0x0B. SSH

DevOpsSSHNetworkSysAdminSecurity

- By: Sylvain Kalache
- Weight: 1
- Project over - took place from Sep 16, 2022 6:00 AM to Sep 19, 2022 6:00 AM
- An auto review will be launched at the deadline

In a nutshell…

- **Auto QA review:** 3.5/8 mandatory & 1.0/2 optional
- **Altogether: 65.63%**
  - Mandatory: 43.75%
  - Optional: 50.0%
  - Calculation: 43.75% + (43.75% * 50.0%) == **65.63%**

Background Context

Along with this project, you have been attributed an Ubuntu server, living in a datacenter far far away. Like level 2 of the application process, you will connect using `ssh`. But contrary to level 2, you will not connect using a password but an RSA key. We've configured your server with the public key you created in the first task of a previous project shared in your intranet profile.

You can access your server information in the my servers section of the intranet, each line with contains the IP and username you should use to connect via `ssh`.

Note: Your server is configured with an Ubuntu 20.04 LTS environment.

Resources

Read or watch:

- What is a (physical) server - text
- What is a (physical) server - video
- SSH essentials
- SSH Config File
- Public Key Authentication for SSH
- How Secure Shell Works
- SSH Crash Course (Long, but highly informative. Watch this if configuring SSH is still confusing. It may be helpful to watch at x1.25 speed or above.)

For reference:

- Understanding the SSH Encryption and Connection Process
- Secure Shell Wiki
- IETF RFC 4251 (Description of the SSH Protocol)
- Internet Engineering Task Force
- Request for Comments

man or help:

- `ssh`
- `ssh-keygen`
- `env`

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone, **without the help of Google**:

General

- What is a server
- Where servers usually live
- What is SSH
- How to create an SSH RSA key pair
- How to connect to a remote host using an SSH RSA key pair
- The advantage of using `#!/usr/bin/env bash` instead of `/bin/bash`

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else’s work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

General

- Allowed editors: `vi`, `vim`, `emacs`
- All your files will be interpreted on Ubuntu 20.04 LTS
- All your files should end with a new line
- A `README.md` file, at the root of the folder of the project, is mandatory
- All your Bash script files must be executable
- The first line of all your Bash scripts should be exactly `#!/usr/bin/env bash`
- The second line of all your Bash scripts should be a comment explaining what is the script doing

Your servers

Name	Username	IP	State	
1609-web-01				Actions Toggle Dropdown

Tasks

0. Use a private key

mandatory

Score: 50.0% (*Checks completed: 100.0%*)

Write a Bash script that uses `ssh` to connect to your server using the private key `~/.ssh/school` with the user `ubuntu`.

Requirements:

- Only use `ssh` single-character flags
- You cannot use `-l`

- You do not need to handle the case of a private key protected by a passphrase

```
sylvain@ubuntu$ ./0-use_a_private_key

ubuntu@server01:~$ exit

Connection to 8.8.8.8 closed.

sylvain@ubuntu$
```

Repo:

- GitHub repository: `alx-system_engineering-devops`
- Directory: `0x0B-ssh`
- File: `0-use_a_private_key`

Done! Help Check your code QA Review

1. Create an SSH key pair

mandatory

Score: 50.0% (*Checks completed: 100.0%*)

Write a Bash script that creates an RSA key pair.

Requirements:

- Name of the created private key must be `school`
- Number of bits in the created key to be created 4096
- The created key must be protected by the passphrase `betty`

Example:

```
sylvain@ubuntu$ ls

1-create_ssh_key_pair

sylvain@ubuntu$ ./1-create_ssh_key_pair

Generating public/private rsa key pair.

Your identification has been saved in school.

Your public key has been saved in school.pub.

The key fingerprint is:

5d:a8:c1:f5:98:b6:e5:c0:9b:ee:02:c4:d4:01:f3:ba vagrant@ubuntu

The key's randomart image is:

+--[ RSA 4096]----+

|      oo...      |

|      .+.o =     |
|     o  + B +    |
|      o. = O     |
|     .. S = .    |
|      .. .       |
|      E.  .      |
|        ..       |
|         ..      |
+-----------------+
sylvain@ubuntu$ ls
1-create_ssh_key_pair school  school.pub
sylvain@ubuntu$
```

Repo:

- GitHub repository: `alx-system_engineering-devops`
- Directory: `0x0B-ssh`
- File: `1-create_ssh_key_pair`

Done! Help Check your code QA Review

2. Client configuration file

mandatory

Score: 50.0% (*Checks completed: 100.0%*)

Your machine has an SSH configuration file for the local SSH client, let's configure it to our needs so that you can connect to a server without typing a password. Share your SSH client configuration in your answer file.

