A8 - Insecure Deserialization

globo.com

GTS 33 | PA 05/2019

Agenda - A8 - Desserialização Insegura

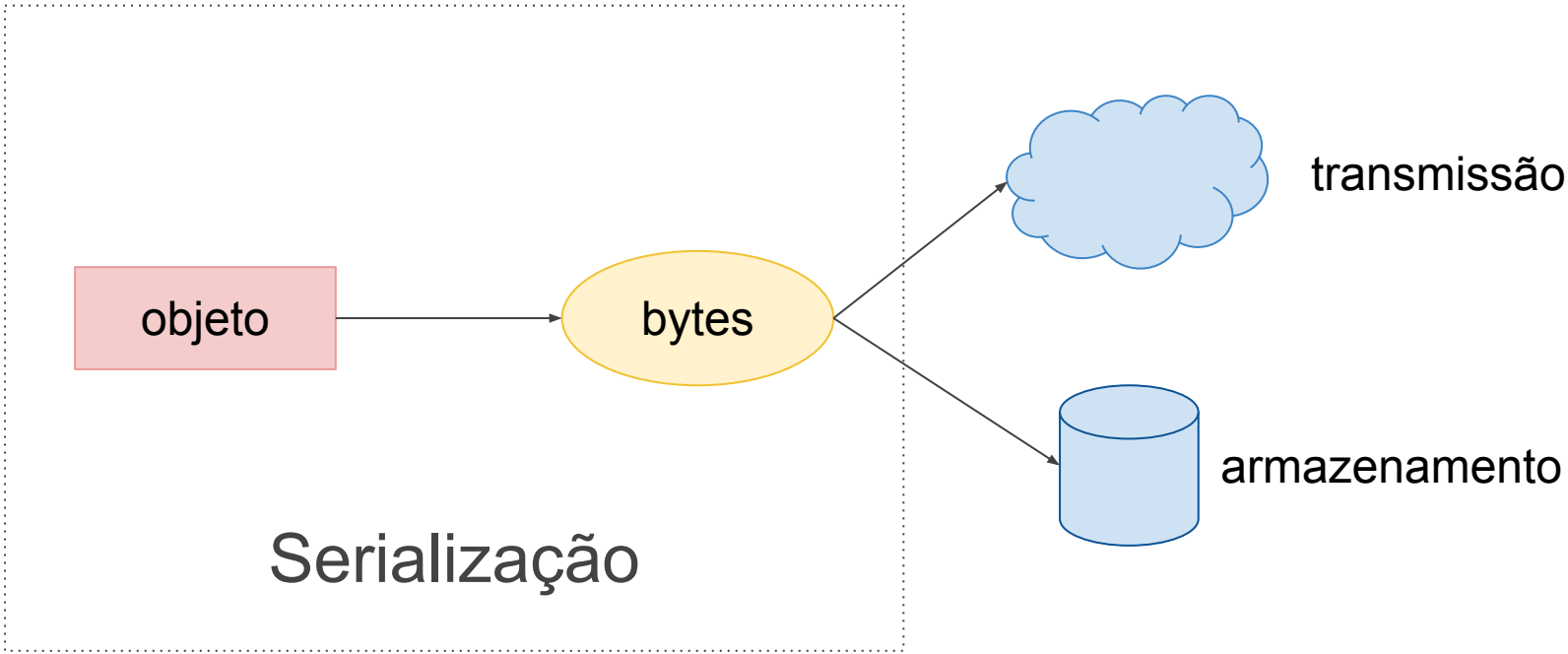
1. Do que se trata?
2. Exemplos
3. Como se proteger?
4. Amarelo Designs

