Bug 1202931 - (CVE-2022-31253) VUL-0: CVE-2022-31253: openldap2: /usr/lib/openldap/start allows ldap user/group to recursively chown arbitrary directory trees to itself

Status:	IN_PROGRESS
Classification:	openSUSE
Product:	openSUSE Tumbleweed
Component:	Security
Version:	Current
Hardware:	Other Other
Priority:	P3 - Medium **Severity**: Normal (vote)
Target Milestone:	---
Assigned To:	Security Team bot
QA Contact:	E-mail List
URL:	https://smash.suse.de/issue/341147/
Whiteboard:	
Keywords:	
Depends on:	
Blocks:	
	Show dependency tree / graph

- Create test case
- Clone This Bug

Reported:	2022-08-30 13:31 UTC by Matthias Gerstner
Modified:	2022-10-27 09:15 UTC (History)
CC List:	4 users (show)
See Also:	
Found By:	---
Services Priority:	
Business Priority:	
Blocker:	---

Attachments

Add an attachment (proposed patch, testcase, etc.)

Note

You need to log in before you can comment on or make changes to this bug.

Matthias Gerstner 2022-08-30 13:31:00 UTC Description

```
+++ This bug was initially created as a clone of ~~Bug #1191283~~

openldap2-2.6.3-1.1.x86_64.rpm
==============================

/usr/lib/systemd/system/slapd.service
-------------------------------------
    credentials: root:root
    ExecStart=/usr/lib/openldap/start
```

```
runs start script as root: there's a lot of happy recursive chown()'ing and
chgrp()'ing that might allow for a symlink attack below /etc/openldap/slap.d,
see chown_database_dir*
```

Matthias Gerstner 2022-09-23 09:26:02 UTC Comment 1

```
I have reviewed the 'start' script and found a local root escalation
vulnerability. There is by default no classical symlink attack in this slapd
"start" script but something even easier to exploit.

1) chown_database_dirs()
========================

This function has no vulnerability by default. This section is only for
documentation purposes.

`chown_database_dirs()` extracts paths from /etc/openldap/slapd.conf (only
writable by root), specified via the 'directory' and 'include' config keys. In
a default openldap2 installation these are:

directory /var/lib/ldap
include   /etc/openldap/schema/core.schema
include   /etc/openldap/schema/cosine.schema
include   /etc/openldap/schema/inetorgperson.schema
include   /etc/openldap/schema/rfc2307bis.schema
include   /etc/openldap/schema/yast.schema

The 'include' files are in turn checked for 'directory' config keys. There are
non in these default schemas. In practice this means by default
`chown_database_dirs()` will execute:

    chown -R ldap /var/lib/ldap
    chgrp -R ldap /var/lib/ldap

This is okay since chown/chgrp recursive operations are safe against race
conditions and symlink attacks in their current versions. The root directory
/var/lib/ldap also cannot be replaced by a symlink by the ldap user or group.

2) chown_database_dirs_bconfig()
================================

This function is only executed if either

- OPENLDAP_CONFIG_BACKEND is set to ldap in the sysconfig file
 or
- /etc/openldap/slapd.conf is missing but /etc/openldap/slap.d is existing

This means by default it will not be executed but only if the openldap2
configuration is changed accordingly by an admin.

If one of the conditions is fulfilled then the following code will run:

```
chown_database_dirs_bconfig "/etc/openldap/slapd.d"

