CENSUS
IT Security Works

DIMITRIOS GLYNOS (@dfunc)
JHUG OCT. 2018 MEETUP @ Eurobank

NOTES ON JAVA SECURITY

PREAMBLE

~

- About me
  - Co-founder & Director of Product Security Services at CENSUS
- Tonight's menu
  - Presentation of 5 security issues that are common in Java codebases

INTRODUCTION

SOFTWARE SECURITY

- Software security lies in the realm of *Information Security* and therefore it covers matters of:
  - **Confidentiality** - Unauthorised users must not be able to read information
  - **Integrity -** Unauthorised users must not be able to alter information
  - **Availability** - Authorised users must *always* be able to access information
- **Software Vulnerabilities**: bugs that may allow attackers to perform actions that lead to the loss of confidentiality, integrity and/or availability in systems
- Software security is concerned with the **identification**, **mitigation** and **management** of **vulnerabilities** in software

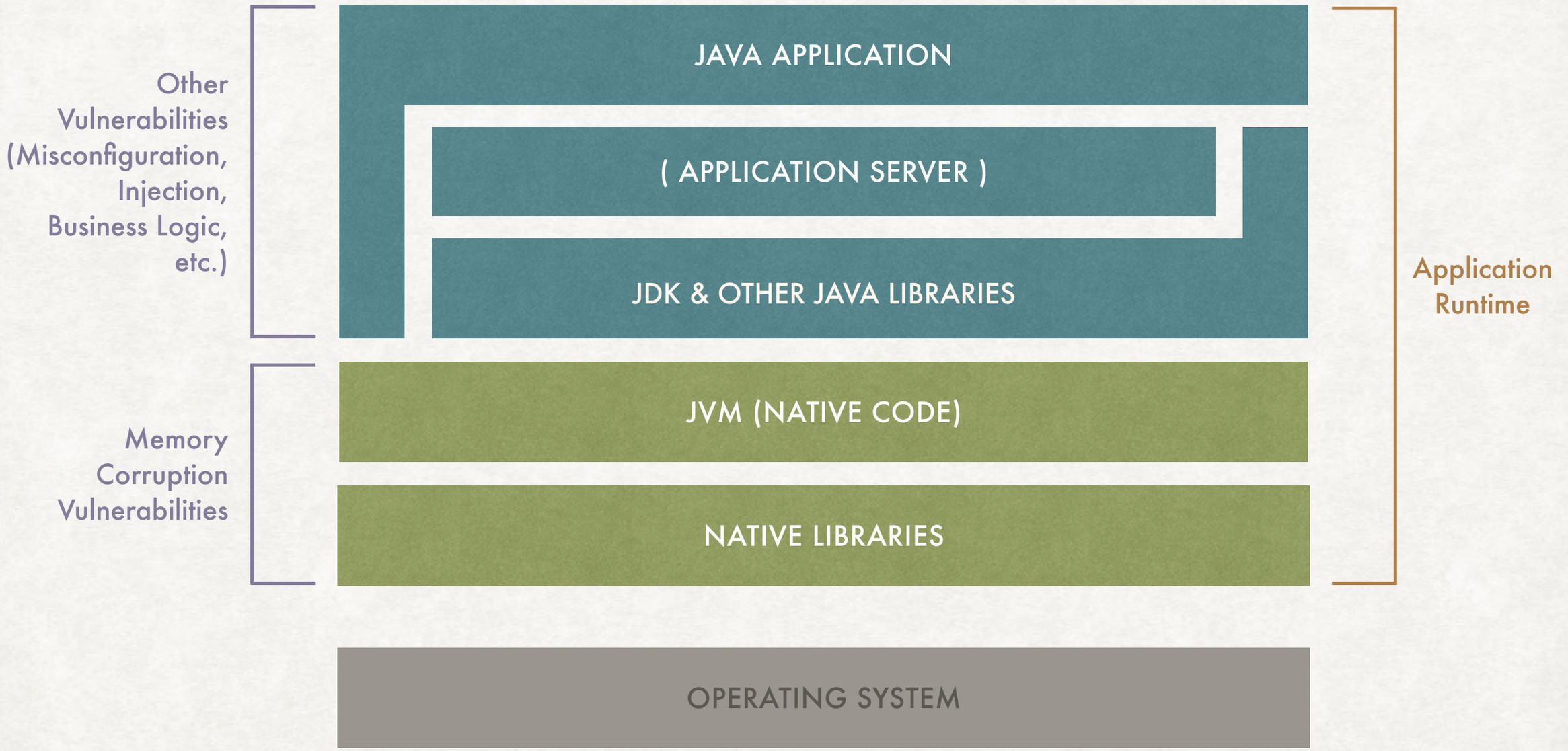
INTRODUCTION

MEET JAVA

- Java is a compiled language that follows the OO paradigm
- Java employs static type checking but allows some implicit type conversions
- Java bytecode executes in a Virtual Machine (JVM)

INTRODUCTION

OVERVIEW OF JAVA STACK ATTACK SURFACE



INTRODUCTION

JAVA SOFTWARE SECURITY

- Java software stacks share many vulnerability types with other software stacks (e.g. code injection, XSS, CSRF, SSRF etc.)
- However, one may note that there are some vulnerabilities that are characteristic of Java applications
- We'll focus on such vulnerabilities in this presentation

HANDLING THE WRONG EXCEPTION

ABOUT JAVA EXCEPTIONS

- **Checked exceptions**
  - They describe error states that the caller of an API must handle (I/O errors etc.)
  - A method that generates a checked exception must either expose it to its callers ("throw") or handle it locally ("catch"), else a **compile-time** error is raised
- **Unchecked exceptions**
  - Any exception class that inherits from java.lang.RuntimeException is an unchecked exception (e.g. null pointer dereference)
  - **At runtime**, an uncaught exception of any type will cause the VM instance to terminate.

HANDLING THE WRONG EXCEPTION

CAN BE ABUSED TO CONTROL PROGRAM FLOW

```
