Unique ways to Hack into a Python Web Service

About

- Senior Solutions Engineer at we45
- Developer of Open-Source Project called Orchestron, ThreatPlaybook
- Part of multiple CTF

https://github.com/we45/orchestron-community

https://github.com/we45/ThreatPlaybook

@ti1akt

Agenda

- Intro to Web-Services
- Common Vulnerabilities
- Unique Vulnerabilities
- Some of Remediation Techniques
- Demo !

What is Web-Service?

- It designed to support interoperable machine to machine over the internet
- It is not tied to any one operating system or programming language

Types of Web-Service?

- SOAP Web Services
- RESTful web services

Python Rest Frameworks

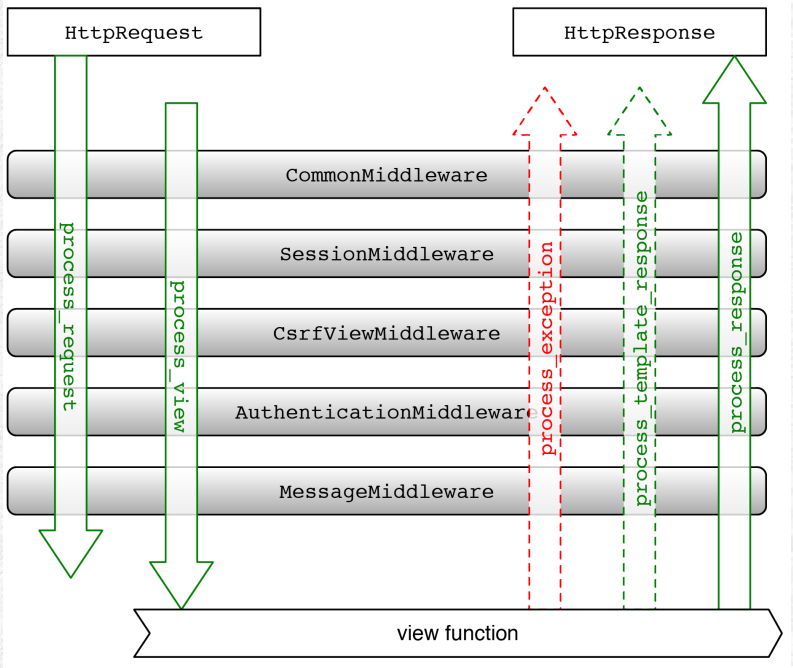
- DjangoRestFramework
- Flask
- Falcon
- Pyramid
- CherryPy
- Bottle

Common Security Threats

- SQL Injection
- Cross Site Scripting
- Broken Authentication
- Security Misconfiguration
- Cross-site Request Forgery
- Many More ..

Django prevents some of these attacks

- SQL Injection
- Cross-Site Request Forgery
- Cross-Site Scripting
- Session Hijacking



What about these

- JWT Manipulation
- Xml External Entity
- InsecureDirectObjectReference
- Server-Side Template Injection
- Etc …

JWT Manipulation

OWASP-2017 A5 Broken Access Control

Why JWT

- Stateless Application
- Authorization Mechanism
- Transfers information between server and client
- Scalable and decoupled