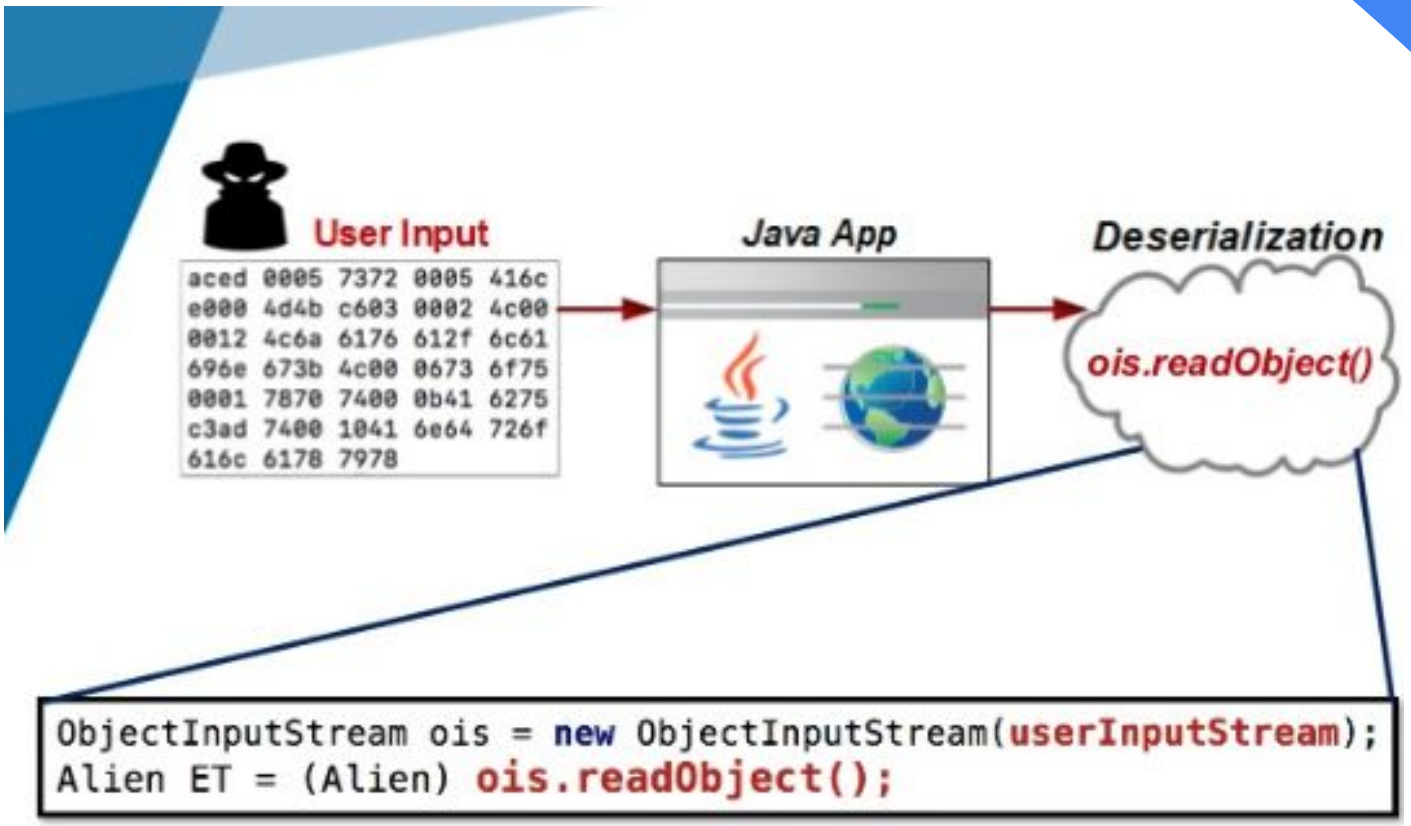
Do que se trata?

