More than Free Schwag

GoSecure's Open Source Contributions to Malware Analysis, Code Hardening and RDP Snooping

Olivier Bilodeau, <obilodeau@gosecure.net>

Agenda

- Our Vision
- Malware Analysis with Malboxes
- Code Hardening with Find Security Bugs
- RDP Snooping with PyRDP
- Future Work

$ whoami

- Cybersecurity Research Director
- Reverse-Engineering and Tools Enthusiast
- International Speaker
  - DefCon, BlackHat, RSAC, DerbyCon, 44CON, etc.
- Co-founder Montrehack (hands-on security workshops)
- VP Training and Hacker Jeopardy at NorthSec

The Vision

Two Types of Products and Services

Polished

- Easy to Use
- Paid For
- Supported
- Proven

Rough

- DIY
- Free, Adaptable
- You Are on Your Own
- Proof of Concept

malboxes

Making Malware Analysis More Accessible

Problems in Malware Analysis

- Malware analysis is not accessible to newcomers
- Easy to mess things up (get infected)
- Building an environment with all the tools installed takes time
- Team work is hard (tools don’t encourage it)

Inspired by DevOps

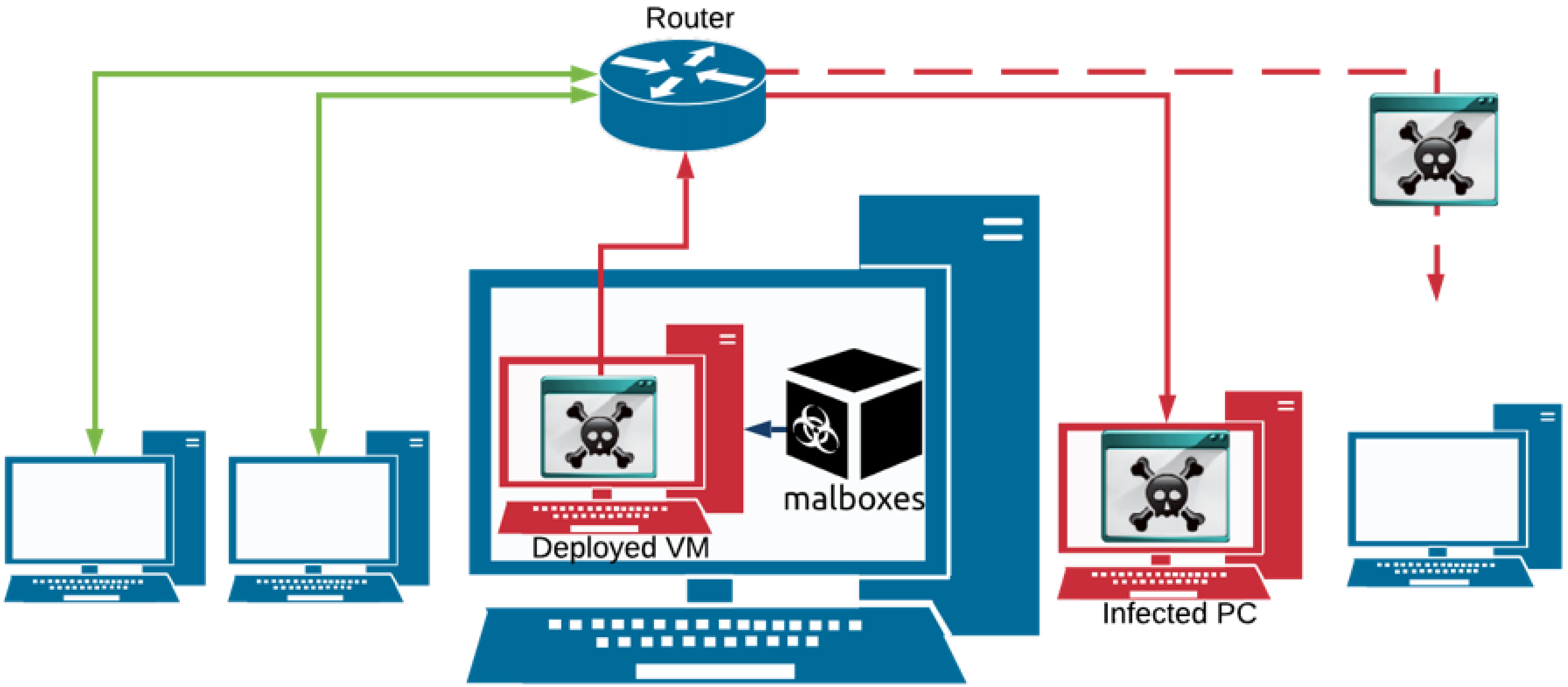
- Core Principle: Infrastructure as Code
- Reproducible
- Throw-away
- Efficient