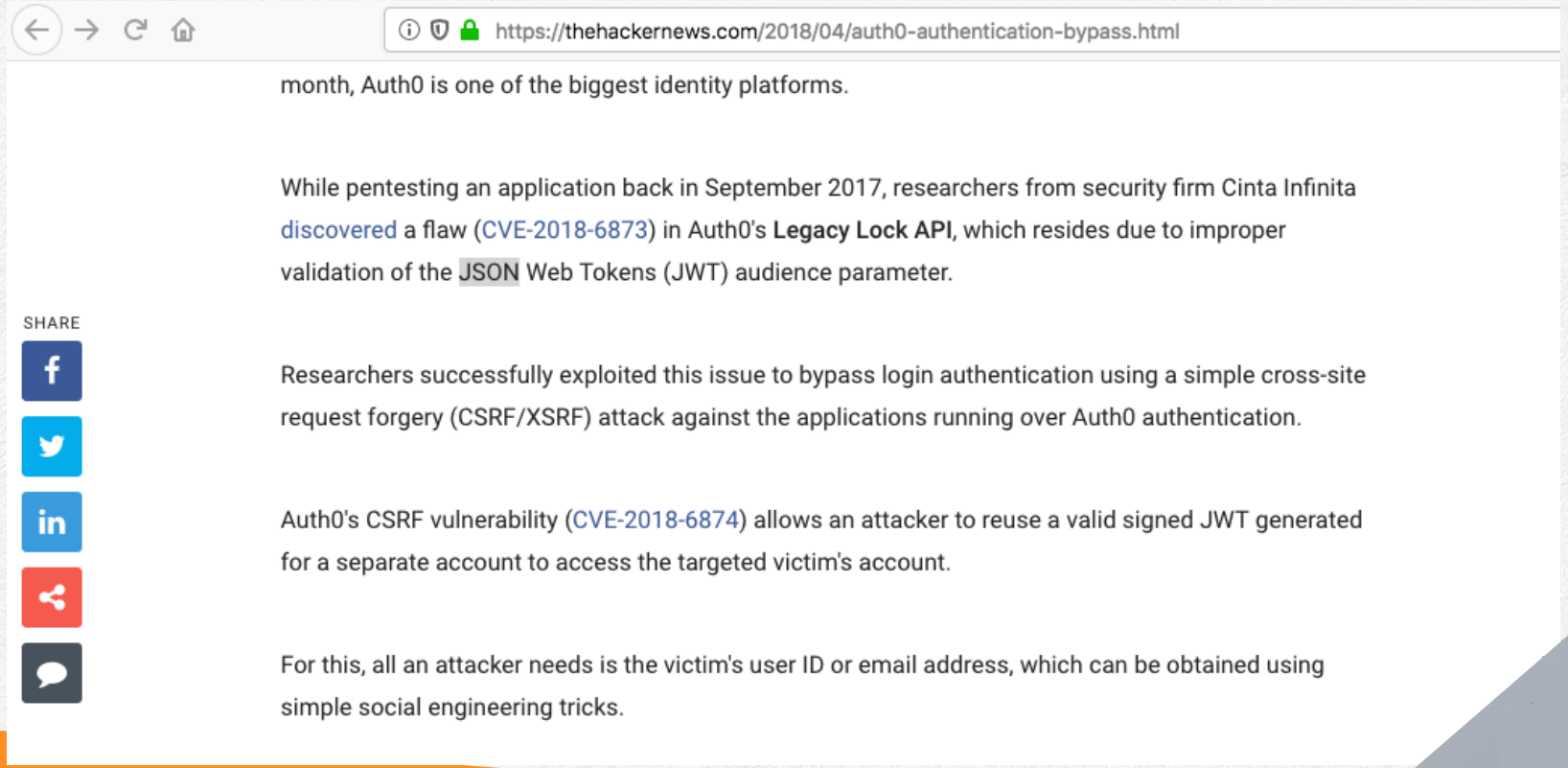
JSON Web Token(JWT)

- The process is relatively simple (typically):
  - Once a user authenticates, the server generates some JSON payload (with some info) and signs the JSON payload with a key
  - This can be a HMAC Based Key (HS256) or a Asymmetric System (RS256)
  - The token is sent by the client (like a session cookie)

Lots of ways to get JWT wrong

- JWT allows for a “none” signature for a token
- Algo Confusion Attacks:
    - CVE-2017-11424
    - CVE-2015-9235
- JWT verification on non-unique private claims

Recent Attack



source: https://thehackernews.com/2018/04/auth0-authentication-bypass.html

Mitigation

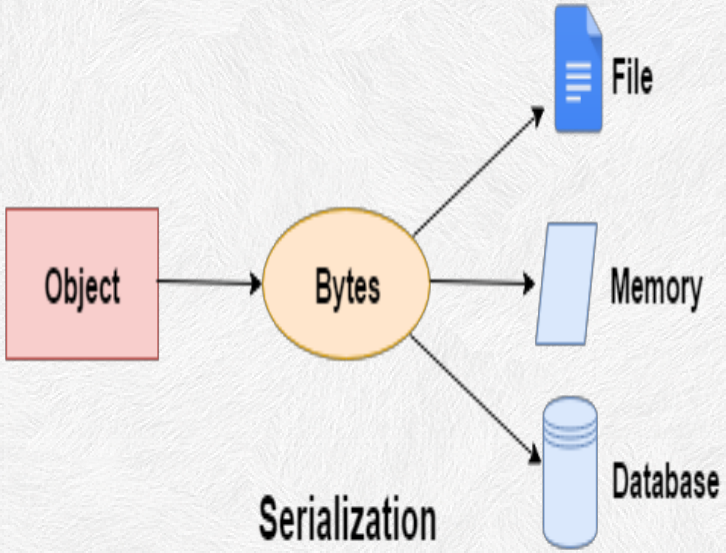
- Validate using Unique ID
- Ensure that JWT implementation doesn't support 'None' signature
- JWT lifetime relatively short
- Check library flaw

