User Creation Password Signin Investigation

Author: Chris Morales

Summary: Wazuh provides a huge help in trying to locate the root cause of an incident while providing key information about the systems. In this scenario, an investigation was launched to find the root cause of an additional user being created on a machine when no other users on the machine would have need of it. Through a series of steps that built upon each other, it was denoted that the root cause was a known password that allowed the attacker to gain access to an account that had sudo privileges that could create an account on the machine.

Investigation

Initial Behavior

This is the problematic log that was produced from our custom Wazuh dashboard that looks for the useradd command.

```
> Mar 7, 2023 @ 18:37:04.416  j-agent1  192.168.0.118  New user added to the system.  Mar  7 18:37:03 j-agent1 useradd[3241861]: new user: name=test-chris, UID=1003, GID=1003,
home=/home/test-chris, shell=/bin/bash, from=/dev/pts/3
```

Here, you can get some crucial information needed for the remediation and the incident response that should be written for this.

1. You notice that this is from the *j-agent1* agent that's registered with the Wazuh server.
2. Next, you notice that the timestamp. This could be the first time that you noticed this action or it could be used as the last time that you noticed an attacker coming in.
3. In the full log on the right, you can see information regarding the user that was created (in terms of UID, GID, login shell, etc).
4. You'll notice that the full log on the right hand side houses the first major investigative note for us; this is the "*from=/dev/pts/3*" field.

Defensive Investigation / Commands

This would indicate to us that this command was run from a shell by a remote user. The only thing that can be stored in the /dev/pts directory are pseudo-terminals that are derived from applications such as xterm or ssh (reference here). And so, we can log into the affected machine and look at the shells that are active.

```
user@j-agent1:~$ who -a
          system boot  2023-02-28 19:11
          run-level 5  2023-02-28 19:11
LOGIN     hvc0         2023-02-28 19:11              1144 id=hvc0
          pts/0        2023-03-01 14:39           1651463 id=ts/0  term=0 exit=0
user     + pts/1        2023-03-07 19:18   .       3304428 (192.168.7.10)
user     + pts/2        2023-03-07 18:36 00:20     3240959 (192.168.7.10)
user     + tty2         2023-03-07 10:58  old      2500049 (tty2)
          pts/3        2023-03-07 18:40           3244681 id=ts/3  term=0 exit=0
          pts/4        2023-03-07 18:40           3244958 id=ts/4  term=0 exit=1
user@j-agent1:~$
```

Here, we can see that there are a few shell sessions that are connected. In this screenshot, you can see that *pts/3* is here. Now, we can see that this is under the pts/2 pseudo-terminal (ignore tty2, this is the actual console on the machine.). And so, *pts/2* is now what we're looking for. You can also look at the PID (the column housing 3244681) and see that the PID is relatively close to it. So it's safe to assume that this pts came from the other pts.

