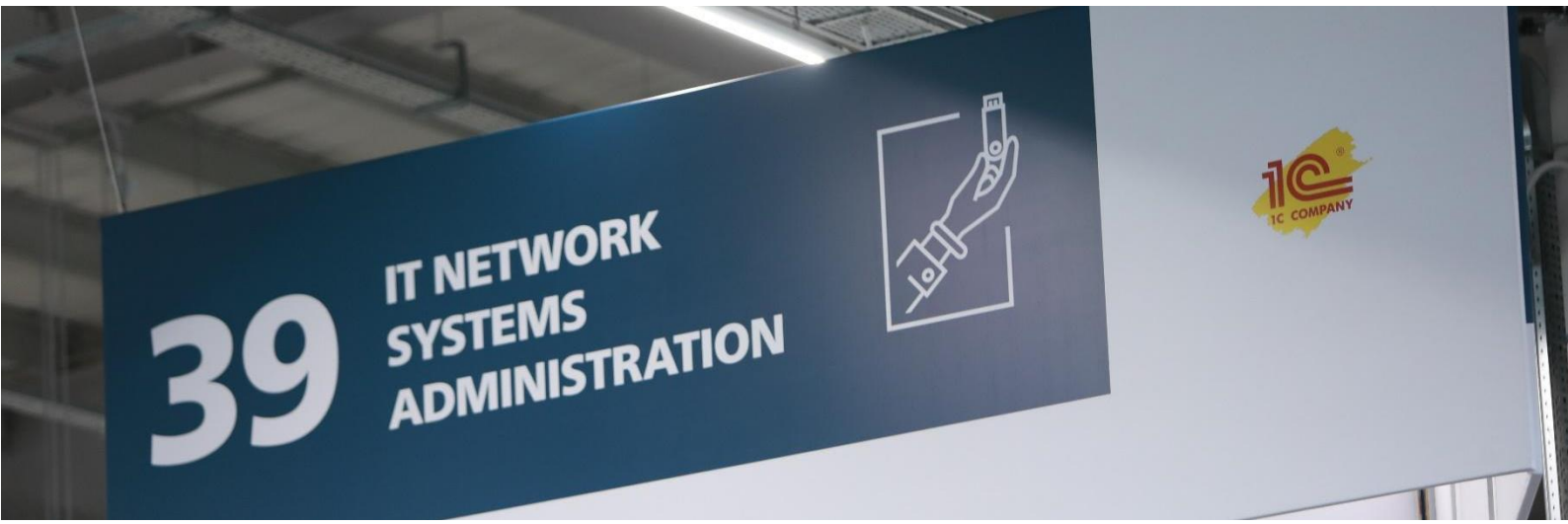
LOMBA KOMPETENSI SISWA
SEKOLAH MENENGAH KEJURUAN
TINGKAT PROVINSI JAWA TENGAH TAHUN 2023

IT NETWORK SYSTEMS ADMINISTRATION



Test Project

MODUL D – INFRASTRUCTURE PROGRAMMABILITY & AUTOMATION

Introduction

The hostname and IP addresses of all VMs are already configured along with sudo access for user “competitor” with password “P@ssw0rd”. Because of the preconfigured sudo, you should never use the root user directly.

Manage your time carefully as the test project has a fixed start and finish time. Good luck.

Tasks

Initial configuration

- Make sure “deployer” can access “srv1” and “srv2”.
- Configure SSH access from “deployer” to “srv1” and “srv2”:
    - Create an Ed25519 SSH key without any passphrase.
    - Add the public key as authorized in “srv1 and “srv2”.
    - Ensure you can seamlessly SSH into “srv1” and “srv2”.
- Set the SSH daemon in “srv1” and “srv2” to listen at port 2222.

Network scripting

Initial setup

- Enable RESTCONF in “router”.
- Make sure python3 is available in “deployer”.

Hostname

- Create file “/home/competitor/set_hostname_via_restconf.py”
    - The script should set a router hostname.
    - Configuration to the router should be done via RESTCONF only.
    - Example usage: /home/competitor/set_hostname_via_restconf.py 10.99.20.254 myhostname

Interfaces

- Create file “/home/competitor/get_interfaces_via_restconf.py”
    - The script should get all interfaces with the IP address. Interfaces without any IP address should be filtered out.
    - Communication with the router should be done via RESTCONF only.
    - Example usage: /home/competitor/get_interfaces_via_restconf.py 10.99.20.254
    - Refer to the appendix for example output.

Ansible

Initial setup

- Install ansible in “deployer”.
- Create an inventory file in “/home/competitor/playbooks/hosts.ini”. Check the appendix for the file content.
- Use the ping module to test connectivity to “srv1” and “srv2”.
- The following tasks should be done in “deployer”.