Insecure Deserialization

OWASP-2017 A8 Insecure Deserialization

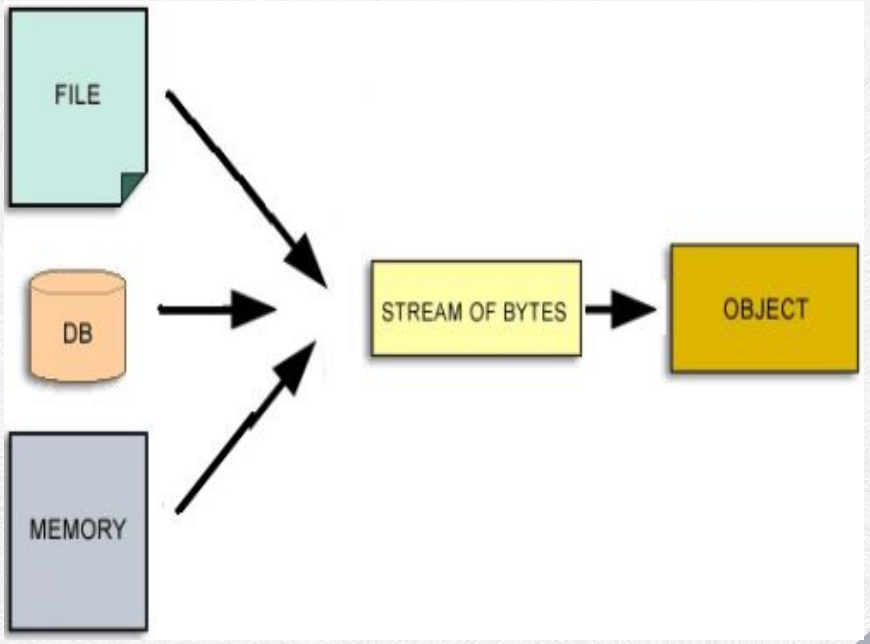
What is Serialization

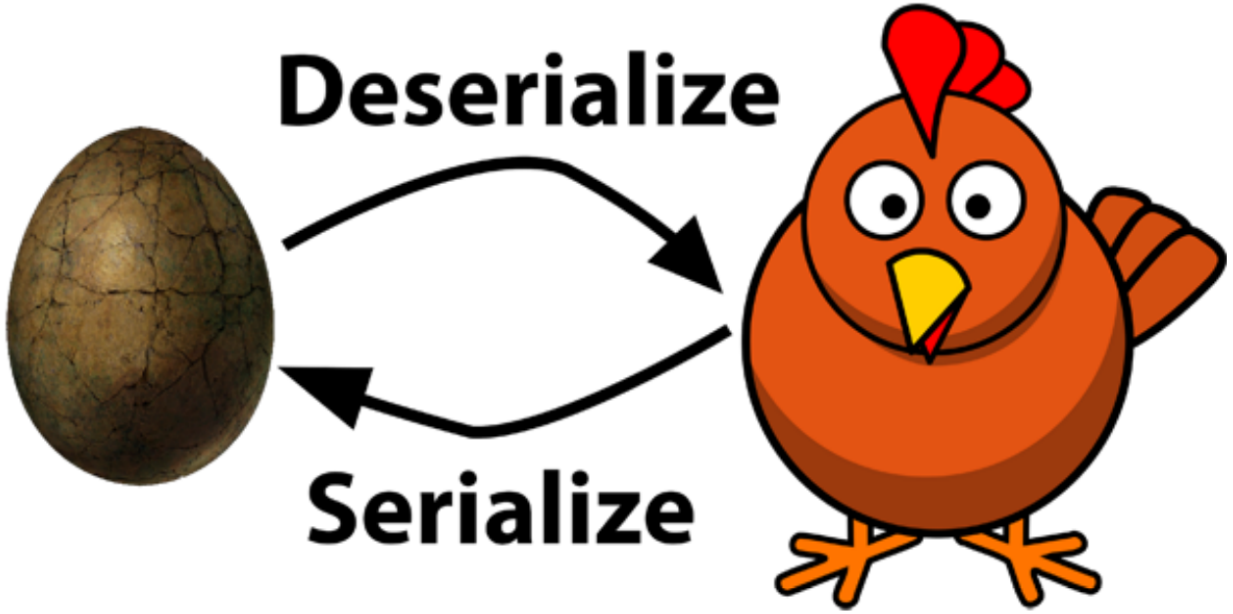
- Serialization means converting object into a binary stream.



What is Deserialization

- Deserialization means converting binary stream into an object.