try {

    signal.encrypt(key, Float.parseFloat(frequency));

} catch (Exception e) {

    /* encrypt sometimes throws SignalTransformException

       on older hardware; let's use the static key and

       failsafe frequency then */

    key = STATIC_KEY;

    signal.encrypt(key, STD_FREQ);

}

signal.transmit();
```

HANDLING THE WRONG EXCEPTION

CAN BE ABUSED TO CONTROL PROGRAM FLOW

- *parseFloat* may raise a *NumberFormatException* if a non-floating number is provided for *frequency*
  - the exception handling code will be triggered in error
  - the program will switch to a static encryption key, which may be known to third parties

HANDLING THE WRONG EXCEPTION

CAN BE ABUSED TO CONTROL PROGRAM FLOW

- Issue usually identified during a source code audit
- Recommendation
  - Audit the exception handling logic
  - Avoid relying on a catch-all for specific Exceptions

RACE CONDITIONS

ON THE FILE SYSTEM

```
File file = new File("/tmp/somewhere");

if (file.exists()) {

        return;

}

FileOutputStream fos = new FileOutputStream(file);

fos.write(data);

fos.close();
```

RACE CONDITIONS

ON THE FILE SYSTEM

- Data of a recently created file may be clobbered
  - A file descriptor will be allocated only when *fos* has been created
  - Time Of Check To Time Of Use (TOCTTOU) bug between *file.exists()* and *fos* object creation
- A symlink attack scenario is also possible

RACE CONDITIONS

ON THE FILE SYSTEM

- The code does not ensure that a new file will be created on the filesystem
- Only the kernel can guaranty that (think O_EXCL)
- Recommendation: starting from JDK 1.7 (circa 2011) there are two methods to help in this
  - *java.nio.file.Files.newOutputStream* returning an OutputStream
  - *java.nio.file.Files.newInputStream* returning an InputStream

RACE CONDITIONS

ON THE FILE SYSTEM

```
Path p = Paths.get("/tmp/somewhere");

// this will raise a (checked) FileAlreadyExistsException

// if the file is already there

OutputStream out = Files.newOutputStream(p, CREATE_NEW);

out.write(data);

out.close();
```

WHEN THE API LETS YOU SHOOT YOURSELF
IN THE FOOT

- javax.mail.internet.MimeMessage.setSubject(String subject)
- *"The application must ensure that the subject does not contain any line breaks."*
- Hmmm.

