Obscuring network proxy password in plain text files on Linux/UNIX-likes

Asked 16 years, 4 months ago Modified 7 years, 6 months ago

Viewed 14k times

13

Typically in a large network a computer needs to operate behind an authenticated proxy - any connections to the outside world require a username/password which is often the password a user uses to log into email, workstation etc.

This means having to put the network password in the `apt.conf` file as well as typically the `http_proxy`, `ftp_proxy` and `https_proxy` environment variables defined in `~/.profile`

I realise that with `apt.conf` that you could set `chmod 600` (which it isn't by default on Ubuntu/Debian!) but on our system there are people who need root priveleges .

I also realise that it is technically impossible to secure a password from someone who has root access, however I was wondering if there was a way of *obscuring* the password to prevent accidental discovery. Windows operates with users as admins yet somehow stores network passwords (probably stored deep in the registry obscured in some way) so that in typical use you won't stumble across it in plain text

I only ask since the other day, I entirely by accident discovered somebody elses password in this way when comparing configuration files across systems.

@monjardin - Public key authentication is not an alternative on this network I'm afraid. Plus I doubt it is supported amongst the majority of commandline tools.

@Neall - I don't mind the other users having web access, they can use my credentials to access the web, I just don't want them to happen across my password in plain text.

linux security unix proxy apt

Share

Improve this question

Follow

edited Aug 24, 2008 at 0:22

asked Aug 20, 2008 at 21:56

Brendan

19.3k ● 19 ● 89 ● 114

9 Answers

Sorted by: Highest score (default)

8

With the following approach you never have to save your proxy password in plain text. You just have to type in a password interactively as soon as you need http/https/ftp access:

- Use openssl to encrypt your plain text proxy password into a file, with e.g. AES256 encryption:

```
openssl enc -aes-256-cbc -in pw.txt -out pw.bin
```

- Use a (different) password for protecting the encoded file
- Remove plain text pw.txt
- Create an alias in e.g. ~/.alias to set your http_proxy/https_proxy/ftp_proxy environment variables (set appropriate values for $USER/proxy/$PORT)

```
alias myproxy='PW=`openssl aes-256-cbc -d -in pw.bin`; PROXY="http://$USER:$PW@proxy:$PORT"; export http_proxy=$PROXY; export https_proxy=$PROXY; export ftp_proxy=$PROXY'
```

- you should source this file into your normal shell environment (on some systems this is done automatically)
- type 'myproxy' and enter your openssl password you used for encrypting the file
- done.

Note: the password is available (and readable) inside the users environment for the duration of the shell session. If you want to clean it from the environment after usage you can use another alias:

> alias clearproxy='export http_proxy=; export https_proxy=; export ftp_proxy='

Share Improve this answer Follow

edited Jan 8, 2013 at 13:39

answered Jan 8, 2013 at 13:18

lumpidu
759 ● 7 ● 13

2 note that you could use `read -s PW` in "myproxy" to achieve similar result (just read the password from command line every time you need it) – FabienAndre Nov 9, 2015 at 9:51

6

I did a modified solution:

edit `/etc/bash.bashrc` and add following lines:

```
alias myproxy='read -p "Username: " USER;read -s -
p "Password: " PW
PROXY="$USER:$PW@proxy.com:80";
export http_proxy=http://$PROXY;export
Proxy=$http_proxy;export
https_proxy=https://$PROXY;export
ftp_proxy=ftp://$PROXY'
```

From next logon enter `myproxy` and input your user/password combination! Now work with `sudo -E`

> -E, --preserve-env Indicates to the security policy that the user wishes to reserve their existing environment variables.

e.g. `sudo -E apt-get update`

Remark: proxy settings only valid during shell session

Share Improve this answer

Follow

answered May 31, 2017 at 13:38

leon22

5,609 ● 19 ● 64 ● 104

1 I prefer this solution since it's more simple.
– Radhwane Chebaane Jul 9, 2018 at 8:59

I put it in a separated script to allow reset proxy if I did a mistake typing login or password. Beside I don't really get hat the -E option imply ? – Welgriv Aug 5, 2019 at 8:33

echo $PW gives out your password in plaintext
– helperFunction Mar 9, 2022 at 6:52

3

There are lots of ways to obscure a password: you could store the credentials in rot13 format, or BASE64, or use the same password-scrambling algorithm that CVS uses. The real trick though is making your applications aware of the scrambling algorithm.

