Web App Attacks of the Modern World

Karan Sharma -> @_R00T_

Document Classification: PROTECTED

$id

- Pen Tester (by day) : Web, Mobile & IoT
- Bug Hunter (at night)
- OSWE (WIP), OSCP, eWPTX etc.
- Football, Pool and Ping – Pong
- Twitter: @_R00T_
- YouTube Channel: SCHOOL OF ROOT

Document Classification: PROTECTED

Document Classification: PROTECTED

Topics

- SSRF
- ESI Injection
- Web Cache Deception
- GraphQL

Document Classification: PROTECTED

Server Side Request Forgery (SSRF)

Document Classification: PROTECTED

- A Server Side Request Forgery (SSRF) vulnerability refers to an attack where an attacker can change a parameter used on the web application to create or control requests from the vulnerable server.

- SSRF is mainly used to attack internal systems that are sitting behind a firewall and attacker may not be able to access these systems from external network.

Document Classification: PROTECTED

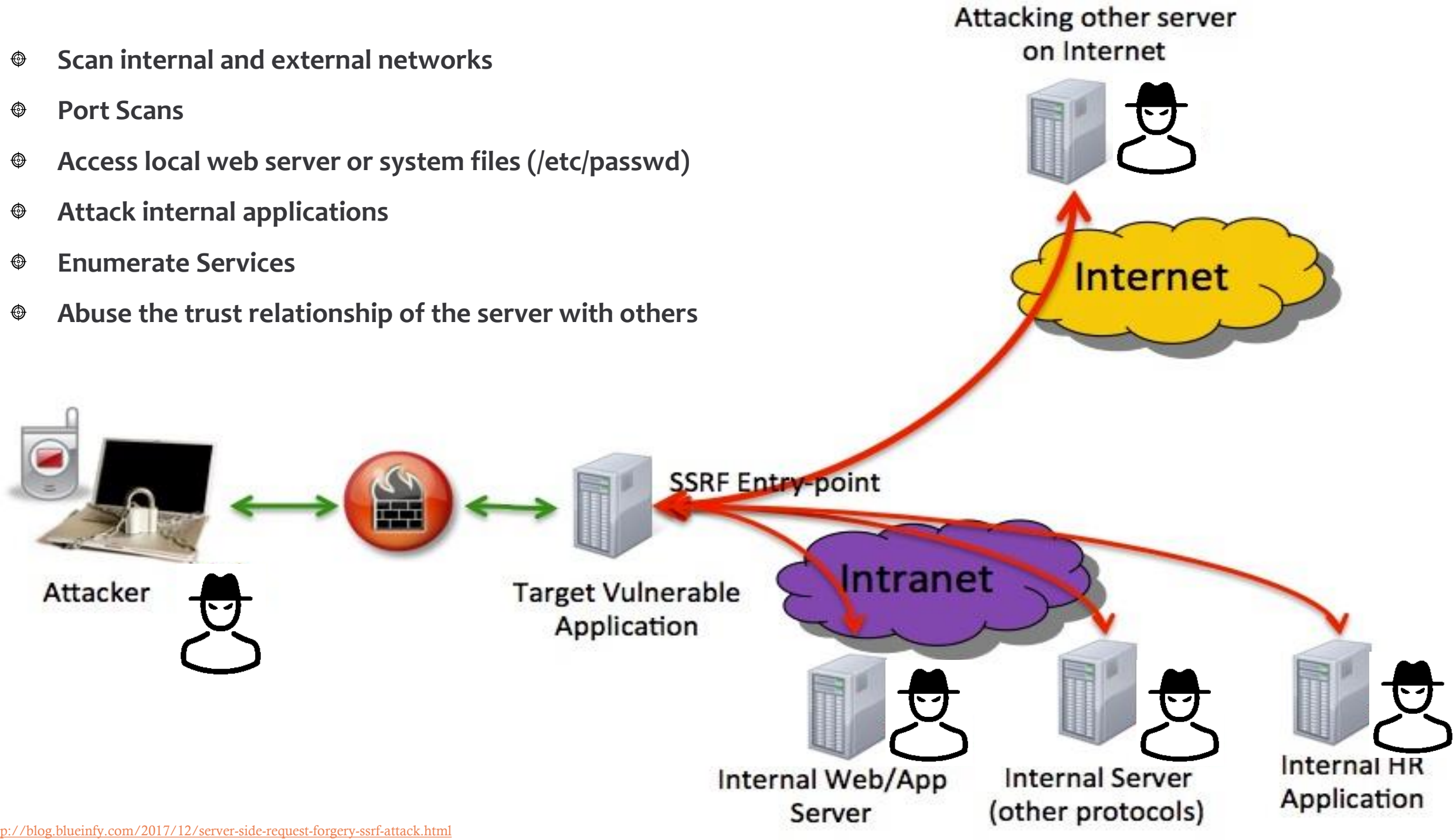
- For example, a developer can use a URL such as https://mysite.com/feed.php?url=https://externalsite.com/feed to retrieve the remote feed.

- If the app is vulnerable, an attacker is able to control this URL parameter.

- If an attacker change this url parameter to localhost, then the attacker will be able to view local resources hosted on the server, making it vulnerable to SSRF.

