Information disclosure vulnerabilities

LAB 56 Information disclosure in error messages

In this lab, any manipulation with productId parameter, for example, entering invalid non-numeric value, will lead to an exemption:

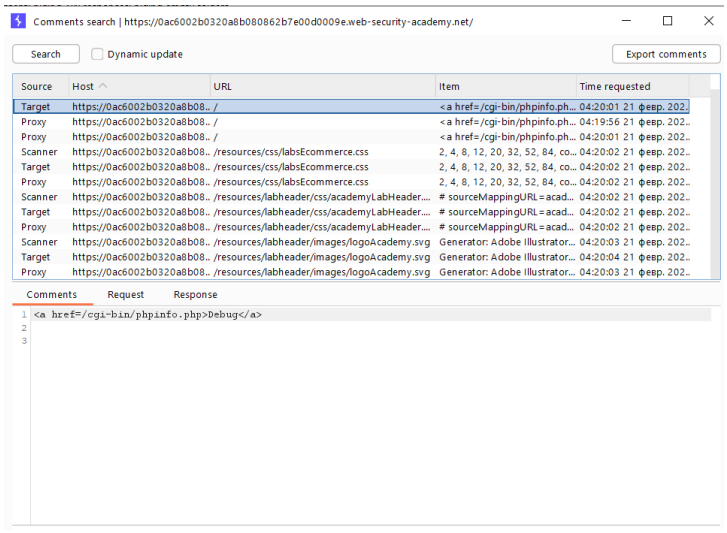
```
Internal Server Error: java.lang.NumberFormatException: For input string: "../"
        at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:67)
        at java.base/java.lang.Integer.parseInt(Integer.java:647)
        at java.base/java.lang.Integer.parseInt(Integer.java:777)
        at lab.j.k.d.s.I(Unknown Source)
        at lab.y.w7.q.i.u(Unknown Source)
        at lab.y.w7.r.p.y.P(Unknown Source)
        at lab.y.w7.r.a.lambda$handleSubRequest$0(Unknown Source)
        at s.x.f.z.lambda$null$3(Unknown Source)
        at s.x.f.z.S(Unknown Source)
        at s.x.f.z.lambda$uncheckedFunction$4(Unknown Source)
        at java.base/java.util.Optional.map(Optional.java:260)
        at lab.y.w7.r.a.v(Unknown Source)
        at lab.server.q.o.dn.M(Unknown Source)
        at lab.y.w7.d.h(Unknown Source)
        at lab.y.w7.d.M(Unknown Source)
        at lab.server.q.o.b.y.W(Unknown Source)
        at lab.server.q.o.b.c.lambda$handle$0(Unknown Source)
        at lab.j.l.d.s.H(Unknown Source)
        at lab.server.q.o.b.c.d(Unknown Source)
        at lab.server.q.o.r.P(Unknown Source)
        at s.x.f.z.lambda$null$3(Unknown Source)
        at s.x.f.z.S(Unknown Source)
        at s.x.f.z.lambda$uncheckedFunction$4(Unknown Source)
        at lab.server.r.l(Unknown Source)
        at lab.server.q.o.r.k(Unknown Source)
        at lab.server.q.b.i.D(Unknown Source)
        at lab.server.q.t.J(Unknown Source)
        at lab.server.q.m.J(Unknown Source)
        at lab.server.z6.l(Unknown Source)
        at lab.server.z6.l(Unknown Source)
        at lab.l.r.lambda$consume$0(Unknown Source)
        at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1144)
        at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:642)
        at java.base/java.lang.Thread.run(Thread.java:1583)

Apache Struts 2 2.3.31
```

The service runs on Apache Struts v.2 (2.3.31)

