ICA0002: IT Infrastructure Services

SSH Basics

Roman Kuchin
Juri Hudolejev
2023

SSH: Secure Shell

Remote shell operated securely over insecure network

Replaced **`telnet`**, **`rsh`**, **`rlogin`** and **`rexec`**

De-facto standard tool to operate remote machines

Default connection protocol in Ansible

More info: https://www.ssh.com/academy/ssh

SSH: Secure Shell

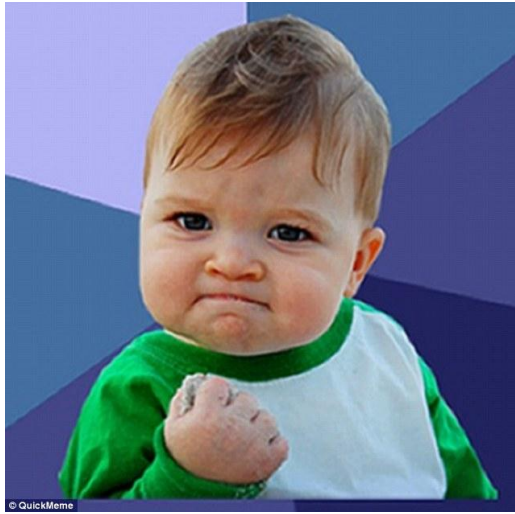
Remote shell operated securely over insecure network

Replaced **`telnet`**, **`rsh`**, **`rlogin`** and **`rexec`**

De-facto standard tool to operate remote machines

Default connection protocol in Ansible

More info: https://www.ssh.com/academy/ssh



Encryption

Symmetric encryption: DES, AES etc.

- Same key for encryption and decryption -- shared secret

Asymmetric (public key) encryption: DSA, RSA etc.

- Public key (openly distributed) + private key (kept secret)
- Message encrypted with one key from the pair
  can only be decrypted with the other key from the same pair

Encryption

Symmetric encryption: DES, AES etc.

Requires secure channel to exchange the shared secret :(

Asymmetric (public key) encryption: DSA, RSA etc.

Is unacceptably inefficient on large data :(

Let's use a **public keys** to compute the shared key securely, and then use that **shared key** to encrypt the data...

That would work!

SSH session initialization

Client sends the connection request to the server

Then client and server both:

- agree algorithms for key exchange, symmetric and public key encryption
- generate shared session key using Diffie-Hellman method (SSH v2)

Then server performs the client authentication

If all good, connection is established

SSH client authentication

Authentication options:

- User password based
- User key based: RSA, DSA, ECDSA etc.
- Host key based
- Interactive -- for one time passwords
- GSSAPI -- for external authentication services such as Kerberos

SSH client authentication

Authentication options:

- User password based ← avoid at all costs!
- User key based: RSA, DSA, ECDSA etc. ← we only use this on this course
- Host key based
- Interactive -- for one time passwords
- GSSAPI -- for external authentication services such as Kerberos

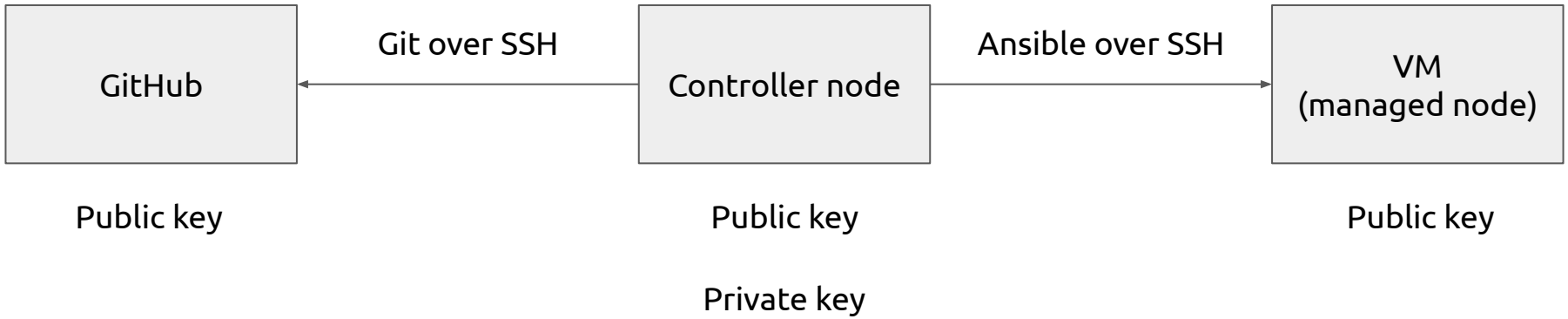
SSH public key based client authentication

In very simple terms:

- Client encrypts (signs) certain data with its **private** key
- **Public** key and signature are sent to SSH server
- SSH server checks if the public key is acceptable (authorized)
- SSH server verifies signature
- If all checks passed -- client is authenticated

More detailed info: https://tools.ietf.org/html/rfc4252

SSH in this course



Your SSH keys in this course

Your public key:

- **`~/.ssh/id_rsa.pub`** file on Controller node (connecting from)
- **`~/.ssh/authorized_keys`** file on your VMs (connecting to)
- In your GitHub account: https://github.com/<username>.keys (lab 1)

Your private key:

- **`~/.ssh/id_rsa`** file on Controller node: should never leave your machine!

Public key may also be computed from the private key file, but not vice versa!