Document Classification: PROTECTED

- Scan internal and external networks
- Port Scans
- Access local web server or system files (/etc/passwd)
- Attack internal applications
- Enumerate Services
- Abuse the trust relationship of the server with others



http://blog.blueinfy.com/2017/12/server-side-request-forgery-ssrf-attack.html

Request

Raw Params Headers Hex JSON Beautifier

```
POST /api/export/ HTTP/1.1
Host: [redacted]
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:63.0) Gecko/20100101 Firefox/63.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: [redacted]/Dashboard/fb58662c-1f60-4d07-a26a-5d436935e80e?e=false&vo=none
Content-Type: application/json; charset=utf-8
X-Requested-With: XMLHttpRequest
Content-Length: 4094
Connection: close
Cookie: dundas_webapp_clientid=cab1c9bb-07d6-4c27-94d8-466edf86806a; dundas_webapp_sessionid=0b89b113-3367-4d34-be35-f52dc8afc0f0

{"__classType":"dundas.export.ExportRequest","id":"08f41699-e344-812b-a5c8-eaf55acda35d","viewUrl":"/Link/?shortLink=zwee6x6ojtxrjnu
6lnajra8fdr","viewId":"fb58662c-1f60-4d07-a26a-5d436935e80e","viewData":{"__classType":"dundas.entities.Dashboard","ignoreAspectRati
oOnResize":false,"isTemplate":false,"templateCells":[],"templateCellHorizontalSpacing":5,"templateCellVerticalSpacing":5,"adapters":
[],"groups":[],"layers":[{"__classType":"dundas.view.controls.Layer","isTemplateLayer":false,"baseZIndex":1001,"locked":false,"opaci
ty":1,"hidden":false,"id":"5fd61721-93cd-7261-c51e-6fb9e16a31f6","name":"Layer1","friendlyName":"Layer
1"}],"viewParameters":[{"__classType":"dundas.view.ViewParameter","id":"486f1c82-5360-430d-bb74-4a1f4f88591d","name":"BrushViewParam
eter","friendlyName":"Brush View
Parameter","elementParameterLinks":[]}],"brushViewParameterId":"486f1c82-5360-430d-bb74-4a1f4f88591d","disableAutomaticBrushing":fal
se,"loadingActions":[],"readyActions":[],"beforeExportActions":[],"resizeActions":[],"contextMenuShowingActions":[],"resizeMode":"Sc
roll","resizeModeFit":"Both","background":{"__classType":"dundas.controls.SolidColorBrush","color":"white","brushType":"SolidColor"}
,"width":1024,"height":768,"viewType":"Dashboard","version":6,"initialViewOptions":"none","exportTimeout":"00:00:00","revision":-1,"
checkedOutTo":{"__classType":"dundas.account.MemberInfo","id":"92ace8aa-757c-44d1-a701-6234ed210fbb","memberKind":"Account"},"isChec
kedOut":true,"isCheckedOutToCaller":true,"currentRevision":-1,"isTransient":true,"isInactive":false,"id":"fb58662c-1f60-4d07-a26a-5d
436935e80e","objectType":"Dashboard","subtype":0,"name":"Dashboard 1","fullName":"/User Projects
Root/92ace8aa-757c-44d1-a701-6234ed210fbb/DashboardsRootFolder/Dashboard 1","location":"/User Projects
Root/92ace8aa-757c-44d1-a701-6234ed210fbb/DashboardsRootFolder","friendlyName":"Dashboard 1","friendlyFullName":"[My Project]
Dashboards Root Folder/Dashboard 1","friendlyLocation":"[My Project] Dashboards Root
Folder","description":"","tags":[],"parentId":"b2715b35-2e45-4051-abfe-e905d9a433e9","projectId":"5ae447bb-0738-404a-8159-b73dcb5a2c
34","isSubentry":false,"isProtected":false,"childCount":0,"createdBy":{"__classType":"dundas.account.MemberInfo","id":"92ace8aa-757c
-44d1-a701-6234ed210fbb","memberKind":"Account"},"createdTime":"2018-09-18T15:11:28.096Z","lastModifiedTime":"2018-09-18T15:11:43.69
8Z","files":[],"folders":[],"children":[],"allChildren":[],"allFiles":[],"allFolders":[],"privilegeInheritanceBehavior":"Inherit","p
rivileges":[{"__classType":"dundas.filesystem.PrivilegeAssignment","privilegeId":"1423fbbd-b6d9-429f-a9ea-d68a29b81cc9","assignmentK
ind":"Grant","assigneeKind":"Account","assignee":{"__classType":"dundas.account.MemberInfo","id":"92ace8aa-757c-44d1-a701-6234ed210f
bb","name":"Admin","displayName":"System
Administrator","memberKind":"LocalUserAccount"},"inheritedFrom":"5ae447bb-0738-404a-8159-b73dcb5a2c34"}],"effectiveCallerPrivileges"
:["1423fbbd-b6d9-429f-a9ea-d68a29b81cc9","cdad8747-3384-4aac-b101-f961671dcdbf","8c1786de-b6ff-4cce-bc09-f7b07e187896","e0d06d65-6ae
9-4ff2-ba4b-eb724487bda3","2a3562bb-4eb2-42ea-843a-6458332e28ac","0eaaa109-75f6-41cf-95ec-8c505de93921","f06aa0cb-2e0d-44d7-93e3-055
212dd6e07"],"metadata":[]},"isLegacyExport":true,"providerId":"9a15b21e-fa9e-4ece-8b31-9e12f1c6134a","parameterValues":[{"__classTyp
e":"dundas.data.SingleNumberValue","value":1,"parameterId":"63A56F50-6A01-4733-82AA-6F5531888B8D","isInverted":false,"parameterValue
Type":"SingleNumber"},{"__classType":"dundas.data.SingleStringValue","parameterId":"D1ABB1E7-6EFE-45CD-B11F-AB9D06532DDD","isInverte
d":false,"parameterValueType":"SingleString"},{"__classType":"dundas.data.SingleBooleanValue","value":false,"parameterId":"4ABB1086-
0857-427F-9D22-BAFB9ABE9EF4","isInverted":false,"parameterValueType":"SingleBoolean"}],"requests":[],"viewParameters":[{"__classType
":"dundas.view.ViewParameter","id":"486f1c82-5360-430d-bb74-4a1f4f88591d","name":"BrushViewParameter","friendlyName":"Brush View
Parameter","elementParameterLinks":[]}],"utcOffset":300,"adapterRequests":[]}
```


https://www.shorebreaksecurity.com/blog/ssrfs-up-real-world-server-side-request-forgery-ssrf/

Real world exploitation

Document Classification: PROTECTED

- Slack – SSRF in ‘Event Subscription’ parameter
  https://hackerone.com/reports/386292

- U.S. Department of Defense
  https://hackerone.com/reports/382048

- SEMursh – SSRF in ‘Get Video Contents’
  https://hackerone.com/reports/643622

- Omise – SSRF in ‘Webhooks’ leads to AWS Private keys disclosure
  https://hackerone.com/reports/508459

Document Classification: PROTECTED

How do I fix this?