For the environment variables in `~/.profile` you could store them encoded and then decode them before setting the variables, e.g.:

```
encodedcreds="sbbone:cnffjbeq"
creds=`echo "$encodedcreds" | tr n-za-mN-ZA-M a-zA-Z`
```

That will set `creds` to `foobar:password`, which you can then embed in `http_proxy` etc.

I assume you know this, but it bears repeating: this doesn't add any security. It just protects against inadvertently seeing another user's password.

Share Improve this answer

Follow

answered Aug 25, 2008 at 21:06

Jason Day

8,839 ● 1 ● 42 ● 46

2

Prefer applications that integrate with Gnome Keyring. Another possibility is to use an SSH tunnel to an external machine and run apps through that. Take a look at the `-D` option for creating a local SOCKS proxy interface, rather than single-serving `-L` forwards.

Share Improve this answer

Follow

answered Aug 21, 2008 at 14:57

T Percival

8,654 ● 4 ● 44 ● 43

1

Unless the specific tools you are using allow an obfuscated format, or you can create some sort of workflow to go from obfuscated to plain on demand, you are probably out of luck.

One thing I've seen in cases like this is creating per-server, per-user, or per-server/per-user dedicated credentials that only have access to the proxy from a specific IP. It doesn't solve your core obfuscation problem but it mitigates the effects of someone seeing the password because it's worth so little.

Regarding the latter option, we came up with a "reverse crypt" password encoding at work that we use for stuff like this. It's only obfuscation because all the data needed to decode the pw is stored in the encoded string, but it prevents people from accidentally seeing passwords in plain text. So you might, for instance, store one of the above passwords in this format, and then write a wrapper for apt that builds apt.conf dynamically, calls the real apt, and at exit deletes apt.conf. You still end up with the pw in plaintext for a little while, but it minimizes the window.

Share Improve this answer

Follow

answered Aug 21, 2008 at 15:13

jj33

7,633 ● 2 ● 40 ● 43

Is public key authentication a valid alternative for you?

Share Improve this answer

answered Aug 20, 2008 at 22:31

0

Follow

Judge Maygarden

27.5k ● 9 ● 83 ● 100

0

As long as all three of these things are true, you're out of luck:

1. Server needs web access
2. Users need absolute control over server (root)
3. You don't want users to have server's web access

If you can't remove #2 or #3, your only choice is to remove #1. Set up an internal server that hosts all the software updates. Keep that one locked down from your other users and don't allow other servers to have web access.

Anything else you try to do is just fooling yourself.

Share Improve this answer

Follow

answered Aug 21, 2008 at 12:26

Neall

27.1k ● 5 ● 50 ● 49

0

we solved this problem by not asking for proxy passwords on rpm, apt or other similar updates (virus databases,

windows stuff etc) That's a small whitelist of known repositories to add to the proxy.

Share Improve this answer

Follow

answered Sep 17, 2008 at 14:57

Gunstick

431 ● 4 ● 3

> Unfortunately I don't have any say in the proxying policy on the network. I have had this same situation in both universities I have worked in. – Brendan Sep 20, 2008 at 15:47

0

I suppose you could create a local proxy, point these tools through that, and then have the local proxy interactively ask the user for the external proxy password which it would then apply. It could optionally remember this for a few minutes in obfuscated internal storage.

An obvious attack vector would be for a privileged user to modify this local proxy to do something else with the entered password (as they could with anything else such as an email client that requests it or the windowing system itself), but at least you'd be safe from inadvertent viewing.

Share Improve this answer

Follow

answered Nov 18, 2010 at 21:01

Chris Stratton

40.3k ● 6 ● 87 ● 119