Session IPA

Sessions' Interesting Protection Anomalies

Disclaimer:

We cannot guarantee that no beers were harmed during the making of this presentation

About Luke

Security Research @ elttam

Running CTFs as BitcoinCTF

Playing CTFs with TheGoonies

About Louis

Security Engineer

Founder of PentesterLab.com

Maker of Silvio stickers

PentesterLab

Why a talk on sessions?

From a developer's point of view it all looks the same but implementation matters. The devil is in the details

```php
<?php
  session_start();
  $_SESSION['admin'] = 1;
?>
```

versus

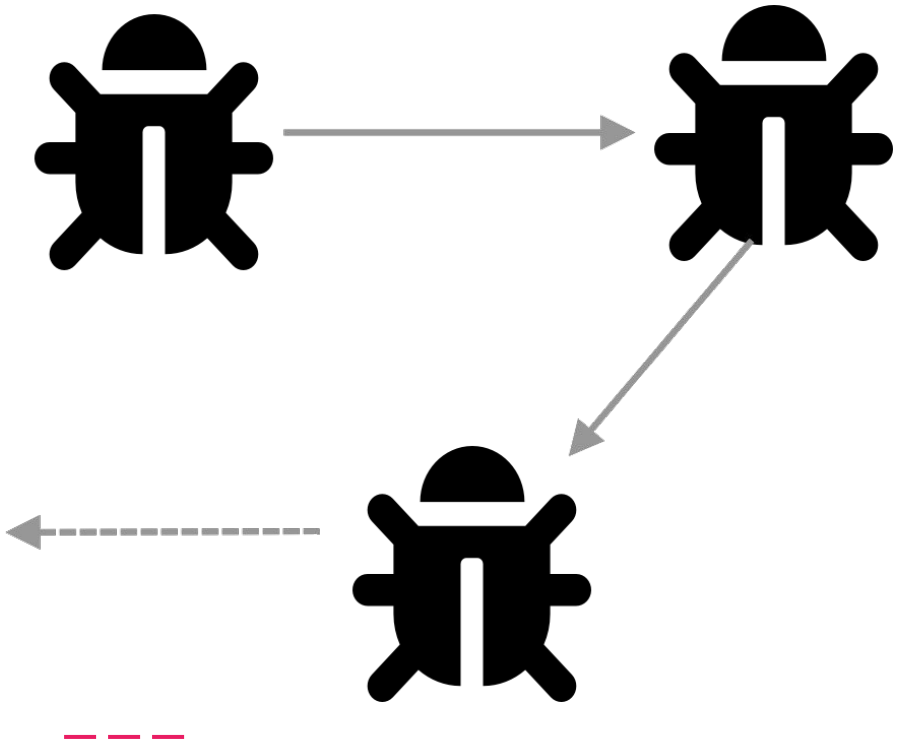
```
session[:admin] = 1
```

Why a talk on sessions?

Developers and security professionals often assume session mechanisms have been heavily tested/reviewed

Why a talk on sessions?

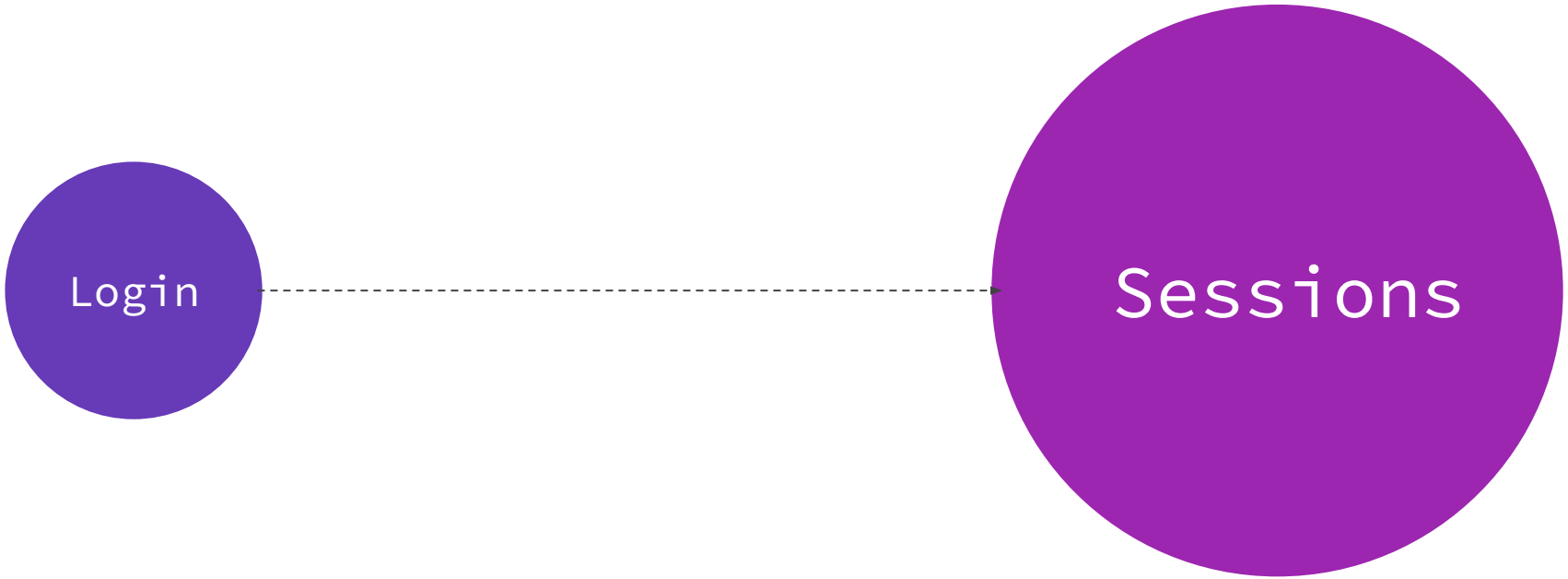
Often a very useful link in a chain of bugs



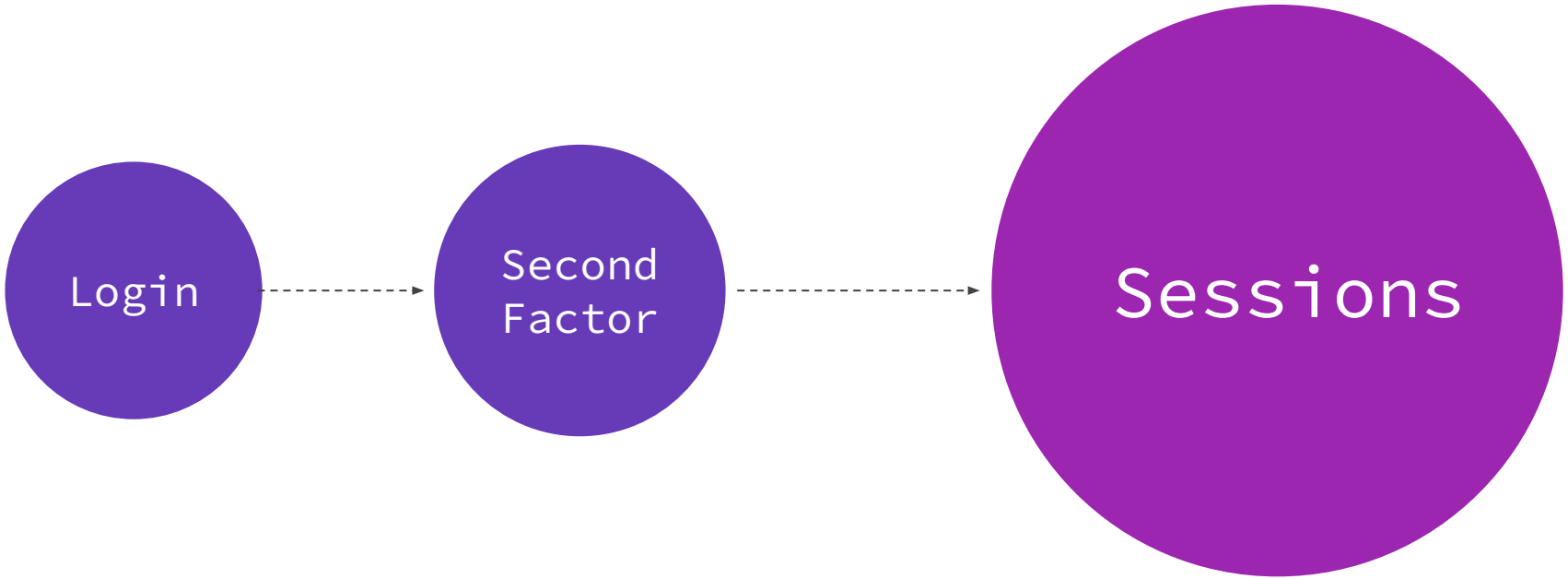
Why a talk on sessions?