Document Classification: PROTECTED

- Whitelist DNS name or IP addresses that your app needs to talk to.
- DO NOT trust user input and always perform input validation.
- Ensure that the response received by the remote server is indeed what the server is expecting.

Document Classification: PROTECTED

- If your application is only making use of HTTP or HTTPS to make requests, only allow those URL schemas.
- Disable unused URL schemes such as file:///, dict://, ftp:// and gopher://
- Remember there is NO universal “fix” to SSRF.

Document Classification: PROTECTED

Edge Side Include Injection (ESI)

Document Classification: PROTECTED

- Research by Louis Dion – Marcil and Philippe Arteau of GoSecure.

- ESI language is based on a small set of XML tags.

- It is used in many popular HTTP surrogate (Cache Servers & Reverse Proxies) solutions to tackle performance issues by enabling caching of Web content.

Document Classification: PROTECTED

- These XML tags instruct a reverse proxy or a caching server to fetch more information on a web page for which a template is already cached.
- This information may come from another server before it gets served to the client.
- This allows fully cached pages to include dynamic content.

Document Classification: PROTECTED

Weather Website

Weather For Auckland

Saturday 25°C 

Sunday 28°C 

Monday 27°C 

Cached Content

Dynamic Content

```
<body>
 <b>The Weather Website</b>
 Weather for <esi:include src="/weather/name?id=$(QUERY_STRING{city_id})" />
 Saturday: <esi:include src="/weather/week/saturday?id=$(QUERY_STRING{city_id})" />
 Sunday: <esi:include src="/weather/week/sunday?id=$(QUERY_STRING{city_id})" />
[...]
```

Document Classification: PROTECTED

What's the Vulnerability?

Document Classification: PROTECTED

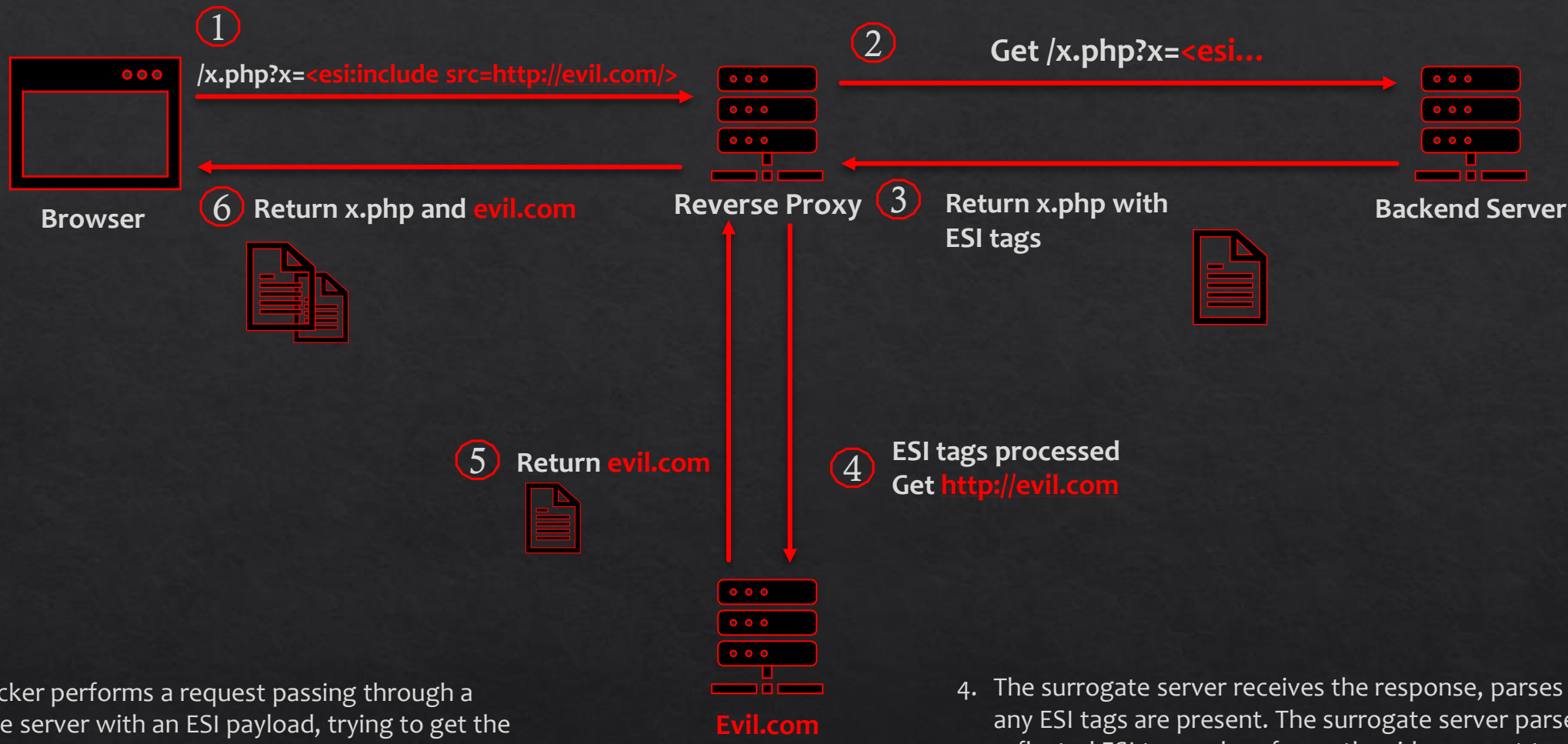
- When an ESI-capable surrogate parses non-sanitized user inputs, it can lead to ESI Injection.