Architecture

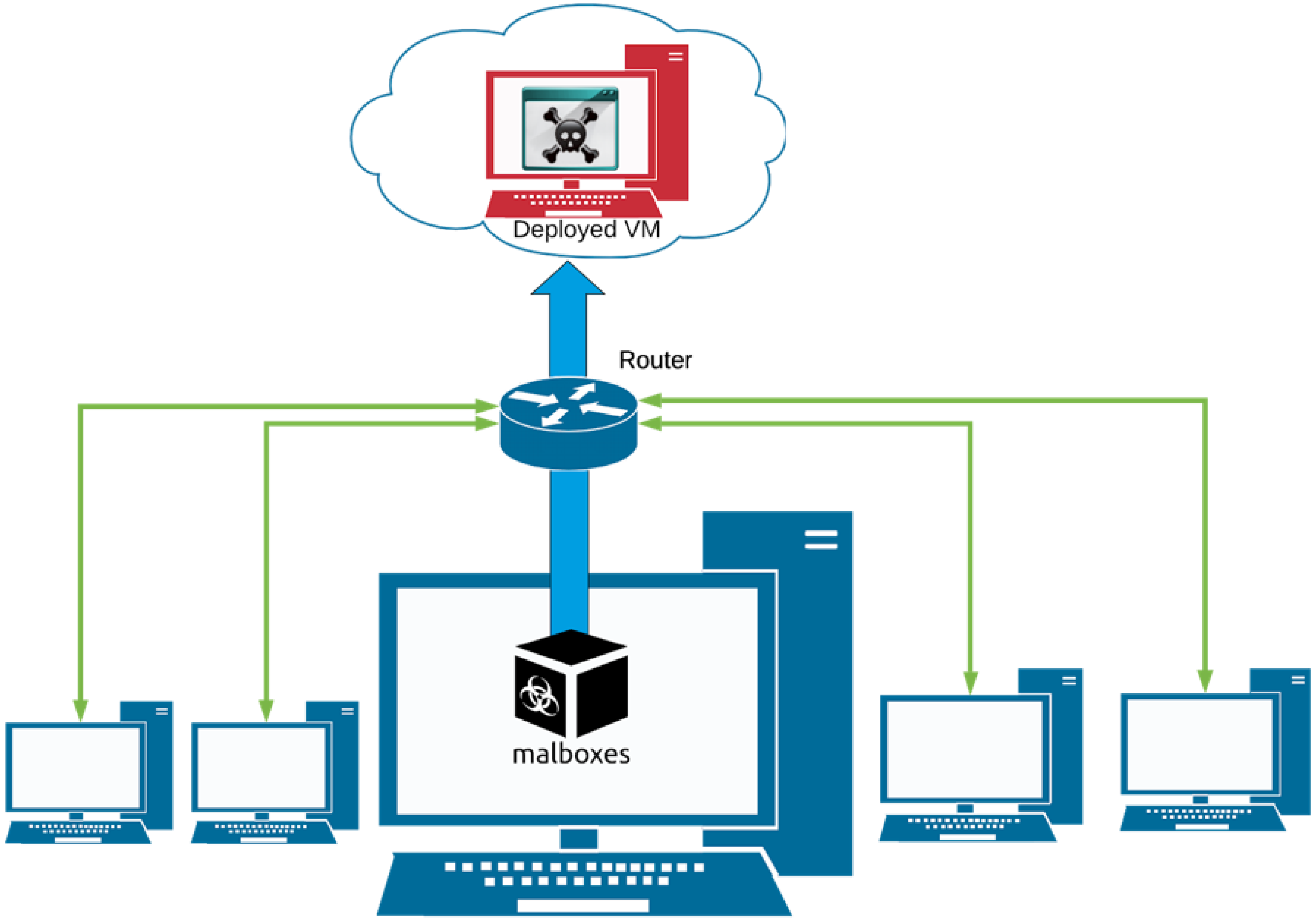
```
          FRONT-END           BACK-END
      +---------------+   +---------------+     +-------------------+
      |               |   |               +---> |  Autounattend.xml |
      |               +-> |  packer       |     +-------------------+
      |               |   |               +-+
      |               |   +---------------+ |   +--------------------+
      |  malboxes.py  |                     +-> |                    |
      |               |   +---------------+     |          PowerShell|
      |               |   |               |     |   WinRM  winrmcp   |
      |               +-> |  vagrant      +---> |          Shell     |
      |               |   |               |     |          Chocolatey|
      +---------------+   +---------------+     +--------------------+

                          +------------------------------------------+
                          |                                          |
                          | VirtualBox / vSphere (ESXi) / KVM / AWS* |
                          |                                          |
                          +------------------------------------------+
*: NEW!
```