function chown_database_dirs_bconfig() {
        ldapdir=$(find $1 -type f -name "olcDatabase*" | xargs grep -i
olcdbdirectory | awk '{print $2}')
        for dir in $ldapdir; do
                [ -d "$dir" ] && [ -n "$OPENLDAP_USER" ] && \
                        chown -R $OPENLDAP_USER $dir 2>/dev/null
                [ -d "$dir" ] && [ -n "$OPENLDAP_GROUP" ] && \
                        chgrp -R $OPENLDAP_GROUP $dir 2>/dev/null
        done
}
```

The directory /etc/openldap/slapd.d is owned by ldap:ldap mode 770. This means
both the ldap user and group can write in there. The code above extracts
configuration key 'olcdbdirectory' from any files named 'olcDatabase*' within
this directory. Thus it is very easy to exploit:
    root# sudo -u ldap -g ldap bash -c "echo 'olcdbdirectory /root'
>/etc/openldap/slapd.d/olcDatabase"
    root# systemctl start slapd
    [...]
    root# ls -lhd /root
    drwx------ 9 ldap ldap 4.0K Sep 23 10:34 /root/

3) Affectedness
===============

This 'start' script appears to be SUSE packaging specific. Thus the security
team can assign one of the SUSE CNA CVEs for the vulnerability. We will not
need to involve upstream. If this assessment is incorrect please tell us.

From what I see only Tumbleweed is affected by this vulnerability, while
in SUSE:SLE-15-SP2:Update and older the functions discussed in 1) and 2) are
not yet existing. There are only recursive chown and chgrp calls for
/etc/openldap/slapd.d.

4) Conclusion
=============

The function discussed in 2) allows for a simple local root exploit for a
compromised ldap user or group account, *if* the specific configuration
preconditions are fulfilled. Only openSUSE Tumbleweed seems affected.

The rest of the recursive chown and chgrp invocations as well as extracting
the 'directory' keys from /etc/openldap/slapd.conf offer by default no
exploitable vulnerabilities, but it still is very dynamic logic that could
lead to a vulnerability in unexpected configuration scenarios, or if the code
of the script is changed in the future.

Finding more solid ground to base this on would be good. For maintaining
certain file permissions e.g. in dynamic runtime directories systemd-tmpfiles
could be used. If the motivation of this logic is to "fix" broken setups or to
adapt to changed configuration settings then involving the admin interactively
would be the safer option instead of running this chown/chgrp logic during
every service start.

We could also add some hardening settings to the systemd service e.g.
ProtectSytem=full and then a few ReadWritePaths=, if that is feasible. Such
hardening should not be the main mechanism to make the script safe, though.

5) Fixing Prodedure
===================

If my assessment that only Tumbleweed is affected is correct then doing the
fix in the open in a less bureaucratic manner should be possible. I leave that
up to reactive security to judge (and also the maintainers, of course).
```

Thomas Leroy 2022-09-23 10:47:50 UTC Comment 2

```
I can confirm that only openSUSE:Factory is affected.
Since this issue is still embargoed and that we can't submit to OBS if the embargo
is not over, I suggest William to prepare a patch, that will be submitted to OBS
right after we drop the embargo.
William, just let us know when you are ready to submit the patch to OBS :)
```

William Brown 2022-09-26 05:23:13 UTC Comment 3

```
I'll need the CVE number assigned to do the OBS submission.

I have a patch, but I'm not 100% sure about some of the extra hardening:


Index: openldap2.changes
===================================================================
--- openldap2.changes   (revision fd35db8a85b5a926ac2712fe9f6a2902)
```

```
+++ openldap2.changes   (working copy)
@@ -1,3 +1,9 @@
+-------------------------------------------------------------------
+Mon Sep 26 05:16:18 UTC 2022 - William Brown <william.brown@suse.com>
+
+- bsc#1202931 - CVE- - Openldap start script allowed the ldap user
+  to privilege escalate to root due to unbound chown commands.
+
 -------------------------------------------------------------------
 Thu Jul 14 21:22:41 UTC 2022 - Michael Ströder <michael@stroeder.com>

Index: slapd.service
===================================================================
--- slapd.service       (revision fd35db8a85b5a926ac2712fe9f6a2902)
+++ slapd.service       (working copy)
@@ -6,6 +6,22 @@
 Type=forking
 ExecStart=/usr/lib/openldap/start

+# Hardening to prevent security escalation.
+ProtectSystem=full
+ReadWritePaths=/etc/openldap/slapd.d /var/lib/ldap
+
+RestrictSUIDSGID=true
+NoNewPrivileges=true
+PrivateTmp=true
+PrivateDevices=true
+ProtectHostname=true
+ProtectClock=true
+ProtectKernelTunables=true
+ProtectKernelModules=true
+ProtectKernelLogs=true
+ProtectControlGroups=true
+MemoryDenyWriteExecute=true
+
 [Install]
 WantedBy=multi-user.target

Index: start
===================================================================
--- start       (revision fd35db8a85b5a926ac2712fe9f6a2902)
+++ start       (working copy)
@@ -81,10 +81,16 @@
 function chown_database_dirs_bconfig() {
         ldapdir=$(find $1 -type f -name "olcDatabase*" | xargs grep -i
olcdbdirectory | awk '{print $2}')
         for dir in $ldapdir; do
+            if [[ $dir =~ ^/var/lib/ldap$|^/var/lib/ldap/.* ]]; then
                 [ -d "$dir" ] && [ -n "$OPENLDAP_USER" ] && \
                         chown -R $OPENLDAP_USER $dir 2>/dev/null
                 [ -d "$dir" ] && [ -n "$OPENLDAP_GROUP" ] && \
                         chgrp -R $OPENLDAP_GROUP $dir 2>/dev/null
+            else
+                echo "Skipping chown of external directory. You must manually
run:"
+                echo "# chown -R $OPENLDAP_USER $dir"
+                echo "# chgrp -R $OPENLDAP_GROUP $dir"
+            fi
         done
 }
```

This has two parts. First, the hardening of the dirs_bconfig script to only chown/chgrp if the prefix is /var/lib/ldap.