WHEN THE API LETS YOU SHOOT YOURSELF

IN THE FOOT

POST /sendmail HTTP/1.1

…

{ subject: “Hey\r\nContent-Type: application/octet-stream; name=malware.zip\r\n … “ }

- Arbitrary content injection to e-mail message

WHEN THE API LETS YOU SHOOT YOURSELF
IN THE FOOT

- Why was this functionality there in the first place?
  - Probably to enable *folded* subject fields

```
Subject: some text
    spanning two rows
```

WHEN THE API LETS YOU SHOOT YOURSELF
IN THE FOOT

- The API developer passed the risk handling to the application developer
  - Remember C strcpy() ?
  - Do not do this if you can - create “safe” APIs
- Beware for 3rd party code that “hides” such unmanaged risks

INSECURE DESERIALISATION

INTRO

- Object serialisation
  - Transfer the object state into a stream of bytes
- Object deserialization
  - Create an object instance from the serialised object form

https://census-labs.com

INSECURE DESERIALISATION

WHAT CAN GO WRONG?

- Guess what happens if an attacker can modify the serialised object
  - Set object properties to interesting values
    - role=“admin”
- Instantiate Objects in the Classpath
  - usually when the class type can be controlled
- Call methods!

INSECURE DESERIALISATION

REMOTE CODE EXECUTION

- *ysoserial, a* tool for generating deserialisation “gadgets”!

  https://github.com/frohoff/ysoserial

```
$ java -jar ysoserial.jar CommonsCollections1 calc.exe | xxd

0000000: aced 0005 7372 0032 7375 6e2e 7265 666c  ....sr.2sun.refl
0000010: 6563 742e 616e 6e6f 7461 7469 6f6e 2e41  ect.annotation.A
0000020: 6e6e 6f74 6174 696f 6e49 6e76 6f63 6174  nnotationInvocat
...
0000550: 7672 0012 6a61 7661 2e6c 616e 672e 4f76  vr..java.lang.Ov
0000560: 6572 7269 6465 0000 0000 0000 0000 0000  erride..........
0000570: 0078 7071 007e 003a                      .xpq.~.:
```

INSECURE DESERIALISATION

A PLAGUE?

"Oracle Plans to Drop Java Serialization Support, the Source of Most Security Bugs", Bleepingcomputer May 2018