It’s not always clear who is in charge of what (application vs framework/library)

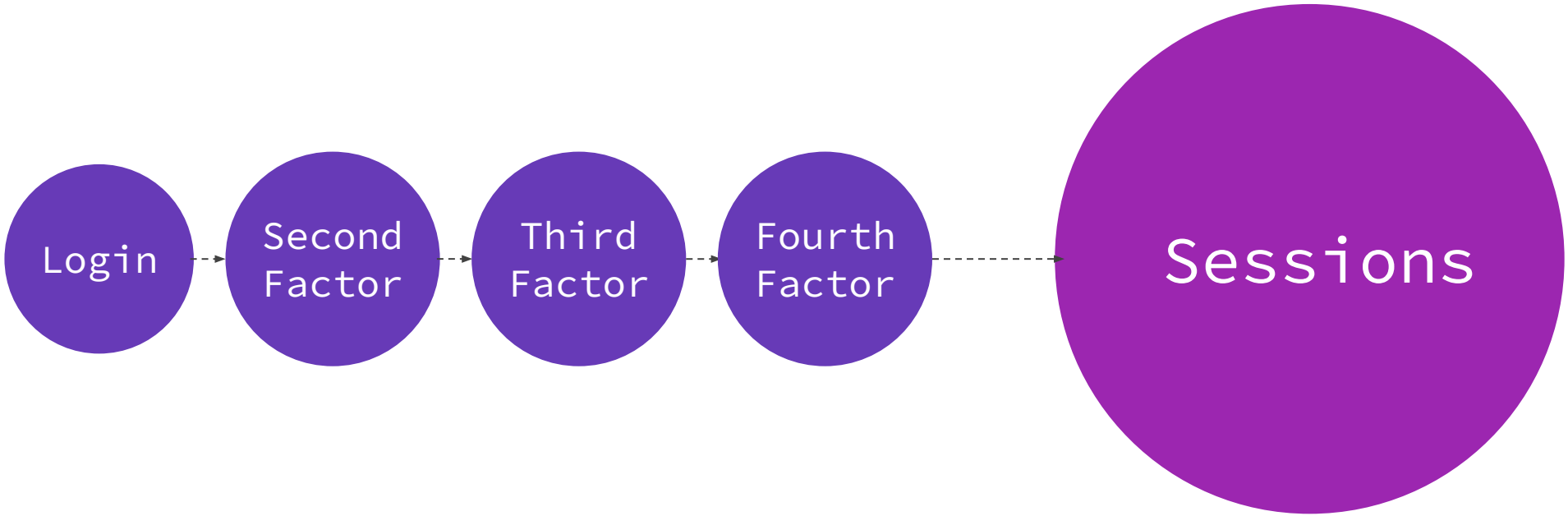
Why a talk on sessions?



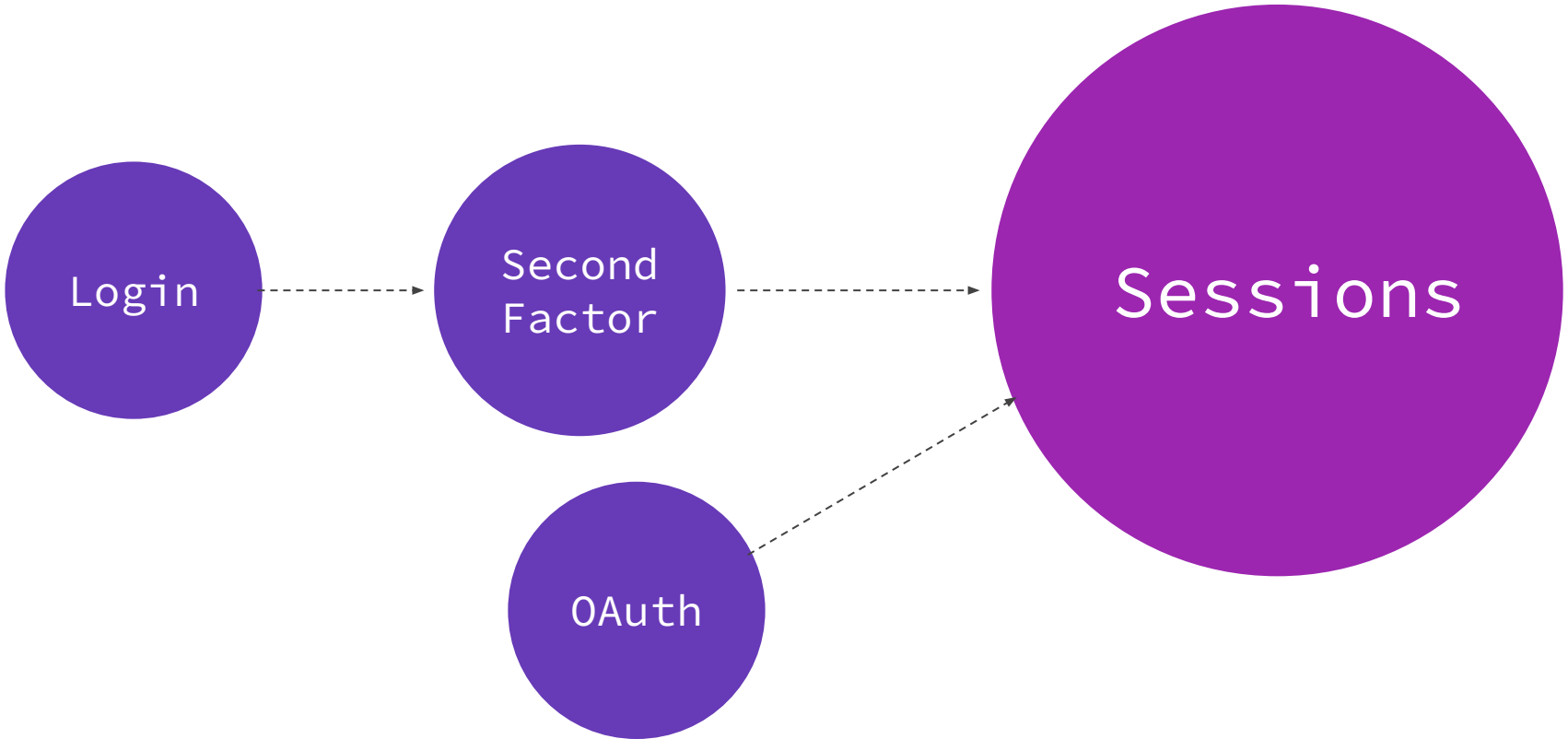
Why a talk on sessions?



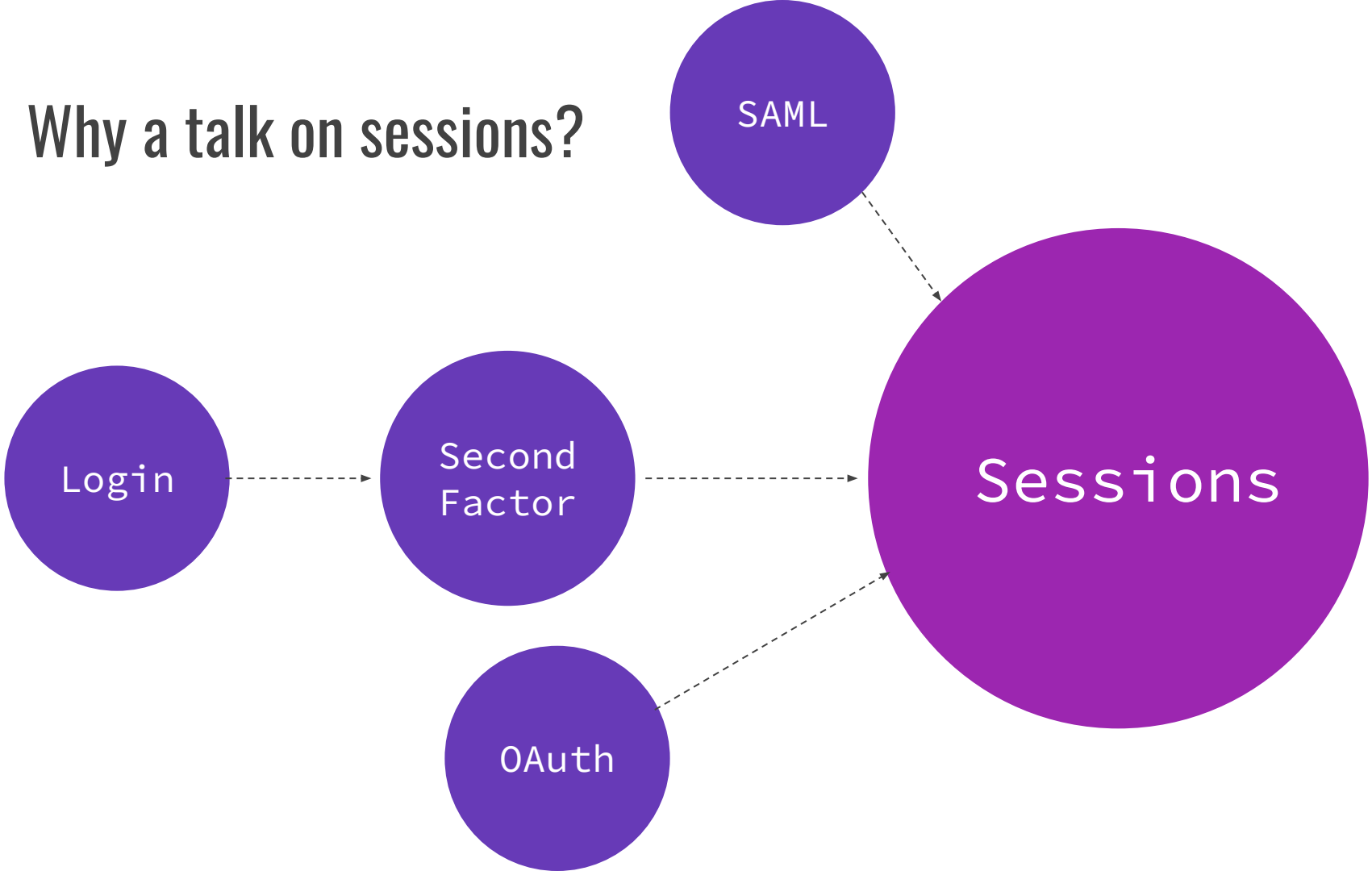
Why a talk on sessions?