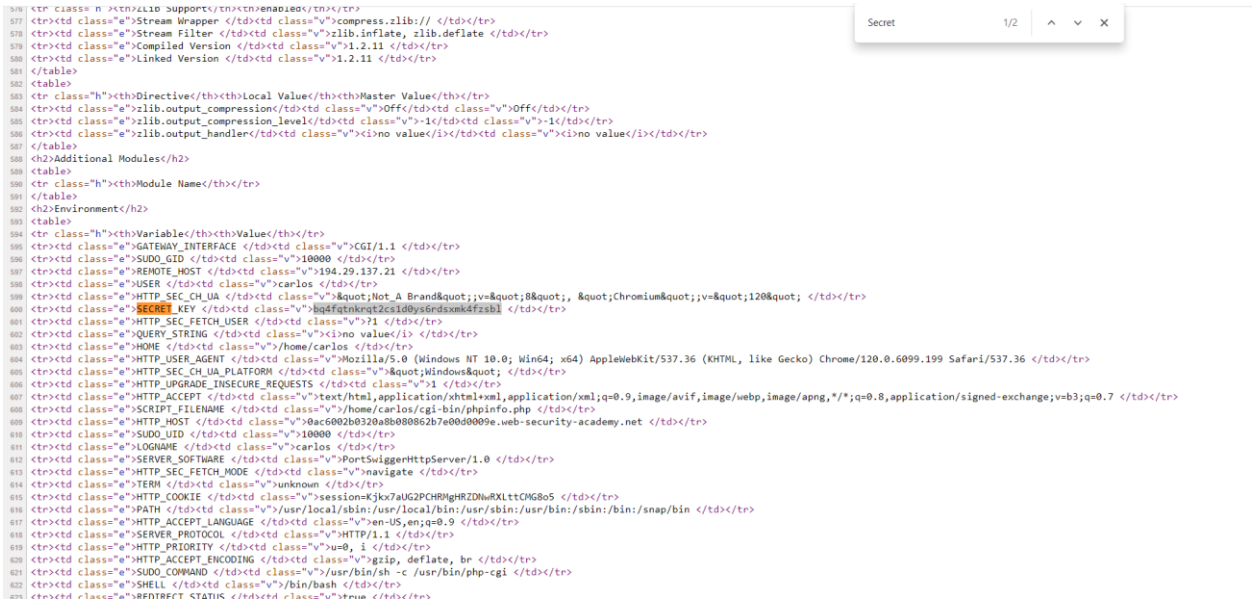
Congratulations, you solved the lab!

LAB 57 Information disclosure on debug page

Having used “Find comments” in engagement tools, I have discovered a comment “Debug” with link to /cgi-bin/phpinfo.php



Here is the content of phpinfo.php:



This file exposes a lot of information about environment variables that are used for debugging, including secret key:

```
bq4fqtnkrqt2cs1d0ys6rdsxmk4fzsbl
```

Congratulations, you solved the lab!

LAB 58 Source code disclosure via backup files

Contents of robots.txt:

```
User-agent: *
Disallow: /backup
```

/backup keeps the backup file of the service, so the source code is free to read:

```
package data.productcatalog;

import common.db.JdbcConnectionBuilder;

import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.Serializable;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class ProductTemplate implements Serializable
{
    static final long serialVersionUID = 1L;

    private final String id;
    private transient Product product;

    public ProductTemplate(String id)
    {
        this.id = id;
    }

    private void readObject(ObjectInputStream inputStream) throws IOException, ClassNotFoundException
    {
        inputStream.defaultReadObject();

        ConnectionBuilder connectionBuilder = ConnectionBuilder.from(
                "org.postgresql.Driver",
                "postgresql",
                "localhost",
                5432,
                "postgres",
                "postgres",
                "9rvl0ndq157jt77ukama166srzhgl6s3"
        ).withAutoCommit();
        try
        {
            Connection connect = connectionBuilder.connect(30);
            String sql = String.format("SELECT * FROM products WHERE id = '%s' LIMIT 1", id);
            Statement statement = connect.createStatement();
            ResultSet resultSet = statement.executeQuery(sql);
            if (!resultSet.next())
            {
                return;
            }
            product = Product.from(resultSet);
```