New Feature: Deploy to AWS

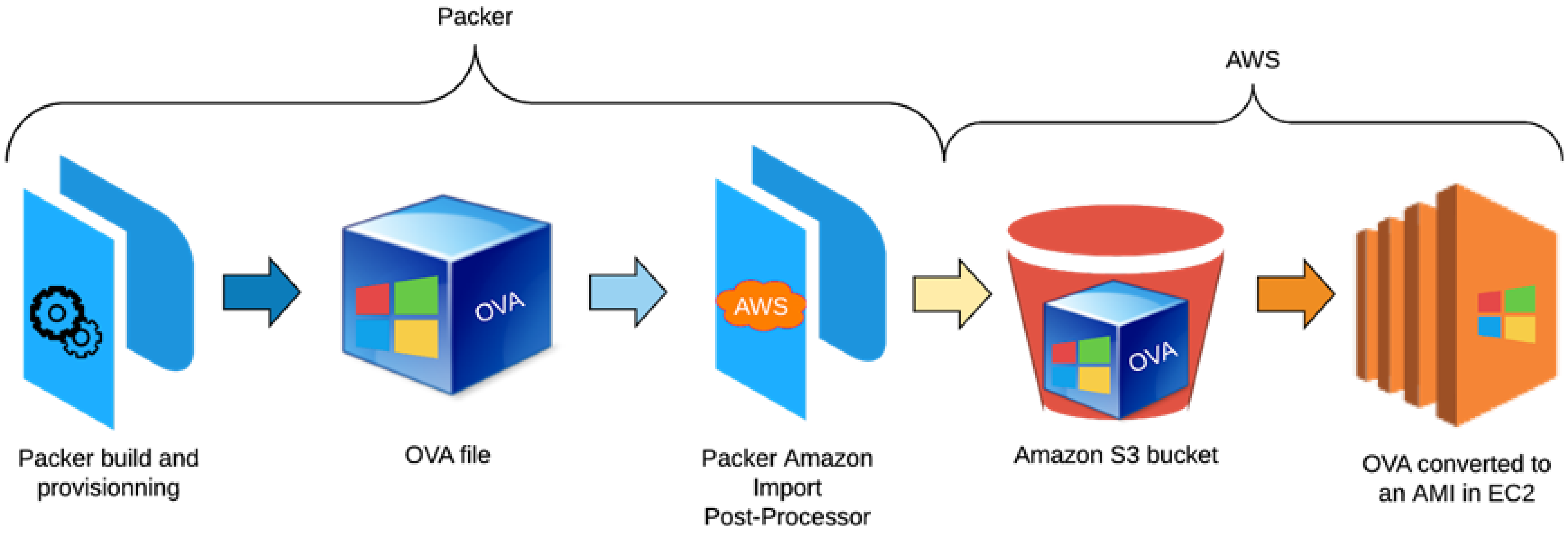
Background



Reducing Risks



The Process

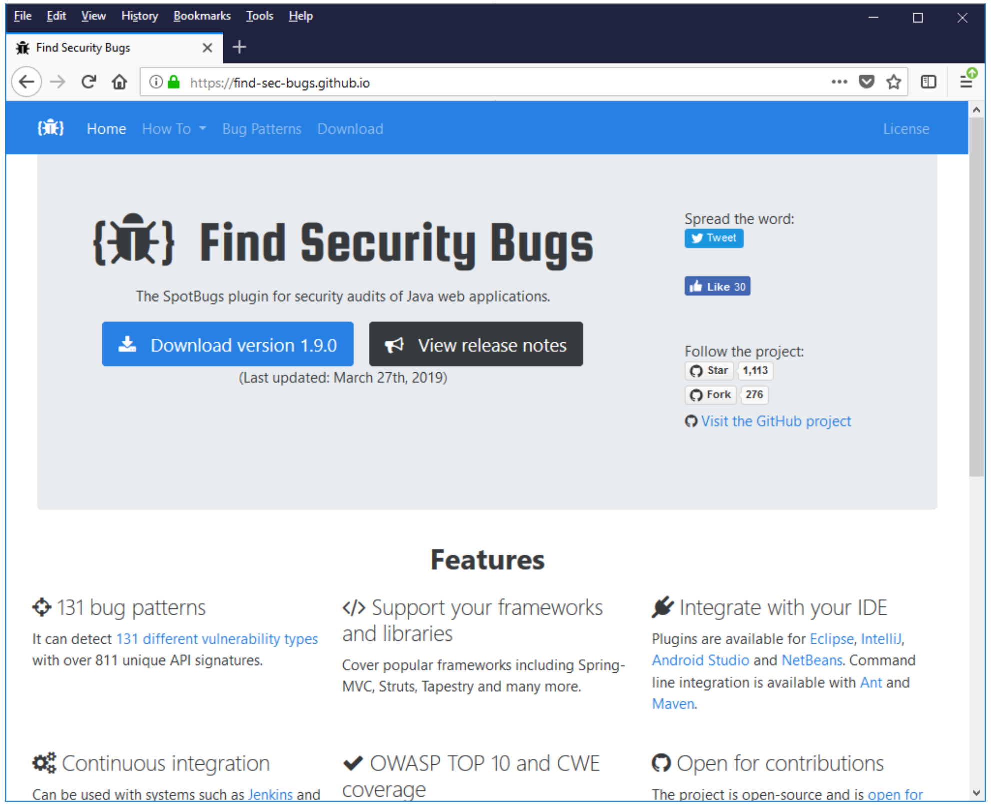

Packer
AWS
Packer build and provisionning
OVA
OVA file
AWS
Packer Amazon Import Post-Processor
OVA
Amazon S3 bucket
OVA converted to an AMI in EC2

Malboxes - Deploy to AWS Cloud Demo

FindSecBugs

Find Security Bugs in a Nutshell

- Detectors built around the SpotBugs engine with a focus on security issues
- Open-source
- OWASP project since 2019
- 131 bug patterns
- Works great with Java, Kotlin and JSP



Find Security Bugs

The SpotBugs plugin for security audits of Java web applications.

(Last updated: March 27th, 2019)

Visit the GitHub project

Features

131 bug patterns

It can detect 131 different vulnerability types with over 811 unique API signatures.

Support your frameworks and libraries

Cover popular frameworks including Spring-MVC, Struts, Tapestry and many more.

Integrate with your IDE

Plugins are available for Eclipse, IntelliJ, Android Studio and NetBeans. Command line integration is available with Ant and Maven.

Continuous integration

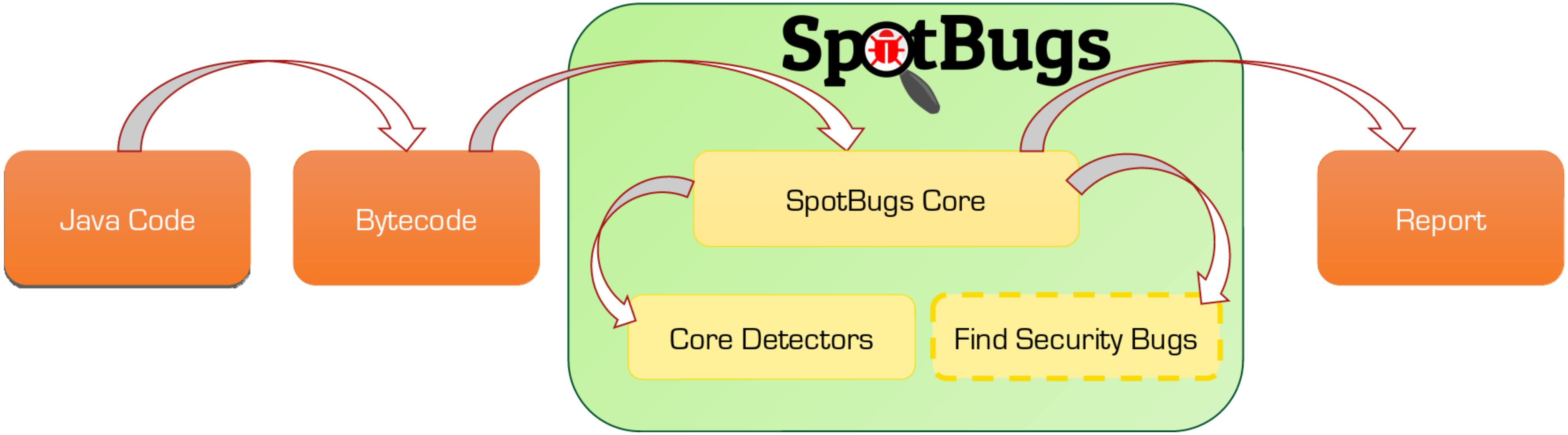
Can be used with systems such as Jenkins and

OWASP TOP 10 and CWE coverage

Open for contributions

The project is open-source and is open for

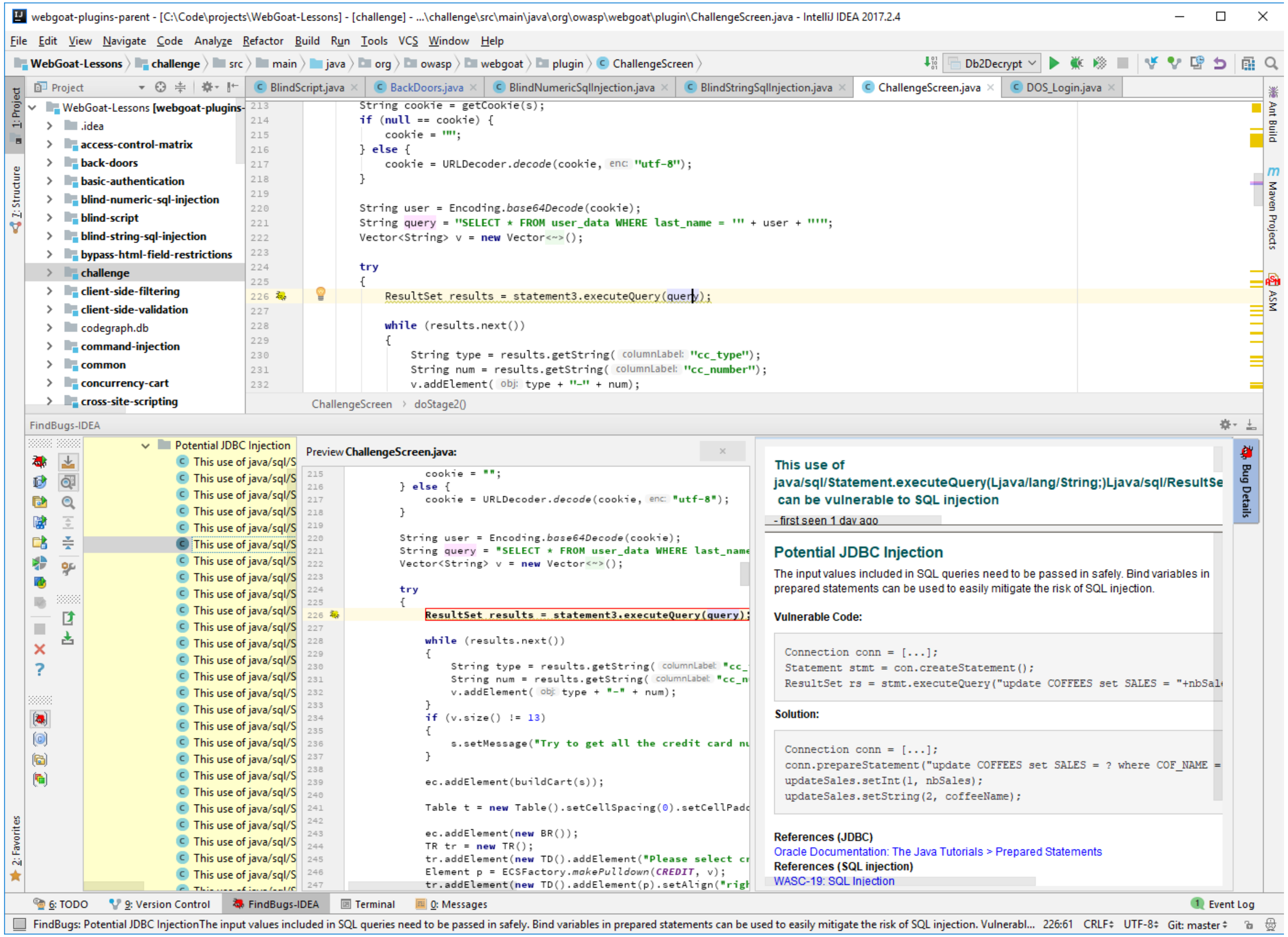
How Does It Work?



