

## Project 2: Enterprise Authentication

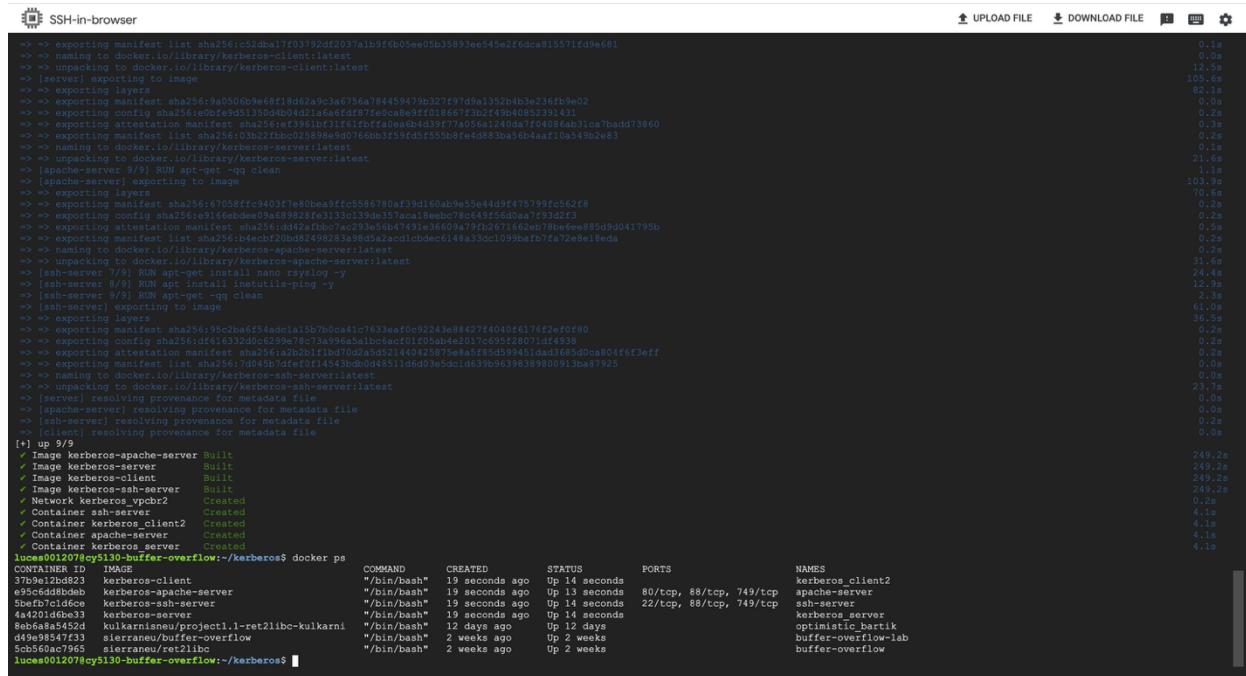
### PART 1: SETTING UP DOCKER CONTAINERS

**Screenshot 1:** This screenshot shows I have successfully made a new directory called “Kerberos” and inside have the subdirectory like “server”, “client”, “ssh-server”, “apache-server” and “docker-compose.yaml”. All these will need to use it next to build the separate container.

```
lucas@01207cy5130-buffer-overflow:~/kerberos$ ls
apache-server client docker-compose.yaml server ssh-server
lucas@01207cy5130-buffer-overflow:~/kerberos$ tree -a kerberos
kerberos
├── apache-server
│   ├── Dockerfile
│   └── Dockerfile
├── client
│   └── Dockerfile
├── docker-compose.yaml
├── server
│   ├── Dockerfile
│   └── Dockerfile
└── ssh-server
    └── Dockerfile

4 directories, 5 files
lucas@01207cy5130-buffer-overflow:~$
```

**Screenshot 2:** Docker compose building all four containers (server, kerberos\_client2, ssh-server, apache-server) from Ubuntu 18.04 base images with required packages (preassigned static IP address and DNS resolution). This also shows successful creation of Docker network "kerberos\_vpcbr2" and all four containers starting with status "done".



```
SSH-in-browser
SSH in browser: 127.0.0.1:22
SSH port: 22
SSH user: lucas
SSH password: lucas

lucas@01207cy5130-buffer-overflow:~/kerberos$ docker-compose up --build
[+] Building 0 services
[+] Building [kerberos-client]: Done
[+] Building [kerberos-server]: Done
[+] Building [apache-server]: Done
[+] Building [ssh-server]: Done
[+] Creating network [kerberos_vpcbr2] with driver [bridge]
[+] Creating [kerberos-client2] using dockerfile
[+] Creating [kerberos-client] using dockerfile
[+] Creating [kerberos-server] using dockerfile
[+] Creating [apache-server] using dockerfile
[+] Creating [ssh-server] using dockerfile
[+] Creating [kerberos-client2] using dockerfile
[+] Creating [kerberos-client] using dockerfile
[+] Creating [kerberos-server] using dockerfile
[+] Creating [apache-server] using dockerfile
[+] Creating [ssh-server] using dockerfile
[*] Started service [kerberos-client] in container [kerberos-client2]
[*] Started service [kerberos-client] in container [kerberos-client]
[*] Started service [kerberos-server] in container [kerberos-server]
[*] Started service [apache-server] in container [apache-server]
[*] Started service [ssh-server] in container [ssh-server]
[*] Started service [kerberos-client2] in container [kerberos-client2]
[*] Started service [kerberos-client] in container [kerberos-client]
[*] Started service [kerberos-server] in container [kerberos-server]
[*] Started service [apache-server] in container [apache-server]
[*] Started service [ssh-server] in container [ssh-server]
```

The screenshot shows an SSH session with the title "SSH-in-browser". The command `docker-compose up --build` is run, building all four services: kerberos-client, kerberos-server, apache-server, and ssh-server. Each service is built using its respective Dockerfile. The resulting containers are named kerberos-client2, kerberos-client, kerberos-server, and apache-server respectively. The status of each container is shown as "done".