Deserialization vulnerability

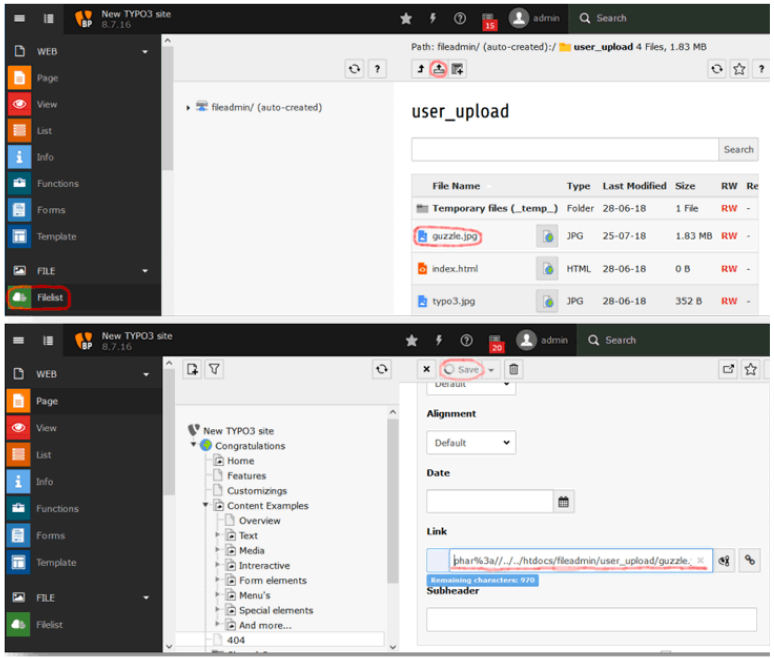
Security Gotcha !!!

- Malformed data deserialisation
- Abusive of an application logic
- Denial-of-Service
- Remote code execution

Recent Attack

https://thehackernews.com/2018/08/php-deserialization-wordpress.html

Exploiting PHP Deserialization Attack Against WordPress Sites



In a detailed paper released at Black Hat conference last week, Thomas demonstrated how this attack can be executed against Wordpress sites using an author account to take full control over the web

SHARE

Popular News

New KickAss Torrent
Best Torrent Sites of

The Pirate Bay Altern
Best Torrent Sites | F
Download

New Cold Boot Attac
Disk Encryption On N
Modern PCs

Source https://thehackernews.com/2018/08/php-deserialization-wordpress.html

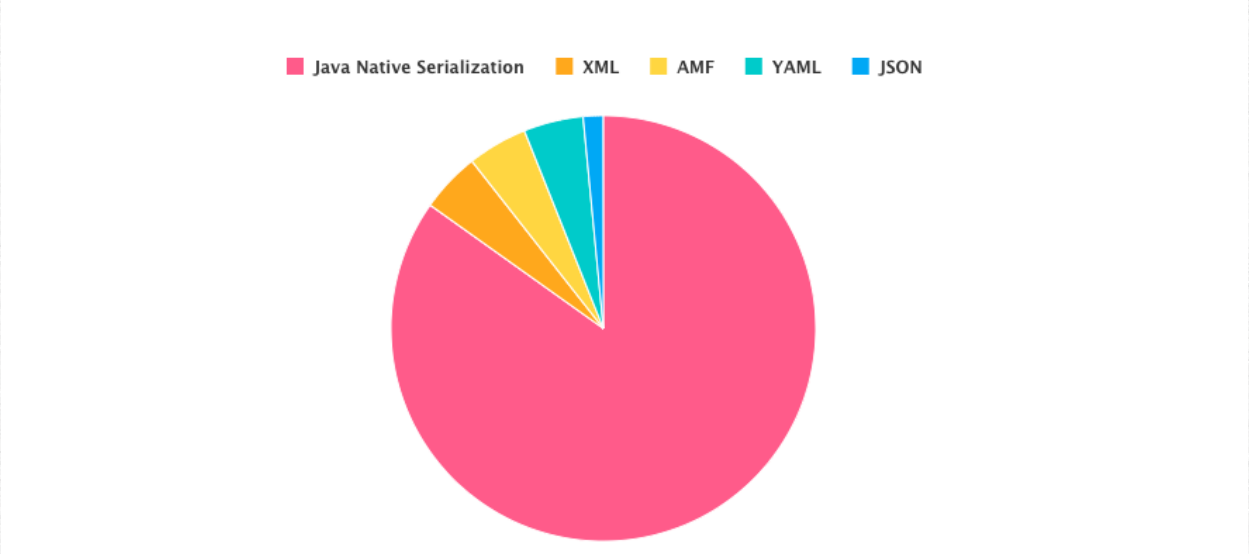
Sourcehttps://threatpost.com/java-serialization-bug-crops-up-at-paypal/116054/

Threatpost, Inc (US) | https://threatpost.com/java-serialization-bug-crops-up-at-paypal/116054/