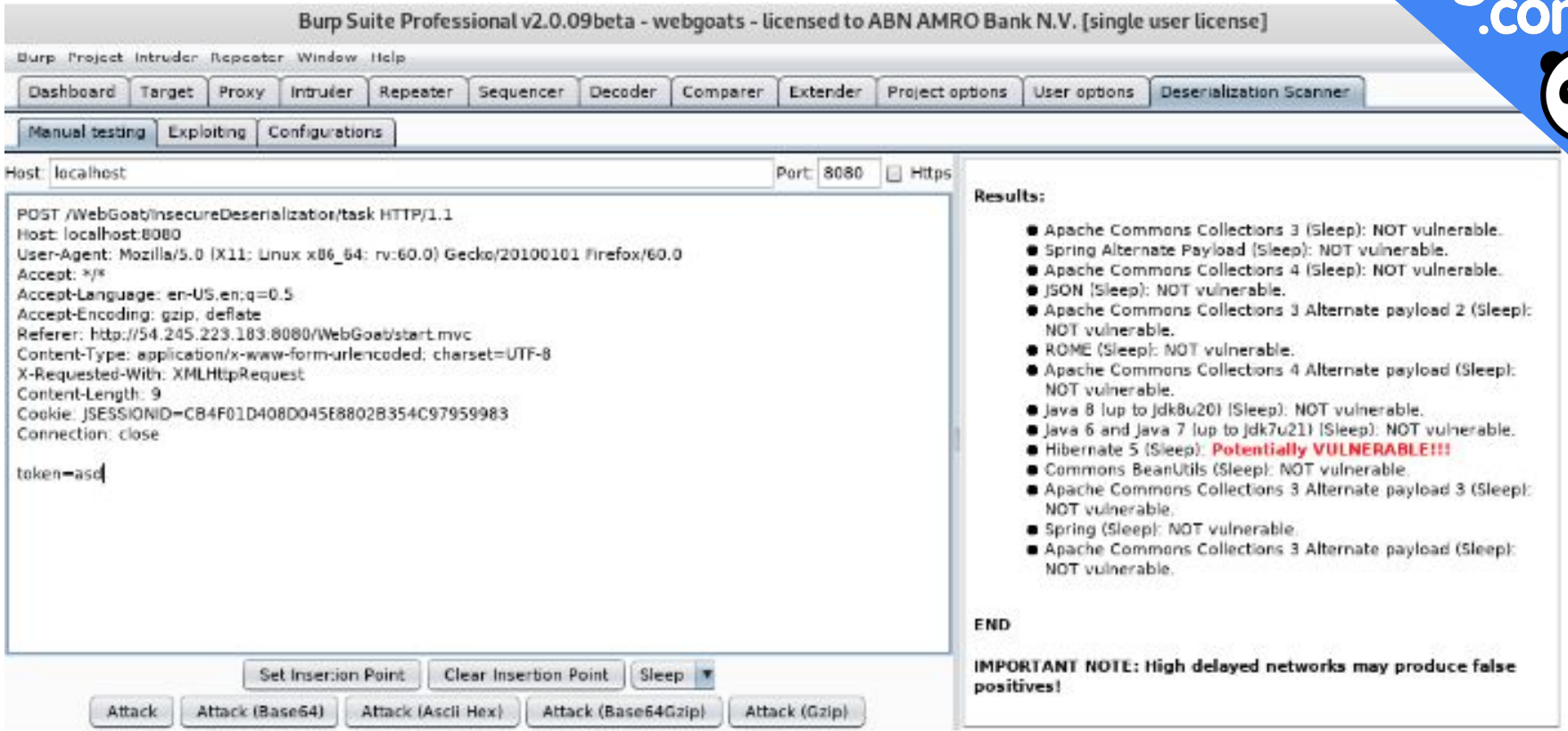
globo.com

"Desserializar" significa recuperar dados ou um estado de um objeto a partir de um conjunto de *bytes*, garantindo que eles representem as mesmas informações. O problema ocorre quando esse processo atua sobre entradas não confiáveis do usuário.

Do que se trata?




