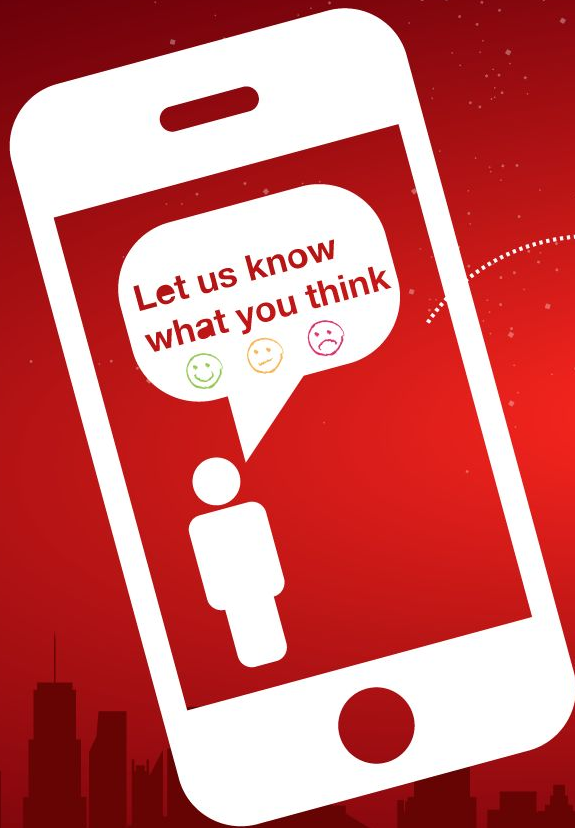




SECURING THE JVM

Neither for fun nor for profit,
but do you have a choice?

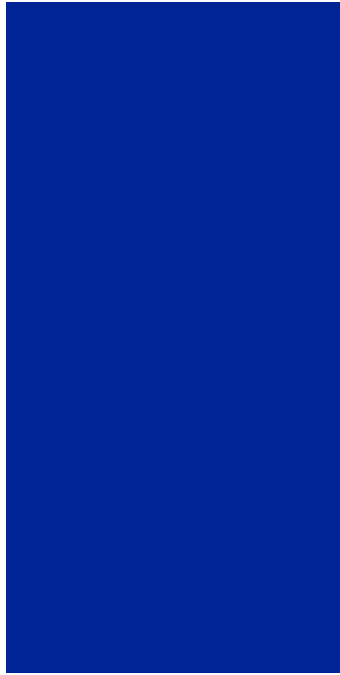
goto;
chicago



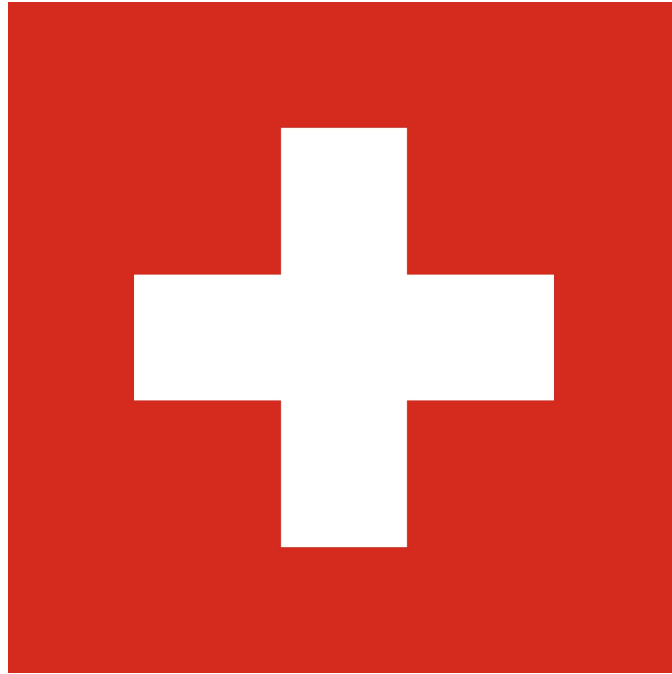
Click 'Rate Session'