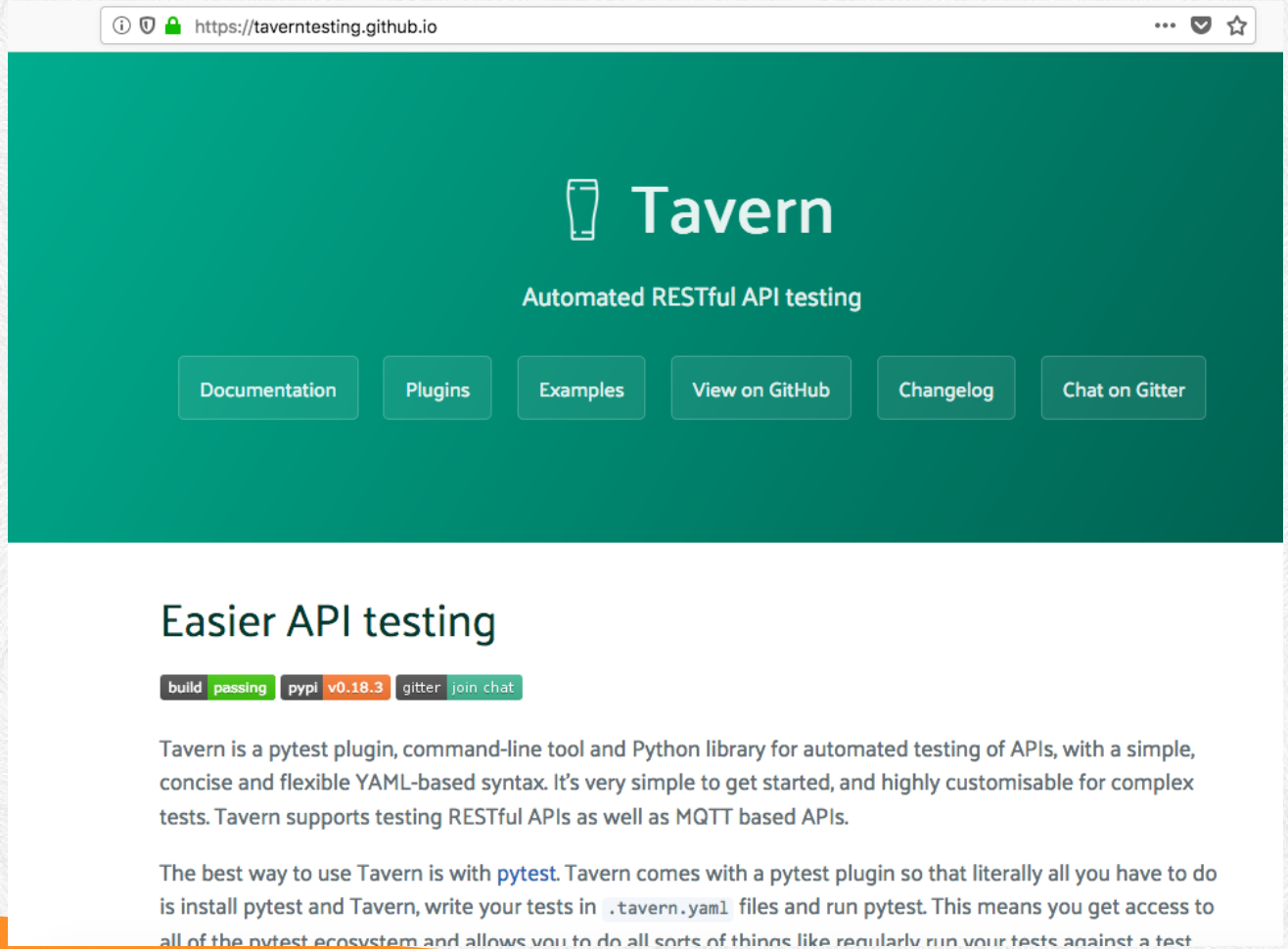
threatpost | Cloud Security / Malware / Vulnerabilities / Privacy