**Screenshot 3:** This shows an output of `docker ps` showing all four containers running with preassigned IP addresses (10.6.0.3-6) and exposed ports. I also have done the ping tests from `kerberos_client2` container to other three containers and it shows successfully reaching all servers, confirming network connectivity and DNS resolution.

```

SSH-in-browser
[luces001207@cy5130-buffer-overflow:~/kerberos$ docker exec -it kerberos_client2 bash
root@client:/# ping -c 2 server.kerberosproject8429.com
PING server.kerberosproject8429.com (10.6.0.5) 56(84) bytes of data.
64 bytes from server.kerberosproject8429.com (10.6.0.5): icmp_seq=1 ttl=64 time=0.392 ms
64 bytes from server.kerberosproject8429.com (10.6.0.5): icmp_seq=2 ttl=64 time=0.073 ms

--- server.kerberosproject8429.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1044ms
rtt min/avg/max/mdev = 0.073/0.232/0.392/0.160 ms
root@client:/# ping -c 2 apache.kerberosproject8429.com
PING apache.kerberosproject8429.com (10.6.0.4) 56(84) bytes of data.
64 bytes from ssh-server.kerberosproject8429.com (10.6.0.4): icmp_seq=1 ttl=64 time=0.259 ms
64 bytes from ssh-server.kerberosproject8429.com (10.6.0.4): icmp_seq=2 ttl=64 time=0.066 ms

--- ssh-server.kerberosproject8429.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1062ms
rtt min/avg/max/mdev = 0.066/0.162/0.259/0.097 ms
root@client:/# ping -c 2 apache.kerberosproject8429.com
PING apache.kerberosproject8429.com (10.6.0.3) 56(84) bytes of data.
64 bytes from apache.kerberosproject8429.com (10.6.0.3): icmp_seq=1 ttl=64 time=0.284 ms
64 bytes from apache.kerberosproject8429.com (10.6.0.3): icmp_seq=2 ttl=64 time=0.091 ms

--- apache.kerberosproject8429.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1031ms
rtt min/avg/max/mdev = 0.091/0.187/0.284/0.097 ms
root@client:/# exit
exit
[luces001207@cy5130-buffer-overflow:~/kerberos$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
379e12bd823 kerberos-client "/bin/bash" 2 minutes ago Up 2 minutes 80/tcp, 88/tcp, 799/tcp, 22/tcp, 89/tcp, 749/tcp kerberos_client2
e95cdd8de8 kerberos-apache-server "/bin/bash" 2 minutes ago Up 2 minutes 80/tcp, 88/tcp, 799/tcp, 22/tcp, 89/tcp, 749/tcp apache-server
85b6a85452d kerberos-kafka-server "/bin/bash" 2 minutes ago Up 2 minutes 22/tcp, 89/tcp, 749/tcp kafka
4a4201d8e33 kerberos-server "/bin/bash" 2 minutes ago Up 2 minutes 80/tcp, 88/tcp, 799/tcp, 22/tcp, 89/tcp, 749/tcp kerberos_server
8eb6a8a5452d kulkarnine/project1.1-ret2l1bc-kulkarni "/bin/bash" 12 days ago Up 12 days 80/tcp, 88/tcp, 799/tcp, 22/tcp, 89/tcp, 749/tcp optimistic_bartik
d49e98547f33 sieranev/buffer-overflow "/bin/bash" 2 weeks ago Up 2 weeks 80/tcp, 88/tcp, 799/tcp, 22/tcp, 89/tcp, 749/tcp buffer-overflow-lab
5cb560a7c7965 sieranev/ret2l1bc "/bin/bash" 2 weeks ago Up 2 weeks 80/tcp, 88/tcp, 799/tcp, 22/tcp, 89/tcp, 749/tcp buffer-overflow
[luces001207@cy5130-buffer-overflow:~/kerberos$ docker network inspect kerberos_vpcbr2 | grep -A 15 "Containers"
"Containers": {
    "379e12bd82312f891d2064357bdd9d0fe2edc6249bf52adde752e0ab9090": {
        "Name": "kerberos_client2",
        "EndpointID": "de2b03065f3f08daa75517f5dd823f3d39cd2e2d1ce7244ca74036b7b083",
        "MacAddress": "02:5a:99:1c:9e:90",
        "IPv4Address": "10.6.0.6/16",
        "IPv6Address": ""
    },
    "4a4201d8e331e536e5cbe379c4a38d545c8de718615796214914f15ea5539f": {
        "Name": "kerberos_server",
        "EndpointID": "de220a10-02-01716:58-02.5038764675",
        "MacAddress": "02:5a:99:1c:9e:90",
        "IPv4Address": "10.6.0.5/16",
        "IPv6Address": ""
    },
    "5cb560a7c79657c1d6ce5c30ab78ff76a21edba0b9ebe59bf5b9caad55ab4327450feb": {
        "Name": "buffer-overflow-lab",
        "EndpointID": "de2b03065f3f08daa75517f5dd823f3d39cd2e2d1ce7244ca74036b7b083",
        "MacAddress": "02:5a:99:1c:9e:90",
        "IPv4Address": "10.6.0.4/16",
        "IPv6Address": ""
    }
},
[luces001207@cy5130-buffer-overflow:~/kerberos$ ]

```

**Screenshot 4:** This shows the successful inspection of `kerberos_vpcbr2` in Docker network.

```

SSH-in-browser
[luces001207@cy5130-buffer-overflow:~/kerberos$ docker network inspect kerberos_vpcbr2
[
    {
        "Name": "kerberos_vpcbr2",
        "Id": "4a4201d8e331e536e5cbe379c4a38d545c8de718615796214914f15ea5539f",
        "Created": "2022-02-01T16:58:02.503876467Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "EnableBridge": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "10.6.0.0/16",
                    "IPRange": "",
                    "Gateway": "10.6.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "Containers": [
            {
                "Container": "379e12bd82312f891d2064357bdd9d0fe2edc6249bf52adde752e0ab9090",
                "Name": "kerberos_client2",
                "EndpointID": "de2b03065f3f08daa75517f5dd823f3d39cd2e2d1ce7244ca74036b7b083",
                "MacAddress": "02:5a:99:1c:9e:90",
                "IPv4Address": "10.6.0.6/16",
                "IPv6Address": ""
            },
            {
                "Container": "4a4201d8e331e536e5cbe379c4a38d545c8de718615796214914f15ea5539f",
                "Name": "kerberos_server",
                "EndpointID": "de220a10-02-01716:58-02.5038764675",
                "MacAddress": "02:5a:99:1c:9e:90",
                "IPv4Address": "10.6.0.5/16",
                "IPv6Address": ""
            },
            {
                "Container": "5cb560a7c79657c1d6ce5c30ab78ff76a21edba0b9ebe59bf5b9caad55ab4327450feb",
                "Name": "buffer-overflow-lab",
                "EndpointID": "de2b03065f3f08daa75517f5dd823f3d39cd2e2d1ce7244ca74036b7b083",
                "MacAddress": "02:5a:99:1c:9e:90",
                "IPv4Address": "10.6.0.4/16",
                "IPv6Address": ""
            }
        ],
        "Status": {}
    }
]

```

## PART 2: KERBEROS SERVER CONFIGURATION IN SERVER CONTAINER

**Screenshot 5:** KDC configuration file **/etc/krb5kdc/kdc.conf** showing that the realm name in the file is the same as the desired realm name (kerberosproject8429.com) for the project and granted with 10-hour ticket lifetime.

```
root@server:/# cat /etc/krb5kdc/kdc.conf
[kdcdefaults]
  kdc_ports = 750,88
[realms]
  kerberosproject8429.com = {
    database_name = /var/lib/krb5kdc/principal
    admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
    acl_file = /etc/krb5kdc/kadm5.acl
    key_stash_file = /etc/krb5kdc/stash
    kdc_ports = 750
    max_life = 10h 0m 0s
    max_renewable_life = 7d 0h 0m 0s
    master_key_type = des3-hmac-sha1
    supported_enctype = aes256-cts:normal aes128-cts:normal
    default_principal_flags = +preauth
  }
root@server:/#
```

## PART 3: KERBEROS DATABASE CONFIGURATION

**Screenshot 6:** Creating Kerberos database and adding user's principals (en (user1), en2 (user2)) and host principal (SSH service, Apache service) using **kadmin.local**.

```
SSH-in-browser
[UPLOAD FILE] [DOWNLOAD FILE] [ ] [ ] [ ] [ ]
kadmin.local: listprincs
KerberosProject8429.com
en2@kerberosproject8429.com
en@kerberosproject8429.com
kadmin/admin@kerberosproject8429.com
kadmin/ssh@kerberosproject8429.com
kadmin/server.kerberosproject8429.com@kerberosproject8429.com
kprop/ssh.kerberosproject8429.com@kerberosproject8429.com
krb5/kerberosproject8429.com@kerberosproject8429.com
kadmin.local: addprinc host@ssh-server.kerberosproject8429.com@kerberosproject8429.com
WARNING: no policy specified for host@ssh-server.kerberosproject8429.com@kerberosproject8429.com; defaulting to no policy
Principal "host@ssh-server.kerberosproject8429.com@kerberosproject8429.com" created.
kadmin.local: addprinc host@HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
WARNING: no policy specified for HTTP/apache.kerberosproject8429.com@kerberosproject8429.com; defaulting to no policy
Principal "HTTP/apache.kerberosproject8429.com@kerberosproject8429.com" created.
kadmin.local: listprincs
HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
KerberosProject8429.com
en2@kerberosproject8429.com
en@kerberosproject8429.com
enkerberosproject8429.com
host@ssh-server.kerberosproject8429.com@kerberosproject8429.com
host@HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
kadmin/admin@kerberosproject8429.com
kadmin/changepw@kerberosproject8429.com
kadmin/server.kerberosproject8429.com@kerberosproject8429.com
kprop/ssh.kerberosproject8429.com@kerberosproject8429.com
kprop/HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
kadmin.local: ktadd -k /etc/ssh-server.keytab host@ssh-server.kerberosprojectNUID.com@kerberosprojectNUID.com does not exist.
kadmin.local: Principal host@ssh-server.kerberosprojectNUID.com@kerberosprojectNUID.com does not exist.
kadmin.local: ktadd -k /etc/ssh-server.keytab host@ssh-server.kerberosprojectNUID.com@kerberosprojectNUID.com does not exist.
kadmin.local: Principal host@ssh-server.kerberosprojectNUID.com@kerberosprojectNUID.com does not exist.
kadmin.local: ktkt -k /etc/ssh-server.keytab
Keytab name: FILE:/etc/ssh-server.keytab
Kvno Timestamp Principal
-----
2 02/01/26 17:11:53 host@ssh-server.kerberosproject8429.com@kerberosproject8429.com
2 02/01/26 17:11:53 host@ssh-server.kerberosproject8429.com@kerberosproject8429.com
root@server:/# cp /etc/ssh-server.keytab /etc/apache.keytab
Keytab name: FILE:/etc/apache.keytab
Kvno Timestamp Principal
-----
2 02/01/26 17:12:14 host@HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
2 02/01/26 17:12:14 host@HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
root@server:/# /etc/init.d/krb5kdc restart
 * Restarting Kerberos KDC krb5kdc
root@server:/# /etc/init.d/krb5-admin-server restart
 * Restarting Kerberos administrative servers kadm5
root@server:/# service krb5-admin-server status
 * krb5kdc is running
root@server:/# service krb5-admin-server status
 * kadm5 is running
root@server:/#
```