Let's confirm where this shell came from with:

```
ps -wauxf | grep -a4 -b4 ssh
```

```
user@j-agent1:~$ ps -wauxf | grep -a4 -b4 ssh
8541-root         509  0.0  0.2 392912  8980 ?        Ssl  Feb28   0:00 /usr/libexec/udisks2/udisksd
8637-root         510  0.0  0.0  16492  2968 ?        Ss   Feb28   0:02 /sbin/wpa_supplicant -u -s -O /run/wpa_supplicant
8754-root         581  0.0  0.1 317040  6948 ?        Ssl  Feb28   0:00 /usr/sbin/ModemManager
8844-root         614  0.0  0.2 126788  9208 ?        Ssl  Feb28   0:00 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-upg
rade-shutdown --wait-for-signal
9005:root         618  0.0  0.1  15420  4476 ?        Ss   Feb28   0:00 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
9128:root       38046  0.0  0.1  17288  4620 ?        Ss   Feb28   0:00  \_ sshd: jenkins [priv]
9220:jenkins    38161  0.0  0.0  17720  3644 ?        S    Feb28   0:01  |   \_ sshd: jenkins@notty
9315-jenkins    38247  0.5  1.8 3605180 75456 ?       Ssl  Feb28  54:31  |       \_ java -jar remoting.jar -workDir /home/jenkins -jar
-cache /home/jenkins/remoting/jarCache
9483:root     3240959  0.0  0.2  17448 11028 ?        Ss   18:36   0:00  \_ sshd: user [priv]
9572:user     3241147  0.0  0.1  17448  7952 ?        S    18:36   0:00  |   \_ sshd: user@pts/2
9664-user     3241148  0.0  0.1  19912  5380 pts/2    Ss+  18:36   0:00  |       \_ -bash
9749:root     3304428  0.0  0.2  17452 11040 ?        Ss   19:17   0:00  \_ sshd: user [priv]
9838:user     3304579  0.0  0.2  17452  8040 ?        S    19:18   0:00      \_ sshd: user@pts/1
9930-user     3304580  0.0  0.1  19912  5412 pts/1    Ss   19:18   0:00          \_ -bash
10015-user    3327690  0.0  0.1  21996  4084 pts/1    R+   19:33   0:00              \_ ps -wauxf
10108:user    3327691  0.0  0.0  17732  2368 pts/1    S+   19:33   0:00              \_ grep --color=auto -a4 -b4 ssh
10221-root        620  0.0  0.1 249984  7060 ?        Ssl  Feb28   0:00 /usr/sbin/gdm3
10303-root    2499884  0.0  0.2 179356  9296 ?        Sl   10:58   0:00  \_ gdm-session-worker [pam/gdm-password]
10412-user    2500049  0.0  0.1 171036  5592 tty2     Ssl+ 10:58   0:00      \_ /usr/libexec/gdm-wayland-session env GNOME_SHELL_SESS
ION_MODE=ubuntu /usr/bin/gnome-session --session=ubuntu
10596-user    2500059  0.0  0.2 231684  8140 tty2     Sl+  10:58   0:00          \_ /usr/libexec/gnome-session-binary --session=ubunt
u
```

Notice that the pts/2 is attached to the *user* account which has PID 3241147. Now, we can take a remediation step and kill that session that the attacker still has.

```
sudo kill -9 3241147
```

```
user@j-agent1:~$ who -a
           system boot  2023-02-28 19:11
           run-level 5  2023-02-28 19:11
LOGIN      hvc0         2023-02-28 19:11              1144 id=hvc0
           pts/0        2023-03-01 14:39           1651463 id=ts/0  term=0 exit=0
user     + pts/1        2023-03-07 19:18   .       3304428 (192.168.7.10)
user     + pts/2        2023-03-07 18:36 00:25     3240959 (192.168.7.10)
user     + tty2         2023-03-07 10:58  old      2500049 (tty2)
           pts/3        2023-03-07 18:40           3244681 id=ts/3  term=0 exit=0
           pts/4        2023-03-07 18:40           3244958 id=ts/4  term=0 exit=1
user@j-agent1:~$ sudo kill -9 3241147
[sudo] password for user:
user@j-agent1:~$
```

In the image above, you can see that we ran this same command to kill the session.

```
user@j-agent1:~$ Connection to 192.168.0.118 closed by remote host.
Connection to 192.168.0.118 closed.
```

On the attacker side, we can see that the connection was closed. And once this is done, the attacker's session will have to be redone. It buys the blue team some time. But it also helps the SOC team look for a persistent

authentication ssh method. This will help find an avenue that an attacker can get into the machine by.

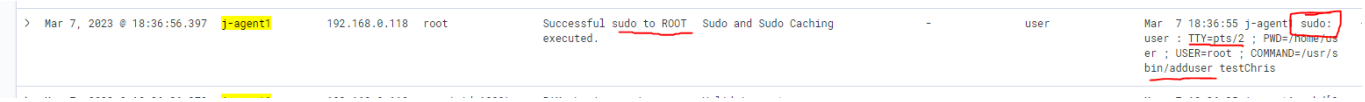
Retracing Steps

Now that we have bought ourselves some time by kicking out the attacker, we can now see how they got in the first place.