Second is hardening in the systemd service file. This hardening is modelled on the systemd hardening the security team have undertaken in other projects. my concern is the protect system and read write paths segments. LDAP deployments historically for some reason have users who *love* to put their database in weird and esoteric locations. Saying this protect system full only protects /usr, /efi, /boot, /etc so it's pretty likely this will be okay as an upgrade for most users.

```
Any comments/thoughts/feedback?
```

William Brown 2022-10-04 00:15:36 UTC Comment 4

```
Please review proposed change.
```

Johannes Segitz 2022-10-10 10:20:36 UTC Comment 7

```
Please use CVE-2022-31253
```

Matthias Gerstner 2022-10-10 14:12:55 UTC Comment 8

(In reply to william.brown@suse.com from comment #3)

Thanks for working on this William.

```
> Index: start
> ===================================================================
> --- start       (revision fd35db8a85b5a926ac2712fe9f6a2902)
> +++ start       (working copy)
> @@ -81,10 +81,16 @@
>  function chown_database_dirs_bconfig() {
>         ldapdir=$(find $1 -type f -name "olcDatabase*" | xargs grep -i olcdbdire
>         for dir in $ldapdir; do
> +           if [[ $dir =~ ^/var/lib/ldap$|^/var/lib/ldap/.* ]]; then
>                 [ -d "$dir" ] && [ -n "$OPENLDAP_USER" ] && \
>                         chown -R $OPENLDAP_USER $dir 2>/dev/null
>                 [ -d "$dir" ] && [ -n "$OPENLDAP_GROUP" ] && \
>                         chgrp -R $OPENLDAP_GROUP $dir 2>/dev/null
> +           else
> +               echo "Skipping chown of external directory. You must manually ru
> +               echo "# chown -R $OPENLDAP_USER $dir"
> +               echo "# chgrp -R $OPENLDAP_GROUP $dir"
> +           fi
>         done
>  }
>
>
>
> This has two parts. First, the hardening of the dirs_bconfig script to only chowr
```

Is this safe against relative path components like
/var/lib/ldap/../../../root? It doesn't look like.

```
> Second is hardening in the systemd service file. This hardening is modelled on th
```

I fear us from security can't don't have much more insight into how well that
will work with end users. Having it in Tumbleweed only for the start will
probably provide some testing ground to hear about any immediate complaining.

Matthias Gerstner 2022-10-11 14:13:42 UTC Comment 10

(In reply to william.brown@suse.com from comment #9)

> Updating the patch, will inline below.

[...]

```
> +        for dir in $(realpath ${ldapdir}); do
> +            if [[ $dir =~ ^/var/lib/ldap$|^/var/lib/ldap/.* ]]; then
```

okay that should cover the immediate issues. It still allows for exploiting
race conditions though for paths beneath /var/lib/ldap. Sadly with the
current setup this is not really avoidable. Shell code cannot use safe path
operations and since the ownership needs to be changed privileges can also not
be dropped. Also chown and chgrp have no ideal behaviour with regards to
symlinks they encounter in command line arguments.

I dislike that this is run *every* time regardless of whether any change in
configuration occured. I suppose this has been introduced to reduce issues
with users shooting themselves in the foot.

```
> +                echo "Skipping chown of external directory. You must manually ru
```

Maybe you can also add here "for security reasons".

William Brown 2022-10-25 23:03:10 UTC Comment 11

> I dislike that this is run *every* time regardless of whether any change in
> configuration occured. I suppose this has been introduced to reduce issues
> with users shooting themselves in the foot.

OpenLDAP is well known for it's terrible user experience

```
>
> > +                echo "Skipping chown of external directory. You must manually
>
> Maybe you can also add here "for security reasons".
```

Will do. Any issues with me making the submission from here then?

PS: Sorry about the delay update, clearly I missed the email notification :(

Matthias Gerstner 2022-10-26 10:16:30 UTC Comment 12

(In reply to william.brown@suse.com from comment #11)

> Will do. Any issues with me making the submission from here then?

You can go ahead an we can make the bug public once the submission is out.
Thank you.

Matthias Gerstner 2022-10-26 11:15:31 UTC Comment 13

(In reply to william.brown@suse.com from comment #11)

> Will do. Any issues with me making the submission from here then?

A colleague has come up with one more bit: Please also add the `-h` flag to
the chown and chgrp invocations. This at least prevents following immediate
symlinks.

William Brown 2022-10-27 00:53:27 UTC Comment 14

```
Added the -h and tested it as working, submit is here, I'm waiting for it to build
then I'll accept it. https://build.opensuse.org/request/show/1031422
```

Matthias Gerstner 2022-10-27 08:35:37 UTC Comment 15

```
Thanks for working on the fix. I'm publishing the bug and re-assigning to
security. Security will close once we're sure the process is complete.
```