Types of Vulnerabilities

- SQL/HQL Injection
- Command Injection
- Cryptographic Weaknesses
- Cross-Site Scripting
- Path Traversal
- Template Injection
- Hard-Coded Password
- Insecure Configuration
- XML External Entity
- Predictable Random Number Generator

Integrated in IDEs


webgoat-plugins-parent - [C:\Code\projects\WebGoat-Lessons] - [challenge] - ...\challenge\src\main\java\org\owasp\webgoat\plugin\ChallengeScreen.java - IntelliJ IDEA 2017.2.4
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
WebGoat-Lessons › challenge › src › main › java › org › owasp › webgoat › plugin › ChallengeScreen
Db2Decrypt
Project
BlindScript.java
BackDoors.java
BlindNumericSqlInjection.java
BlindStringSqlInjection.java
ChallengeScreen.java
DOS_Login.java
WebGoat-Lessons [webgoat-plugins-
.idea
access-control-matrix
back-doors
basic-authentication
blind-numeric-sql-injection
blind-script
blind-string-sql-injection
bypass-html-field-restrictions
challenge
client-side-filtering
client-side-validation
codegraph.db
command-injection
common
concurrency-cart
cross-site-scripting
1: Project
7: Structure
2: Favorites
Ant Build
Maven Projects
ASM
ChallengeScreen › doStage2()
FindBugs-IDEA
Potential JDBC Injection
This use of java/sql/S
Preview ChallengeScreen.java:
Bug Details
6: TODO
9: Version Control
FindBugs-IDEA
Terminal
0: Messages
Event Log

```
String cookie = getCookie(s);
if (null == cookie) {
    cookie = "";
} else {
    cookie = URLDecoder.decode(cookie, enc: "utf-8");
}

String user = Encoding.base64Decode(cookie);
String query = "SELECT * FROM user_data WHERE last_name = '" + user + "'";
Vector<String> v = new Vector<~>();

try
{
    ResultSet results = statement3.executeQuery(query);

    while (results.next())
    {
        String type = results.getString( columnLabel: "cc_type");
        String num = results.getString( columnLabel: "cc_number");
        v.addElement( obj: type + "-" + num);
```

```
        cookie = "";
    } else {
        cookie = URLDecoder.decode(cookie, enc: "utf-8");
    }

    String user = Encoding.base64Decode(cookie);
    String query = "SELECT * FROM user_data WHERE last_name
    Vector<String> v = new Vector<~>();

    try
    {
        ResultSet results = statement3.executeQuery(query);

        while (results.next())
        {
            String type = results.getString( columnLabel: "cc_
            String num = results.getString( columnLabel: "cc_n
            v.addElement( obj: type + "-" + num);
        }
        if (v.size() != 13)
        {
            s.setMessage("Try to get all the credit card nu
        }

        ec.addElement(buildCart(s));

        Table t = new Table().setCellSpacing(0).setCellPadd

        ec.addElement(new BR());
        TR tr = new TR();
        tr.addElement(new TD().addElement("Please select cr
        Element p = ECSFactory.makePulldown(CREDIT, v);
        tr.addElement(new TD().addElement(p).setAlign("righ
```

This use of java/sql/Statement.executeQuery(Ljava/lang/String;)Ljava/sql/ResultSe can be vulnerable to SQL injection

- first seen 1 day ago

Potential JDBC Injection

The input values included in SQL queries need to be passed in safely. Bind variables in prepared statements can be used to easily mitigate the risk of SQL injection.

Vulnerable Code:

```
Connection conn = [...];
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("update COFFEES set SALES = "+nbSal
```

Solution:

```
Connection conn = [...];
conn.prepareStatement("update COFFEES set SALES = ? where COF_NAME =
updateSales.setInt(1, nbSales);
updateSales.setString(2, coffeeName);
```

References (JDBC)