**Screenshot 7:** Generating keytab files for SSH and Apache services with AES256/AES128 encryption keys.

```

SSH-in-browser
Enter password for principal "en@kerberosproject8429.com":
Re-enter password for principal "en@kerberosproject8429.com":
Principal "en@kerberosproject8429.com" created.
kadmin.local: addprinc en
WARNING: no policy specified for en@kerberosproject8429.com, defaulting to no policy
kadmin.local: addprinc -randkey en@kerberosproject8429.com
Re-enter password for principal "en@kerberosproject8429.com":
Principal "en@kerberosproject8429.com" created.
kadmin.local: listprincipal
Kerberos Principal: en@kerberosproject8429.com
en@kerberosproject8429.com
en@kerberosproject8429.com
kadmin/admin@kerberosproject8429.com
kadmin@kerberosproject8429.com
kadmin/server.kerberosproject8429.com@kerberosproject8429.com
kripow/server.kerberosproject8429.com@kerberosproject8429.com
krbtgt/kerberosproject8429.com@kerberosproject8429.com
host/sh-server.kerberosproject8429.com@kerberosproject8429.com
WARNING: no policy specified for host/sh-server.kerberosproject8429.com@kerberosproject8429.com, defaulting to no policy
Principal "host/sh-server.kerberosproject8429.com@kerberosproject8429.com" created.
kadmin.local: addprinc -randkey HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
WARNING: no policy specified for host/http/apache.kerberosproject8429.com@kerberosproject8429.com, defaulting to no policy
Principal "HTTP/apache.kerberosproject8429.com@kerberosproject8429.com" created.
kadmin.local: listprincipal
HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
en@kerberosproject8429.com
en@kerberosproject8429.com
host/sh-server.kerberosproject8429.com@kerberosproject8429.com
host/http/apache.kerberosproject8429.com@kerberosproject8429.com
kadmin/charon@kerberosproject8429.com
kadmin/server.kerberosproject8429.com@kerberosproject8429.com
kripow/server.kerberosproject8429.com@kerberosproject8429.com
krbtgt/kerberosproject8429.com@kerberosproject8429.com
host/sh-server.kerberosproject8429.com@kerberosproject8429.com
WARNING: no policy specified for host/sh-server.kerberosproject8429.com@kerberosproject8429.com, defaulting to no policy
Principal "host/sh-server.kerberosproject8429.com@kerberosproject8429.com" created.
kadmin.local: addprinc -randkey HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
WARNING: no policy specified for host/http/apache.kerberosproject8429.com@kerberosproject8429.com, defaulting to no policy
Principal "HTTP/apache.kerberosproject8429.com@kerberosproject8429.com" created.
kadmin.local: listprincipal
host/sh-server.kerberosproject8429.com@kerberosproject8429.com
host/http/apache.kerberosproject8429.com@kerberosproject8429.com
kadmin/charon@kerberosproject8429.com
kadmin/server.kerberosproject8429.com@kerberosproject8429.com
kripow/server.kerberosproject8429.com@kerberosproject8429.com
krbtgt/kerberosproject8429.com@kerberosproject8429.com
host/sh-server.kerberosproject8429.com@kerberosproject8429.com
WARNING: no policy specified for host/sh-server.kerberosproject8429.com@kerberosproject8429.com, defaulting to no policy
Principal "host/sh-server.kerberosproject8429.com@kerberosproject8429.com" created.
kadmin.local: kcat -k /etc/sh-server.keytab
kadmin.local: kcat -k /etc/apache.keytab
Keytab name: FILE:/etc/sh-server.keytab
Keytab name: FILE:/etc/apache.keytab
KWDN Timestamp Principal
-----
2 02/01/26 17:11:55 host/sh-server.kerberosproject8429.com@kerberosproject8429.com
2 02/01/26 17:11:55 host/sh-server.kerberosproject8429.com@kerberosproject8429.com
root@server:~# klist -kt /etc/sh-server.keytab
Keytab name: FILE:/etc/sh-server.keytab
KWDN Timestamp Principal
-----
2 02/01/26 17:12:14 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
2 02/01/26 17:12:14 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
root@server:~#

```

**Screenshot 8:** The Apache and SSH keytab files were successfully copied from the KDC container to the local host for transfer. Verification confirmed that the keytabs contained the correct service principals and that KDC services were running. Because the keytabs are generated inside the kerberos\_server container but must be installed on the SSH and Apache server containers, the host machine serves as an intermediary to move the files between containers.

```

SSH-in-browser
Management: https://landscape.canonical.com
Support: https://ubuntu.com/pro
* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s just raised the bar for easy, resilient and secure K8s cluster deployment.
https://ubuntu.com/engage/secure-kubernetes-at-the-edge

This system has been minimized by removing packages and content that are not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

Expanded Security Maintenance for Applications is not enabled.

8 updates can be applied immediately.

To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

New release '24.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

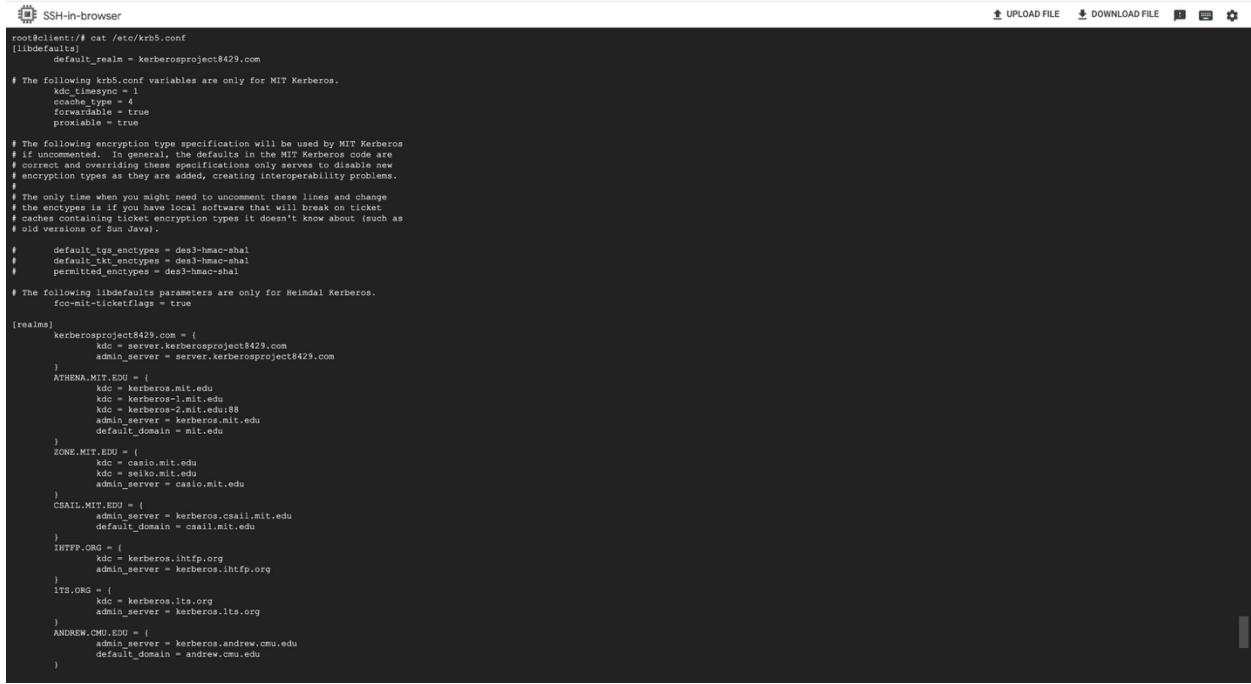
Last login: Sun Feb 1 16:27:30 2026 from 35.235.244.34
lucass001207@y5130-buffer-overflow:~$ cd kerberos
lucass001207@y5130-buffer-overflow:~/kerberos$ docker cp kerberos_server:/etc/sh-server.keytab ..ssh-server.keytab
lucass001207@y5130-buffer-overflow:~/kerberos$ docker cp kerberos_server:/etc/apache.keytab ..apache.keytab
Successfully copied 2.05KB to /home/lucas001207/kerberos/apache.keytab
lucass001207@y5130-buffer-overflow:~/kerberos$ ls -l .keytab
-rw-r--r-- 1 lucass001207 lucass001207 230 Feb 1 17:11 apache.keytab
-rw-r--r-- 1 lucass001207 lucass001207 142 Feb 1 17:11 ssh-server.keytab
lucass001207@y5130-buffer-overflow:~/kerberos$ tree -o kerberos
kerberos
├── apache-server
│   └── Dockerfile
├── apache.keytab
├── client
│   └── Dockerfile
├── docker-compose.yaml
└── server
    ├── Dockerfile
    └── ssh-server
        └── Dockerfile

4 directories, 7 files
lucass001207@y5130-buffer-overflow:~$ cat apache.keytab
cat: apache.keytab: No such file or directory
lucass001207@y5130-buffer-overflow:~$ cat ssh-server.keytab
cat: ssh-server.keytab: No such file or directory
lucass001207@y5130-buffer-overflow:~$ ls -l kerberos
ls: cannot access 'kerberos': No such file or directory
lucass001207@y5130-buffer-overflow:~$ ls -l kerberos
ls: cannot access 'kerberos': No such file or directory
kerberosproject8429.com@HTTPApache.kerberosproject8429.com$ xmllk[4]@Kbkerberosproject8429.comHTTPApache.kerberosproject8429.commin0
NFsd  lucass001207@y5130-buffer-overflow:~$ kerberos$ cat ssh-server.keytab
kerberosproject8429.com$host@ssh-server.kerberosproject8429.com[ s]t-Mhd-N<j5||fkerberosproject8429.com$host@ssh-server.kerberosproject8429.com[Stat28|lucass001207@y5130-buffer-overflow:-/kerberos$ 