User Input
aced 0005 7372 0005 416c
e000 4d4b c603 0002 4c00
0012 4c6a 6176 612f 6c61
696e 673b 4c00 0673 6f75
0001 7870 7400 0b41 6275
c3ad 7400 1041 6e64 726f
616c 6178 7978
Java App
Deserialization
ois.readObject()
ObjectInputStream ois = new ObjectInputStream(userInputStream);
Alien ET = (Alien) ois.readObject();


Burp Suite Professional v2.0.09beta - webgoats - licensed to ABN AMRO Bank N.V. [single user license]
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Deserialization Scanner
Manual testing Exploiting Configurations
Host: localhost
Port: 8080
Https
POST /WebGoat/InsecureDeserialization/task HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://54.245.223.183:8080/WebGoat/start.mvc
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 9
Cookie: JSESSIONID=CB4F01D408D045E88028354C97959983
Connection: close
token=asd
Set Insertion Point
Clear Insertion Point
Sleep
Attack
Attack (Base64)
Attack (Ascii Hex)
Attack (Base64Gzip)
Attack (Gzip)
Results:
Apache Commons Collections 3 (Sleep): NOT vulnerable.
Spring Alternate Payload (Sleep): NOT vulnerable.
Apache Commons Collections 4 (Sleep): NOT vulnerable.
JSON (Sleep): NOT vulnerable.
Apache Commons Collections 3 Alternate payload 2 (Sleep): NOT vulnerable.
ROME (Sleep): NOT vulnerable.
Apache Commons Collections 4 Alternate payload (Sleep): NOT vulnerable.
Java 8 (up to Jdk8u20) (Sleep): NOT vulnerable.
Java 6 and Java 7 (up to Jdk7u21) (Sleep): NOT vulnerable.
Hibernate 5 (Sleep): Potentially VULNERABLE!!!
Commons BeanUtils (Sleep): NOT vulnerable.
Apache Commons Collections 3 Alternate payload 3 (Sleep): NOT vulnerable.
Spring (Sleep): NOT vulnerable.
Apache Commons Collections 3 Alternate payload (Sleep): NOT vulnerable.
END
IMPORTANT NOTE: High delayed networks may produce false positives!

GitHub, Inc. [US] | https://github.com/frohoff/ysoserial

ysoserial

chat on gitter | download master | build passing | build passing

A proof-of-concept tool for generating payloads that exploit unsafe Java object deserialization.

Description

Originally released as part of AppSecCali 2015 Talk "Marshalling Pickles: how deserializing objects will ruin your day" with gadget chains for Apache Commons Collections (3.x and 4.x), Spring Beans/Core (4.x), and Groovy (2.3.x). Later updated to include additional gadget chains for JRE <= 1.7u21 and several other libraries.

ysoserial is a collection of utilities and property-oriented programming "gadget chains" discovered in common java libraries that can, under the right conditions, exploit Java applications performing **unsafe deserialization** of objects. The main driver program takes a user-specified command and wraps it in the user-specified gadget chain, then serializes these objects to stdout. When an application with the required gadgets on the classpath unsafely deserializes this data, the chain will automatically be invoked and cause the command to be executed on the application host.