Requirements:

- Your SSH client configuration must be configured to use the private key `~/.ssh/school`
- Your SSH client configuration must be configured to refuse to authenticate using a password

Example:

```
sylvain@ubuntu$ ssh -v ubuntu@98.98.98.98
OpenSSH_6.6.1, OpenSSL 1.0.1f 6 Jan 2014
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 47: Applying options for *
debug1: Connecting to 98.98.98.98 port 22.
debug1: Connection established.
```

```
debug1: identity file /home/sylvain/.ssh/school type -1
debug1: identity file /home/sylvain/.ssh/school-cert type -1
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_8.1
debug1:Remote protocol version 2.0, remote software version OpenSSH_7.6p1 Ubuntu-4ubuntu0.5
debug1: match: OpenSSH_7.6p1 Ubuntu-4ubuntu2.1 pat OpenSSH* compat 0x04000000
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: kex: server->client aes128-ctr hmac-sha1-etm@openssh.com none
debug1: kex: client->server aes128-ctr hmac-sha1-etm@openssh.com none
debug1: sending SSH2_MSG_KEX_ECDH_INIT
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: Server host key: ECDSA bd:03:f8:6a:12:28:d6:17:85:c1:b6:91:f1:da:0f:37
debug1: Host '98.98.98.98' is known and matches the ECDSA host key.
debug1: Found key in /home/sylvain/.ssh/known_hosts:1
debug1: ssh_ecdsa_verify: signature correct
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: SSH2_MSG_NEWKEYS received
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Trying private key: /home/sylvain/.ssh/school
debug1: key_parse_private2: missing begin marker
debug1: read PEM private key done: type RSA
debug1: Authentication succeeded (publickey).
Authenticated to 98.98.98.98 ([98.98.98.98]:22).
debug1: channel 0: new [client-session]
debug1: Requesting no-more-sessions@openssh.com
debug1: Entering interactive session.
debug1: client_input_global_request: rtype hostkeys-00@openssh.com want_reply 0
debug1: Sending environment.
```

```
debug1: Sending env LANG = en_US.UTF-8

ubuntu@magic-server:~$
```

In the example above, we can see that `ssh` tries to authenticate using `school` and does not try to authenticate using a password. You can replace `98.98.98.98` by the IP of your server for testing purposes.

Repo:

- GitHub repository: `alx-system_engineering-devops`
- Directory: `0x0B-ssh`
- File: `2-ssh_config`

Done! Help Check your code QA Review

3. Let me in!

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Now that you have successfully connected to your server, we would also like to join the party.

Add the SSH public key below to your server so that we can connect using the `ubuntu` user.

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDNdtrNGtTXe5Tp1EJQop8mOSAuRGLjJ6DW4PqX4wId/Kawz
35ESampIqHSOTJmbQ8UlxdJuk0gAXKk3Ncle4safGYqM/VeDK3LN5iAJxf4kcaxNtS3eVxWBE5iF3FbIjOqwx
w5Lf5sRa5yXxA8HfWidhbIG5TqKL922hPgsCGABIrXRlfZYeC0FEuPWdr6smOElSVvIXthRWp9cr685KdCI+C
Oxlj1RdVsvIo+zunmLACF9PYdjB2s96Fn0ocD3c5SGLvDOFCyvDojSAOyE70ebIElnskKsDTGwfT4P6jh9OBz
TyQEIS2jOaE5RQq4IB4DsMhvbjDSQrP0MdCLgwkN
```

Repo:

- GitHub repository: `alx-system_engineering-devops`
- Directory: `0x0B-ssh`

Done! Help Check your code Ask for a new correction Get a sandbox QA Review

4. Client configuration file (w/ Puppet)

#advanced

Score: 50.0% (*Checks completed: 100.0%*)

Let’s practice using Puppet to make changes to our configuration file. Just as in the previous configuration file task, we’d like you to set up your client SSH configuration file so that you can connect to a server without typing a password.

Requirements:

- Your SSH client configuration must be configured to use the private key `~/.ssh/school`
- Your SSH client configuration must be configured to refuse to authenticate using a password

Example:

```
vagrant@ubuntu:~$ sudo puppet apply 100-puppet_ssh_config.pp
Notice: Compiled catalog for ubuntu-xenial in environment production in 0.11 seconds
Notice: /Stage[main]/Main/File_line[Turn off passwd auth]/ensure: created
Notice: /Stage[main]/Main/File_line[Declare identity file]/ensure: created
Notice: Finished catalog run in 0.03 seconds
vagrant@ubuntu:~$
```

Repo:

- GitHub repository: `alx-system_engineering-devops`
- Directory: `0x0B-ssh`
- File: `100-puppet_ssh_config.pp`

Done! Help Check your code QA Review