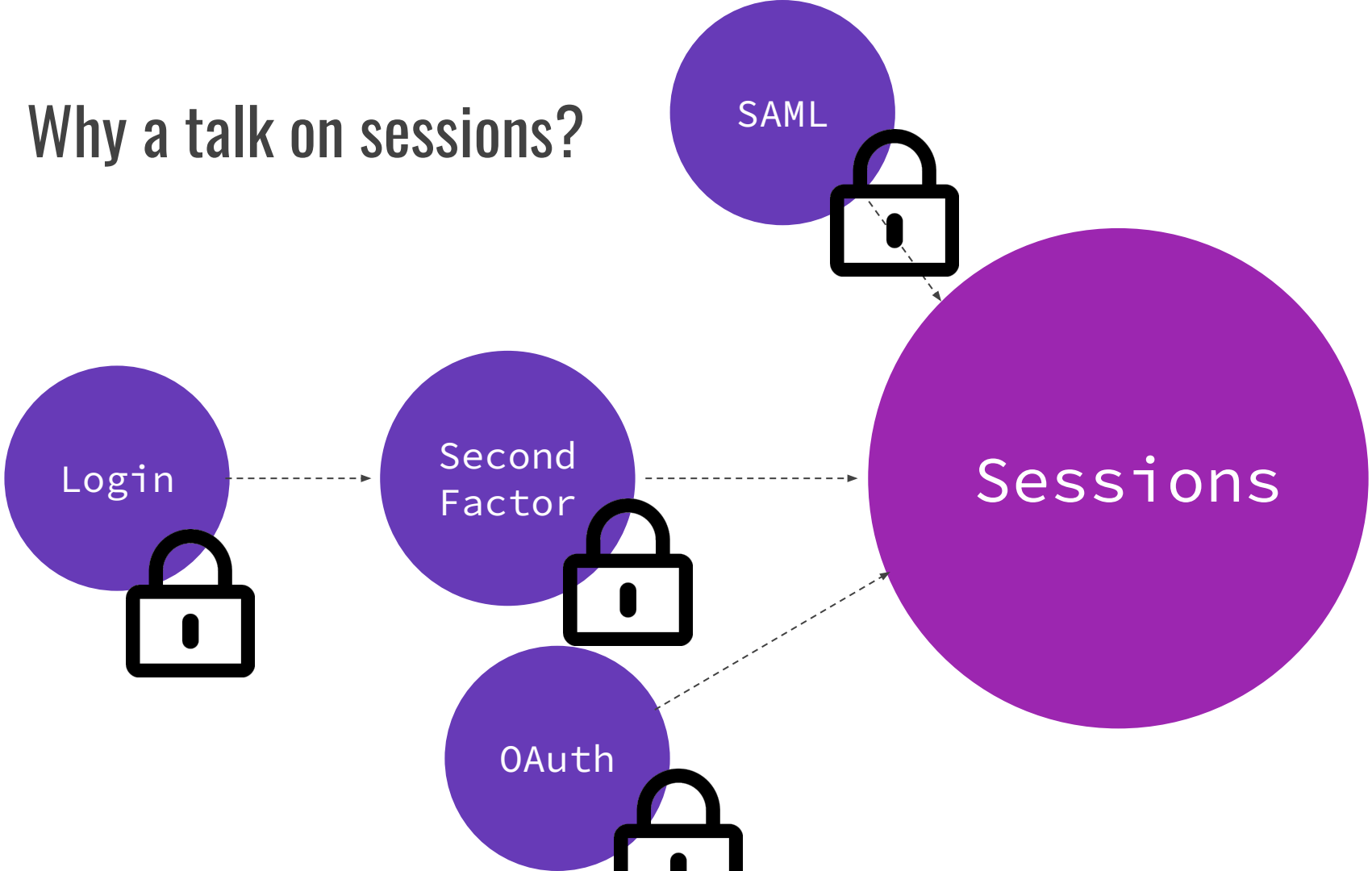
Why a talk on sessions?



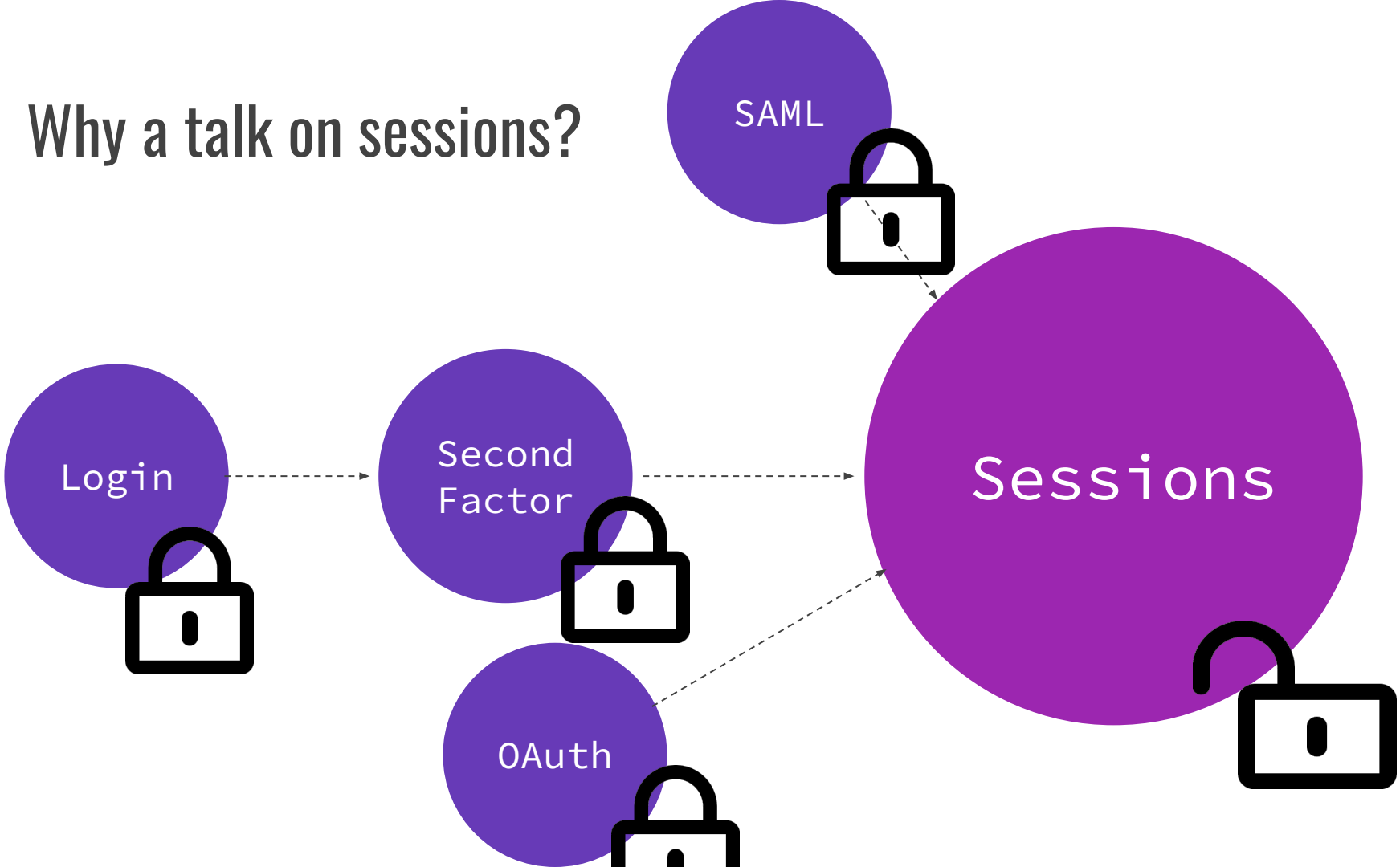
Why a talk on sessions?



Why a talk on sessions?

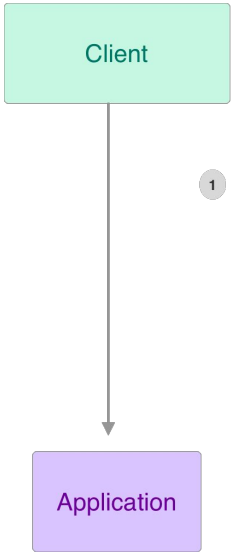


Why a talk on sessions?



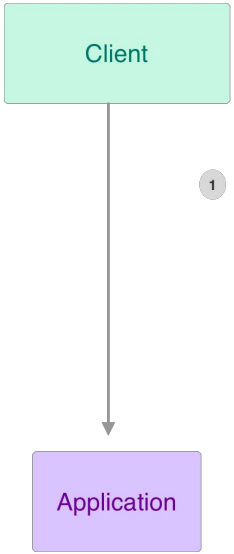
Sessions 101

Stored Sessions 101 - Login




```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

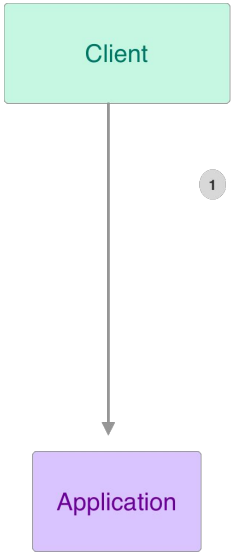
Stored Sessions 101 - Login




```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

2 **Verify the credentials**

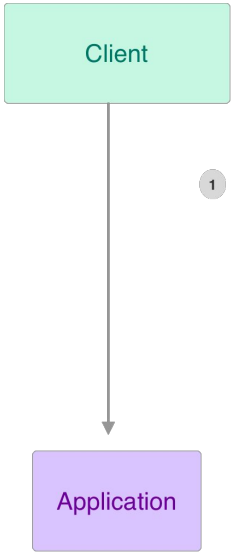
Stored Sessions 101 - Login




```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

2. **Verify the credentials**
3. **Create a file /var/lib/php5/sessions/ sess_edtbb70qto8dm73ip93larn587**

Stored Sessions 101 - Login