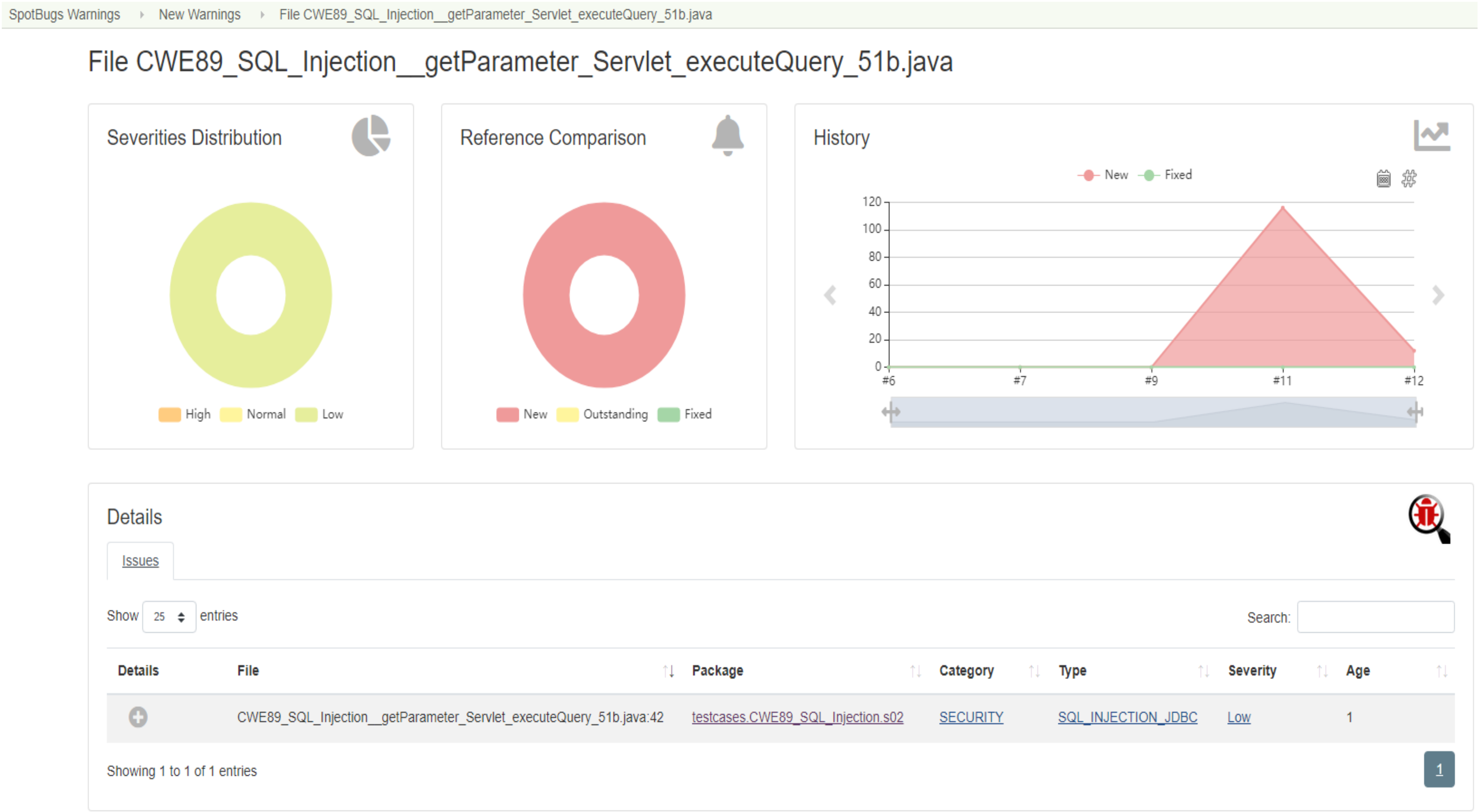
Oracle Documentation: The Java Tutorials > Prepared Statements

References (SQL injection)

WASC-19: SQL Injection

FindBugs: Potential JDBC InjectionThe input values included in SQL queries need to be passed in safely. Bind variables in prepared statements can be used to easily mitigate the risk of SQL injection. Vulnerabl... 226:61 CRLF UTF-8 Git: master

and in Continous Integration (CI)



SpotBugs Warnings › New Warnings › File CWE89_SQL_Injection__getParameter_Servlet_executeQuery_51b.java

File CWE89_SQL_Injection__getParameter_Servlet_executeQuery_51b.java

Details

Issues

Show 25 entries

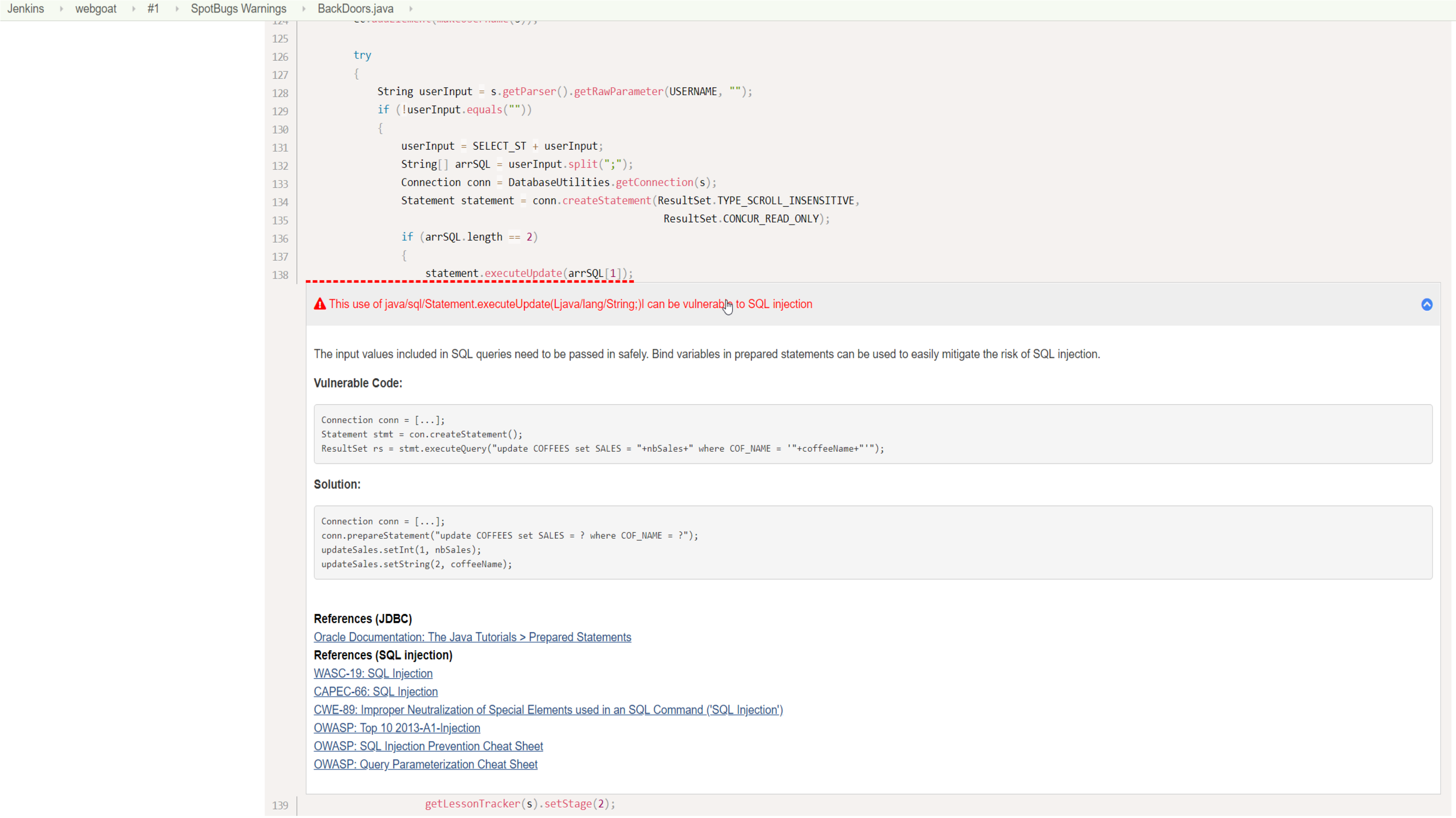
Search:

Details	File	Package	Category	Type	Severity	Age
⊕	CWE89_SQL_Injection__getParameter_Servlet_executeQuery_51b.java:42	testcases.CWE89_SQL_Injection.s02	SECURITY	SQL_INJECTION_JDBC	Low	1

Showing 1 to 1 of 1 entries

1

and in Continous Integration (CI)



Jenkins › webgoat › #1 › SpotBugs Warnings › BackDoors.java ›

```
        try
        {
            String userInput = s.getParser().getRawParameter(USERNAME, "");
            if (!userInput.equals(""))
            {
                userInput = SELECT_ST + userInput;
                String[] arrSQL = userInput.split(";");
                Connection conn = DatabaseUtilities.getConnection(s);
                Statement statement = conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
                                                           ResultSet.CONCUR_READ_ONLY);
                if (arrSQL.length == 2)
                {
                    statement.executeUpdate(arrSQL[1]);
```

⚠ This use of java/sql/Statement.executeUpdate(Ljava/lang/String;)I can be vulnerable to SQL injection

The input values included in SQL queries need to be passed in safely. Bind variables in prepared statements can be used to easily mitigate the risk of SQL injection.

Vulnerable Code:

```
Connection conn = [...];
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("update COFFEES set SALES = "+nbSales+" where COF_NAME = '"+coffeeName+"'");
```

Solution:

```
Connection conn = [...];
conn.prepareStatement("update COFFEES set SALES = ? where COF_NAME = ?");
updateSales.setInt(1, nbSales);
updateSales.setString(2, coffeeName);
```

References (JDBC)

Oracle Documentation: The Java Tutorials > Prepared Statements

References (SQL injection)

WASC-19: SQL Injection
CAPEC-66: SQL Injection
CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
OWASP: Top 10 2013-A1-Injection
OWASP: SQL Injection Prevention Cheat Sheet
OWASP: Query Parameterization Cheat Sheet

```
                    getLessonTracker(s).setStage(2);
```

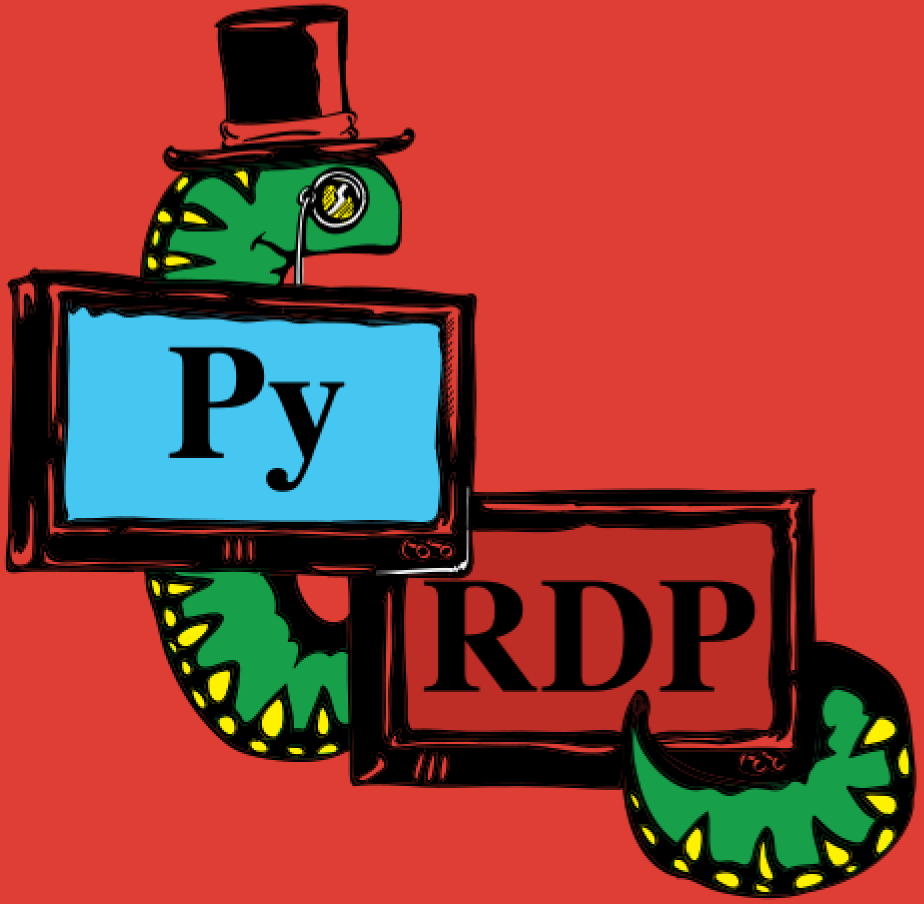
Integrations

Many free and open-source options

- SonarQube (with Sonar-FindBugs)
- Jenkins (with Warnings-NG)

