

**Title:** Finding Vulnerable Telerik Instances with Scytode.py and Telerik\_RCE\_Scan.py

**Architecture:** NA

**Environment:** NIX Environment Preferred

**Dependencies:** python3

**Difficulty:** Easy

**Steps:**

**Application:** Identifying Reachable MS/IIS Targets and assessing them for Telerik RCE

**Competencies:**

Contents

Background .....	2
Targeted Enumeration .....	4
Execution:.....	5
Vulnerability Identification Algorithm: .....	2
Unit Testing .....	8
Unit Test Summary.....	8

## Background

Telerik UI for ASP.NET AJAX contains a security vulnerability that if exploited exposes users to remote code execution. This vulnerability exists in Telerik's UI file handler RadAsyncUpload, which is the IOE (indicator of Exploitability) that the scanning tool is based upon.

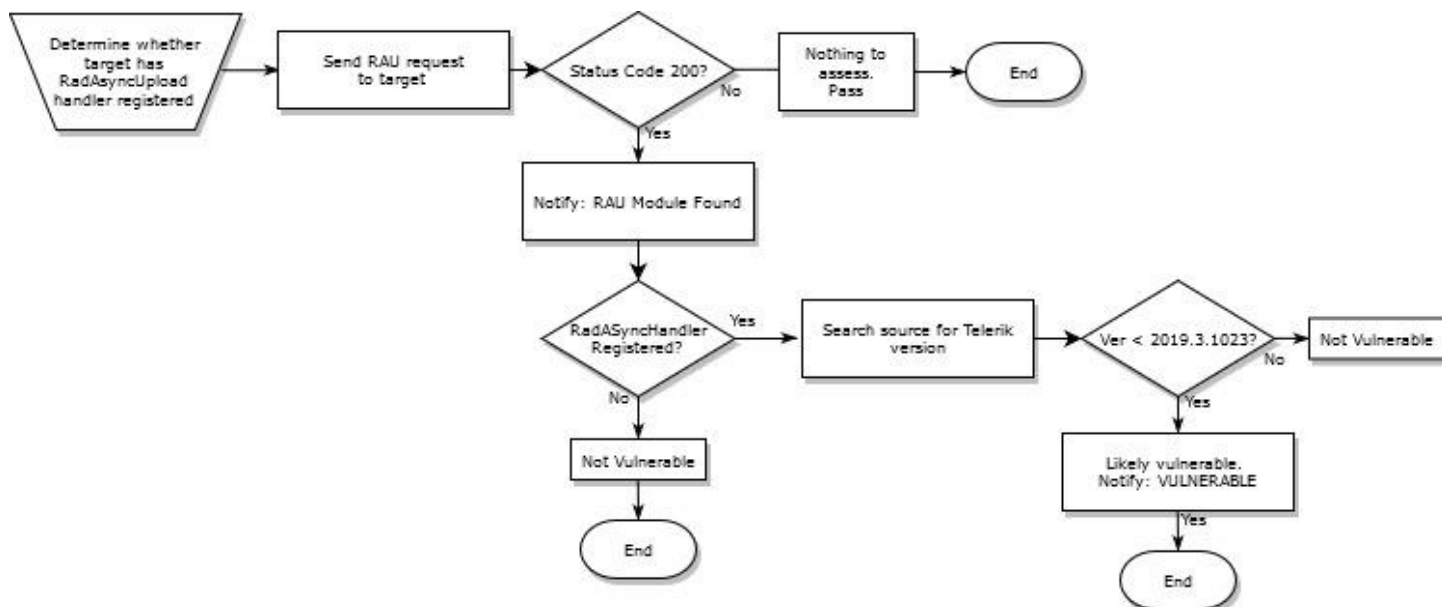
Telerik UI instances susceptible to exploitation must have the file handler registered, which can be confirmed by accessing the path to the RadAsyncUpload UI web resource:



The next step is to identify the Telerik ASP.NET AJAX version running on the target, as versions through 2019.3.1023 require a non-default setting to remediate the vulnerability. Simply put, finding an ASP.NET AJAX release prior to 2019.3.1023, indicates a high probability of exploitation.

It's important to note that this search and evaluate technique is completely unauthenticated. If the target in question does require authentication to identify its version of Telerik, there's a brute force method of guessing the correct version of Telerik in order to upload a file to a vulnerable instance. That technique is not covered in this demonstration.