Users

- Create a playbook in “/home/competitor/playbooks/01_create_users.yaml”:
    - Create 1000 users on “srv1” and “srv2”.
    - The username is “userXXXX” where XXXX is 0000-1000.
    - All passwords should be set to “P@ssw0rd”.
    - The home directory should be set to “/home/userXXXX”.

DNS

- Create a playbook in “/home/competitor/playbooks/02_dns_server.yaml”:
    - Configure DNS server on “srv1” and “srv2”.
    - The first server should be the master for the domain “itnsa.id”.
    - The second server should be a secondary DNS server for the domain “itnsa.id”.
    - Make sure all hosts in the logical topology have an A record for <hostname>.itnsa.id.
    - www.itnsa.id should resolve both to “srv1” and “srv2” using A records.
- Configure “deployer” to use “srv1” and “srv2” as nameservers.

App

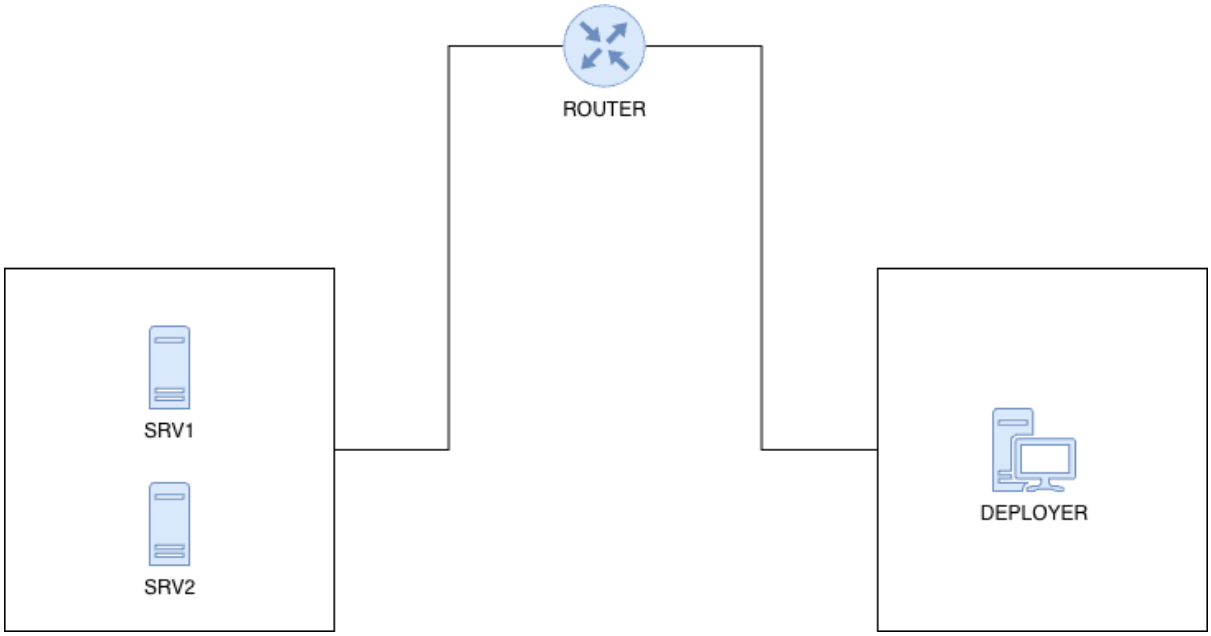
- Create a playbook in “/home/competitor/playbooks/03_app_server.yaml”:
    - Create a file named “/home/competitor/myapp.py”. Check the appendix for the file content.
    - Create a systemd service named “myapp.service” to run the script.
    - Make sure the systemd service is running and automatically started when booting.

Web

- Create a playbook in “/home/competitor/playbooks/04_web_server.yaml”:
    - Configure web server on “srv1” and “srv2”.
    - The web server used should be determined by the host variable “package”.
    - The web server should use the myapp service as its backend.

Appendix

Logical topology



Example output for get_interfaces_via_restconf.py

```
{
  "interfaces": [
    {
      "GigabitEthernet0": "10.99.10.254"
    },
    {
      "GigabitEthernet1": "10.99.20.254"
    }
  ]
}
```

Ansible inventory file

```
[dns]
srv1
srv2

[app]
srv1
srv2

[web]
srv1 package=apache2
srv2 package=nginx
```

MyApp source code

```
from http.server import BaseHTTPRequestHandler, HTTPServer

class Handler(BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.send_header("Content-Type", "text/plain")
        self.end_headers()
        self.wfile.write(bytes(f"MyApp", "utf-8"))

if __name__ == "__main__":
    host = "0.0.0.0"
    port = 8080
    server = HTTPServer((host, port), Handler)
    print(f"Server listening on {host}:{port}")

    try:
        server.serve_forever()
    except KeyboardInterrupt:
        pass

    server.server_close()
```