- HTTP Surrogates are not able to distinguish between the legitimate ESI tags provided by the upstream server and malicious ones injected in the HTTP response.

- In other words, if an attacker can successfully reflect ESI tags in the HTTP response, then the surrogate will blindly parse and evaluate them, believing they are legitimate tags served from the upstream server.

Document Classification: PROTECTED

Document Classification: PROTECTED



1. The attacker performs a request passing through a surrogate server with an ESI payload, trying to get the backend server to reflect it in the response.
2. The surrogate server receives the request and forwards it to the appropriate backend server.
3. The application server reflects the ESI payload in the response, sends that response to the surrogate server.
4. The surrogate server receives the response, parses it to check if any ESI tags are present. The surrogate server parses the reflected ESI tag and performs the side-request to evil.com.
5. The surrogate server receives the side-request from evil.com and adds it to the initial response from the backend server.
6. The surrogate server sends the full response back to the client.

Document Classification: PROTECTED

Real world exploitation

Document Classification: PROTECTED

- ## Squid HTTP Caching Proxy

  CVE – 2018 – 1000024

  https://access.redhat.com/security/cve/cve-2018-1000024

  CVE – 2018 – 1000027

  https://access.redhat.com/security/cve/cve-2018-1000027

- ## Oracle Fusion Middleware

  CVE – 2019 – 2438

  https://nvd.nist.gov/vuln/detail/CVE-2019-2438

Document Classification: PROTECTED

How do I fix this?

Document Classification: PROTECTED

- Review the config of your reverse proxies.
- Your existing XSS related mitigations can help you here.
- Limit or disable what Content – Types are processed.
- Make sure your framework does proper XML/HTML encoding/escaping.
- You can partially fix this by whitelisting your hosts and domains that can be reached by ESI Includes.

Document Classification: PROTECTED

\Path\ Confusion: Web Cache Deception Attack

Document Classification: PROTECTED

- First documented in 2017 by security researcher Omer Gil.

- It is a type of attack that forces caching servers to store and reveal sensitive user information.

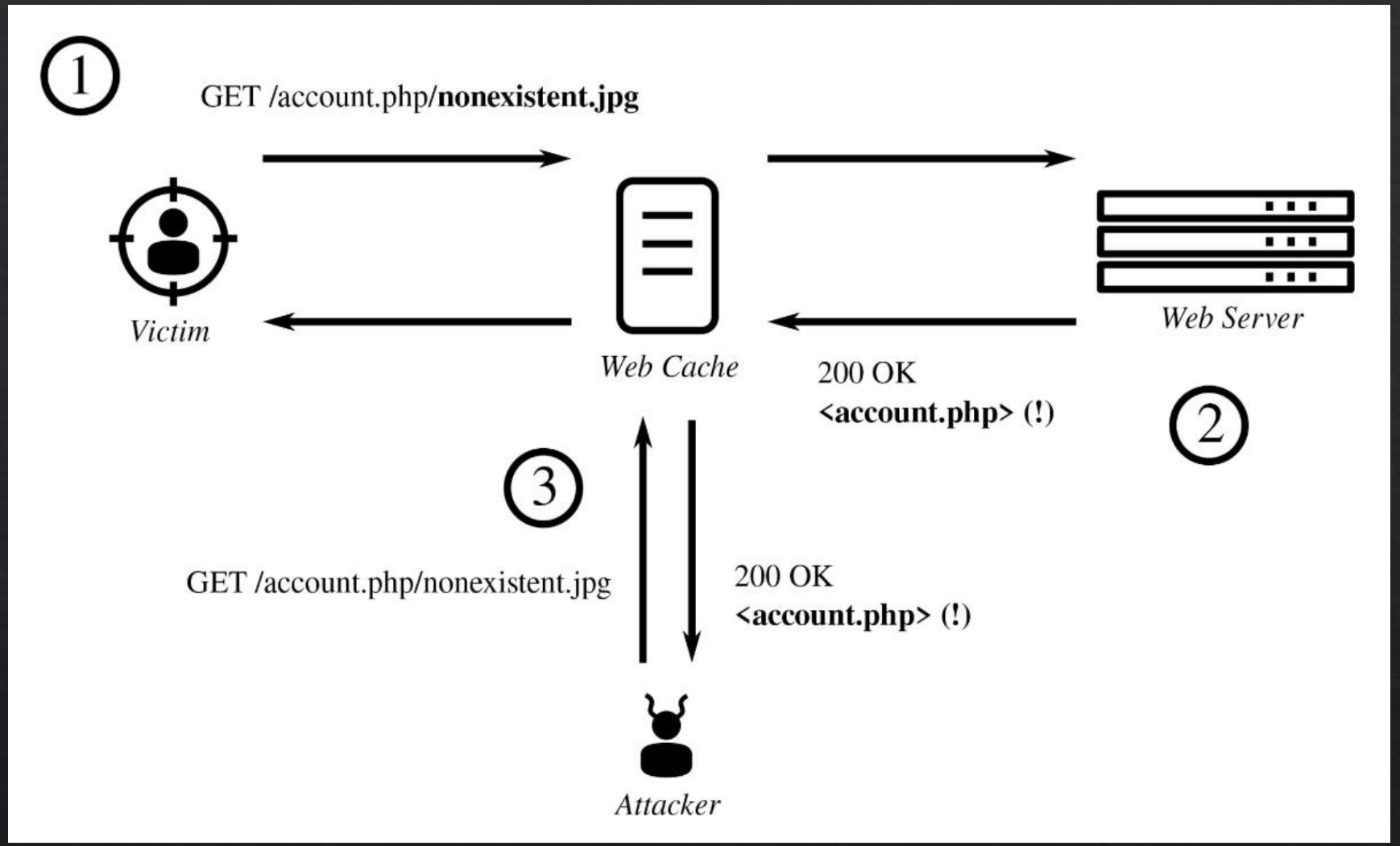
- The attack hinges on "path confusion" – manipulating URL paths to confound the cache server into classifying sensitive HTTP responses as public, cacheable documents.