Rate **5** sessions to get the
supercool GOTO reward

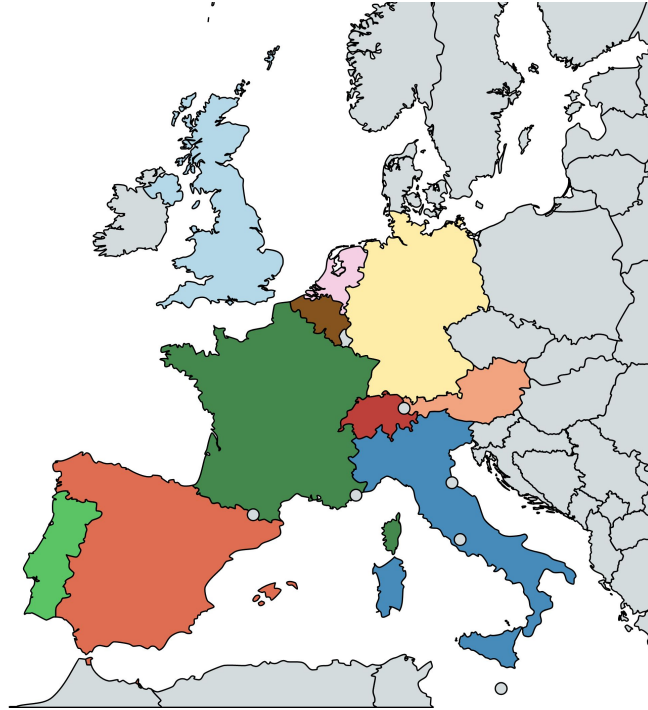
LITTLE TRIVIA TO GET STARTED



LITTLE TRIVIA TO GET STARTED



WHERE IS ~~WALDO~~ SWITZERLAND?



EXOSCALE



@nicolas_frankel

FUN FACT #1



EXOSCALE



@nicolas_frankel

FUN FACT #2



FUN FACT #3



EXOSCALE



@nicolas_frankel

ME, MYSELF AND I

- **Security-minded developer**
- **Developer advocate**
 - Previously
developer/architect
consultant





European alternative to the “big” cloud-computing players

- **Privacy-minded**
- **Great support**



@nicolas_frankel

WHO DOESN'T KNOW THAT TRICK?

```
public class Foo {  
    private String hidden = "This should remain inaccessible!";  
}  
  
public class Oops {  
    public static void main(String[] args) throws Exception {  
        Foo foo = new Foo();  
        Class<? extends Foo> clazz = foo.getClass();  
        Field field = clazz.getDeclaredField( "hidden");  
        field.setAccessible( true);  
        Object hidden = field.get(foo);  
        System.out.println(hidden);  
    }  
}
```

REFLECTION API

- **Part of the JRE**
 - `java.lang.reflect`



USED BY A LOT OF LANGUAGES/Frameworks

- **Dynamic languages**
 - Groovy
- **JPA frameworks**
 - Hibernate
 - Others
- **Spring**
- **Etc.**



EXOSCALE



@nicolas_frankel

YOU CAN DO PRETTY STUPID STUFF

```
public class Bar {  
    private int hidden = 5;  
}  
  
public class OopsAgain {  
    public static void main(String[] args) throws Exception {  
        Bar bar = new Bar();  
        Class<? extends Bar> clazz = bar.getClass();  
        Field field = clazz.getDeclaredField("hidden");  
        Field type = Field.class.getDeclaredField("type");  
        AccessibleObject.setAccessible(  
            new AccessibleObject[] { field, type }, true);  
        type.set(field, String.class);  
        field.set(bar, "This should print 5!");  
        Object hidden = field.get(bar);  
        System.out.println(hidden);  
    }  
}
```



BUT EVERYTHING IS POSSIBLE

- **Make networks calls**
- **Compile code on the fly**
- **Execute code**
- **Etc.**