```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

2. **Verify the credentials**
3. **Create a file /var/lib/php5/sessions/ sess_edtbb70qto8dm73ip93larn587**
4. **Serialize the data**

Stored Sessions 101 - Login

1

```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

2 **Verify the credentials**

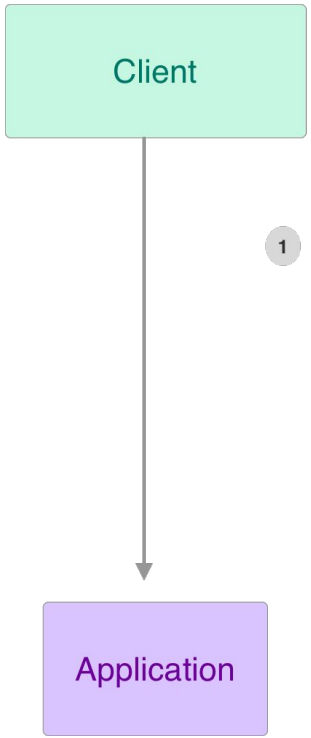
3 **Create a file /var/lib/php5/sessions/ sess_edtbb70qto8dm73ip93larn587**

4 **Serialize the data**

5

```
HTTP/1.1 302 Found
Location: /
Set-Cookie: PHPSESSID=edtbb70qto8…
```

Stored Sessions 101 - Access

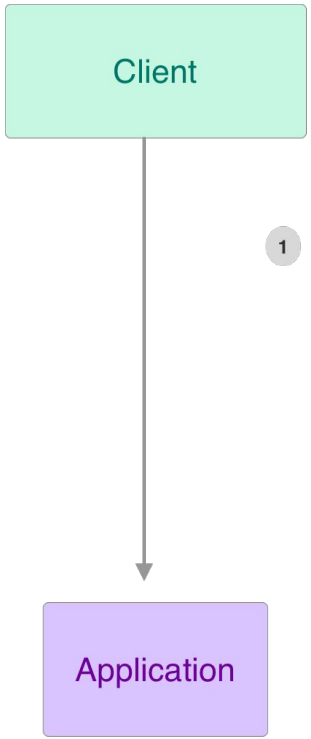


1

```
GET / HTTP/1.1
Host: pentesterlab.com
Cookie: PHPSESSID=edtbb70qto8dm73ip931arn587
```

Stored Sessions 101 - Access

Stored Sessions 101 - Access



1 `GET / HTTP/1.1`
`Host: pentesterlab.com`
`Cookie: PHPSESSID=`**`edtbb70qto8dm73ip93larn587`**

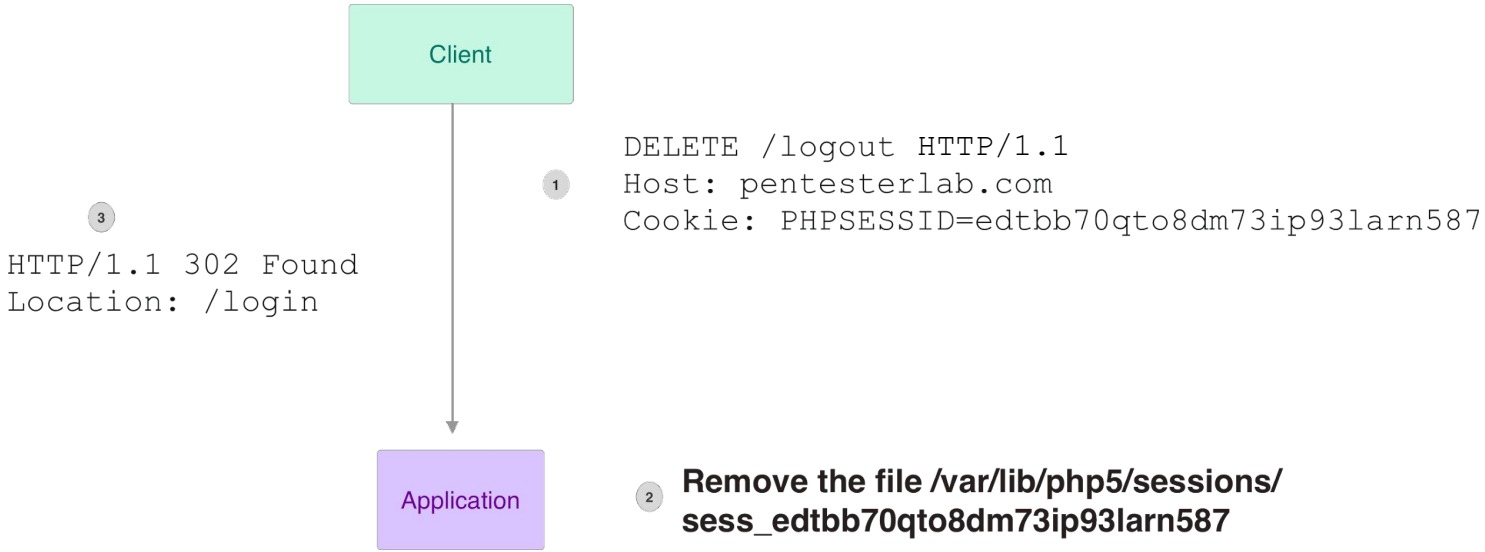
2 **Read the file**
/var/lib/php5/sessions/sess_edtbb70qto8dm73ip93larn587

3 **Unserialize the data**

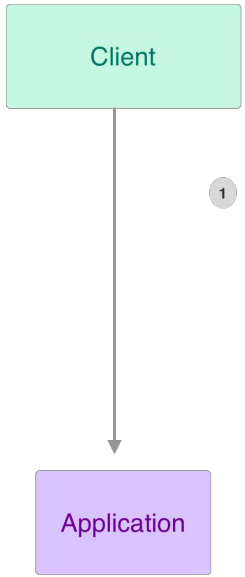
Stored Sessions 101 - Logout

Stored Sessions 101 - Logout

Stored Sessions 101 - Logout

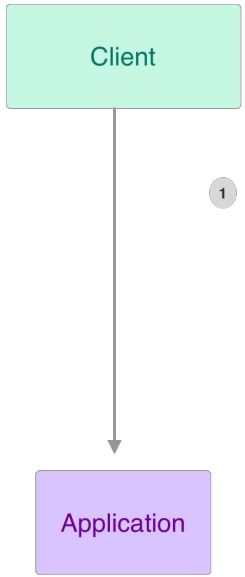


Signed Sessions 101 - Login




```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

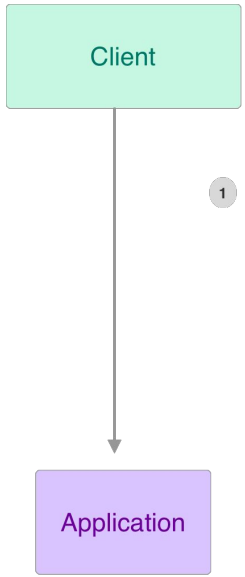
Signed Sessions 101 - Login




```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

2 **Verify the credentials**

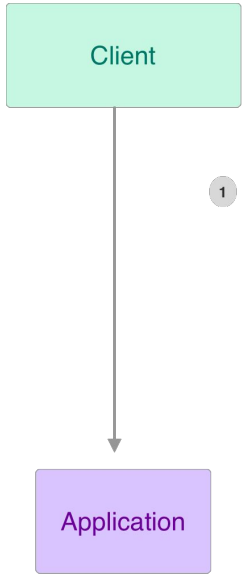
Signed Sessions 101 - Login