Document Classification: PROTECTED

Let’s take a look…

Document Classification: PROTECTED

1. For instance, consider a dynamic, non-cacheable page that contains sensitive user account information, say /account.php.
2. To get the Caching server to store a cached copy of the page, a malicious user might add a suffix to the path to make it look like a static, public asset. For e.g. /account.php/nonexistent.jpg
3. If the victim visits the attacker's website, the attacker can then manipulate the victim's browser into loading a specific URL on the target website, which will cause their account data to be cached on the Caching Server.
4. After that, all the attacker has to do is send a GET request for the forged URL to receive a copy of the cached data.



Document Classification: PROTECTED

Real world exploitation

Document Classification: PROTECTED

- **Original Research by Omer Gil**
  http://omergil.blogspot.com/2017/02/web-cache-deception-attack.html

- **Postmates – User information disclosure**
  https://hackerone.com/reports/492841

- **Chaturbate – Exposed Token Information**
  https://hackerone.com/reports/397508/

- **Vanilla – Access to user's private messages**
  https://hackerone.com/reports/593712

Document Classification: PROTECTED

How do I fix this?

Document Classification: PROTECTED

- Configure the caching solution to only cache files if their HTTP caching headers allow this.
- Review the configuration of your caching solution and make sure that your origin and cache agrees with each other.
- Disable features that could result in confusion about the file extension between the proxy server and the origin server.

Document Classification: PROTECTED

- Configure the web server so that for pages such as *https://non-existentsite.com/home.php/non-existent.css*, the web server doesn't return the content of "home.php" with this URL. Instead, for example, the server should respond with a 404 or 302 response.

- There is no silver bullet solution to this attack since the caching configuration varies from site to site.

- Test the bugs outta this thing prior to deploying it in production.

Document Classification: PROTECTED

GraphQL

Document Classification: PROTECTED

What is this now?

Document Classification: PROTECTED

- Created by Facebook for internal use back in 2012.
- They Open - Source it in 2015.
- It is a Data Query Language (DQL) for your APIs.
- One of its key feature is to provide more efficiency compared to REST architectures.

Document Classification: PROTECTED

- Let's say for example you want to find a User's name, their favorite post, and pull data from their Profile and Settings.

- With REST, you would need four separate requests to grab the information you need.

- With GraphQL, you make a single request, and the structured data allows you to pull all the pieces of data you need – how convenient.

Document Classification: PROTECTED

REST

GET www.example.com/users/user1/posts

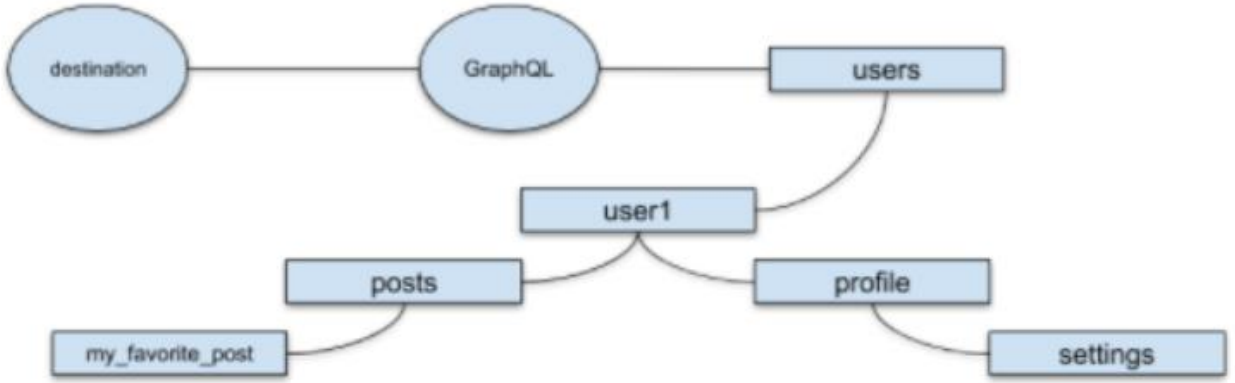
GET www.example.com/users/user1/posts/my_favorite_post