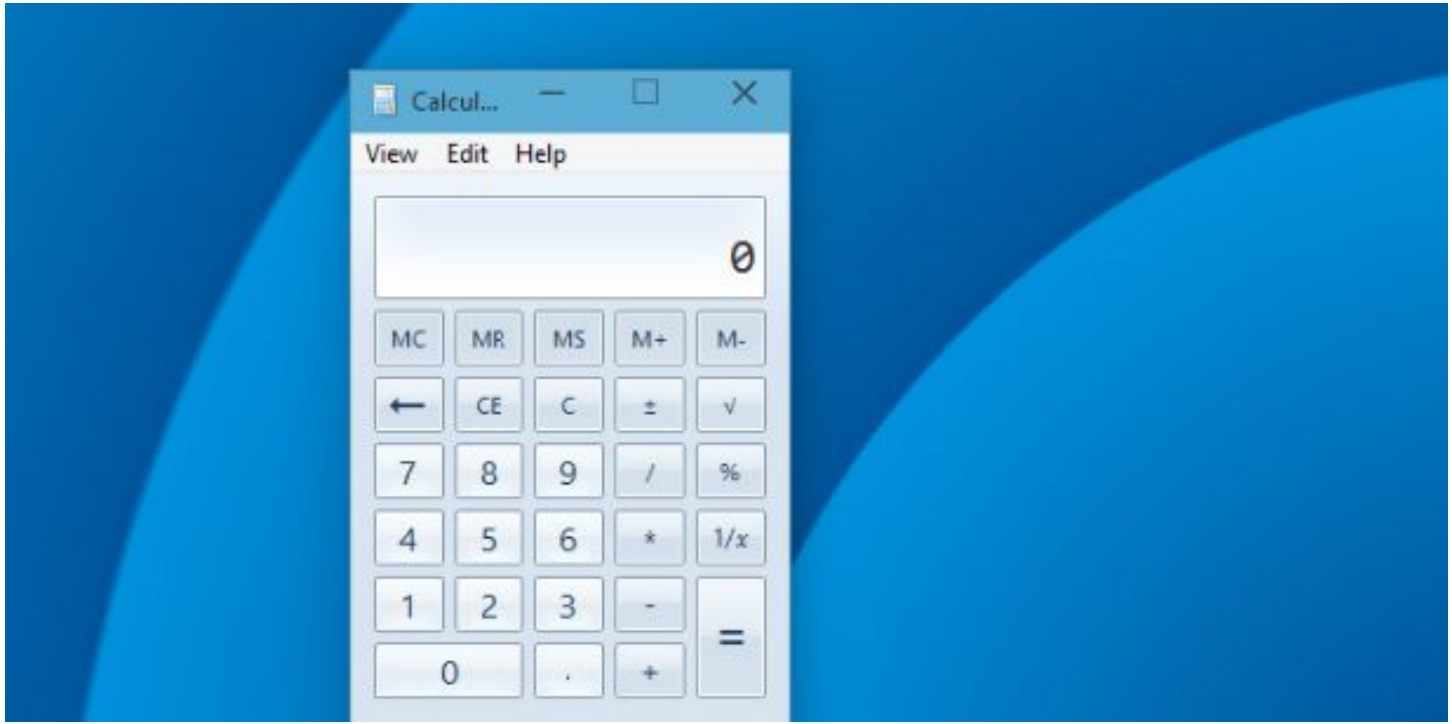
It should be noted that the vulnerability lies in the application performing unsafe deserialization and NOT in having gadgets on the classpath.

Disclaimer

This software has been created purely for the purposes of academic research and for the development of effective