## Vulnerability Identification Algorithm



## Scoping for Ranges

To target relevant systems, some DNS recon is in order. A no-nonsense approach to generating a solid list of targets, IP's, and interesting network ranges is to the DNS enumeration tool Fierce. Let's find some interesting Aetna and CVSCaremark ranges to assess:

### Aetna

```
$ fierce -dns aetna.com
```

```
Subnets found (may want to probe here using nmap or unicornscan):  
13.111.153.0-255 : 1 hostnames found.  
172.27.208.0-255 : 1 hostnames found.  
199.15.213.0-255 : 1 hostnames found.  
206.213.179.0-255 : 1 hostnames found.  
206.213.209.0-255 : 6 hostnames found.  
206.213.211.0-255 : 20 hostnames found.  
206.213.246.0-255 : 2 hostnames found.  
206.213.251.0-255 : 9 hostnames found.  
206.213.253.0-255 : 19 hostnames found.  
54.81.116.0-255 : 1 hostnames found.
```

### CVS

```
$ fierce -dns cvscaremark.com
```

### CVS Results:

```
Subnets found (may want to probe here using nmap or unicornscan):  
12.171.214.0-255 : 1 hostnames found.  
12.46.112.0-255 : 2 hostnames found.  
12.46.114.0-255 : 2 hostnames found.  
12.46.119.0-255 : 1 hostnames found.  
204.99.17.0-255 : 4 hostnames found.  
207.21.210.0-255 : 1 hostnames found.  
63.131.135.0-255 : 2 hostnames found.
```

This gives us something to work with. On the aetna side, we have the following network ranges:

```
13.111.153.0/24  
172.27.208.0/24  
199.15.213.0/24  
206.213.179.0/24  
206.213.209.0/24  
206.213.211.0/24  
206.213.246.0/24  
206.213.251.0/24
```

206.213.253.0/24

54.81.116.0/24

On the CVS side, we have the ranges below:

12.171.214.0/24

12.46.112.0/24

12.46.114.0/24

12.46.119.0/24

204.99.17.0/24

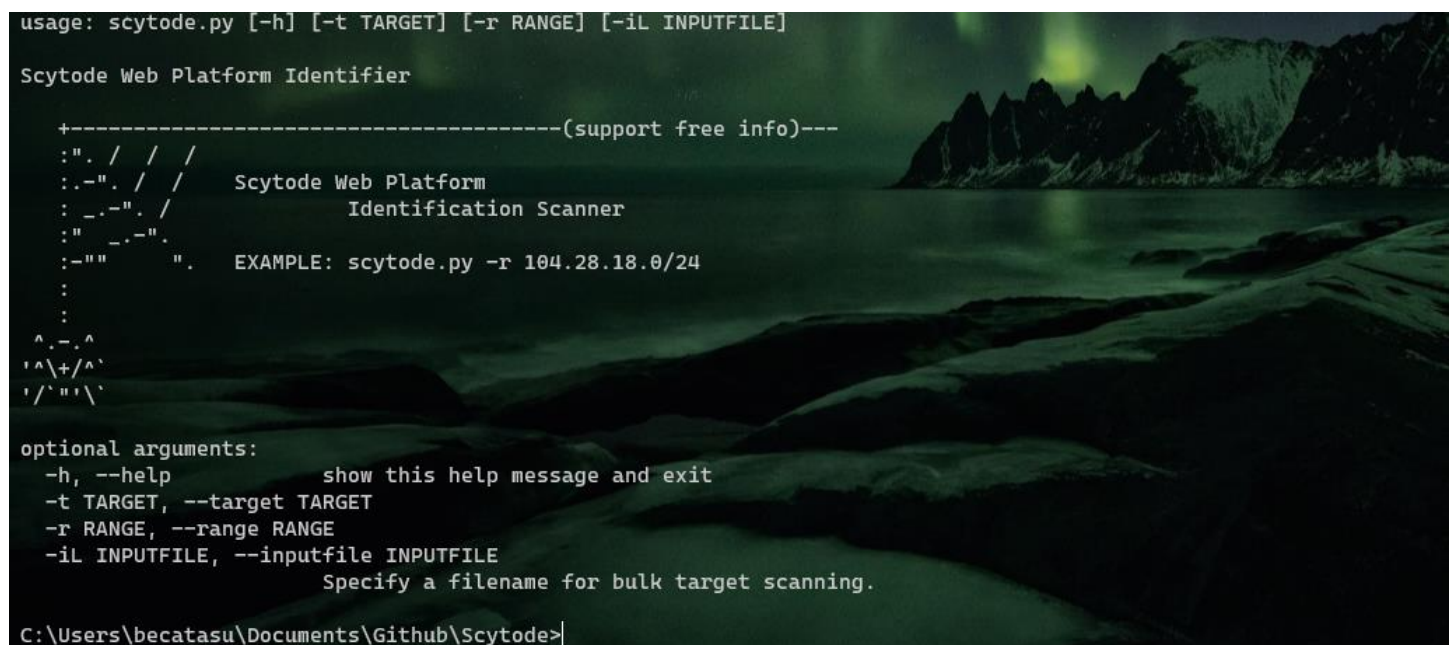
207.21.210.0/24

63.131.135.0/24

## Targeted Enumeration

Because Telerik UI instances run on ASP.NET AJAX, web servers running Microsoft IIS are the targets of interest. There are a lot of ways to identify Microsoft IIS web servers, but, I of course like to roll my own.

A *Scytodes Lugubris* is a species of spitting spider found in Tropical Asia, Hawaii and Mexico. Recently I became aware that they're now relatively common in Arizona as well, and I found the Genus *Scytodes* an appropriate name for my new scanner.



```
usage: scytode.py [-h] [-t TARGET] [-r RANGE] [-iL INPUTFILE]

Scytode Web Platform Identifier

+------(support free info)---
: ". / / /
: ". / / Scytode Web Platform
: ". / Identification Scanner
: " ".
: "- " " EXAMPLE: scytode.py -r 104.28.18.0/24
:
:
: ^.^
: ^.^
: /' "' \

optional arguments:
-h, --help show this help message and exit
-t TARGET, --target TARGET
-r RANGE, --range RANGE
-iL INPUTFILE, --inputfile INPUTFILE
Specify a filename for bulk target scanning.

C:\Users\becatasu\Documents\Github\Scytode>
```

Scytode evaluates a single target, a CIDR notation range, or a list of hosts via an input file for their respective server types. To find a list of Microsoft IIS targets, we can supply it with a network range and let it do its thing. It's a multi-threaded script, which speeds up the process from its original iterative approach, reducing the average time of completion to scan a C class subnet from 2 minutes to about 15 seconds.

After saving the identified Aetna network ranges to a file (aetna\_ext.txt), the file is used as an argument to scytode to scan, identify, and sort the findings by server type:

```
C:\Users\becatasu\Documents\Github\Scytode>type aetna_ext.txt
12.171.214.0/24
12.46.112.0/24
12.46.114.0/24
12.46.119.0/24
204.99.17.0/24
207.21.210.0/24
63.131.135.0/24
```

Execution:

```
$ python3 scytode.py -iL aetna_ext.txt
```

```
C:\Users\becatasu\Documents\Github\Scytode>python3 scytode.py -il aetna_ext.txt
Scytode Web Identifier ( github.com/becrevex/Telerik_CVE-2019-18935 )
```

```
+------(support free info)-----
: ". / / /
:.. ". / /      Scytode Web Platform
: _.- ". /      Identification Scanner
: " _.- "
: -" " " "      EXAMPLE: scytode.py -r 104.28.18.0/24
:
:
^ _ ^
r ^ \ + / ^ \
r / ^ " \ \
```

```
Starting platform identification scan at 2020-07-21 19:21:27.704737
```

```
12.171.214.30    [+] Server: Apache-Coyote/1.1
12.171.214.33    [+] Server: Apache-Coyote/1.1
12.171.214.31    [+] Server: Apache-Coyote/1.1
12.171.214.32    [+] Server: Apache-Coyote/1.1
12.46.112.39     [+] Server: Apache
12.46.112.180    [+] Server: Apache
12.46.112.183    [+] Server: Apache
12.46.114.71     [+] Server: BigIP
12.46.114.91     [+] Server: BigIP
12.46.114.166    [+] Server: Apache-Coyote/1.1
12.46.114.165    [+] Server: Apache-Coyote/1.1
12.46.114.167    [+] Server: Apache-Coyote/1.1
12.46.114.163    [+] Server: Apache
12.46.114.175    [+] Server: Apache-Coyote/1.1
12.46.114.211    [+] Server: Microsoft-IIS/8.5
12.46.114.229    [+] Server: Apache
12.46.114.237    [+] Server: Apache
12.46.114.242    [+] Server: Microsoft-IIS/8.5
```