Important!

If your private key is lost or compromised,

1. Delete the corresponding public key from your GitHub account immediately!
2. Generate a new key pair (see lab 1)
3. Contact the teachers to reset the keys on your VMs

Questions?

SSH session initialization

Client sends the connection request to the server

~~~ What step is missing here? ~~~

Then client and server both:

- agree algorithms for key exchange, symmetric and public key encryption
- generate shared session key using Diffie-Hellman method (SSH v2)

Then server performs the client authentication

If all good, connection is established
~~~

SSH session initialization

Client sends the connection request to the server

Client performs the server authentication

Then client and server both:

- agree algorithms for key exchange, symmetric and public key encryption
- generate shared session key using Diffie-Hellman method (SSH v2)

Then server performs the client authentication

If all good, connection is established

Host SSH keys

Host key:

- Identifies the host (server) that client is connecting to
- Verified by client on connection initialization (earlier stage)

Host SSH keys

Host key:

- Identifies the host (server) that client is connecting to
- Verified by client on connection initialization (earlier stage)

Client key (discussed earlier):

- Identifies the client that connect to the host
- Verified by server on client authentication (later stage)

Host SSH keys

Host key:

- Identifies the host (server) that client is connecting to
- Verified by client on connection initialization (earlier stage)

Client key (discussed earlier):

- Identifies the client that connect to the host
- Verified by server on client authentication (later stage)

"Host key verification" ≠ "Host key based client authentication"

^-- Server check client's host key

^-- Client checks server's host key

Host SSH keys: first connection

Host key is approved manually on the client's first connection to this host:

```
$ ssh -p9022 ubuntu@193.40.156.67
The authenticity of host '193.40.156.67 (193.40.156.67)' can't be established.
ECDSA key fingerprint is SHA256:{...key fingerprint...}.
Are you sure you want to continue connecting (yes/no)?
```

Host SSH key is changed

Client will discard SSH connection if the host key is different from previously approved value:

```
$ ssh -p9022 ubuntu@193.40.156.67
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
```

Host SSH key files

Host public key:

`/etc/ssh/ssh_host_ecdsa_key.pub` file on your VMs (connecting to)

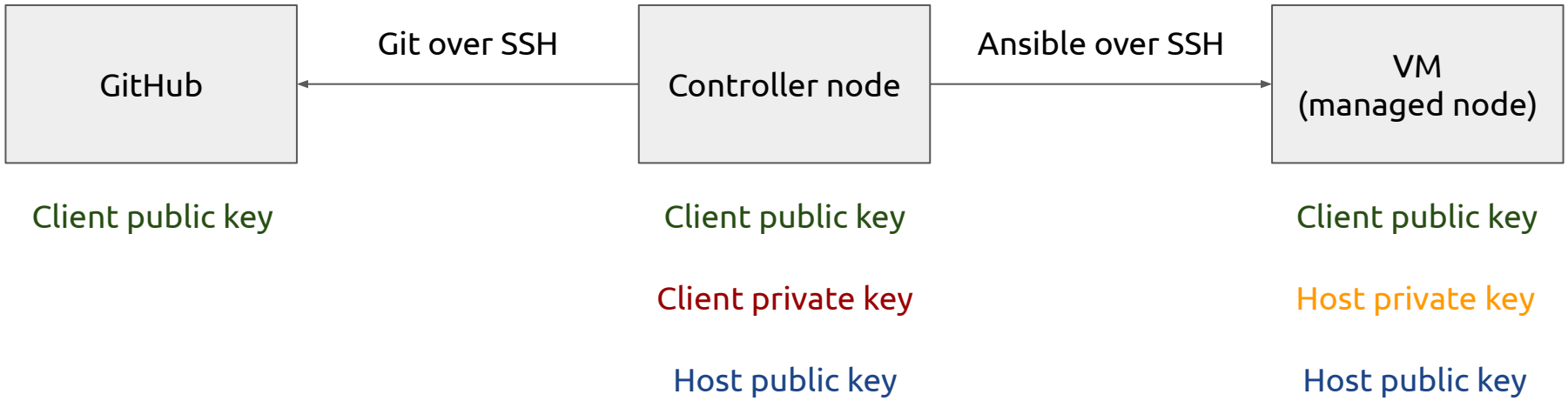
`~/.ssh/known_hosts` file on Controller node (connecting from)

Host private key:

`/etc/ssh/ssh_host_ecdsa_key` file on your VMs

On this course: do NOT touch host private keys!

SSH in this course



Questions?

What key is stored in this file?

1. `~/.ssh/authorized_keys`
2. `~/.ssh/id_rsa`
3. `~/.ssh/id_rsa.pub`
4. `~/.ssh/known_hosts`
5. `/etc/ssh/ssh_host_rsa_key`
6. `/etc/ssh/ssh_host_rsa_key.pub`
7. `https://github.com/elvis.keys`

What key is stored in this file?

1. `~/.ssh/authorized_keys` ← client public key on server
2. `~/.ssh/id_rsa` ← client private key
3. `~/.ssh/id_rsa.pub` ← client public key on client machine
4. `~/.ssh/known_hosts` ← server public key on client
5. `/etc/ssh/ssh_host_rsa_key` ← server private key
6. `/etc/ssh/ssh_host_rsa_key.pub` ← server public key on server
7. `https://github.com/elvis.keys` ← client public key in GitHub