```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

2. **Verify the credentials**
3. **Serialize information about the user**

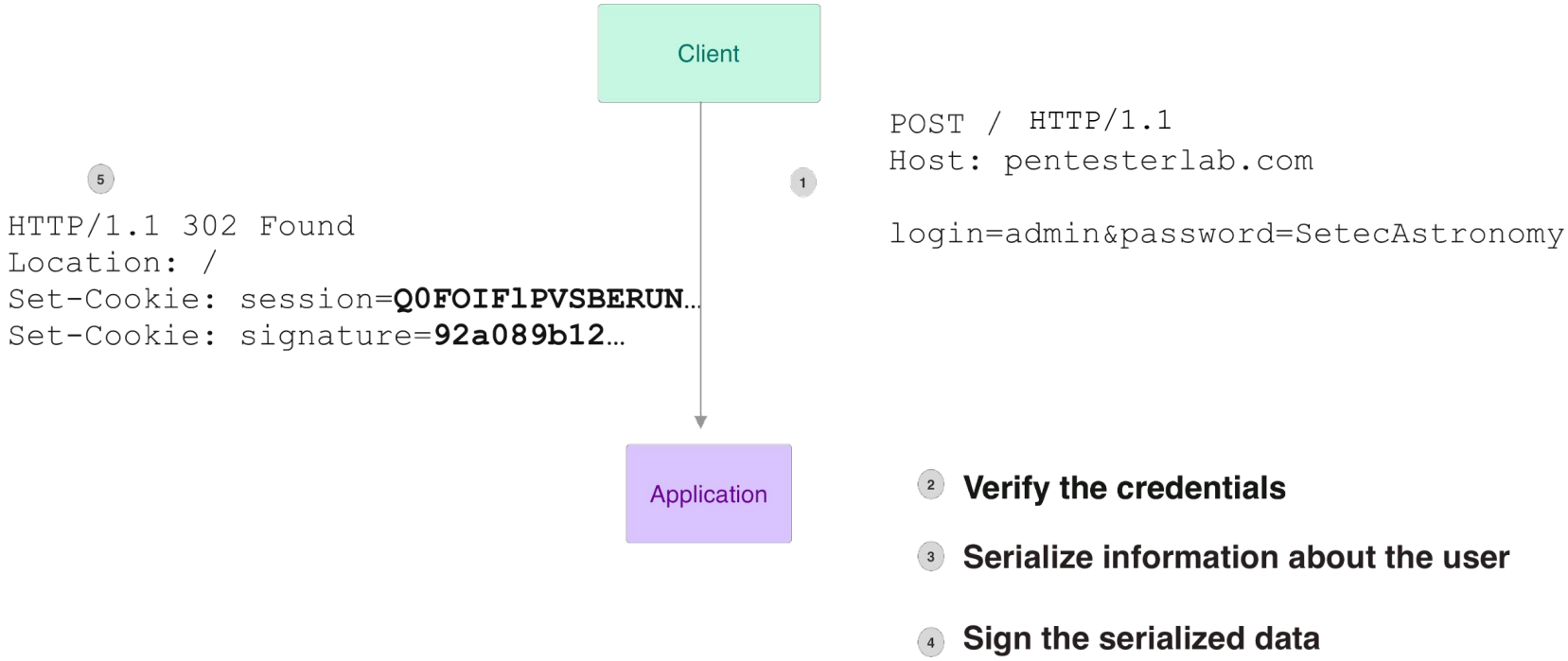
Signed Sessions 101 - Login



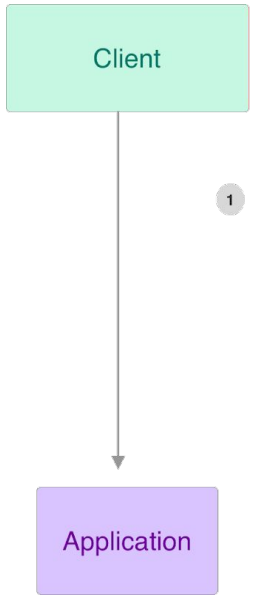

```
POST /  HTTP/1.1
Host: pentesterlab.com

login=admin&password=SetecAstronomy
```

2. **Verify the credentials**
3. **Serialize information about the user**
4. **Sign the serialized data**

Signed Sessions 101 - Login



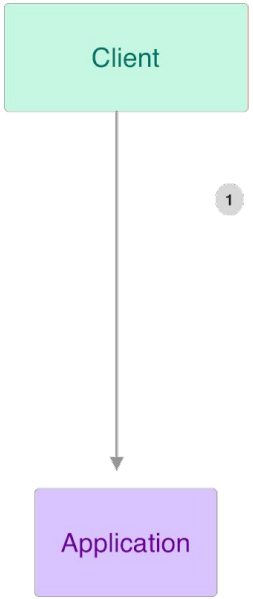
Signed Sessions 101 - Access



1

```
GET /  HTTP/1.1
Host: pentesterlab.com
Cookie: session=Q0FOIFlPVSBERUNSWVBUIFRIQVQK; signature= 92a089b1243628228
```

Signed Sessions 101 - Access

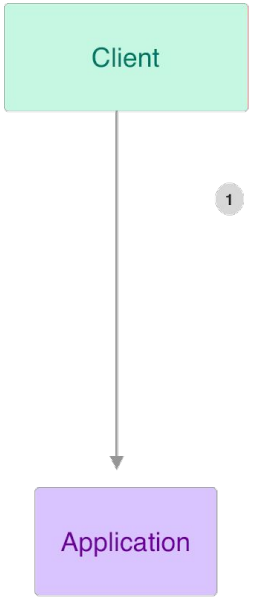



```
GET / HTTP/1.1
Host: pentesterlab.com
Cookie: session=Q0FOIFlPVSBERUNSWVBUIFRIQVQK; signature= 92a089b1243628228
```

1

2 **Verify the signature**

Signed Sessions 101 - Access

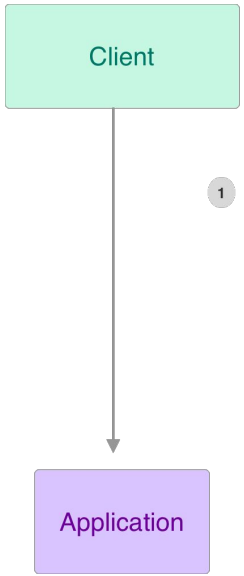


1
```
GET / HTTP/1.1
Host: pentesterlab.com
Cookie: session=Q0FOIFlPVSBERUNSWVBUIFRIQVQK; signature= 92a089b1243628228
```

2 **Verify the signature**

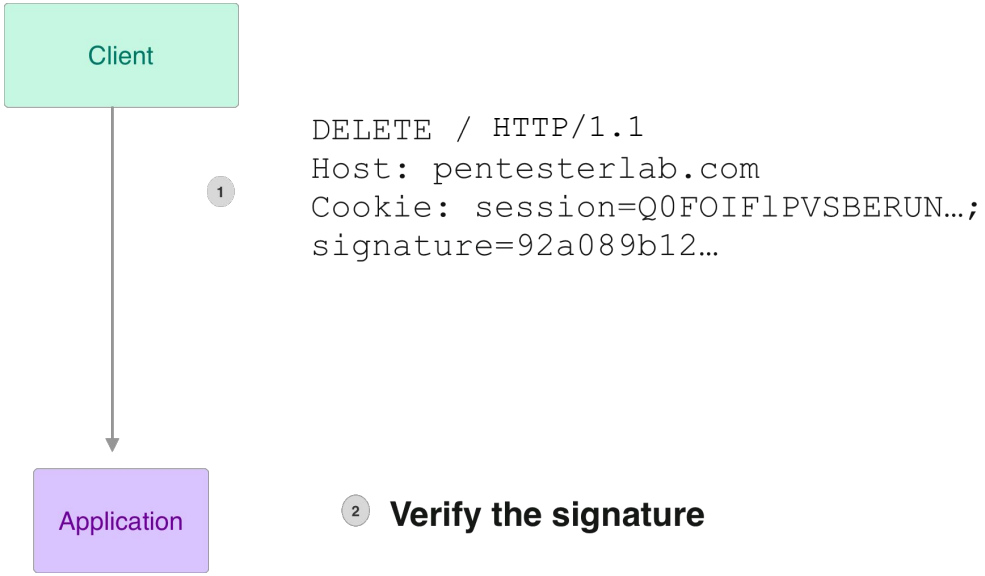
3 **Unserialize the data if the signature is valid**

Signed Sessions 101 - Logout

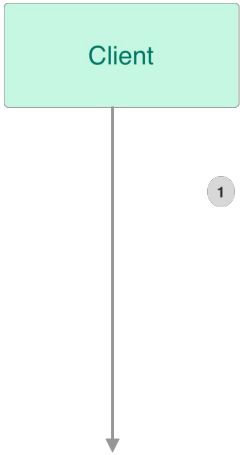



```
DELETE / HTTP/1.1
Host: pentesterlab.com
Cookie: session=Q0FOIFlPVSBERUN…;
signature=92a089b12…
```

Signed Sessions 101 - Logout



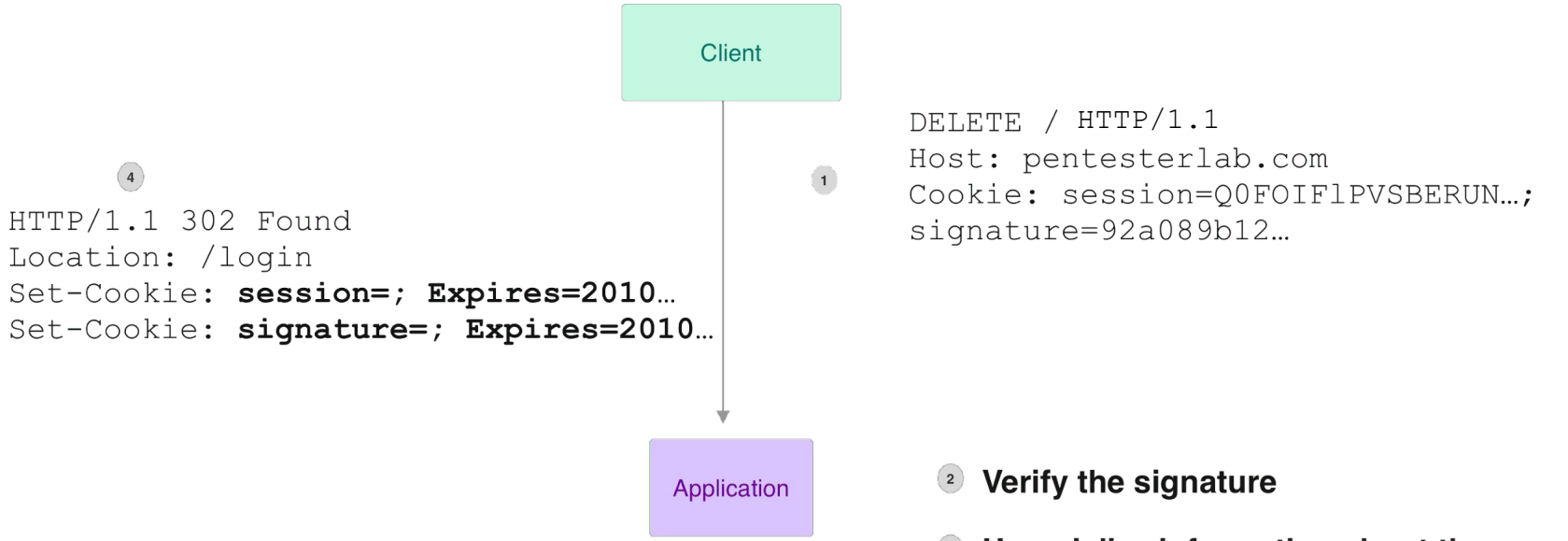
Signed Sessions 101 - Logout



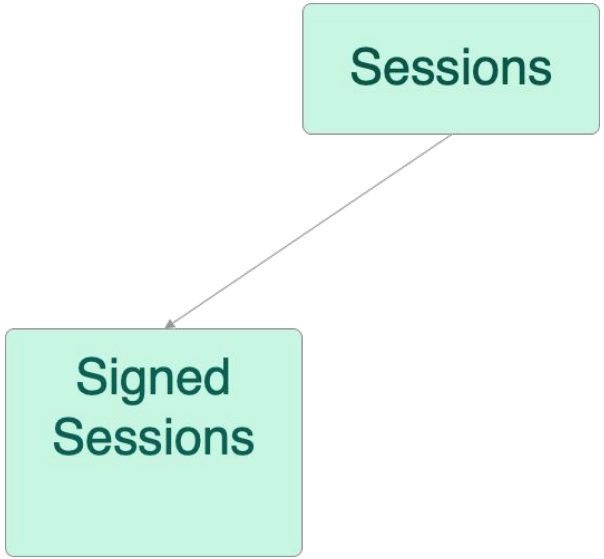

```
DELETE / HTTP/1.1
Host: pentesterlab.com
Cookie: session=Q0FOIFlPVSBERUN…;
signature=92a089b12…
```