```

## PART 4: CLIENT SETUP WITHIN THE CONTAINER

**Screenshot 9:** Client's */etc/krb5.conf* configuration showing default realm (kerberosproject8429.com) and KDC/admin server information is the same as those desired for this project.



```
SSH-in-browser
root@client:/# cat /etc/krb5.conf
[libdefaults]
    default_realm = kerberosproject8429.com

# The following krb5.conf variables are only for MIT Kerberos.
# If uncommented, in general, the defaults in the MIT Kerberos code are
# correct and overriding these specifications only serves to disable new
# encryption types as they are added, creating interoperability problems.
#
# The only time when you might need to uncomment these lines and change
# the encyptypes is if you have local software that will break on ticket
# caches containing ticket encryption types it doesn't know about (such as
# old versions of Sun Java).
#
#     default_tgs_enctypes = des3-hmac-sha1
#     default_tkt_enctypes = des3-hmac-sha1
#     permitted_enctypes = des3-hmac-sha1

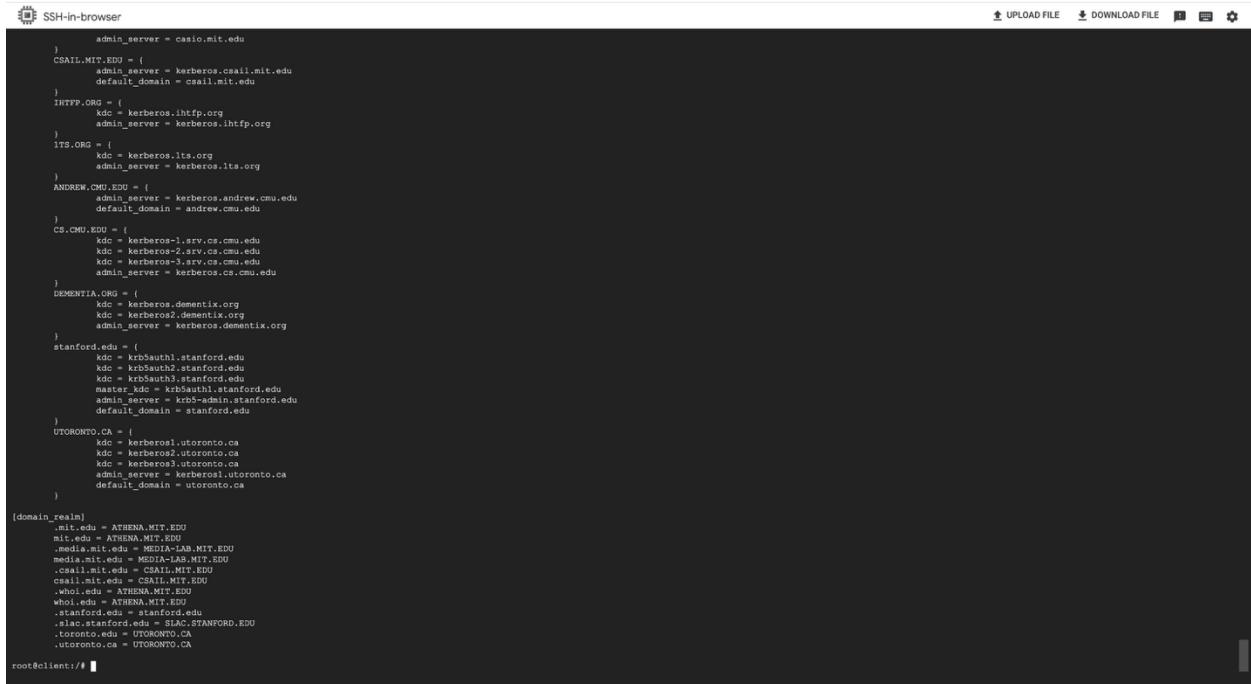
# The following libdefaults parameters are only for Heimdal Kerberos.
    fcc-mit-ticketflags = true

[realms]
    kerberosproject8429.com = {
        kdc = server.kerberosproject8429.com
        admin_server = server.kerberosproject8429.com
    }
    ATHENA.MIT.EDU = {
        kdc = kerberos.mit.edu
        kdc = kerberos-1.mit.edu
        kdc = kerberos-2.mit.edu:88
        admin_server = kerberos.mit.edu
        default_domain = mit.edu
    }
    ZONE.MIT.EDU = {
        kdc = casio.mit.edu
        kdc = seiko.mit.edu
        admin_server = casio.mit.edu
    }
    CSAIL.MIT.EDU = {
        admin_server = kerberos.csail.mit.edu
        default_domain = csail.mit.edu
    }
    INTFP.ORG = {
        kdc = kerberos.intfp.org
        admin_server = kerberos.intfp.org
    }
    ITS.ORG = {
        kdc = kerberos.its.org
        admin_server = kerberos.its.org
    }
    ANDREW.CMU.EDU = {
        admin_server = kerberos.andrew.cmu.edu
        default_domain = andrew.cmu.edu
    }
}

[domain_realm]
    .mit.edu = ATHENA.MIT.EDU
    mit.edu = ATHENA.MIT.EDU
    .media.mit.edu = MEDIA-LAB.MIT.EDU
    media.mit.edu = MEDIA-LAB.MIT.EDU
    .csail.mit.edu = CSAIL.MIT.EDU
    csail.mit.edu = CSAIL.MIT.EDU
    .whoi.edu = ATHENA.MIT.EDU
    whoi.edu = ATHENA.MIT.EDU
    .stanford.edu = stanford.edu
    .slac.stanford.edu = SLAC.STANFORD.EDU
    .utoronto.edu = UTORONTO.CA
    .utoronto.ca = UTORONTO.CA

root@client:/#
```