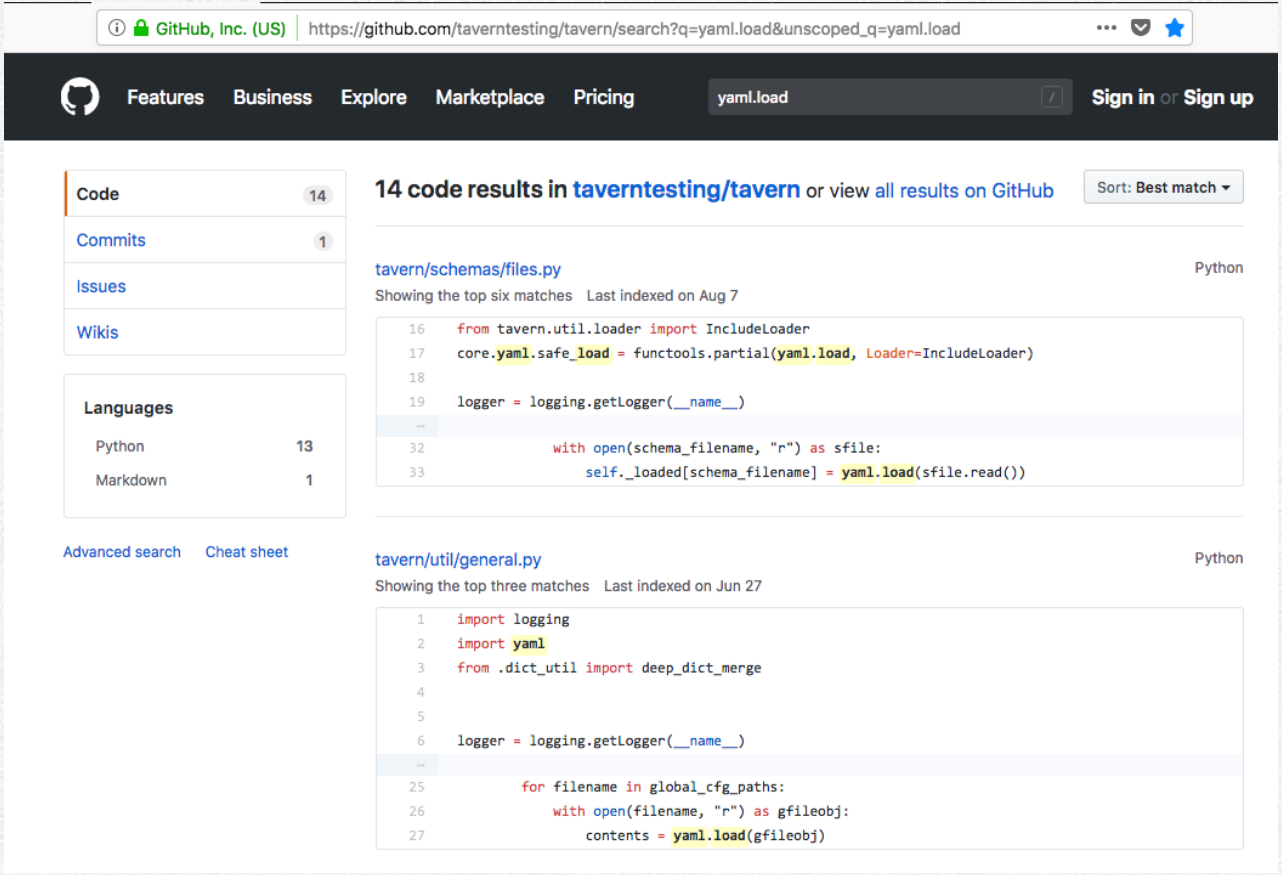
Java Serialization Bug Crops Up At PayPal

Vulnerabilities Serialization formats



Source https://thehackernews.com/2018/08/php-deserialization-wordpress.html





Mitigations

- Integrate integrity check such as digital signature
- Isolate then deserialise the data
- Monitor incoming and outgoing network connectivity
- Instead of ‘yaml.load’ use yaml.safe_load