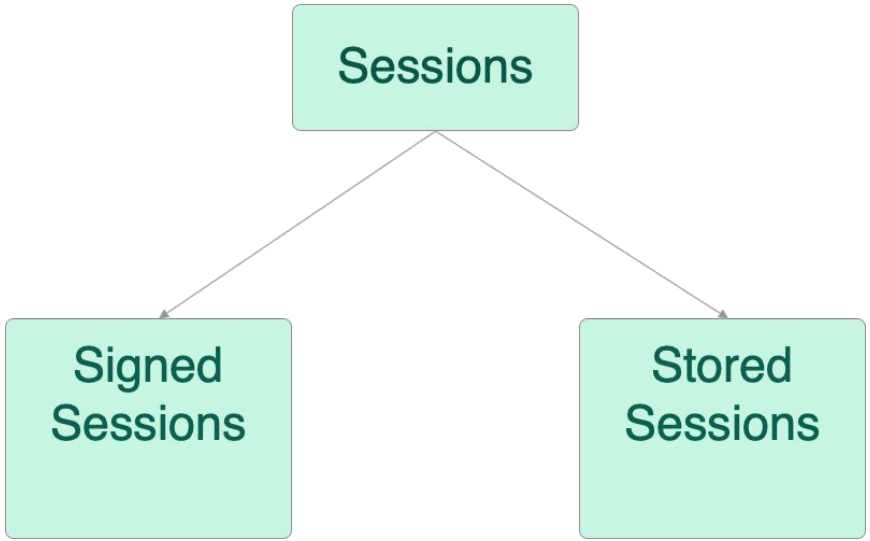
2. **Verify the signature**
3. **Unserialize information about the user**

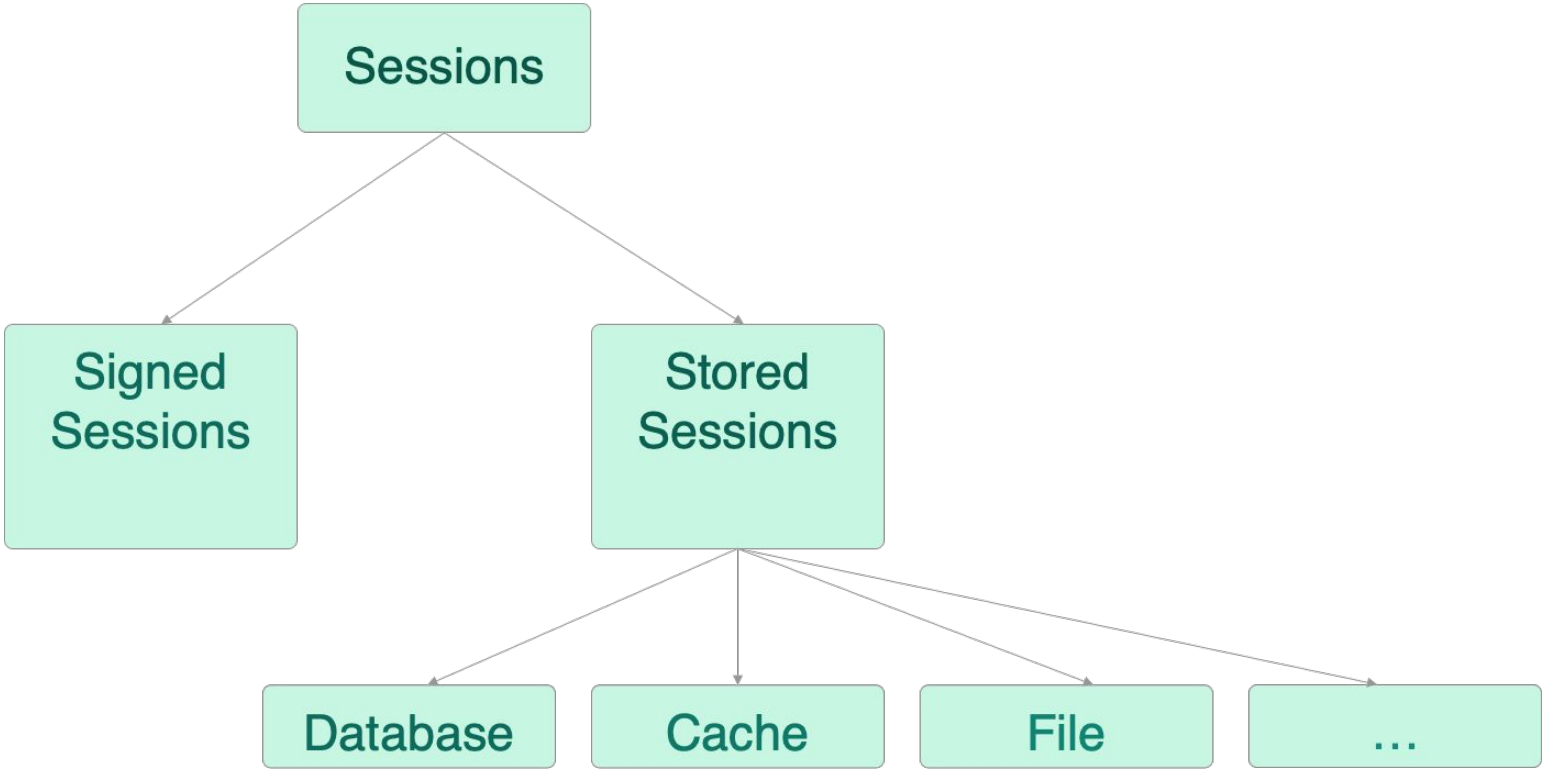
Signed Sessions 101 - Logout

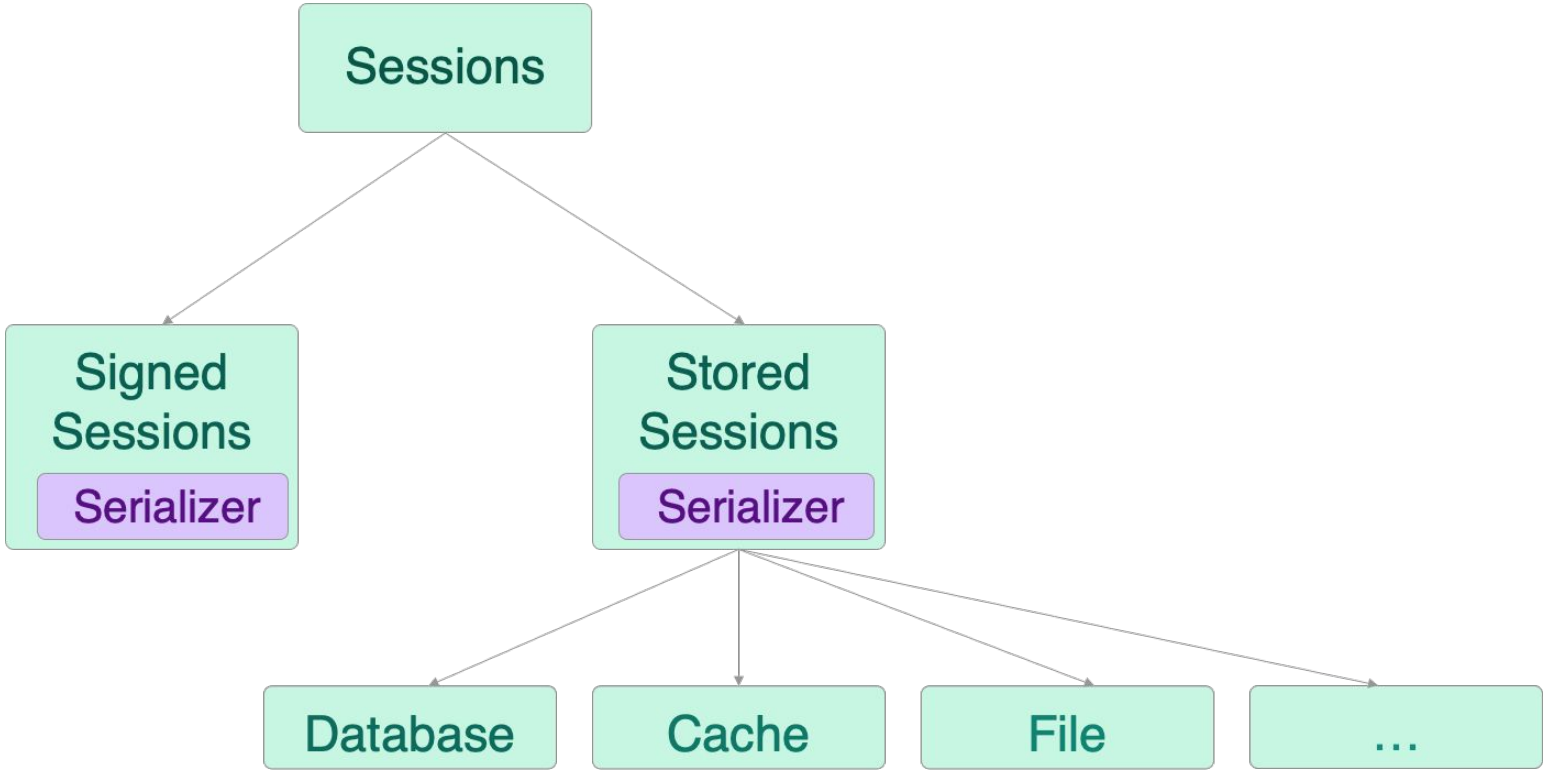


Sessions


Sessions
Signed Sessions


Sessions
Signed Sessions
Stored Sessions


Sessions
Signed Sessions
Stored Sessions
Database
Cache
File
…


Sessions
Signed Sessions
Serializer
Stored Sessions
Serializer
Database
Cache
File
…

Issues with Stored Sessions

Injection on the retrieval of the data based on the session identifier

- SQL Injection
- Directory Traversal
- ...