**Screenshot 10:** Continuation of client */etc/krb5.conf* showing domain-to-realm mappings for automatic realm detection.



```
SSH-in-browser
root@client:/# cat /etc/krb5.conf
    .admin_server = casio.mit.edu
}
    CSAIL.MIT.EDU = {
        admin_server = kerberos.csail.mit.edu
        default_domain = csail.mit.edu
    }
    INTFP.ORG = {
        kdc = kerberos.intfp.org
        admin_server = kerberos.intfp.org
    }
    ITS.ORG = {
        kdc = kerberos.its.org
        admin_server = kerberos.its.org
    }
    ANDREW.CMU.EDU = {
        admin_server = kerberos.andrew.cmu.edu
        default_domain = andrew.cmu.edu
    }
    CS.CMU.EDU = {
        kdc = kerberos-1.cs.cmu.edu
        kdc = kerberos-2.srv.cs.cmu.edu
        kdc = kerberos-3.srv.cs.cmu.edu
        admin_server = kerberos.cs.cmu.edu
    }
    DEMENTIA.ORG = {
        kdc = kerberos.dementix.org
        kdc = kerberos2.dementix.org
        admin_server = kerberos.dementix.org
    }
    stanford.edu = {
        kdc = krb5auth1.stanford.edu
        kdc = krb5auth2.stanford.edu
        kdc = krb5auth3.stanford.edu
        master_kdc = krb5auth1.stanford.edu
        admin_server = krb5-admin.stanford.edu
        default_domain = stanford.edu
    }
    UTORONTO.CA = {
        kdc = kerberos1.utoronto.ca
        kdc = kerberos2.utoronto.ca
        kdc = kerberos3.utoronto.ca
        admin_server = kerberos1.utoronto.ca
        default_domain = utoronto.ca
    }

[domain_realm]
    .mit.edu = ATHENA.MIT.EDU
    mit.edu = ATHENA.MIT.EDU
    .media.mit.edu = MEDIA-LAB.MIT.EDU
    media.mit.edu = MEDIA-LAB.MIT.EDU
    .csail.mit.edu = CSAIL.MIT.EDU
    csail.mit.edu = CSAIL.MIT.EDU
    .whoi.edu = ATHENA.MIT.EDU
    whoi.edu = ATHENA.MIT.EDU
    .stanford.edu = stanford.edu
    .slac.stanford.edu = SLAC.STANFORD.EDU
    .utoronto.edu = UTORONTO.CA
    .utoronto.ca = UTORONTO.CA

root@client:/#
```

## PART 5: SSH-SERVER SETUP WITHIN THE CONTAINER

**Screenshot 11:** I created a user account (en) on the ssh-server container and set a password for this account. After creating the user, I tested it by logging into the server container as this user. I then edited the /etc/ssh/sshd\_config file to enable GSSAPI/Kerberos authentication, with KerberosTicketCleanup set to “no.”

- I configure /etc/ssh/sshd\_config to enable Kerberos authentication because SSH doesn't use Kerberos tickets by default. Without these settings (GSSAPIAuthentication yes, KerberosAuthentication yes), SSH will ignore the user's Kerberos ticket and always prompt for a password, defeating Single Sign-On. The configuration tells SSH to check for and validate Kerberos tickets first using the server's keytab, enabling passwordless authentication when a valid ticket is presented.

```
SSH-in-browser
[stanford.edu]
stanford.edu = {
    kdc = krb5auth1.stanford.edu
    kdc = krb5auth2.stanford.edu
    kdc = krb5auth3.stanford.edu
    master_kdc = krb5auth1.stanford.edu
    admin_kdc = krb5auth1.stanford.edu
    default_domain = stanford.edu
}

[UTORONTO.CA]
UTORONTO.CA = {
    kdc = kerberos1.utoronto.ca
    kdc = kerberos2.utoronto.ca
    kdc = kerberos3.utoronto.ca
    admin_kdc = kerberos1.utoronto.ca
    default_domain = utoronto.ca
}

[domain_realm]
.mit.edu = ATHENA/MIT.EDU
.mit.edu = ATHENA/MIT.EDU
.media.mit.edu = MEDIA-LAB/MIT.EDU
.media.mit.edu = MEDIA-LAB/MIT.EDU
.csail.mit.edu = CSAIL/MIT.EDU
.csail.mit.edu = CSAIL/MIT.EDU
.whois.edu = ATHENA/MIT.EDU
.whois.edu = ATHENA/MIT.EDU
.stanford.edu = STANFORD.EDU
.stanford.edu = STANFORD.EDU
.utoronto.edu = UTORONTO.CA
.utoronto.ca = UTORONTO.CA

root@ssh-server:/# adduser en
Adding user 'en' ...
Adding new group 'en' (1000) ...
Adding new user 'en' (1000) with group 'en' ...
Creating home directory '/home/en' ...
Copying files from '/etc/skel' ...
Enter new UNIX password:
Reenter new password:
password: password updated successfully
Changing the user information for en
Enter the new value, or press ENTER for the default
Full Name []:
Room Number []:
Work Phone []:
Home Phone []:
Other []:
Is the information correct? [y/n] y
root@ssh-server:/# su - en
en@ssh-server:~$ whoami
en
en@ssh-server:~$ exit
logout
root@ssh-server:/# nano /etc/ssh/sshd_config
root@ssh-server:/# grep "GSSAPIAuthentication|KerberosAuthentication|KerberosTicketCleanup|UsePAM" /etc/ssh/sshd_config | grep -v "no"
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
KerberosAuthentication yes
KerberosTicketCleanup no
UsePAM yes
root@ssh-server:/#
```

**Screenshot 12:** SSH keytab (*/etc/krb5.keytab*) file permissions showing 600 (owner read/write only) for security. The krb5.keytab file has 600 permissions (owner read/write only), owned by root:root. These restrictive permissions are critical for security because the krb5.keytab contains the SSH server's secret key used to decrypt Kerberos tickets. If this file were compromised, an attacker could impersonate the SSH server.

```
root@ssh-server:/# service ssh restart
* Restarting OpenBSD Secure Shell server sshd
root@ssh-server:/# service ssh status
* sshd is running
root@ssh-server:/# chmod 600 /etc/krb5.keytab
root@ssh-server:/# chown root:root /etc/krb5.keytab
root@ssh-server:/# ls -la /etc/krb5.keytab
-rw-r----- 1 root root 230 Feb 1 17:11 /etc/krb5.keytab
root@ssh-server:/#
```

**Screenshot 13:** SSH keytab contents displaying host principal with AES256/AES128 encryption keys. Multiple encryption types ensure compatibility with various Kerberos clients. The KVNO

(Key Version Number) of 2 indicates this is the second version of the keytab, allowing for key rotation without breaking existing tickets.

```
root@ssh-server:~# cat /etc/krb5.keytab
kerberosproject8429.comhost"ssh-server.kerberosproject8429.com[ 4]t=Mhd4-NcJ5)fkerberosproject8429.comhost"ssh-server.kerberosproject8429.com[8tatZ8/root@ssh-server:#
root@ssh-server:~# klist -kt /etc/krb5.keytab
Keytab File: /etc/krb5.keytab
KVNO Timestamp Principal
-----+
 2 02/01/26 17:11:55 host/ssh-server.kerberosproject8429.com@kerberosproject8429.com
 2 02/01/26 17:11:55 host/ssh-server.kerberosproject8429.com@kerberosproject8429.com
root@ssh-server:~#
```