All discovered server types are grouped and saved to their respective files in the `./targets/` directory.



```
C:\Users\becatasu\Documents\Github\Scytode\targets>dir
Volume in drive C is Windows
Volume Serial Number is 4AE6-EB12

Directory of C:\Users\becatasu\Documents\Github\Scytode\targets

07/21/2020  07:24 PM    <DIR>          .
07/21/2020  07:24 PM    <DIR>          ..
07/21/2020  07:24 PM                165 Apache-Coyote.txt
07/21/2020  07:24 PM                409 Apache.txt
07/21/2020  07:24 PM                 43 BigIP.txt
07/21/2020  07:24 PM                 15 JBoss-EAP.txt
07/21/2020  07:24 PM                208 Microsoft-IIS.txt
               5 File(s)                840 bytes
               2 Dir(s)  1,386,828,357,632 bytes free
```

This allows us to target just the Microsoft-IIS servers for the Telerik RCE vulnerability.

```
$ python3 telerik_rce_scan.py -iL ..\Scytode\targets\Microsoft-IIS.txt
```

```
C:\Users\becatasu\Documents\Github\Telerik_CVE-2019-18935>python3 telerik_rce_scan.py -iL ..\Scytode\targets\Microsoft-IIS.txt
TelerikUI RCE Scan ( github.com/becrevex/Telerik_CVE-2019-18935 )
EXAMPLE: telerik_rce_check -t 104.28.18.139
Starting CVE-2019-18935 (Telerik RCE) scan at 2020-07-21 19:26:40.811511

Checking: 12.46.114.211
Telerik RAU Module not found on 12.46.114.211. Status (404)
Telerik RAU Module not found on 12.46.114.211. Status (404)
Checking: 12.46.114.242
Telerik RAU Module not found on 12.46.114.242. Status (404)
Telerik RAU Module not found on 12.46.114.242. Status (404)
Checking: 12.46.114.243
Telerik RAU Module not found on 12.46.114.243. Status (404)
Telerik RAU Module not found on 12.46.114.243. Status (404)
Checking: 12.46.119.241
Telerik RAU Module not found on 12.46.119.241. Status (404)
Checking: 12.46.119.232
Telerik RAU Module not found on 12.46.119.232. Status (404)
Checking: 204.99.17.100
Telerik RAU Module not found on 204.99.17.100. Status (404)
Telerik RAU Module not found on 204.99.17.100. Status (404)
Checking: 204.99.17.150
Telerik RAU Module not found on 204.99.17.150. Status (404)
Telerik RAU Module not found on 204.99.17.150. Status (404)
Checking: 204.99.17.122
Telerik RAU Module not found on 204.99.17.122. Status (404)
Telerik RAU Module not found on 204.99.17.122. Status (404)
Checking: 204.99.17.98
Checking: 204.99.17.97
Checking: 204.99.17.184
Telerik RAU Module not found on 204.99.17.184. Status (404)
Telerik RAU Module not found on 204.99.17.184. Status (404)
Checking: 204.99.17.180
Telerik RAU Module not found on 204.99.17.180. Status (404)
Telerik RAU Module not found on 204.99.17.180. Status (404)
Checking: 207.21.210.67
Telerik RAU Module not found on 207.21.210.67. Status (404)
Telerik RAU Module not found on 207.21.210.67. Status (404)
Checking: 63.131.135.18
```

And it looks like the Aetna Network Ranges assessed are clean.

To show the identification of a vulnerable instance, some GoogleFu was used to ID URL's in the wild that contain the term Telerik.