globo.com

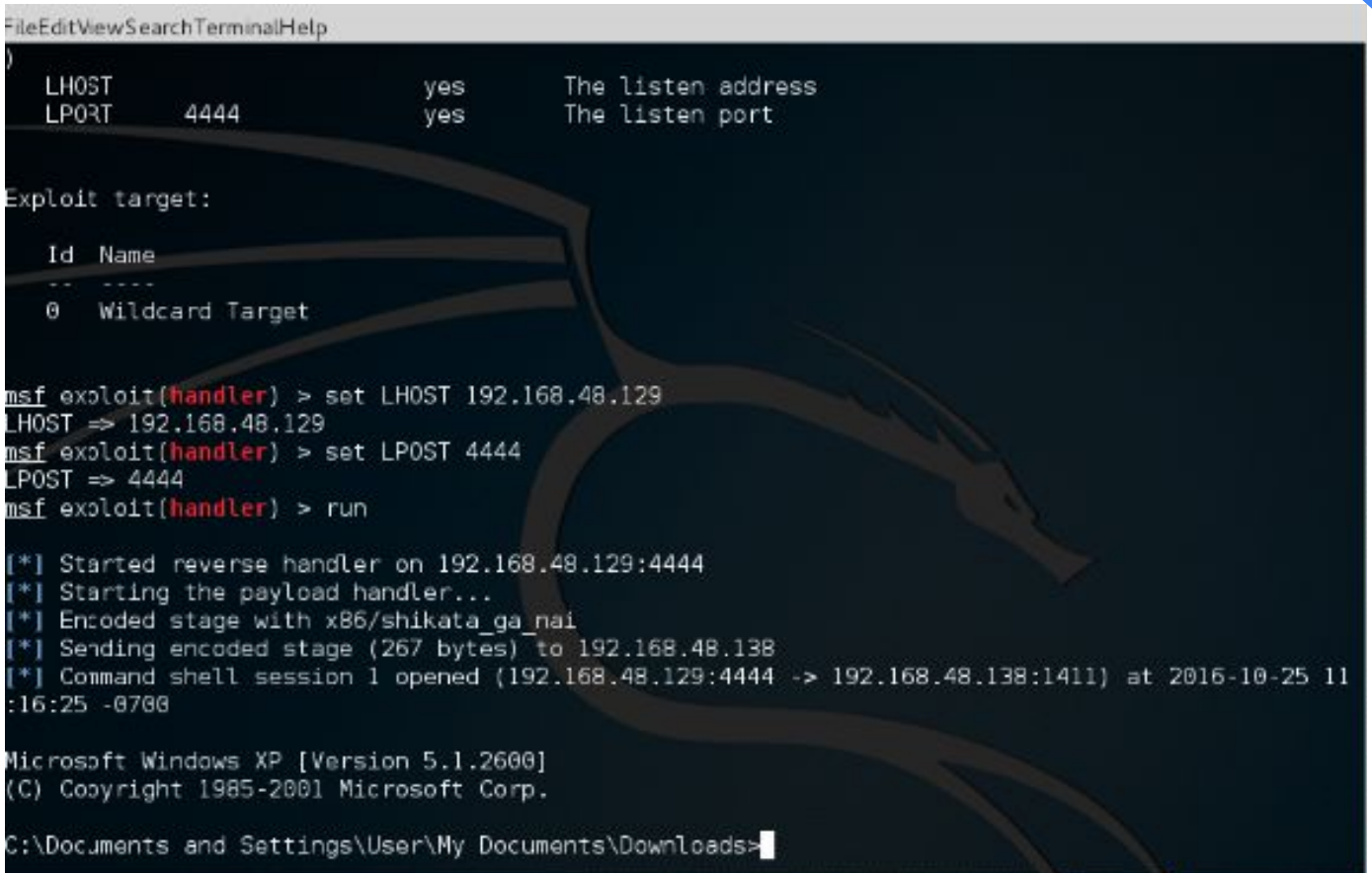
Java ysoserial



globo.com

Java

GTS 33 | PA 05/2019



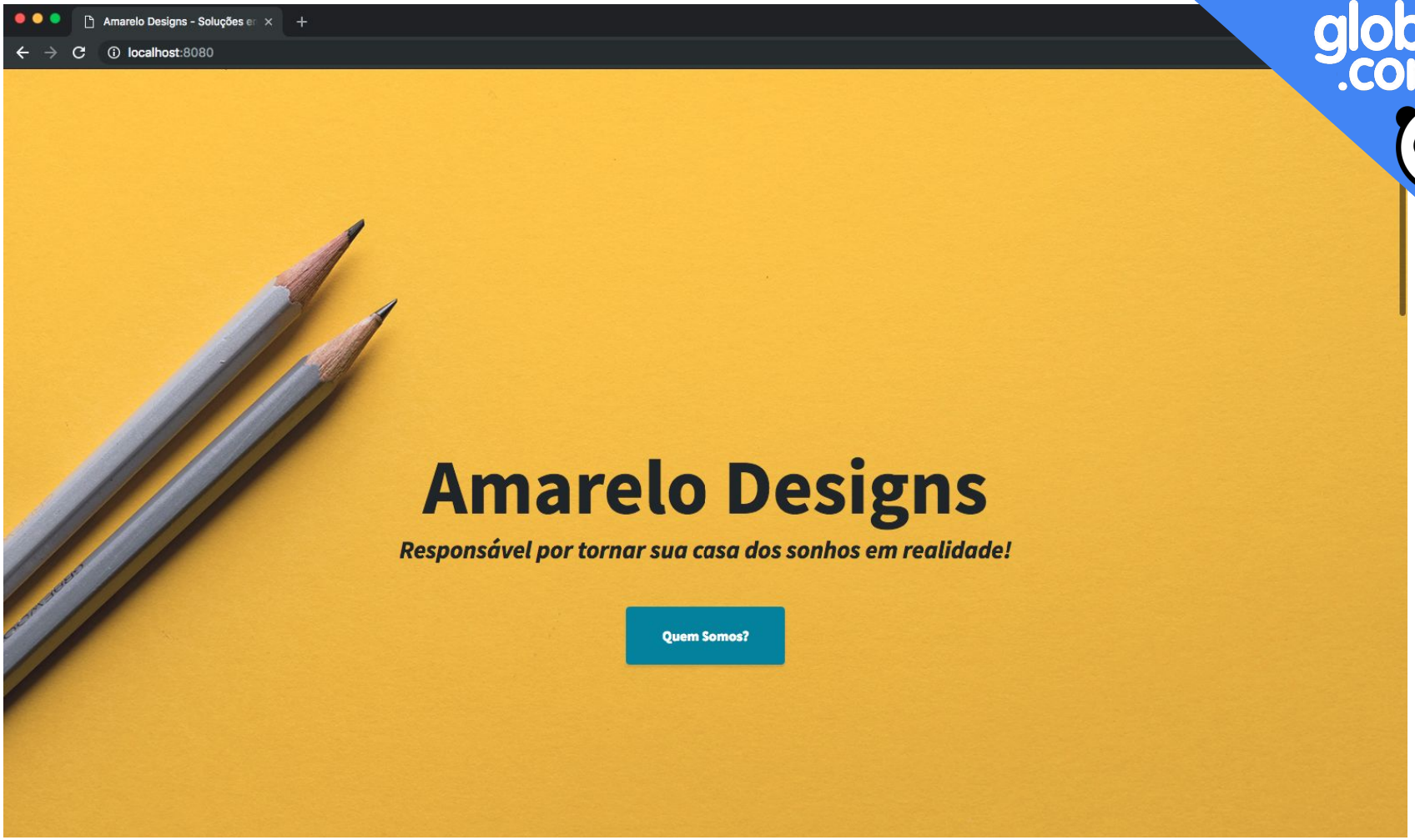
globo.com

Java

GTS 33 | PA 05/2019

Como se prevenir?

1. Não aceitar objetos serializados de **fonte não-confiáveis**. 👎
2. Validar a **integridade** do objeto serializado:
   a. Usar método de assinatura no artefato serializado, como *cookies* assinados. 🍪
3. Utilizar uma serialização mais segura, como **JSON**. 🔓
4. Executar as funções de desserialização em um usuário com o **mínimo de permissões possível**. 👀


Amarelo Designs - Soluções
localhost:8080
Amarelo Designs
Responsável por tornar sua casa dos sonhos em realidade!
Quem Somos?

Narrativa do ataque

Hands on!

1. Entrar na pasta da app

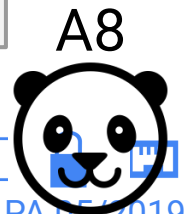
```
$ cd owasp-top10-2017-apps/a8/amarelo-designs
```

2. Inicializar o container

```
$ make install
```

3. Acessar a página

```
localhost:5000
```



globo.com

A8 - Desserialização Insegura