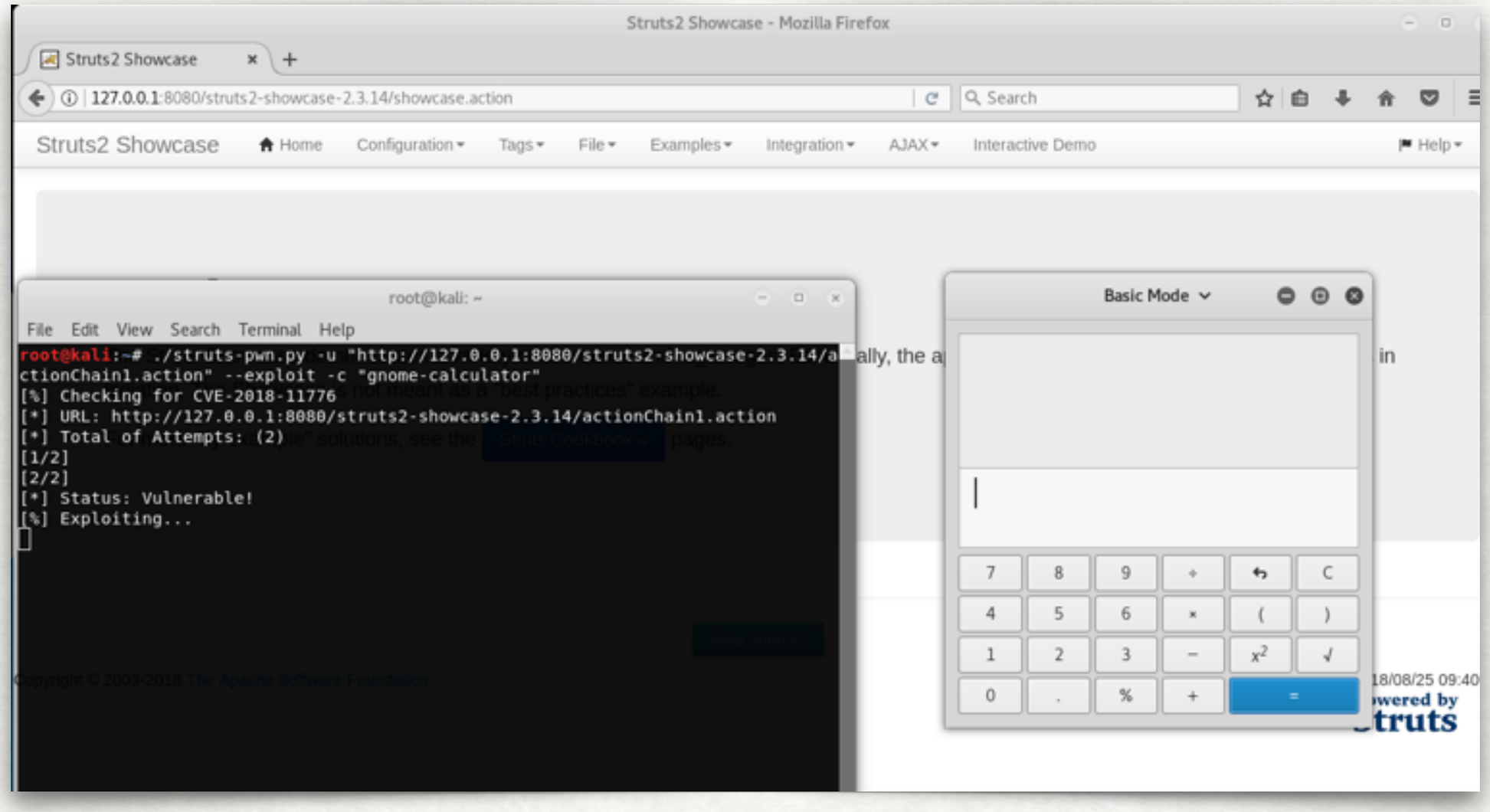
"Just one Apache Struts (Java) deserialization bug from last year affected an estimated 65% of all Fortune 100 companies, showing how widespread the practice of serializing data is, and how one bug could bring down the security of the world's biggest companies."

...

Source: https://www.bleepingcomputer.com/news/security/oracle-plans-to-drop-java-serialization-support-the-source-of-most-security-bugs/

INSECURE DESERIALISATION

APACHE STRUTS EXPLOITATION DEMO



Example exploit from https://github.com/mazen160/struts-pwn_CVE-2018-11776

https://census-labs.com

INSECURE DESERIALISATION

WHAT CAN YOU DO?

- Do deserialisation the same as object construction:
  - Put default values to fields that don't have a value
  - Do validity checking on the values imported via readObject()
- Do not permit arbitrary class loading
- Want to pass server objects to web clients? Encrypt, sign and replay-protect them!

INCORPORATING THIRD PARTY CODE

AKA DEPENDENCY HELL

- The JVM now supports many programming languages
  - Groovy, Scala, Clojure etc.
- These languages come with frameworks
- The frameworks come with dependencies
- Some of these projects standing in the outer rim of the Java galaxy have untracked vulnerabilities that concern them

