Di cosa tratta il lavoro?

03

SIEGE

Cosa presenta il lavoro?

04

CONCLUSIONI

Cosa abbiamo appreso?



CONCLUSIONI

LEZIONI APPRESE

- SIEGE mostra risultati promettenti
- Esistono vulnerabilità più complesse di altre





CONCLUSIONI

LEZIONI APPRESE

- SIEGE mostra risultati promettenti
- Esistono vulnerabilità più complesse di altre



PROBLEMI APERTI

EvoSuite inadatto:
reimplementare SIEGE





CONCLUSIONI

PROSPETTIVE

- Assunzione di linea vulnerabile limitativa
- Studio empirico larga scala su progetti reali

LEZIONI APPRESE

- SIEGE mostra risultati promettenti
- Esistono vulnerabilità più complesse di altre



PROBLEMI APERTI

EvoSuite inadatto:
reimplementare SIEGE



Catch Me If You Can: Toward Automatic Exploit Generation of Known API Vulnerabilities

Tesi di Laurea Magistrale in
Informatica

Relatori

Prof. Fabio Palomba
Prof. Andrea De Lucia

Candidato

Emanuele Iannone
Matr. 0522500588

Anno Accademico 2019-2020