Issues with Stored Sessions

Leveraging arbitrary file upload with directory traversal, to pollute session pool

- Create fake session
- Exploit deserialization

Issues with Stored Sessions

Predictable session identifier

- Session ID based on username
- Weak RNG (rand(), java.util.Random, seed(time()), lcg_value, …)

Issues with Stored* Sessions

Predictable session identifier : Session ID based on username

- plain
- hashed (md5/sha/...)
- “Encrypted” (xor/rot13/...)
- …

https://github.com/tweksteen/exonumia

Issues with Stored Sessions

Predictable session identifier: Weak RNG

- rand()
- Lack of entropy
- java.util.Random
- seed(time())
- lcg_value
- ...

https://github.com/altf4/untwister

Java: https://github.com/votadlos/JavaCG

Example attack: https://news.ycombinator.com/item?id=639976

Issues with Stored Sessions

DoS via inode exhaustion...

Issues with Stored Sessions

Race conditions/TOCTOU

- Shopping cart balances

Issues with Stored Sessions

Shared pool of sessions between applications

https://www.slideshare.net/ZendCon/lesser-known-security-problems-in-php-applications-presentation

Almost Vulnerabilities

Express Session File Store

https://github.com/valery-barysok/session-file-store/blob/master/lib/session-file-helpers.js#L19-L22

```
  sessionPath: function (options, sessionId) {
    //return path.join(basepath, sessionId + '.json');
    return path.join(options.path, sessionId + options.fileExtension);
  },
```

Express Session File Store

Not Exploitable :/

https://github.com/valery-barysok/session-file-store/blob/master/lib/session-file-helpers.js#L19-L22

```
sessionPath: function (options, sessionId) {
  //return path.join(basepath, sessionId + '.json');
  return path.join(options.path, sessionId + options.fileExtension);
},
```

- Express sessions are signed by default
- NULL bytes can't be used to get rid of the extension
  ("The argument 'path' must be a string or Uint8Array without null bytes")

Flask Session

https://github.com/fengsp/flask-session/blob/master/flask_session/sessions.py#L319-L335

```
def open_session(self, app, request):
    sid = request.cookies.get(app.session_cookie_name)

    ...

    data = self.cache.get(self.key_prefix + sid)
```

Flask Session

Not Exploitable :/

https://github.com/fengsp/flask-session/blob/master/flask_session/sessions.py#L319-L335

```
def open_session(self, app, request):
    sid = request.cookies.get(app.session_cookie_name)

    ...

    data = self.cache.get(self.key_prefix + sid)
```

Not exploitable! They rely on FileSystemCache that md5s the “file name” (https://github.com/pallets/cachelib/blob/master/cachelib/file.py#L112-L116)

CVE-2018-18925

Gogs/Gitea RCE: CVE-2018-18925

Gogs/Gitea RCE: CVE-2018-18925

Gogs/Gitea RCE: CVE-2018-18925

Gogs/Gitea RCE: CVE-2018-18925

- Gogs/Gitea does not check the session identifier for directory traversal
- An attacker can use a malicious session id to load a previously uploaded file
- Use “admin” session and get RCE via git hooks

CVE-2018-20303

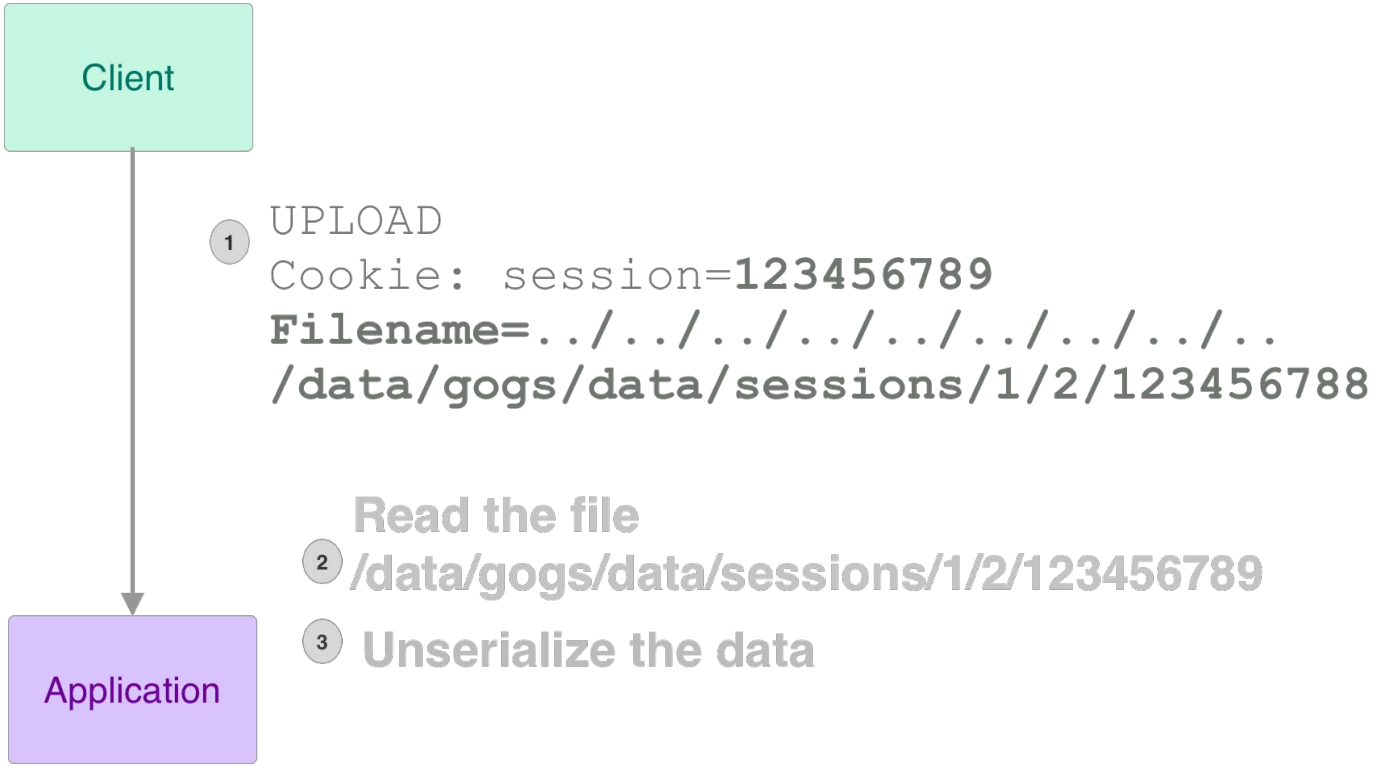
Gogs/Gitea RCE II: CVE-2018-20303

- Gogs/Gitea does not check the filename when you upload a file
- An attacker can use a malicious filename to add a session in the session pool
- Use “admin” session and get RCE via git hooks