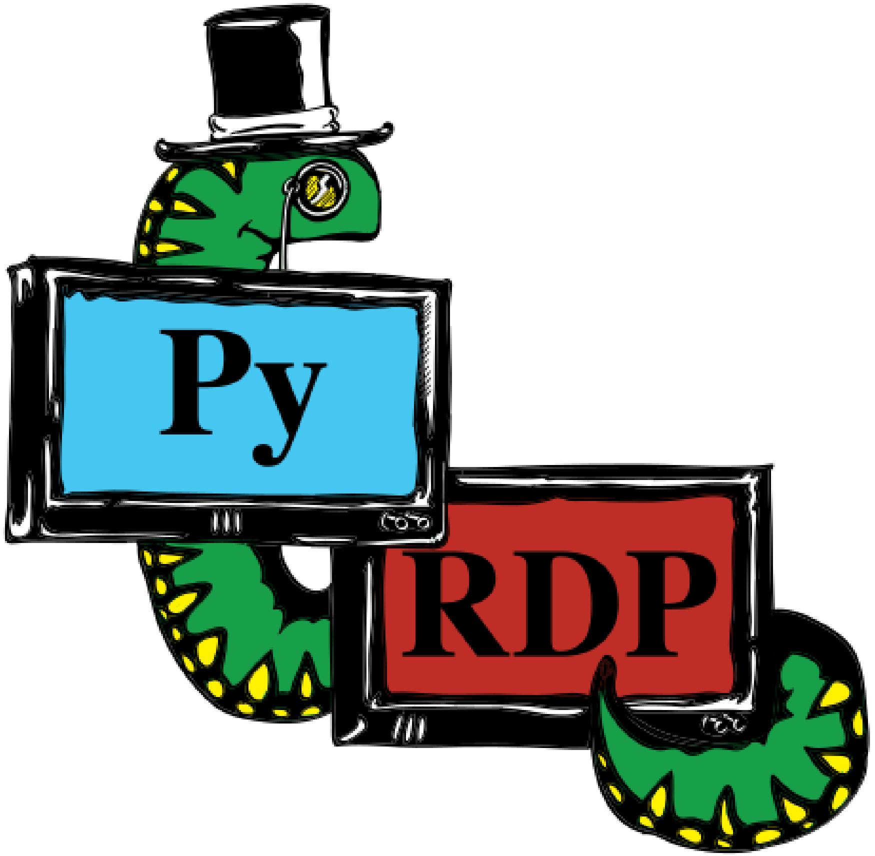
Integrated in many commercial solutions

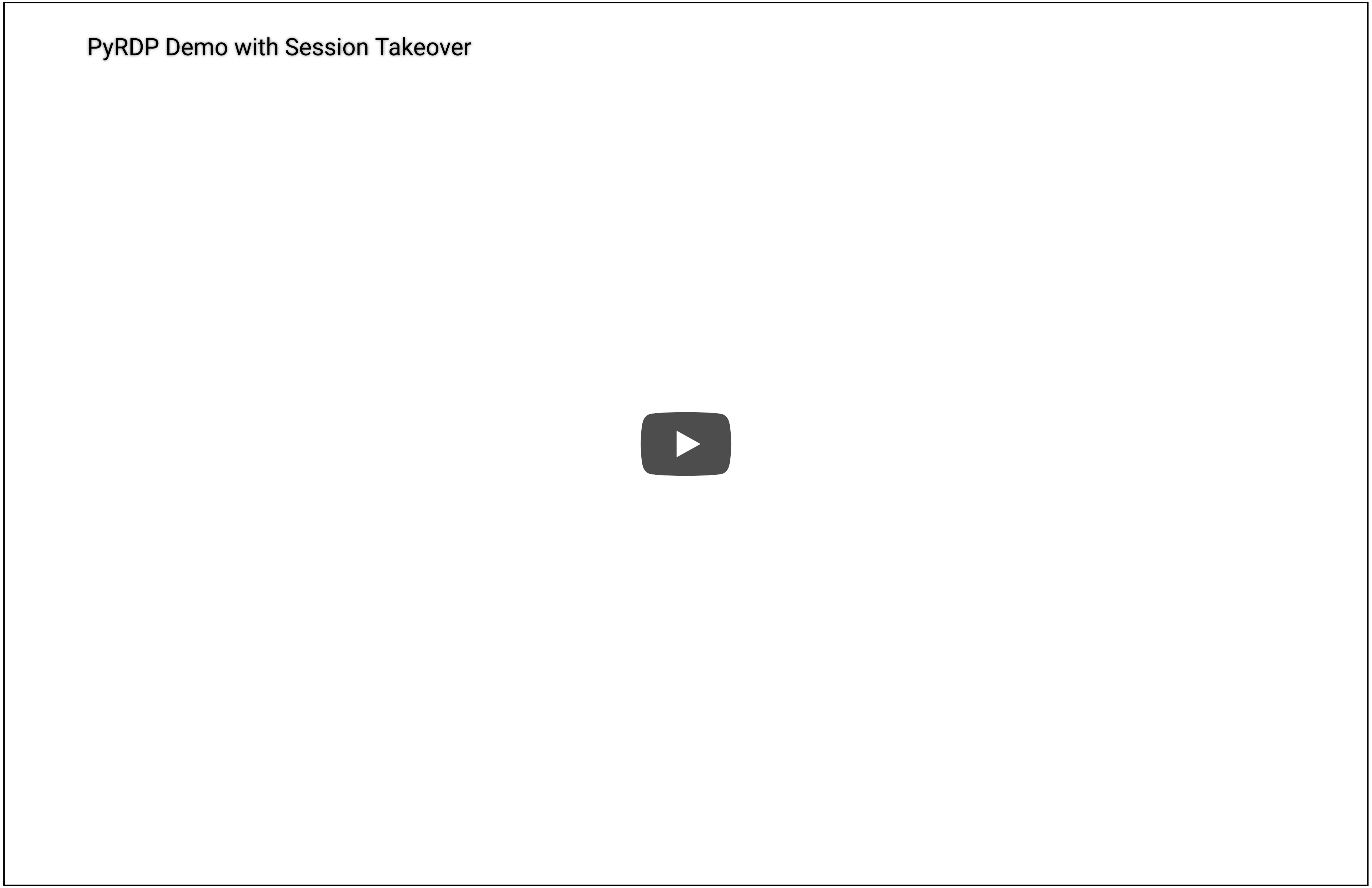
- Gitlab
- CodeDX


Py
RDP

PyRDP Is

- Remote Desktop Protocol MITM
- Active Clipboard Stealer
- PowerShell / cmd Injection on Login
- Take Control of the Remote Session
- Client-Side File Browsing