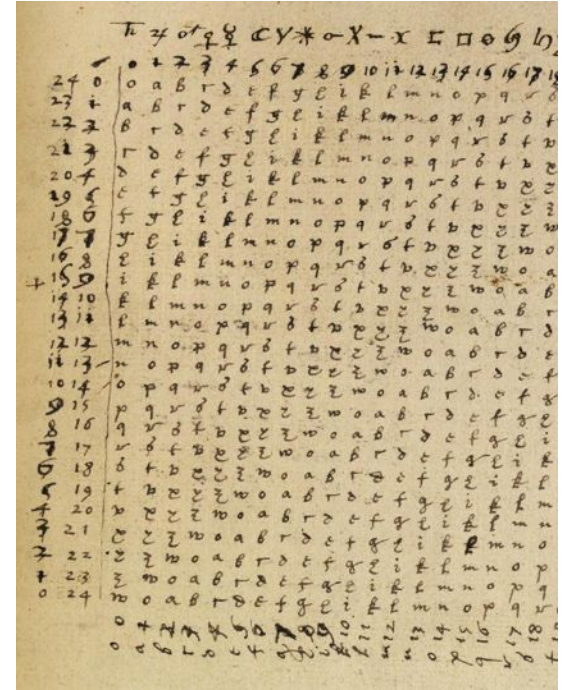
SAFEGUARDS?

- **Static analyzers**
- **Byte-code analyzers**
- **Code reviews**
- **Security team**
- **Etc.**



STEGANOGRAPHY

“Steganography is the practice of concealing a file, message, image, or video within another file, message, image, or video”



EXOSCALE



@nicolas_frankel

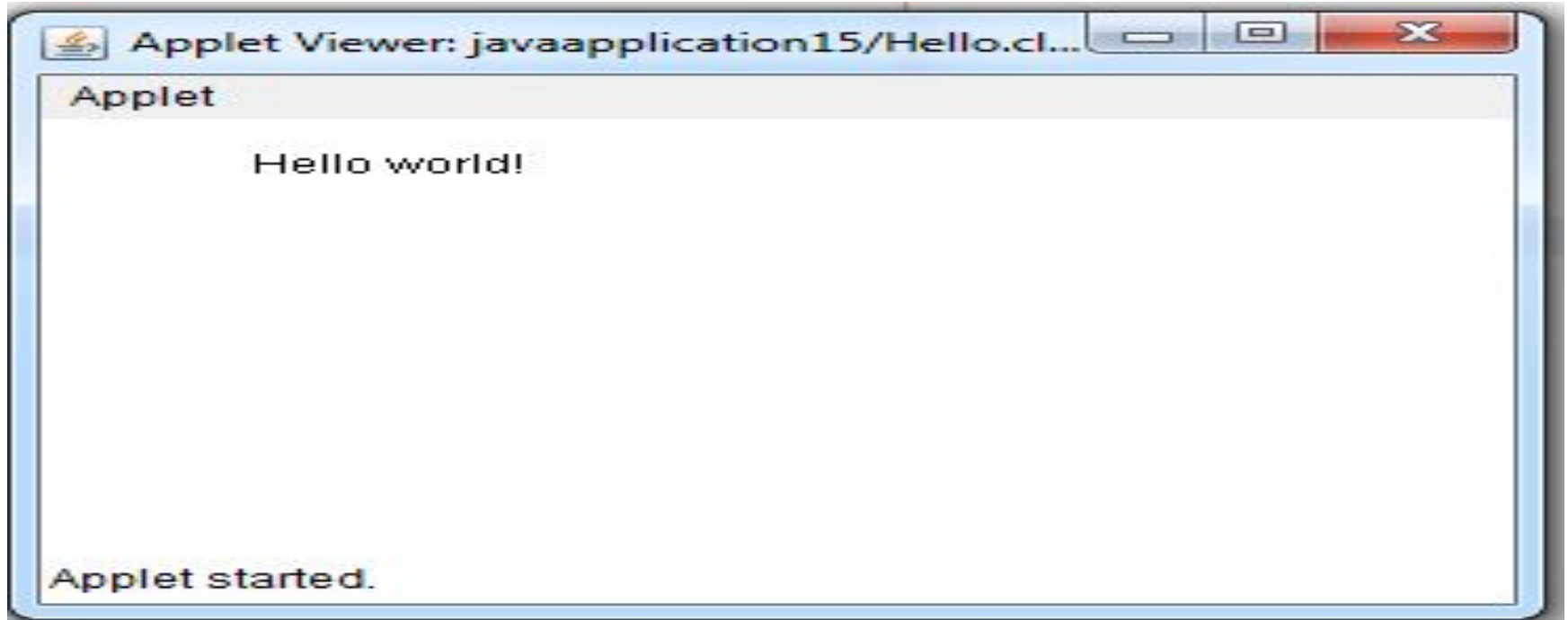
Time for **DEMO**

A SIMPLE PROCESS

1. **Publish a new library to central**
 - With hidden steganographic/compile/run features
2. **Add the library to your POM**
3. **Add an image with embedded code**
4. **Call the library**
 - Pretend to read the image
5. **Enjoy**



REMEMBER?