GET www.example.com/users/user1/profile

GET www.example.com/users/user1/profile/settings

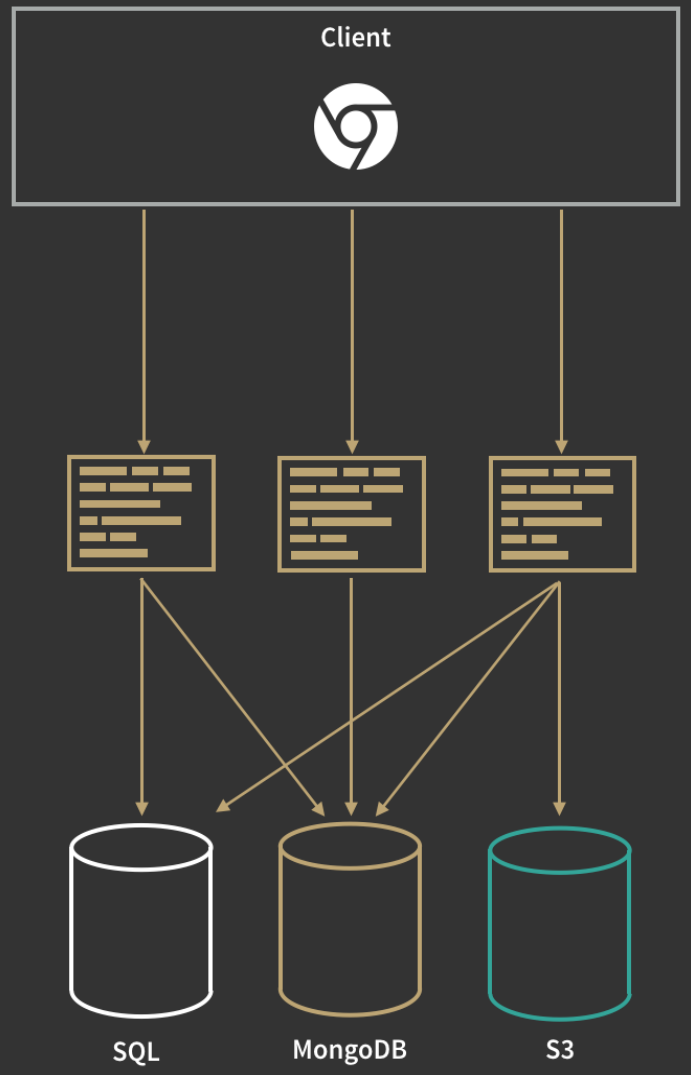
GraphQL

GET www.example.com/graphql

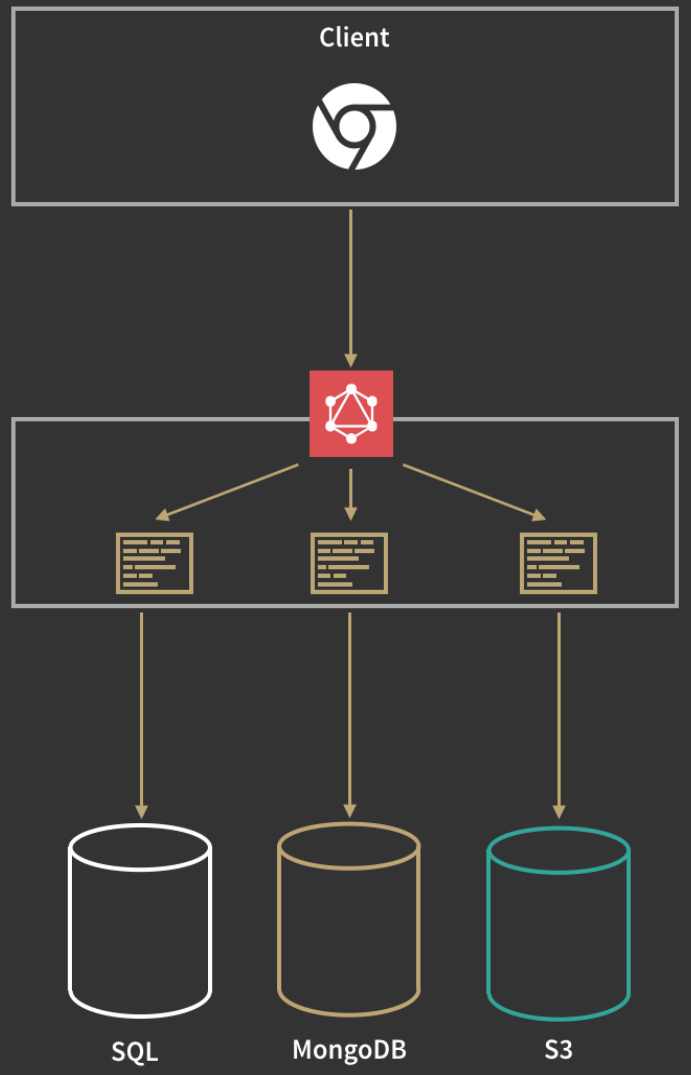


https://labs.detectify.com/2018/03/14/graphql-abuse/

REST



GRAPHQL



= Arbitrary code

http://bearcatjs.org/graphql-versus-rest-api/

Document Classification: PROTECTED

Request

```
query {
    allUsers {
        id
        name
        createdAt
        email
  }
}
```

Response


```
{
  "data": {
    "allUsers": [
      {
        "id": "ck2cqgmny048h0126ijoctgdz",
        "name": "Matt LeBlanc",
        "createdAt": "2019-10-30T03:41:26.000Z",
        "email": "admin@gmail.com"
      },
      {
        "id": "ck2cqj2lm04h40102u8du1u92",
        "name": "Paul Rudd",
        "createdAt": "2019-10-30T03:43:20.000Z",
        "email": "admin1@gmail.com"
      },
      {
        "id": "ck2k7pk7f0yfj0167q5a7iyfr",
        "name": "Nguyet Tran #",
        "createdAt": "2019-11-04T09:18:40.000Z",
        "email": "nguyet@gmail.com"
      },
      {
        "id": "ck2ks14gy1pgi0155u83ek72f",
        "name": "Foo",
        "createdAt": "2019-11-04T18:47:31.000Z",
        "email": "s"
      },
```


Document Classification: PROTECTED

How can we Hackz this?

Document Classification: PROTECTED

- What's next when you have a GraphQL endpoint?
- Well there is a thing called **Introspection**.
- It allows you to query GraphQL **Schema** for information such as available queries, types, fields and mutations etc.
- In the presence of an interactive GraphQL endpoint, you can simply go to **Docs** section to get everything you need to construct a query.

Document Classification: PROTECTED

- If not, then this may come in handy ☺
  - Do it manually
  - Use GraphQL IDE (IDE for exploring GraphQL API's)
  - Or you can use a Python script written by Doyensec

Document Classification: PROTECTED

Vulns...