PyRDP Demo with Session Takeover

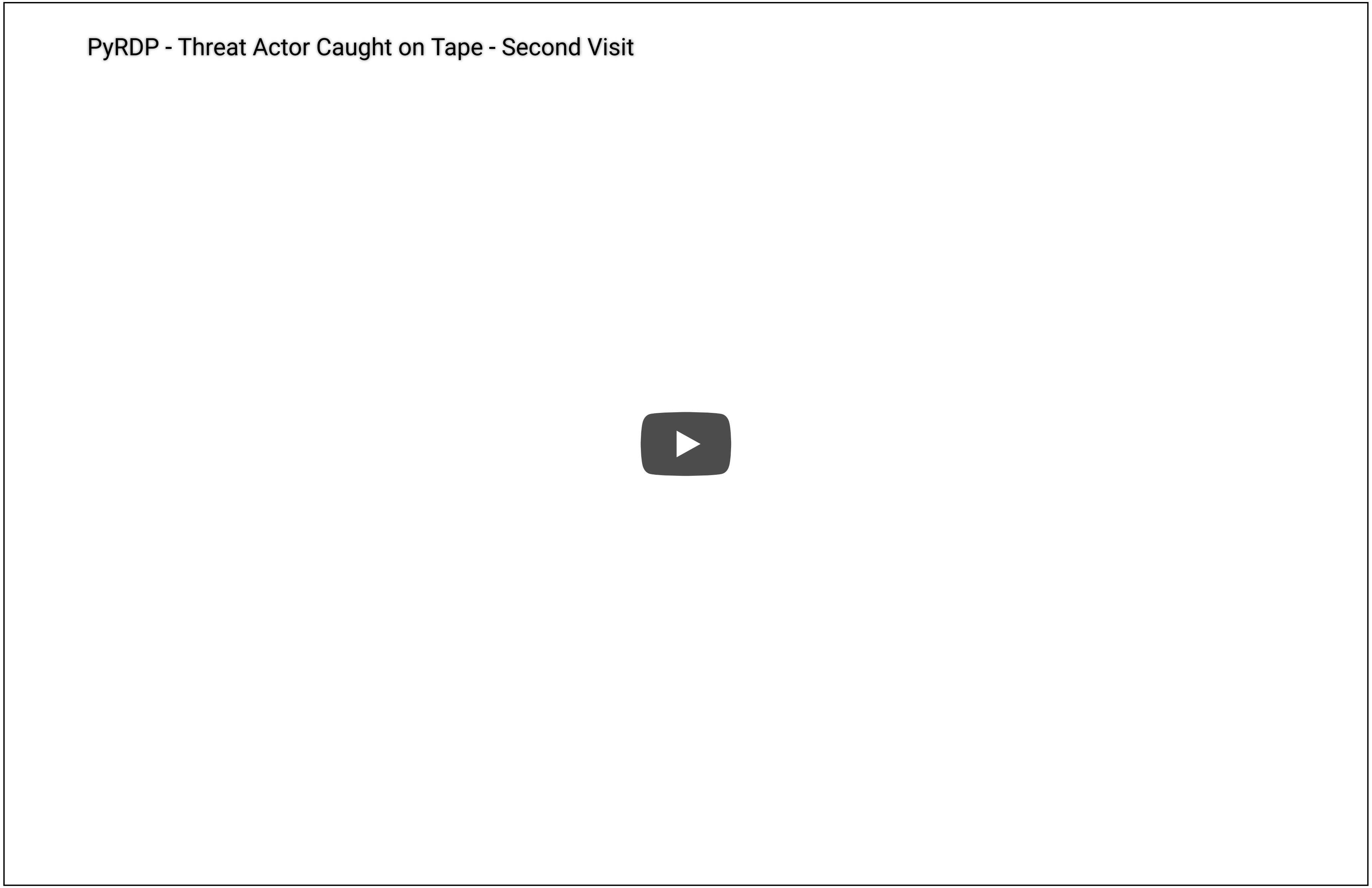
Latest Features

On Autopilot!

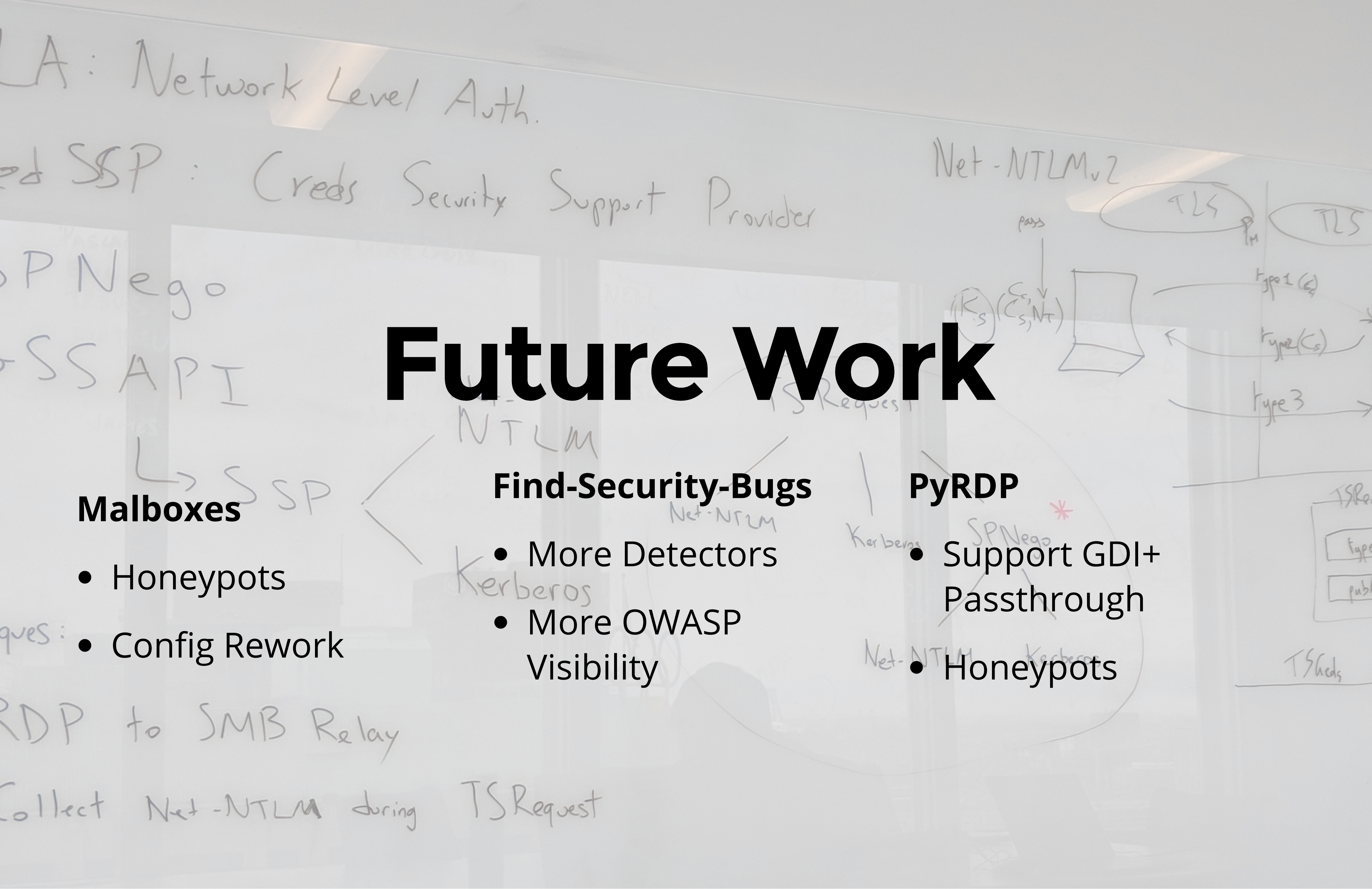
- NLA Downgrade Attack
- Heuristic-based Credential Harvester
- Integration with Bettercap
- Active File Crawler / Downloader

0:00 / 0:45

0:00 / 0:47



PyRDP - Threat Actor Caught on Tape - Second Visit



Future Work

Malboxes

- Honeypots
- Config Rework

Find-Security-Bugs

- More Detectors
- More OWASP Visibility

PyRDP

- Support GDI+ Passthrough
- Honeypots

Pointers

- Learn More About GoSecure Research
- malboxes Malboxes Project | Deploy to AWS Blog | Chat room
- FindSecBugs Find Security Bugs Project | AppSec Global Presentation
- PyRDP Project | PyRDP on Autopilot Blog

Thanks to All External Contributors!

Hugo Genesse, Gregory Leblanc, @snakems, @pix, Camille Moncelier, @xambroz, @malwarenights, Mathieu Tarral, Maxime Carbonneau, Etienne Lacroix, Emilio Gonzalez, Francis Labelle, Humoud, Ondrej Gersl, @tothi, François Labrèche, Sanket Shah

Questions?

- Our Blog | Our GitHub
- Corporate 🐦 @GoSecure_Inc
- Personal 🐦 @obilodeau
- Email: obilodeau@gosecure.net
- Send me malware samples!