## PART 6: APACHE server SETUP WITHIN THE CONTAINER

**Screenshot 14:** Apache Kerberos module installation and keytab permissions showing 640 (root:www-data). The keytab has 640 permissions with root:www-data ownership because Apache runs as the www-data user and needs to read the keytab to validate tickets, but shouldn't be able to write or modify it for security.

```
SSH-in-browser
UPLOAD FILE DOWNLOAD FILE
root@apache:~# cat /etc/krb5.conf
kdc = kerberos1.utoronto.ca
kdc = kerberos2.utoronto.ca
kdc = kerberos3.utoronto.ca
admin_server = kerberos1.utoronto.ca
default_domain = utoronto.ca
}

[domain_realm]
".mit.edu" = ATHENA.MIT.EDU
.mit.edu = ATHENA.MIT.EDU
"media-lab.mit.edu" = MEDIA-LAB.MIT.EDU
.media.mit.edu = MEDIA-LAB.MIT.EDU
.csail.mit.edu = CSAIL.MIT.EDU
.csail.mit.edu = CSAIL.MIT.EDU
"web.mit.edu" = WEB.MIT.EDU
.web.mit.edu = WEB.MIT.EDU
.stanford.edu = stanford.edu
.siac.stanford.edu = SIAC.STANFORD.EDU
.toronto.edu = TORONTO.CA
.utoronto.ca = UTORONTO.CA

root@apache:~# apt-get update
Hit:1 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:3 http://archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
 libapache2-mod-auth-kerb
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
 libapache2-mod-auth-kerb
libapache2-mod-auth-kerb
0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
Need to get 34.0 kB of additional disk space will be used.
After this operation, 101 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/universe libapache2-mod-auth-kerb amd64 5.4-2.3 [34.0 kB]
Fetched 34.0 kB in 0s (76.6 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libapache2-mod-auth-kerb.
(Reading database ... 10629 files and directories currently installed.)
Preparing to unpack .../libapache2-mod-auth-kerb_5.4-2.3_amd64.deb ...
Unpacking libapache2-mod-auth-kerb (5.4-2.3) ...
Setting up libapache2-mod-auth-kerb (5.4-2.3) ...
apache2 invoke-rc.d: Enable module auth_kerb
invoke-rc.d: could not determine current runlevel
invoke-rc.d: policy-rc.d denied execution of restart.
root@apache:~# ls -l /etc/apache2/mods-available/auth_kerb.load
lrwxrwxrwx 1 root root 37 Feb 1 18:06 auth_kerb.load > ../../mods-available/auth_kerb.load
root@apache:~# chmod 640 /etc/apache.keytab
root@apache:~# klist -kt /etc/apache.keytab
Keytab Name: FILE:/etc/apache.keytab
KVNO Timestamp Principal
-----+
 2 02/01/26 17:12:14 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
 2 02/01/26 17:12:14 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
root@apache:~#
```

**Screenshot 15:** Complete Apache Kerberos configuration in */etc/apache2/sites-enabled/000-default.conf*. Create a file to check if Kerberos authentication is working and restrict access to all kerberos principals and allow only en2 to access the website test directory because this user is the only one in the ACL. Setting permissions to 755 for directories and 644 for files ensures www-data can read and serve the content while maintaining security (only root can modify).

```

SSH-in-browser
root@apache:/# klist -kt /etc/apache.keytab
Keytab file: /etc/apache.keytab
KVNO Timestamp
-----+
 2 02/07/26 17:12:44 http@kerberosproject8429.com@kerberosproject8429.com
root@apache:/# mkdir -p /var/www/html/test
root@apache:/# echo "Restricted Access to Kerberos Principals in <8429>" >/var/www/html/test/restricted.test
root@apache:/# chmod 444 /var/www/html/test/restricted.test
root@apache:/# cat /var/www/html/test/restricted.test
root@apache:/# cat /var/www/html/test/restricted.test
Restricted Access to Kerberos Principals in <8429>
root@apache:/# nano /etc/apache2/sites-enabled/000-default.conf
root@apache:/# nano /etc/apache2/sites-enabled/000-default.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # external redirections (with mod_rewrite) and is also used as a default hostname
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    <Directory "/var/www/html/test">
        AuthType Kerberos
        AuthName "KERBEROS AUTHENTICATION"
        KrbAuthRealms kerberosproject8429.com
        KrbKeyTabFile /etc/apache2/keys.keytab
        KrbMethodK5Passwd On
        KrbSaveCredentials Off
        #KrbVerifyRKC Off
        #Require user en2@kerberosproject8429.com
    </Directory>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info as:warn
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables configuration from conf-available/ for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
# vim syntax=apache ts=4 sw=4 sts=4 sr noet
root@apache:/#

```

**Screenshot 16:** This shows a successful restart after the configuration was modified.

```

root@apache:/# apache2ctl configtest
Syntax OK
root@apache:/# service apache2 restart
  Restarting Apache httpd web server apache2
root@apache:/# service apache2 status
 apache2 is running
root@apache:/#
root@apache:/#

```

**Screenshot 17:** I use en (user1) to test the Kerberos authentication, although en(user1) granted the server ticket but because it's not in the ACL. Thus, en (user1) can't be successfully authorized and it returns access denied.

```

SSH-in-browser


By default, Ubuntu does not allow access through the web browser to <a href="anywhere"> file apart of those located in <t>/var/www</t>, <a href="http://www.ubuntu.org/docs/2/howto.html" rel="nofollow">public_html</a> directory (unless enabled via <a href="http://www.ubuntu.org/docs/2/howto.html" rel="nofollow">share</a> [for web applications]. If your site is using a web document root located elsewhere (such as in <t>/src</t>) you may need to whitelist your document root directory in <t>/etc/apache2/apache2.conf</t>.



## Reporting Problems



Please use the <t>ubuntu-bug</t> tool to report bugs in the Apache2 package with Ubuntu. However, check <a href="https://bugs.launchpad.net/ubuntu/+source/apache2" rel="nofollow">existing bug reports</a> before reporting a new bug.



Please report bugs specific to modules (such as PHP and others) to respective packages, not to the web server itself.



root@apache:/# curl http://localhost/test/restricted.test
<!DOCTYPE html PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html>
<head>
<title>401 Unauthorized</title>
</head>
<body>
<h1>Unauthorized</h1>
<p>This server could not verify that you are authorized to access the document requested. Either you supplied the wrong credentials (e.g., bad password), or your browser doesn't understand how to supply the credentials required.</p>
<hr>
<address>Apache/2.4.29 (Ubuntu) Server at localhost Port 80</address>
</body>
</html>
root@apache:/#


```

## PART 7: SSH KERBEROS AUTHENTICATION FROM CLIENT (DOCKER CONTAINER)

**Screenshot 18:** Request a TGT from the KDC for the user (en) principal, and verify if the TGT is being granted with klist. Once the TGT is verified, I initiate an SSH connection using ssh -vvv showing successful GSSAPI authentication without password prompt.

```

SSH-in-browser
lucas001207@en5130-buffer-overflow:~$ docker exec -it kerberos_client2 bash
root@client2:~# kinit en
Password for en@kerberosproject8429.com:
root@client2:~# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: en@kerberosproject8429.com

Valid starting       Expires             Service principal
02/01/26 18:24:08  02/02/26 04:24:08  krbtgt/kerberosproject8429.com@kerberosproject8429.com
root@client2:~# ssh -vvv en@kerberosproject8429.com
OpenSSH_7.6p1 Ubuntu-4ubuntu0.7, OpenSSL 1.0.2n 7 Dec 2017
debug1: Reading configuration data /etc/ssh/sshd_config
debug1: Applying options for *
debug2: reading "sshd_server.kerberosproject8429.com" port 22
debug3: private key: /root/.ssh/id_rsa
debug1: Connecting to en@sshd_server.kerberosproject8429.com [10.6.0.4] port 22.
debug1: Connection established.
debug1: permanently_set_uid: 0/0
debug1: Local version string: OpenSSH_7.6p1 Ubuntu-4ubuntu0.7
debug1: Remote protocol version 2.0, remote software version OpenSSH_7.6p1 Ubuntu-4ubuntu0.7
debug1: match: OpenSSH_7.6p1 Ubuntu-4ubuntu0.7 pat OpenSSH* compat 0x04000000
debug1: Setting up PAM
debug1: Authentication to en@sshd_server.kerberosproject8429.com:22 as 'en'
debug3: send packet: type 20
debug1: SSH2_MSG_KEXINIT sent
debug3: receive packet: type 20
debug1: SSH2_MSG_KEXINIT received
debug2: local client KEXINIT proposal
debug2: KEX algorithms: curve25519-sha256,curve25519-sha256@libssh.org,ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-exchange-sha256,diffie-hellman-group16-sha512,diffie-hellman-group18-sha512,diffie-hellman-group-exchange-sha1,diffie-hellman-group-exchange-sha1,diffie-hellman-group-exchange-sha1-dss,diffie-hellman-group-exchange-sha1-dss+sha1
debug2: host key algorithms: ecdsa-sha2-nistp256,ecdsa-sha2-nistp384,ecdsa-sha2-nistp521,rsa-sha2-256,rsa-sha2-512
debug2: cipher ctoe: chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes192-gcm@openssh.com,aes256-gcm@openssh.com
debug2: cipher stoc: chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,aes192-gcm@openssh.com,aes256-gcm@openssh.com
debug2: MACs ctos: umac-64-etm@openssh.com,umac-128-etm@openssh.com,hmac-sha2-256,hmac-sha2-512,hmac-sha1
debug2: MACs stoc: umac-64-etm@openssh.com,umac-128-etm@openssh.com,hmac-sha2-256-etm@openssh.com,hmac-sha1-etm@openssh.com,umac-64@openssh.com,umac-128@openssh.com,hmac-sha2-256,hmac-sha1
debug2: compression ctos: none,zlib@openssh.com,zlib
debug2: compression stoc: none,zlib@openssh.com,zlib
debug2: languages ctos:
debug2: languages stoc:

```



SSH-in-browser

```
Debug1: receive packet: type 51
Debug1: received packet type 51, contents publickey,gsapi-keyv,gsapi-with-mic,password
Debug1: start over, passed a different list publickey,gsapi-keyv,gsapi-with-mic,password
Debug1: authmethod,loopback,gsapi-with-mic,pubkey,keyboard-interactive,password
Debug1: authmethod,x509v3,gsapi-with-mic,pubkey,publickey,keyboard-interactive,password
Debug1: authmethod,gsapi-with-mic,x509v3,gsapi-with-mic
Debug1: No valid key exchange content in gsapi-keyv
Debug1: we did not send a packet, discard method
Debug1: remaining preferred: publickey,keyboard-interactive,password
Debug1: Next authentication method: gsapi-with-mic
Debug1: we sent a gsapi-with-mic packet, wait for reply
Debug2: send packet type 41
Debug1: send packet type 51
Debug1: send packet type 65
Debug1: Authentication succeeded (gsapi-with-mic).
(Debug) channel 0 new (client-session)
(Debug) channel 0 configured channel_new 0
(Debug) channel 0 open
(Debug) send packet type 90
(Debug) pledge: network
(Debug) send packet type 80
(Debug) client_input_global_request: type hostkeys=0@openssh.com want_reply 0
(Debug) channel_input_confirmation: channel 0: callback start
(Debug) channel_input_confirmation: channel 0: callback done
(Debug) ssh_packet_set_toси set IP_200_0x0
(Debug) channel 0 request pty-req confirm 0
(Debug) send packet type #0
(Debug) ignored env LINES
(Debug) ignored env LOGNAME
(Debug) ignored env MAIL
(Debug) ignored env PWD
(Debug) ignored env TERM
(Debug) ignored env USER
(Debug) ignored env PATH
(Debug) ignore env _PATH
(Debug) channel 0 request shell confirm 1
(Debug) channel_input_confirmation: channel 0: callback done
(Debug) channel_input_confirmation: channel 0: check window 0 read/write
(Debug) receive packet type #0
(Debug) channel 0 request pty-req confirm 0
(Debug) PTY allocation request accepted on channel 0
(Debug) receive packet type #0
(Debug) receive packet type #0
(Debug) channel_input_confirmation: type #0 id 0
(Debug) channel_input_confirmation: type #0 id 0
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-1044-grub #46~14)
* Documentation: https://help.ubuntu.com
* Support: https://ubuntu.com/advantage
This system has been configured for restricted access and content that are
not required on a system that uses do not log in.

To restore this content, you can run the 'unminimize' command.
Last login: Sun Mar 10 01:55:24 2018 from 10.6.0.6
ssh-server@ssh-server:~$
```

**Screenshot 19:** klist output showing both TGT and SSH service ticket after successful connection.

SSH-in-browser

```
To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ensssh-server:~$ hostname
ssh-server.kerberosproject8429.com
ensssh-server:~$ klist
klist: No credentials cache found (filename: /tmp/krb5cc_1000)
ensssh-server:~$ exit
logout
debug3: receive packet: type 98
debug: client_input_channel_req: channel 0 rtype exit-status reply 0
debug1: client input channel req: channel 0 rtype eow@openssh.com reply 0
debug1: channel 0: rcvd eow
debug1: channel 0: close read
debug1: channel 0: output open -> closed
debug3: receive packet: type 96
debug2: channel 0: rcvd eof
debug2: channel 0: output open -> drain
debug2: channel 0: close write
debug2: channel 0: output drain -> closed
debug3: receive packet: type 97
debug1: channel 0: will not send data after close
debug3: channel 0: almost dead
debug3: channel 0: gct notify user
debug1: channel 0: gct notify detached
debug2: channel 0: send close
debug3: send packet: type 97
debug2: channel 0: is dead
debug1: channel 0: destroy collecting
debug1: channel 0: free: client-session, nchannels 1
debug3: channel 0: status: The following connections are open:
 #0 client-session (l4 r0 i3/0 o3/0 fd -1/-1 cc -1)

debug3: send packet: type 1
Connection to ssh-server.kerberosproject8429.com closed.
Transferred: sent 3236, received 4032 bytes, in 51.1 seconds
bytes per second: sent 63.4, received 79.0
debug3: Exit status 1
root@client:/# klist
Ticket cache FILE:/tmp/krb5cc_0
Default principal: en@kerberosproject8429.com

Valid starting Expires Service principal
02/01/26 18:24:08 02/02/26 04:24:08 krbtgt/kerberosproject8429.com@kerberosproject8429.com
02/01/26 18:24:52 02/02/26 04:24:08 host/ssh-server.kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:24:04
02/01/26 18:24:52 02/02/26 04:24:08 host/ssh-server.kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:24:04
root@client:/#
```

## PART 8: APACHE KERBEROS AUTHENTICATION FROM CLIENT (DOCKER CONTAINER)

**Screenshot 20:** Authorized access succeeded for *en2*, and klist displays the final tickets. The curl command successfully accessed the restricted content. Content displayed: "Restricted

Access to Kerberos Principals \n <8429>."The klist output shows both the TGT and the HTTP service ticket for the user *en2*.

```
root@client:/# kinit en2
Password for en2@kerberosproject8429.com:
root@client:/# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: en2@kerberosproject8429.com

Valid starting Expires Service principal
02/01/26 18:32:24 02/02/26 04:32:24 krbtgt/kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:32:21
root@client:/# curl "http://apache.kerberosproject8429.com/test/restricted.test" -u : --negotiate
Restricted Access to Kerberos Principals \n <8429>
root@client:/# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: en2@kerberosproject8429.com

Valid starting Expires Service principal
02/01/26 18:32:24 02/02/26 04:32:24 krbtgt/kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:32:21
02/01/26 18:32:41 02/02/26 04:32:24 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:32:21
02/01/26 18:32:49 02/02/26 04:32:24 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:32:21
root@client:/#
```