It is possible that the attacker has an authentication method but doesn't want to re run it immediately because they now know that we'll be on the lookout for their *in*. So, we need to find out what that original *in* was.

Looking for sudo command in Wazuh

Remember, the session that we killed belonged to the *user* account. The only way to create a new session is to have *elevated privileges* through the "sudo" command with a *non-root* account. And so, we can look for the command that needed sudo to execute. Very likely something along the lines of "*sudo adduser*". Let's look for a log in Wazuh that supports our theory.



In this screenshot, you'll see a few key things:

1. The sudo command was utilized.
2. pts/2 was used to carry this out.
3. The adduser command was attached to this sudo command

These are all indicators of *how* an attack was carried out. There are two possibilites now on how the attacker was able to elevate privileges using sudo:

1. Entering the password upon the prompt.
2. The *user* account was granted access to use sudo without a password. This access is typically given in the sudoers file.

Looking through sudoers

Quickly looking through the sudoers file, there was nothing there (in this case).

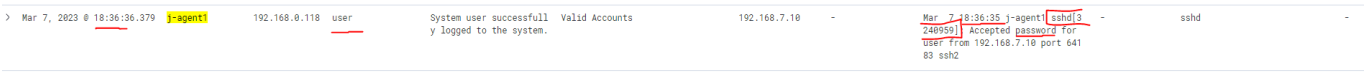
Now, we can pivot to looking at this session came to be. And so, we need to find out when a particular authentication for *user* had occurred.

Finding Authentication Successful in Wazuh

Remember the old `who -a` command that we had run before? It had a key piece of information that we can use to help us digest the logs in Wazuh.

```
user@j-agent1:~$ who -a
           system boot  2023-02-28 19:11
           run-level 5  2023-02-28 19:11
LOGIN      hvc0         2023-02-28 19:11              1144 id=hvc0
           pts/0        2023-03-01 14:39           1651463 id=ts/0  term=0 exit=0
user     + pts/1        2023-03-07 19:18    .      3304428 (192.168.7.10)
user     + pts/2        2023-03-07 18:36 00:25     3240959 (192.168.7.10)
user     + tty2         2023-03-07 10:58  old      2500049 (tty2)
           pts/3        2023-03-07 18:40           3244681 id=ts/3  term=0 exit=0
           pts/4        2023-03-07 18:40           3244958 id=ts/4  term=0 exit=1
```

The key piece of information is the *timestamp*. Let's use this to look at the Wazuh log that matches with this.



We have found a log that looks like it. Let's confirm that the information matches what we know.

1. The timestamp in the log is similar if not the same as the SSH session under the `who -a` command - Yes.
2. The account that was targeted was *user* - Yes.
3. The agent host was *j-agent1*. Yes.
4. In the full log on the right, we can see that the PID that was given to the session was 3240959. This matches with the PID contained in the `who -a` output. This also matches the `ps -wauxf` command from earlier if you match the PID with the *root* SSH process that houses pts/2.

This log showcases that the method in which they had logged into the system was **via a password.**

Putting the story together

Now, that we have all the pieces, we can come up with a timeline of what happened here.

And so, for an executive summary:

On March 7, 2023 @ 18:36:36:379, a malicious party had gained access to one of our machines via a known password. Once inside, the attacker had created another user account (likely to setup persistent access to this machine). Our SOC analysts noticed this and swiftly removed the attacker from our system and removed this user. Upon further analysis of the breach, we do not see any signs of any data exfiltration or modification on our systems. We have contacted the administrator of the machine to immediately change the password of the machine to prevent this from happening. We have not seen any additional traffic from this attacker after March 7, 2023 @ 18:37:04.416.