Insecure Direct Object Reference

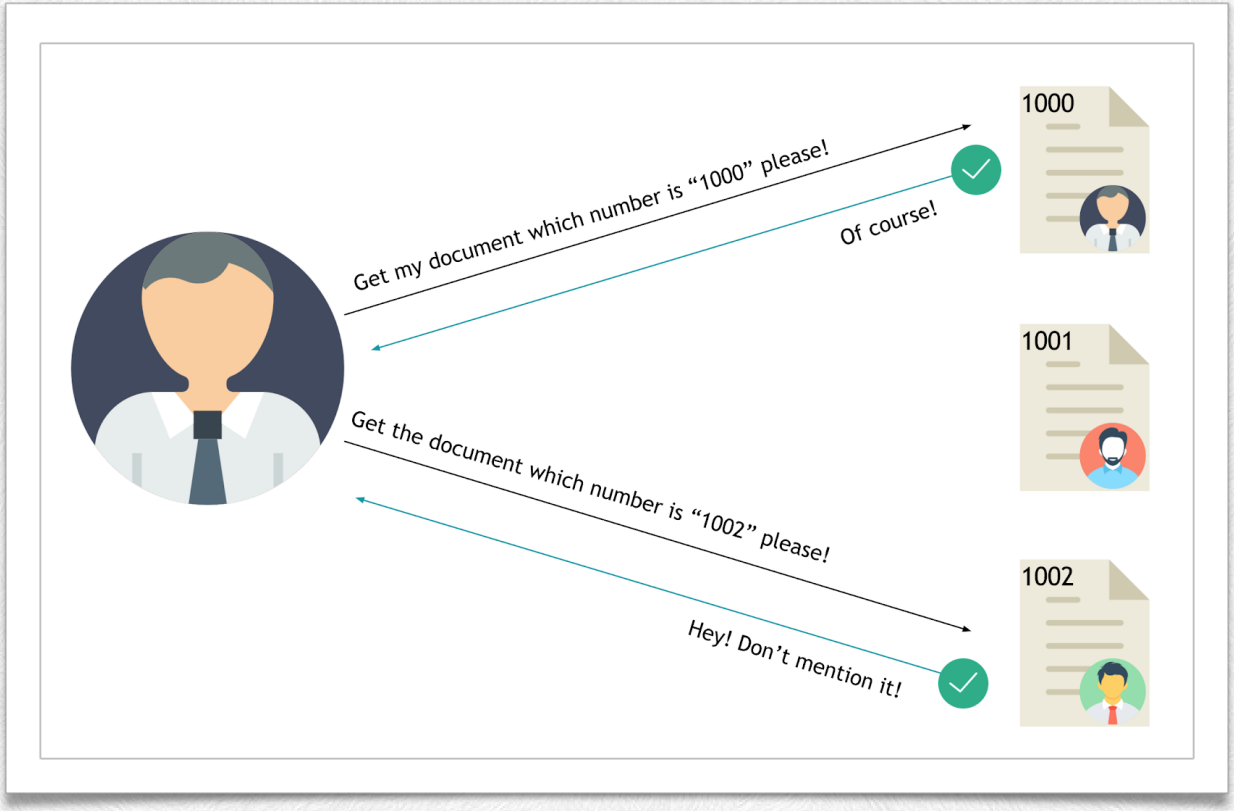
OWASP-2017 A8 Broken Access Control

Insecure Direct Object reference

- “id”,”pid”,”uid” are often seen in HTTP parameter
- Accessing other user privilege
- Backend not properly validated users

```
re_path(r'api/get_user/(?P<pk>[0-9]+)$', IndividualUserView.as_view({'get':'list','post':'update'})
```

How it works



Bad Code

```
class IndividualUserView(viewsets.ModelViewSet):
    serializer_class = UserSerializer
    model_class = User

    def get_queryset(self, pk):
        object_list = self.model_class.objects.get(pk=pk)
        return object_list

    def list(self, request, pk=None):
        queryset = self.get_queryset(pk)
        serializer = self.serializer_class(queryset)
        return Response(serializer.data)

    def update(self, request, pk=None):
        queryset = self.get_queryset(pk)
        serializer = self.serializer_class(queryset, data=request.data)
        if serializer.is_valid(raise_exception=True):
            serializer.save()
            return Response(serializer.data, status=status.HTTP_200_OK)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
```

Good Code

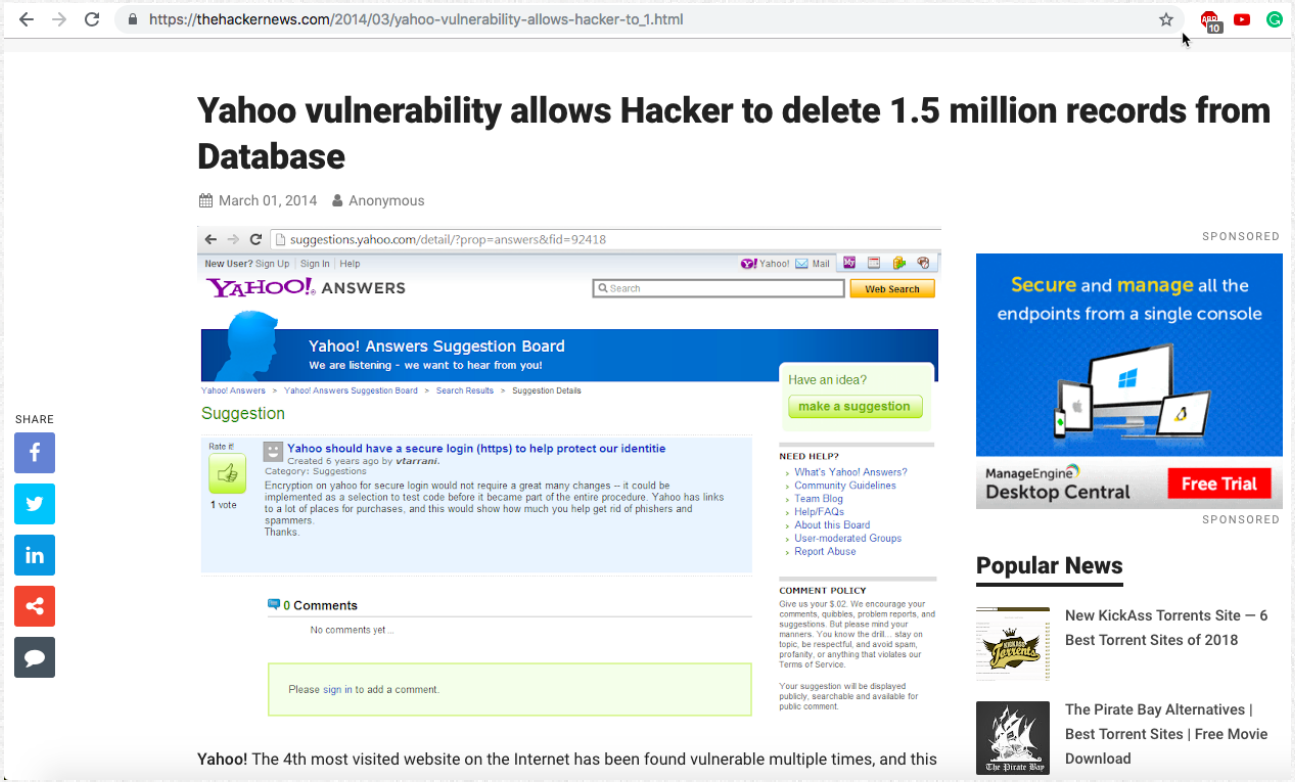
```
class IndividualUserView(viewsets.ModelViewSet):
    serializer_class = UserSerializer
    model_class = User

    def get_queryset(self,pk):
        object_list = self.model_class.objects.get(pk=pk)
        return object_list

    def list(self, request):
        queryset = self.get_queryset(request.user.id)
        return Response(serializer.data)

    def update(self, request):
        queryset = self.get_queryset(request.user.id)
        if serializer.is_valid(raise_exception=True):
            serializer.save()
            return Response(serializer.data, status=status.HTTP_200_OK)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
```