Document Classification: PROTECTED

- Broken Access Controls
- Insecure Direct Object Reference (IDOR)
- SQL/NoSQL Injections
- Parameter Smuggling
- Expensive to compute large nested queries
- Information Disclosure
- Exposing GraphQL implementation details

Document Classification: PROTECTED

IDOR

Query

```
query{
        user(id: 165274){
                id
                email
                firstName
                lastName
                password
        }
}
```

Response

```
{
        "data": {
                "user": {
                        "id": "165274",
                        "email": "johndoe@mail.com",
                        "firstName": "John",
                        "lastName": "Doe"
                        "password": "5F4DCC3B5AA765D61D8327DEB882CF99"
                }
        }
}
```

https://blog.doyensec.com/2018/05/17/graphql-security-overview.html

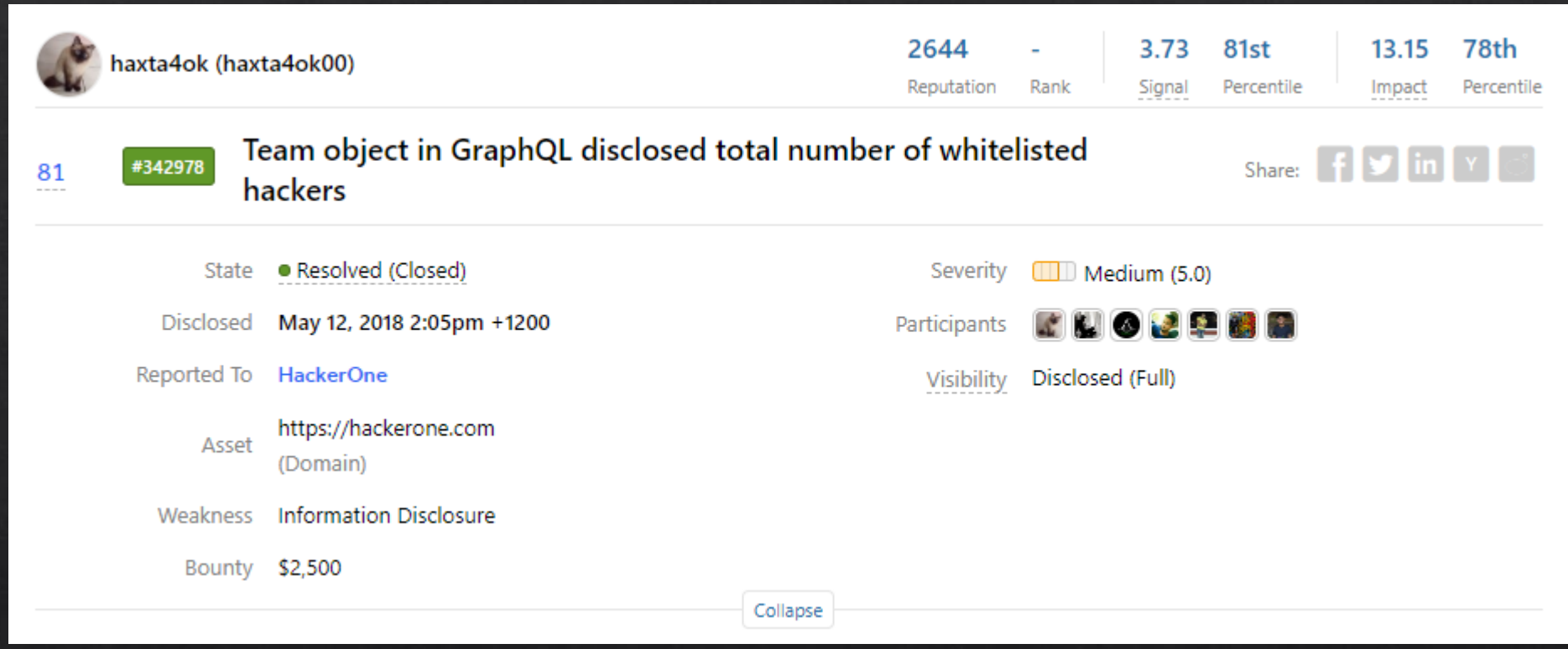
Nested Queries

```
query {
 stories{
  title
  body
  comments{
   comment
   author{
    comments{
     author{
      comments{
       comment
       author{
        comments{
         comment
         author{
          comments{
           comment
           author{
            name
           }
          }
         }
        }
       }
      }
     }
    }
   }
  }
 }
}
```