EXOSCALE



@nicolas_frankel

APPLETS

- **Meant to run on the client machine**
 - Forbidden to do “dangerous” stuff
 - Run in a sandbox



AVAILABLE MODES

- **Standard mode**
 - a.k.a. “God Mode”
- **Sandbox mode**
 - Through a SecurityManager



ACTIVATING THE SECURITY MANAGER

- **On the command line**

- `java -Djava.security.manager ...`

- **Via the API**

- `System.setSecurityManager()`

PERMISSIONS CONFIGURATION

- **Java 9**

- `conf/security/java.security`
- → `conf/security/java.policy`

- **Java 8**

- `jre/lib/security/java.security`
- → `jre/lib/security/java.policy`

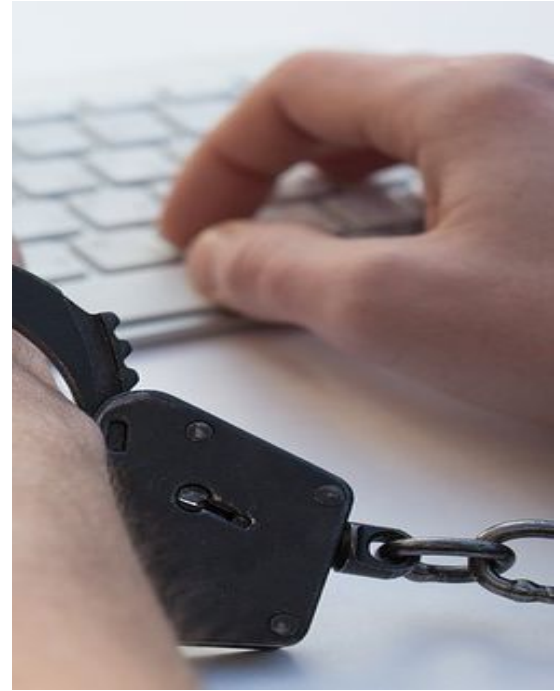


ALTERNATIVE POLICY FILE

- **Setting the policy file**
 - `-Djava.security.policy==my.policy`
- **Adding additional permissions**
 - `-Djava.security.policy=my.policy`

PRINCIPLE OF LEAST PRIVILEGE

“Requires that in a particular abstraction layer of a computing environment, every module must be able to access only the information and resources that are necessary for its legitimate purpose”



EXOSCALE



@nicolas_frankel

DEFAULT PERMISSIONS

- **Listen on ports**
- **Read a few System properties**



PERMISSIONS ARE ENFORCED IN THE JDK!

```
public class AccessibleObject
    implements AnnotatedElement {
    public void setAccessible(boolean flag)
        throws SecurityException {
        SecurityManager sm =
            System.getSecurityManager();
        if (sm != null)
            sm.checkPermission(ACCESS_PERMISSION);
        setAccessible0(this, flag);
    }
}
```

PERMISSIONS

Permission	Details
AllPermission	😊
PropertyPermission	• Read • Write
FilePermission	• Read • Write • Execute • Delete
ReflectPermission	Suppress reflective access checks
RuntimePermission	• More granular reflective access checks • Security-related • Class-loading related • Exit Virtual Machine, shutdown hooks, etc.



PERMISSIONS (CONTINUED)

Permission	Details
SocketPermission	• accept • connect • listen • resolve
SSLPermission	• Get/set SSL context • Set hostname verifier
AuthPermission	
ServicePermission	Kerberos-related
AWTPermission	• Clipboard • Tray • Windows • Pointer • etc.



STRUCTURE OF A POLICY FILE (SIMPLIFIED)

```
grant codebase "file:myjar" {  
    permission Foo "foo";  
    permission Bar "bar", "baz";  
};
```

Time for **DEMO**





Please

**Remember to
rate this session**

Thank you!

THANKS!

- <https://blog.frankel.ch/>
- @nicolas_frankel
- <https://git.io/fx54L>
- <https://git.io/fhZ4t>
- <https://bit.ly/2DQBRrt>