INCORPORATING THIRD PARTY CODE

AKA DEPENDENCY HELL

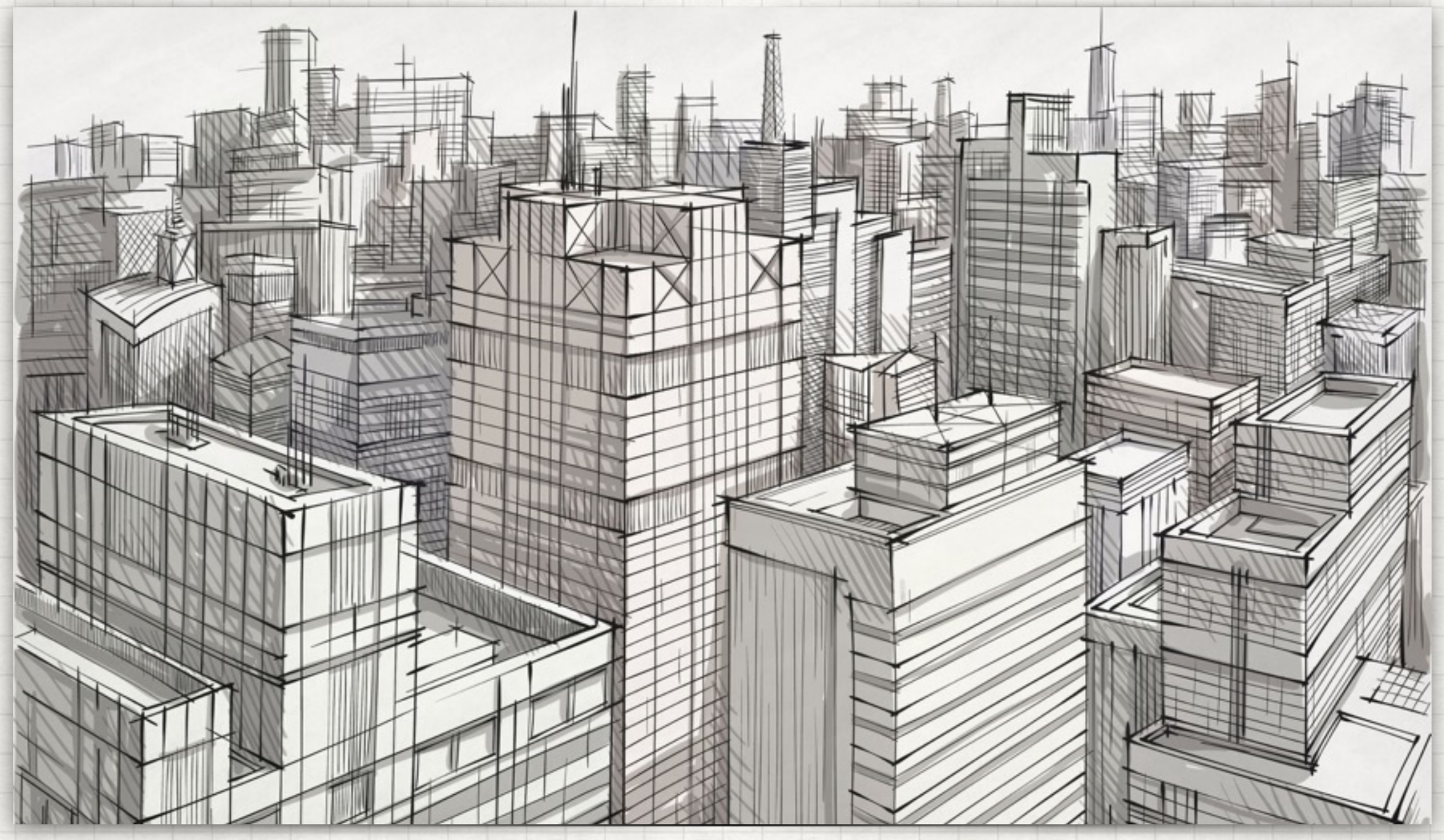
- Many developers choose to stay with a packaged, but not fixed, version of the software
- Others, keep locally patched versions of these projects
- Generally speaking
  - the correct solution to this problem would be to let the maintainer know the issue and propose a patch
  - get involved with projects you rely upon!

SUMMARISING

- Beware of
  - bad exception handlers - they may allow for security control circumvention
  - race conditions - you are not executing alone
  - insecure APIs - protect your users
  - insecure deserialisation - limit your class loading capabilities
  - insecure third party components - get involved!

QUESTIONS?

THANK YOU



https://census-labs.com