https://blog.doyensec.com/2018/05/17/graphql-security-overview.html

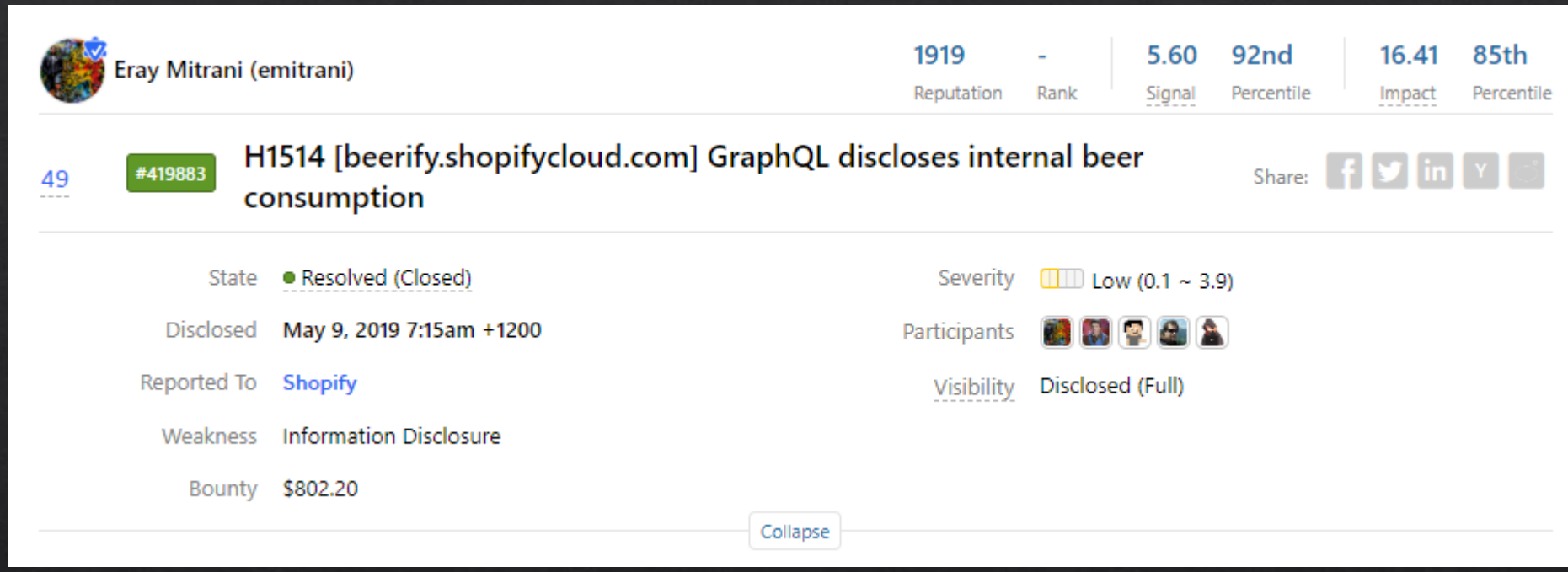
Real world exploitation

Document Classification: PROTECTED



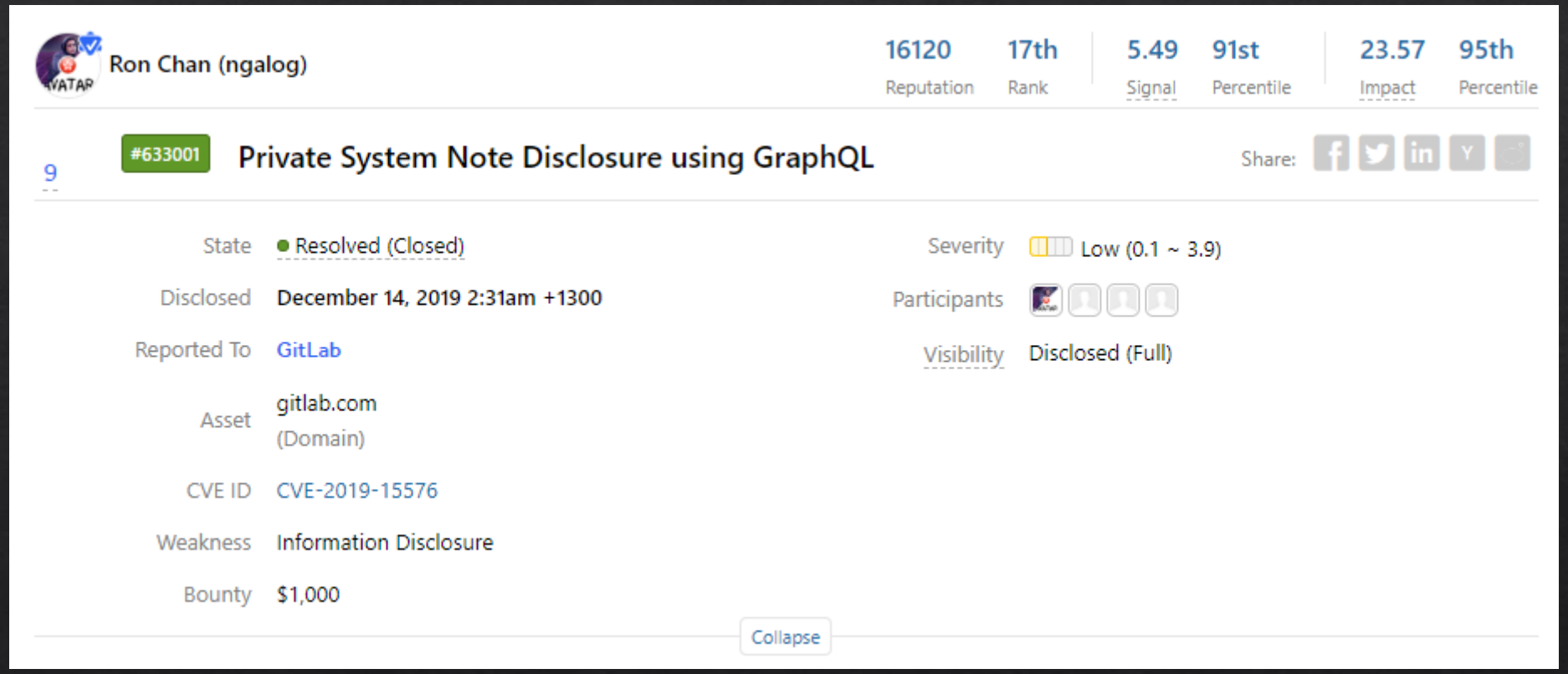
https://hackerone.com/reports/342978

Document Classification: PROTECTED



https://hackerone.com/reports/419883

Document Classification: PROTECTED



https://hackerone.com/reports/633001

Document Classification: PROTECTED

Best Practices

Document Classification: PROTECTED

- Rate limiting
- Disable Introspection in staging and production
- Depth limiting
- Infer input from the user's session where you can
- Query cost limitations

Document Classification: PROTECTED

Resources:

https://medium.com/swlh/protecting-your-graphql-api-from-security-vulnerabilities-e8afdfa6fbe4

https://medium.com/@localh0t/discovering-graphql-endpoints-and-sqli-vulnerabilities-5d39f26cea2e

https://labs.detectify.com/2018/03/14/graphql-abuse/

https://graphql.org/learn/introspection/

https://labs.detectify.com/2018/03/14/graphql-abuse/

https://blog.doyensec.com/2018/05/17/graphql-security-overview.html

https://lucasconstantino.github.io/graphiql-online/

Document Classification: PROTECTED

Thank You

Follow me: @_ROOT_

Document Classification: PROTECTED