Yahoo Breach



Source:https://thehackernews.com/2014/03/yahoo-vulnerability-allows-hacker-to_1.html

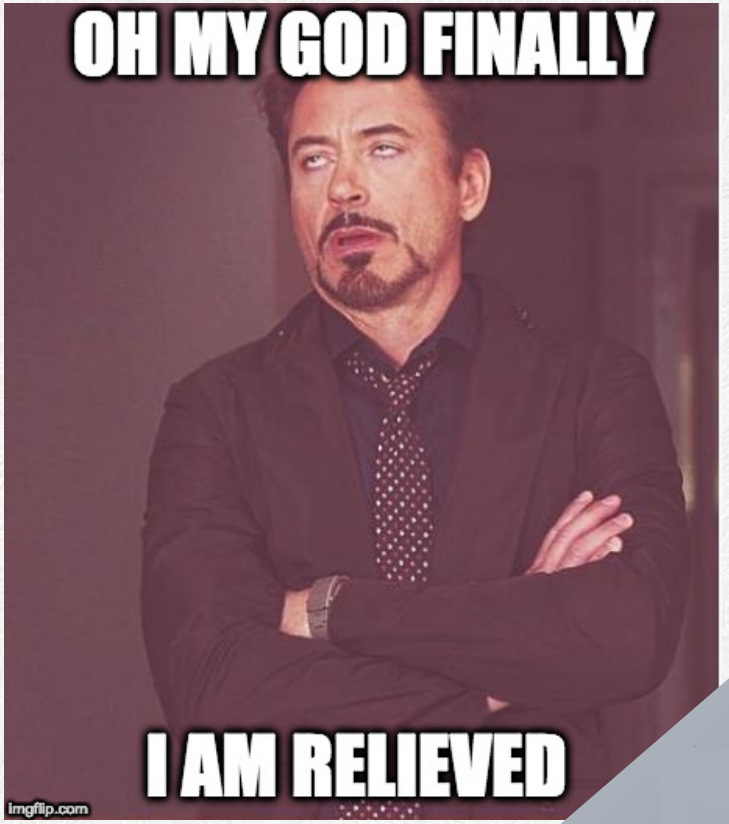
Mitigation

- Validate user using requested query
- Check database is that user is genuine or not
- Custom validation in server side as well as client side
- JWT should be invalidated once the user is logout

Some Tips

To Prevent some of threats

- Run SCA,
  - https://github.com/pyupio/safety
- Run SAST
  - https://github.com/PyCQA/bandit
- Run DAST
  - https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project
- Include security testing in DevOps pipeline



Basic Pipeline Demo

Download Examples and Slides

- You can download it from
  - http://github.com/we45/django con-2018

https://github.com/we45/orchestron-community

https://github.com/we45/ThreatPlaybook

Thank you

@ti1akt