Laboratory Assignment 1: Identification and Authentication

Olof Magnusson and Yu Hsuan Liao

Computer Security EDA263

August 11, 2025

Contents

1 Questions

1. What is password ageing? And what methods exist to implement it?

   Answer: Password ageing is a mechanism that allows the system to enforce specific numbers of minimum and maximum lifetime for passwords. In the Redhat, documentation [1], the authors mention that the password ageing means that after a set amount of time (often referred as 90 days) the user is prompted to create a new password. In the Linux system architecture, there are several ways to implement password ageing controls. More specific, we can look into the `/etc/login.defs`. The login.defs implies the minimum and maximum days that a password possible could be used, the minimum acceptable password length, and a trigger that notifies the user several days before a password expires. An implementation could then look something like this: `PASS_MAX_DAYS_100`, `PASS_MIN_DAYS 60` and `PASS_WARN_AGE 94`. Another way of implementing it is to set `min` and `max` values for automatic uid selection in `useradd`, `groupadd`. Hence, we will have more flexibility than only setting requirements only by the user. However, one should check `/etc/default/useradd` in detail, especially the parameters inactive and expire. These parameters are more interesting since we can terminate an account based on the number of days after a particular password expires. Depending on the Linux distribution, an example of the parameters in the `/etc/default/useradd` can be shown below:

   ```
   #!useradd defaults file
   GROUP=100
   HOME=/home
   INACTIVE=-1
   EXPIRE=
   SHELL=/bin/bash
   SKEL=/etc/skel
   CREATE_MAIL_SPOOL=yes
   ```

   An example that we used in the lab is to increment the password age when access is granted `passwddata -> pwage++;` to keep track of the user's current logins. This code snippet below is used to find out if a particular user has entered the password many times.

   ```
   if (passwddata->pwage == 5) {
       printf("Change your password!");
   }
   ```

2. If you want to increase the security of a system, you can use one-time passwords. What are the advantages? What are the disadvantages?

   Answer: The most significant advantage when using two-factor authentication is to add another layer of security with the authentication process. Mainly, we ask the individual to authorize to the system with his secret together with a unique key. The unique key is randomly generated, and it is continuously refreshing code to keep it up to date for authentication. This indicates that if an adversary finds out the secret (password), it is difficult to get access without the corresponding unique key. The authenticator will discard every attempt to the system that is not considered a trusted party (not validated). In terms of security, this is a

good advantage, both that we can eliminate potential replay-attack and eavesdropping since it is only one-time-use.

However, one more clear advantage of LastPass (a password manager) is that the authenticator provides one-session-key; where we, as a user, do not need to use the master password on an unknown device. Another service the company offer is that we could be using the Yubi key, which provides multi-protocol flexibility. There are some problems that an attacker can cause. One example is that the attacker can freeze the account if the username is revealed. The simply methodology is to exhaust the authentication server by repeatedly sending login requests. Another disadvantage of using a Yubi-key is that we rely on that the hardware does not get stolen or damaged in any way.

3. Authentication systems are often based on some knowledge shared by the computing system and the user. This knowledge can be of three types: something the user knows, has or is. For each of these types, answer the following questions:

   (a) How can the authentication mechanism be implemented?

   Answer: The authentication mechanism can be implemented in several ways. The most trivial is we base the authentication mechanism on *Something that a user knows*, which is, for example, a password. One case is when a system takes the input (password) for that userID and compares it in the system password file. *Something that a user has* could be a physical object such as a USB-stick, PIN, access cards that can be used for authentication to a specific location in a building like a server room. Lastly, *Something that the user is*, could be any physical characteristic that the user choose to authenticate with. Some examples are fingerprint, eyes and voice.

   (b) What advantages and disadvantages does the authentication method have with respect to e.g. user friendliness, cost of introduction, and accuracy.?

   Answer: *Something that a user knows* is user-friendly due it is a straightforward authentication method, e.g. remembering a password. The problem with this authentication method is that people tend to choose a simple password (often to remember them more comfortable). It is usually based on something related to them. Another growing problem that is widely discussed today is that people reuse the same password on different websites [2]. The problem with simplicity and reusing is that there are possibilities to discover the secret's combinations. By finding the secret information, you could access more platforms if they are reusing it. The cost of introduction is low, but the accuracy that it is the right user cannot be guaranteed based on this authentication method.

   Instead, we look into *Something that the user have*. From the first glance, this can still be considered user-friendly and using an access card with a PIN can even be more manageable than using a password. The disadvantage is that the PIN could be discovered easier since it often implemented using a four-digit number. The access card can, besides, be lost or easily be damaged. A side-channel attack could be deployed to bypass the encryption and instead of looking at the different frequencies of the pressed PINS of the card. [3].

   The *Something that the user is* could be considered user-friendly since it is bound to the person's characteristics. In terms of usage, the introduction cost will grow significantly in terms of installing those authentication stations. However, it depends on how good the technique is classifying the biometric authentication correct. A false positive is when the

authentication system matches an individual to someone else's credentials. This method needs more integration with hardware which could raise another level of complexity.

(c) How can the disadvantages be overcome in the following environments?

Answer: *Something that the user is:* we could use a password generator instead of remembering a password and using a one-time-session key for some particular tasks, avoiding reusing the master-key. A clear example is that LastPass warns for password duplication and weakness in the password. *Something that the user has:* we could educate the employees to protect their physical assets better. *Something that the user is:* we have to have some backup if the system is not accurate and does not work as intended. This could be, by using a specific access card, for example.

4. What authentication method would you recommend for use in computer systems

(a) A university

Answer: *Something that the user know* due to the simplicity. We still need to have a password to authenticate to the website server. However, we should enforce the students/employees to use a password of some complexity with symbols, numbers, letters and a maximum length of 10-15 characters. Two-factor authentication would raise another layer of complexity in this environment for authentication. A recommendation could be that we could use two-factor authentication when we are signing essential documents.

(b) A military facility

Answer: *Something that the user is*. This is a sensitive environment which requires maximum security in certain regions. Some information even needs to be delivered in person. For example, the server room needs to be accessed only by biometric identification.

(c) A middle-sized company

Answer: *Something that the user knows*. We recommend that a middle-sized company enforce more challenging policies on how the users create and manage their passwords. *Something that the user has* is also vital since every employee have an access card to enter buildings. Here, it is essential to create guidelines for the employees that are easy to follow and to adapt.

(d) A personal home computer

Answer: *Something that the user knows*, we do not need some biometric identification here (but it depends on whether we have any significant assets). However, we could think about how we are managing our passwords using a password manager.

5. In some systems you do not only implement authentication of the user against the system, but also of the system against the user. What is the purpose of doing this?

Answer: The purpose is to ensure that it is the right user that is authenticated to the system. One good example is remote access to a system/subsystem; for example, when we are connecting to `chalmers.remote11` where the system requires us to enter credentials based on our CID.

6. A user is running a program containing the system call setuid(). Depending on who is running the program and on the value of the argument to setuid(), different

things can happen. Check the manual page for setuid(2) on the lab system (man -s2 setuid).

(a) Study Figure 1. What are the values of the real user ID (ruid) and the effective user ID (euid) at i) and ii)? Refer to Table 1 for the uid.
Answer: i) `euid=20716`, `ruid=20716` ii) `euid=20716 ruid=20716`

(b) Table 1 illustrates six cases where root or a normal user account starts a normal program which uses setuid() with different arguments (user IDs). Fill in the third and fourth column in the table. The third column should state whether the system call will succeed or fail. The fourth column should state the effective user ID (euid) of the program after the setuid() call.

Table 1: Table showing current user, setuid

	User:	setuid(UID)	Sucess/failure	setuid()
1	root	0 (root)	Success	0
2	root	20757(csec069)	Success	20757
3	root	20716 (csec028)	Success	20716
4	csec028	0 (root)	Failure	20716
5	csec028	20757 (csec069)	Failure	20716
6	csec028	20716 (csec028)	Success	20716

7. On Unix systems, file-access permissions on programs may be set to set-user-ID (these are often called SUID programs) as in the passwd program

```
#!/bin/bash
csec@computer ~ \$ ls \-l /usr/bin/passwd
-rwsr-xr-x 1 root root 27888 Jul 26 09:22 /usr/bin/passwd
```

(a) What are the values of the real UID (ruid) and the effective UID (euid) at i), when the user csec028 runs the SUID program /bin/prog
Answer: i) `euid = 0`, `ruid = 20716`

(b) What is the purpose of using SUID on programs?
Answer: It is more flexible in terms of delegating privileges when running different processors in the system. Some tasks in a system need to be done in a high-privilege mode (admin), and low-privilege mode since the users need to have the ability to do basic tasks.

8. Sometimes the setuid() (or perhaps more likely, seteuid(2)) function is used in programs where the set-user-ID bit is activated on the program's file-access permissions. This is done, for example, in the /usr/bin/passwd program.

(a) What are the values of ruid and euid at i) and ii) in Figure 3?
Answer: i) `euid = 0`, `ruid = 20716` ii) `euid = 20716`, `ruid = 20716`.

(b) What is the purpose of doing this?

Answer: According to [4, 5], the purpose of this is that we want to escalate permissions, and thus we need to start with the highest privilege as possible, which is often to be considered the root. When we have finished one permission, it will call `setuid` and downgrade privileges since the consequent task does not require high privilege to execute. Thus, when this process is done, then the original permission is restored.

2 Conclusion and reflections

The lab's theoretic part has provided a general overview of password ageing, using one-time-password with its advantages and disadvantages. For this lab, it has been interesting to understand the different authentication systems - how they work and what its flaws. New reasoning to use the authentication in different applications areas such as university, companies, university and personal home computer. There is no doubt that the cybersecurity market is increasing in its size and that we need to have global thinking regarding how we design systems. Another important thing that is easily overlooked is to educate individuals in order to create user security awareness.

References

[1] Swcp, "Password aging."
Online: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/4/html/security_guide/s3-wstation-pass-org-age Accessed: 2021-01-26, access.redhat.

[2] Digitalguardian, "Uncovering password habits: Are users' password security habits improving."
Online: https://digitalguardian.com/blog/uncovering-password-habits-are-users-password-security-habits-improving-infographic Accessed: 2021-01-23, digitalguardian.com.

[3] F. Koeune and F.-X. Standaert, "A tutorial on physical security and side-channel attacks," *Foundations of Security Analysis and Design III*, pp. 78–108, 2005.

[4] A. Frisch, *Essential system administration: Tools and techniques for linux and unix administration.* " O'Reilly Media, Inc.", 2002.

[5] Swcp, "setuid(2) — linux manual page."
Online: https://man7.org/linux/man-pages/man2/setuid.2.html Accessed: 2021-01-21, man7.org.