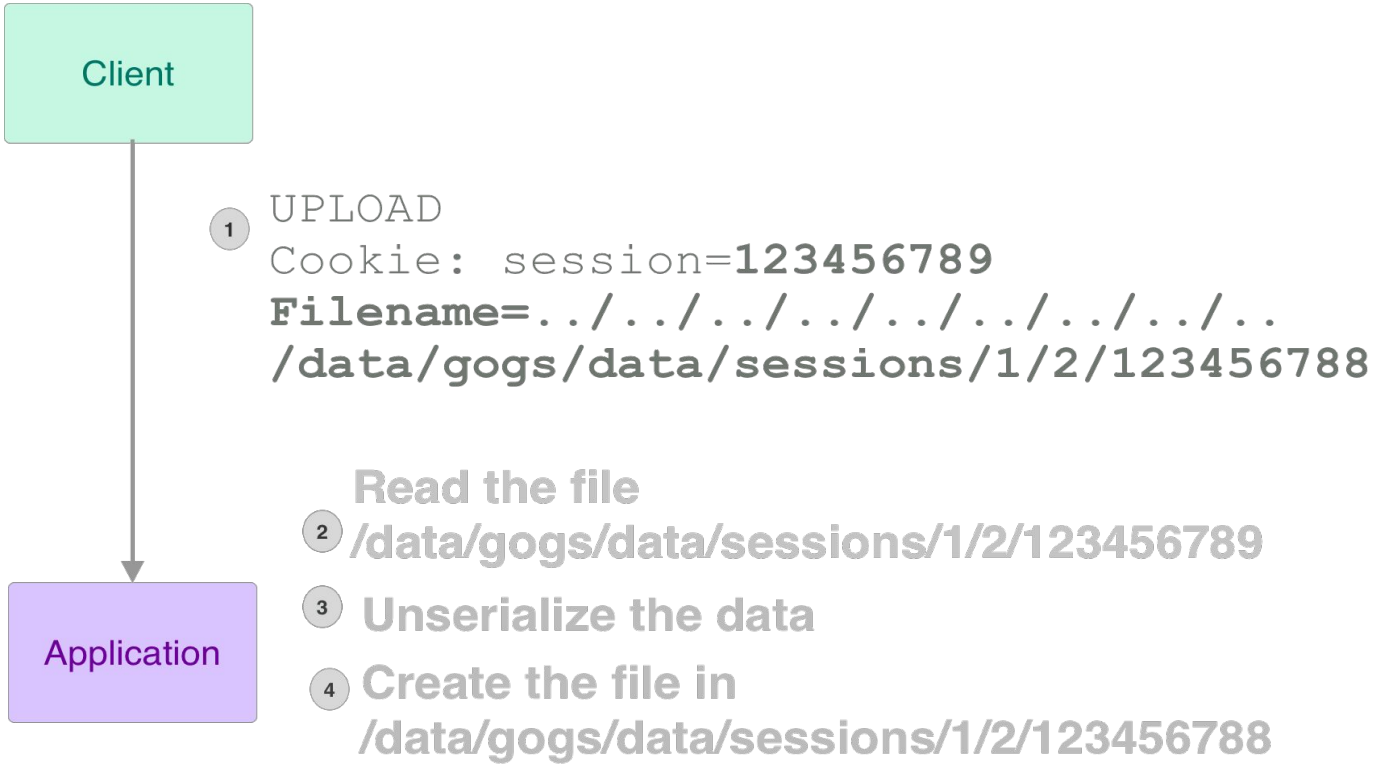
Gogs/Gitea RCE II: CVE-2018-20303

Gogs/Gitea RCE II: CVE-2018-20303

Gogs/Gitea RCE II: CVE-2018-20303



Gogs/Gitea RCE II: CVE-2018-20303



Tomcat Session

https://github.com/apache/tomcat/blob/master/java/org/apache/catalina/session/FileStore.java#L343-L350

```
private File file(String id) throws IOException {
    if (this.directory == null) {
        return null;
    }
    String filename = id + FILE_EXT;
    File file = new File(directory(), filename);
    return file;
}
```

Issues with Signed Sessions

Weak || shared secret/key

CVE-2019-5420

Weak signature verification

Not time constant comparison

Only verifying if a signature is provided

Replay

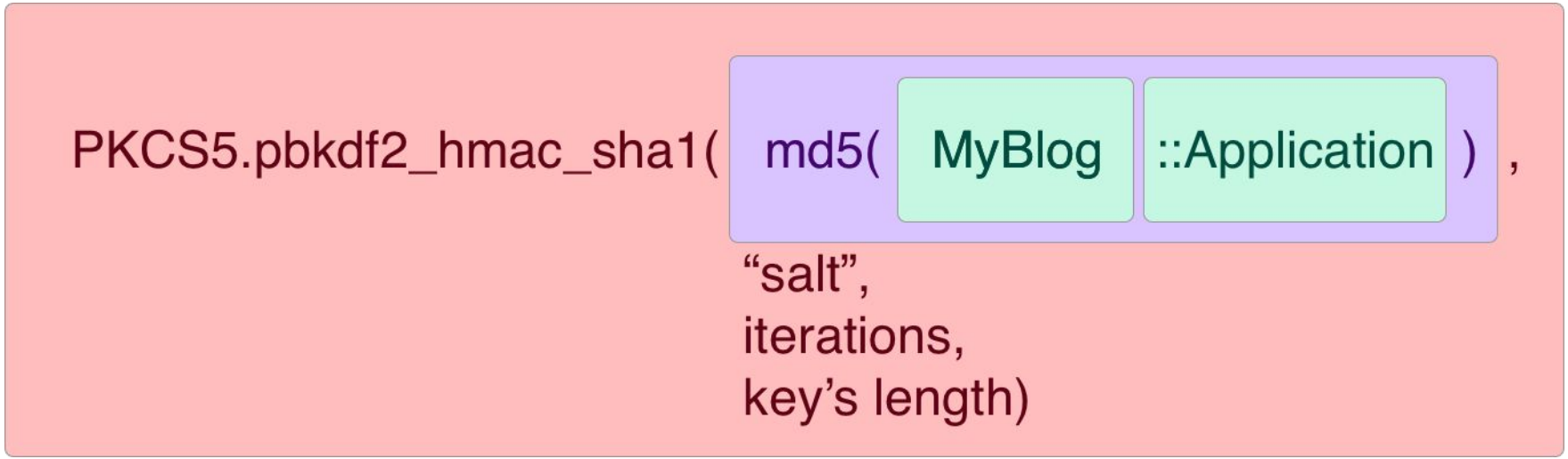
Bypass Anti Bruteforce protection

Logout and Bug Bounty (and Signature Malleability)...

Rails CVE-2019-5420

- In development mode, Rails derives the key used to protect sessions using the application’s name

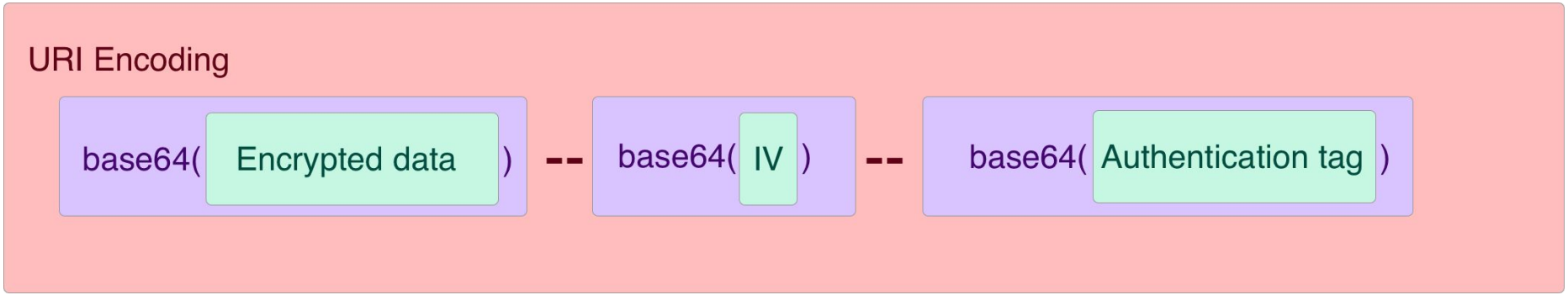
Rails CVE-2019-5420



Rails CVE-2019-5420

```
PKCS5.pbkdf2_hmac_sha1( md5( MyBlog ::Application ) ,
"authenticated encrypted cookie",
1000,
32)
```

Rails CVE-2019-5420



Rails CVE-2019-5420

- In development mode, this can lead to Remote Code Execution if the application supports Marshal(more on that later)

- Even in development mode, CVE-2019-5420 may not impact web applications using the :json serialiser and devise for authentication
  - Devise stores and verifies part of the user’s hashed password in the session.

The Serializers

Issues with Serializers

Injection in the value

Play Framework Session Injection

* this can also impact Stored Sessions

Remote Code Execution

Rails Marshal

Play Session Injection

KEY1:VALUE1 | KEY2:VALUE2 | KEY3:VALUE3

Play Session Injection

%00KEY1:VALUE1%00%00KEY2:VALUE2%00%00KEY3:VALUE3%00

- Play used to have its own parser

```
Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");
```

- Nothing preventing injection of : or NULL byte

Play Session Injection

- Play used to have its own parser

```
Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");
```

- Nothing preventing injection of : or NULL byte

Play Session Injection

KEY1:VALUE1%00%00KEY4:VALUE4 | KEY2:VALUE2 | KEY3:VALUE3

- Play used to have its own parser

```
Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");
```

- Nothing preventing injection of : or NULL byte

Play Session Injection

KEY1:VALUE1 | KEY4:VALUE4 | KEY2:VALUE2 | KEY3:VALUE3

- Play used to have its own parser

```
Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");
```

- Nothing preventing injection of : or NULL byte

Play Session Injection

- Play used to have its own parser

```
Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");
```

- Nothing preventing injection of : or NULL byte

Play Session Injection

- Play used to have its own parser

```
Pattern.compile("\u0000([^:]*):([^\u0000]*)\u0000");
```

- Nothing preventing injection of : or NULL byte

Stored Session Examples

	Serializer	Storage Location	Extras
Django	Pickle	Database	
Tomcat (FileStore)	ObjectInputStream	File system	
PHP	serialize	File system	Encrypted with Suhosin

Signed Session Examples

	Signing Algorithm	Serializer	Extras
Express	HMAC-SHA1	JSON	Signature stored in separate cookie
Go Secure Cookie	HMAC-SHA2	JSON	Opt-in Encryption (AES-CTR)
Phoenix	HMAC-SHA256	External Term Format	Key length > 64 PBKDF2-HMAC-SHA256
Ruby on Rails	AES-GCM	JSON	

Rails Sessions

Version	Date	Serializers	Signature/Encryption	Secret
3.0	August 2010	Marshal	SHA1	Cleartext
4.2	August 2014	Marshal	AES-CBC and SHA1	Cleartext
5.2	June 2018	JSON, Marshal, Hybrid	AES-GCM	Encrypted

- `Rails now supports two serializers: :json, :marshal and :hybrid`
- `Marshal gives you code execution if you can forge a session`

Conclusion

Trust but verify

Make sure you test for this when doing pentest/bug bounty

Sessions are often at the edge between the application’s developer responsibilities and the framework’s developer responsibilities

Questions?

@BitcoinCTF
@PentesterLab