**Screenshot 21:** This screenshot shows the complete authentication and authorization flow for Apache Kerberos access. First, en (user1) obtains a TGT via kinit and attempts to access the restricted directory, but receives a 401 Unauthorized response because en is not in the Apache ACL, demonstrating that authentication alone does not grant authorization. After using kdestroy to clear all tickets (confirmed by klist showing no credentials), en2 (user2) obtains a fresh TGT and successfully accesses the restricted content, returning "Restricted Access to Kerberos Principals <8429>." A final klist confirms that both the TGT and the HTTP service ticket were issued for en2.

```
SSH in-browser
UPLOAD FILE DOWNLOAD FILE
root@client:/# curl "http://apache.kerberosproject8429.com/test/restricted.test" -u : --negotiate
HTTP/1.1 401 Unauthorized
Content-Type: text/html; charset=ISO-8859-1
Content-Length: 20
Date: Wed, 01 Feb 2026 18:32:40 GMT
Server: Apache/2.4.29 (Ubuntu)
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html>
<head>
<title>401 Unauthorized</title>
</head>
<body>
<h1>Unauthorized</h1>
<p>This server could not verify that you
are not a computer program. To protect
our systems from abuse, this action
was required. Either you supplied the wrong
credentials (e.g., bad password), or your
browser doesn't understand how to supply
the credentials required.</p>
<hr>
<address>Apache/2.4.29 (Ubuntu) Server at apache.kerberosproject8429.com Port 80</address>
</body></html>
root@client:/# echo ""
root@client:/# echo "==== DESTROYING on TICKETS ==="
==== DESTROYING on TICKETS ===
root@client:/# kdestroy
root@client:/# klist
klist: No credentials cache found (filename: /tmp/krb5cc_0)
root@client:/# echo ""

root@client:/# echo "==== TEST 2: AUTHORIZED USER (en2) ==="
==== TEST 2: AUTHORIZED USER (en2) ===
root@client:/# curl "http://apache.kerberosproject8429.com/test/restricted.test" -u : --negotiate
HTTP/1.1 200 OK
Content-Type: text/html; charset=ISO-8859-1
Content-Length: 20
Date: Wed, 01 Feb 2026 18:36:00 GMT
Server: Apache/2.4.29 (Ubuntu)
<html>
<head>
<title>Restricted Access to Kerberos Principals \n <8429></title>
</head>
<body>
<h1>Restricted Access to Kerberos Principals \n <8429></h1>
<hr>
<address>Apache/2.4.29 (Ubuntu) Server at apache.kerberosproject8429.com Port 80</address>
</body></html>
root@client:/# echo "==== FINAL TICKET CHECK ==="
==== FINAL TICKET CHECK ===
root@client:/# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: en2@kerberosproject8429.com

Valid starting Expires Service principal
02/01/26 18:36:00 02/02/26 04:35:49 krbtgt/kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:35:45
02/01/26 18:36:00 02/02/26 04:35:49 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:35:45
02/01/26 18:36:00 02/02/26 04:35:49 HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
    renew until 02/02/26 18:35:45
root@client:/#
```

**Screenshot 22:** en2 successfully accesses restricted content showing "NUID: 8429", with klist displaying both TGT and HTTP service ticket.

```

root@client:/# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: en@kerberosproject8429.com

Valid starting       Expires                 Service principal
02/01/26 18:35:49    02/02/26 04:35:49  krbtgt/kerberosproject8429.com@kerberosproject8429.com
02/01/26 18:36:00    02/02/26 04:35:49  HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
        renew until 02/02/26 18:35:48
02/01/26 18:36:00    02/02/26 04:35:49  HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
        renew until 02/02/26 18:35:48

root@client:/# klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: en@kerberosproject8429.com

Valid starting       Expires                 Service principal
02/01/26 18:45:00    02/02/26 04:45:00  krbtgt/kerberosproject8429.com@kerberosproject8429.com
02/01/26 18:45:00    02/02/26 04:45:00  HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
        renew until 02/02/26 18:44:56
02/01/26 18:45:00    02/02/26 04:45:00  HTTP/apache.kerberosproject8429.com@kerberosproject8429.com
        renew until 02/02/26 18:45:48

root@client:/# curl "http://apache.kerberosproject8429.com/test/restricted.test" -u : --negotiate
HTTP/1.1 200 OK
Date: Mon, 01 Feb 2026 18:45:00 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 14
Server: Apache/2.4.29 (Ubuntu) Server at apache.kerberosproject8429.com Port 80</address>
<html><head>
<title>Unauthorized</title>
</head><body>
<h1>Unauthorized</h1>
Sorry, you do not yet verify that you
are authorized to access the document
requested. Either you supplied the wrong
credentials (e.g., bad password), or your
browser does not support how to supply
the credentials required.</p>
<br>
<address>Apache/2.4.29 (Ubuntu) Server at apache.kerberosproject8429.com Port 80</address>
</body></html>
root@client:/#

```

## QUESTIONS:

**Why do we use the docker network? How does the network help us to imitate the described scenario?**

- We use Docker network to create an isolated environment for four containers to communicate like separate servers on an internal network. It assigns static IPs (10.6.0.3-6) and DNS resolution, necessary for Kerberos's hostname-based service principals. This setup mimics an organization with multiple Linux servers (via SSH and Apache containers) managed by administrators (client container) needing centralized KDC authentication. Without Docker network, containers couldn't communicate, preventing Kerberos authentication and Single Sign-On demonstration. The network enables containers to function as an enterprise infrastructure, allowing administrators to authenticate once and access all services, fulfilling the organization's goal of centralized Kerberos authentication.

**You generated a ssh-server.keytab file on the kerberos KDC server.**

**(A) Why did you need to copy it into the SSH server?**

- The SSH server needs the keytab file locally to validate Kerberos tickets without contacting the KDC each time. When a client presents a service ticket, the SSH server decrypts it using the secret key in the keytab to verify and extract the username. The keytab contains the same key used by the KDC, creating a shared secret between them. Without the keytab on the SSH server, ticket validation would be impossible, forcing password fallback and weakening security. Local validation is safer (passwords aren't transmitted) and more efficient (no network to KDC).

**(B) How many of these .keytab files are required in an architecture like ours and where are they located?**

- Our architecture requires one keytab per service, totaling 2: one for the SSH server at /etc/krb5.keytab, with the principal host/ssh server.kerberosproject8429.com@kerberosproject8429.com, and one for the Apache server at /etc/apache.keytab, with the principal HTTP/apache.kerberosproject8429.com@kerberosproject8429.com. Scaling to 7 SSH and 7 Apache servers would need 14 keytabs, as each has a unique hostname and principal. User principals (en, en2) do not have keytabs, as humans authenticate by password, unlike services.

**(C) How are they protected?**

- Keytab files are protected with restrictive permissions because they contain secret keys that could allow impersonation. The SSH server's keytab has 600 permissions (-rw----) with root: root ownership, so only root can access it, as the SSH daemon runs as root. The Apache server's keytab has 640 permissions (-rw-r----) with root:www-data ownership, allowing the www-data user to read it while preventing non-root modifications. Additional protections include physical security, network isolation, SELinux/AppArmor, and regular keytab rotation. If stolen, an attacker could decrypt tickets and impersonate the server, so permissions enforce the principle of least privilege; only authorized services have access to the key.

**The organization described in the scenario has 8 new administrators and extends their apache web servers' architecture from 2 to 7. What should they do? You can use the names of the parts of this assignment to simplify your explanation (e.g.: Part 2 should be completed but including the new host in the configuration file such as: ...)**

- To accommodate 8 new administrators and scale from 2 to 7 Apache servers, the organization should first update the infrastructure (Part 1) by adding 5 new Apache server containers with unique static IPs, hostnames, and DNS entries required for Kerberos principal resolution. The KDC configuration (Part 2) should be verified to ensure the new hostnames are resolvable. Following Part 3, use kadmin.local to create 8 new administrator principals (addprinc admin3 through admin10) and 5 new service principals (addprinc -randkey HTTP/apache3.company.com through apache7.company.com), generating individual keytabs for each new server. Each administrator's client (Part 4) should have /etc/krb5.conf configured to point to the KDC for TGT retrieval. For Part 6, each new Apache server needs Kerberos packages installed (krb5-user, libapache2-mod-auth-kerb), /etc/krb5.conf configured to point to the KDC, the appropriate keytab copied with 640 permissions (root:www-data), the VirtualHost configured with Kerberos authentication directives, and the ACL updated (Require user) to include the new administrators. Finally, local user accounts should be created with

adduser on servers where shell access is needed, though passwords are unnecessary since Kerberos handles authentication centrally.

**Assume Northeastern University is adopting Kerberos when providing services. Explain how it works.**

- If Northeastern University adopts Kerberos, it would establish a Kerberos realm (e.g., NEU.EDU) with centralized Key Distribution Center (KDC) servers that store credentials for all users. Students log in once with kinit, obtaining a Ticket Granting Ticket (TGT) valid for the day. This TGT enables seamless access to email, Canvas, library systems, Wi-Fi, SSH, and printing, without re-entering passwords, because service-specific tickets are automatically requested from the KDC. Each service has a keytab file to verify tickets, confirming user identity without revealing passwords. When accessing a service like Canvas, the browser requests a ticket from the KDC using the TGT, and Canvas validates the ticket with its keytab, granting access transparently. This Single Sign-On system reduces password management, enhances security by transmitting only passwords to the KDC, and centralizes access control, so revoking a single account disables all access.