- [www.kha-net.org](http://www.kha-net.org)
- [www.cityofbeaufort.org/common/admin/Jobs2/](http://www.cityofbeaufort.org/common/admin/Jobs2/)
- [www.cityofjackson.org](http://www.cityofjackson.org)
- [www.townoffarragut.org](http://www.townoffarragut.org) [208.90.188.135] (208.90.188.0/24)

As mentioned above, the scanner can be used to target one instance:

```
$ python3 telerik_rce_scan.py -t www.kha-net.org
```

```
C:\Users\becatasu\Documents\Github\Telerik_CVE-2019-18935>python3 telerik_rce_scan.py -t www.kha-net.org
TelerikUI RCE Scan ( github.com/becrevex/Telerik_CVE-2019-18935 )
EXAMPLE: telerik_rce_check -t 104.28.18.139
Starting CVE-2019-18935 (Telerik RCE) scan at 2020-07-21 19:31:34.032091

Checking: www.kha-net.org
[!] RAU Module Found [ www.kha-net.org/Telerik.Web.UI.WebResource.axd?type=rau ]!
Checking target for vulnerable versions...
Identified version matches:
2018.2.516.45
[!] VULNERABLE

Risk: HIGH          CVE: CVE-2019-18935          CWE: CWE-913
-----
Telerik UI for ASP.NET AJAX through 2019.3.1023 contains a
.NET deserialization RCE vulnerability in the RadAsyncUpload function.

-- As of 2020.1.114, a default setting prevents the exploit.
-- In 2019.3.1023, but not earlier versions, a non-default
-- setting can prevent exploitation.

-- This is exploitable when the encryption keys are known due
-- to the presence of CVE-2017-11317 or CVE-2017-11357.
-- Exploitation can result in remote code execution.
```

CIDR notation ranges:

```
$ python3 telerik_rce_scan.py -r 208.90.188.0/24
```

```
C:\Users\becatasu\Documents\Github\Telerik_CVE-2019-18935>python3 telerik_rce_scan.py -r 208.90.188.0/24
TelerikUI RCE Scan ( github.com/becrevex/Telerik_CVE-2019-18935 )
EXAMPLE: telerik_rce_check -t 104.28.18.139
Starting CVE-2019-18935 (Telerik RCE) scan at 2020-07-21 19:32:41.989325

Running range scan...
Checking: 208.90.188.0
Checking: 208.90.188.1
Telerik RAU Module not found on 208.90.188.1. Status (403)
Telerik RAU Module not found on 208.90.188.1. Status (403)
Checking: 208.90.188.2
Checking: 208.90.188.3
Checking: 208.90.188.4
Checking: 208.90.188.5
```

## Unit Testing

Unit tests will address the core functionality of each function, ensuring the features of the tool perform as expected, generate accurate data points, and handle exceptions gracefully.

Use cases to be tested:

- Single Target Scan of likely vulnerable IP
- Single Target Scan of likely vulnerable Hostname
- Single Target Scan of unresolvable hostname
- Single Target Scan of invalid IP address
- Target Range Scan of IPs
- Target Range Scan of Hostnames via Input File
- Target Range Scan of IP's via Input File

## Unit Test Summary

The testing performed assessed 11 individual controller modules, each containing various features and functions that stood to execute one primary function of the application. Of the 11 cases assessed, 4 did not pass the unit tests performed.

Use Case / Module	Test Case ID	Pass/Fail	Summary of Defect	Comments
Single Target Scan of LV IP check_vuln(target)	Case001	Pass	NA	None

Use Case / Module	Test Case ID	Pass/Fail	Summary of Defect	Comments
Single Target Scan of LV Hostname check_vuln(target)	Case002	Pass	NA	None

Use Case / Module	Test Case ID	Pass/Fail	Summary of Defect	Comments
Single Target Scan of Unresolvable Hostname check_vuln(target)	Case003	Pass	NA	More specific exception handling can be applied to provide information in the event of NewConnectionError, Timeout, or SSL issue.

Use Case / Module	Test Case ID	Pass/Fail	Summary of Defect	Comments
Single Target Scan of invalid IP check_vuln(target)	Case004	Pass	NA	Fails gracefully, but more context can be provided to illustrate why.

Use Case / Module	Test Case ID	Pass/Fail	Summary of Defect	Comments
Target Range Scan of IPs check_vuln(target)	Case005	Pass	NA	None

Implement error handling when providing wrong argument types –especially to ranges.



Use Case / Module	Test Case ID	Pass/Fail	Summary of Defect	Comments
Single Target Scan of Hostnames via File	Case006	Pass Fail		

Use Case / Module	Test Case ID	Pass/Fail	Summary of Defect	Comments
Single Target Scan of IP via File	Case007	Pass		

----- Appendices -----