Hard coded password from the database: 9rvl0ndq157jt77ukama166srzhgl6s3

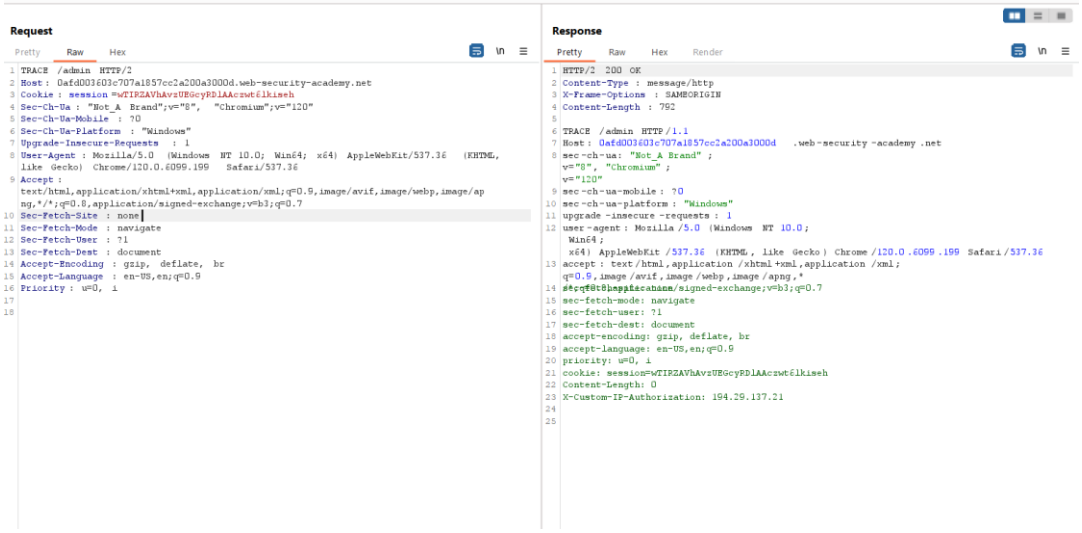
Congratulations, you solved the lab!

LAB 59 Authentication bypass via information disclosure

/admin contents:

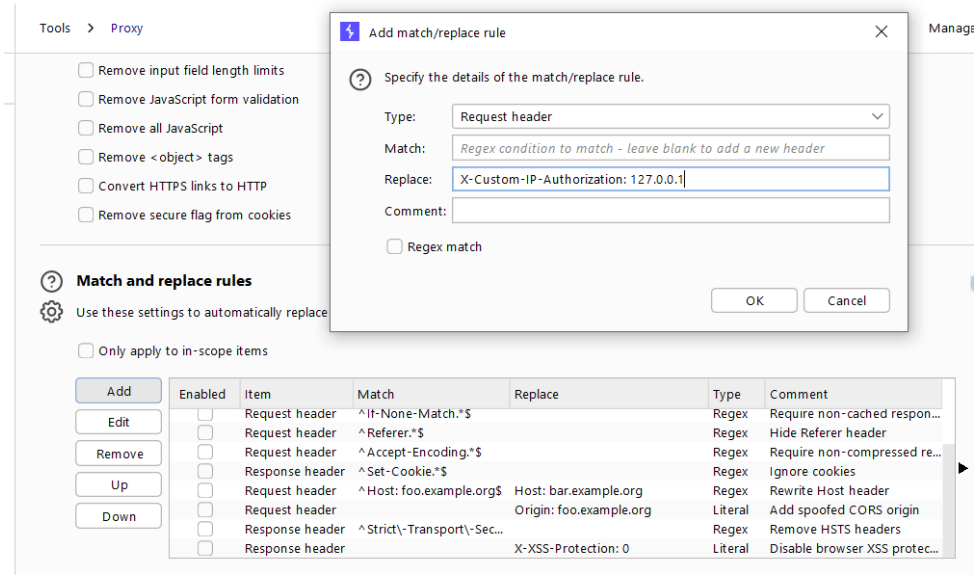
Admin interface only available to local users

I have sent GET /admin request to Burp Repeater but changed HTTP GET with HTTP TRACE method and got a response successfully:

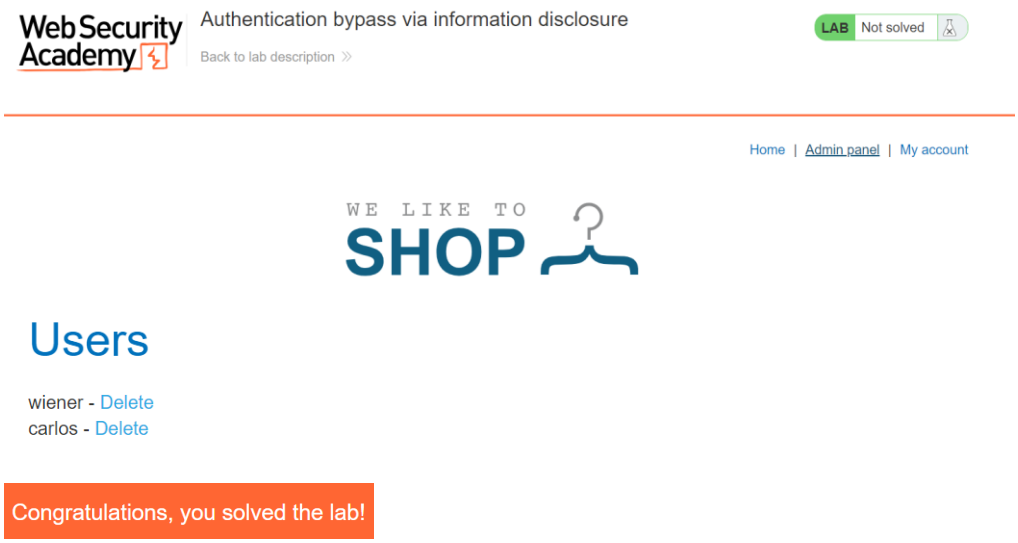


X-Custom-IP-Authorization: 194.29.137.21

This header was appended to my request automatically and it is used to check whether the request came from localhost 127.0.0.1 request. Therefore, I can this header in match/replace rules to localhost IP and trick the host:

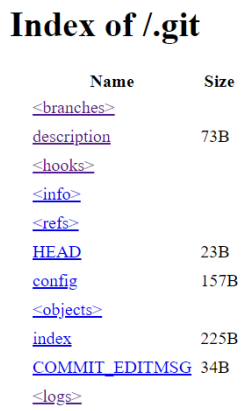


Once done, I can see that Admin Panel is now available and list of users to delete is showed:



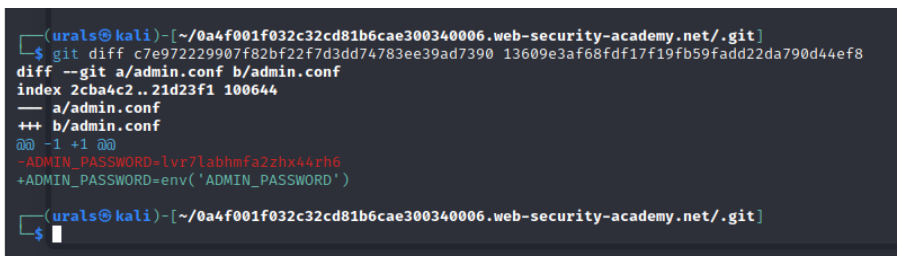
LAB 60 Information disclosure in version control history

/.git contents:



I downloaded the repository with wget and started to inspect it from my Linux environment:

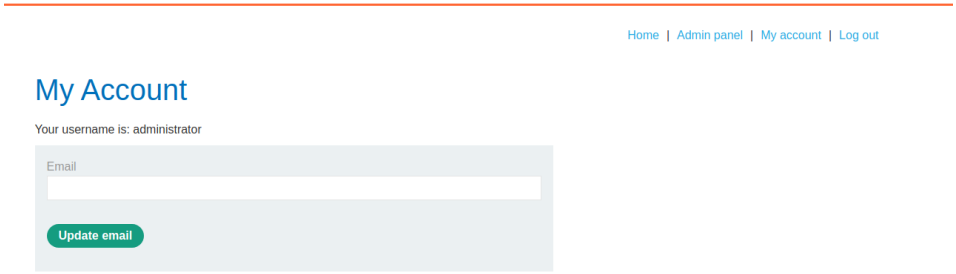
Above is contents of git logs: there is 2 commits. Their diff:



As the message in log file said, the admin password was deleted from configuration file, and commits were done at admin.conf and I can find the password that was deleted (red log):

lvr7labhmfa2zhx44rh6

Let's try this:



Great! The log in was successful and Admin panel is now available: