v1.10

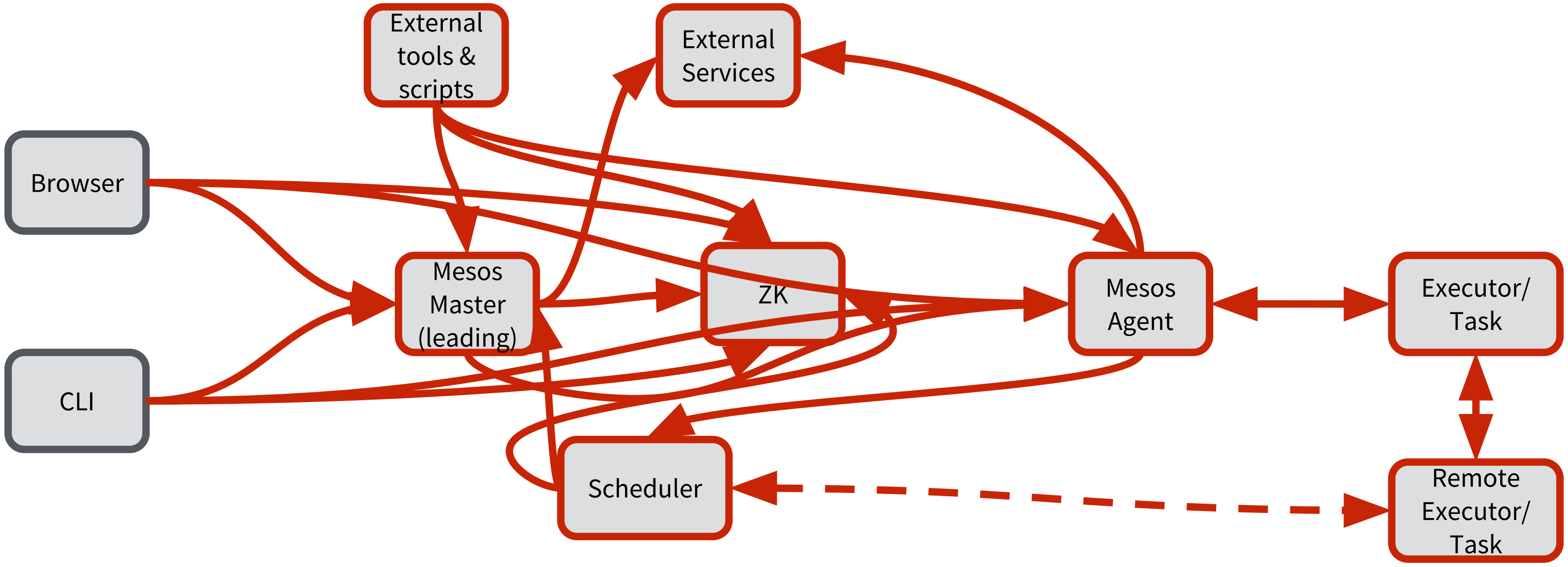
SECURITY

MESOSPHERE

AGENDA

1. Review
2. DC/OS IAM
3. Cluster Security Modes
4. Cluster Security Auditing
5. Secrets
6. CLI

ATTACK VECTORS



SECURITY REQUIREMENTS

Access to Mesos cluster = access to hundreds of root shells!

Multitenancy

- Multiple groups using the same cluster
- Users within a group with different roles

Compliance

- Security regulations (PCI, HIPAA)
- Auditing

ENTERPRISE DC/OS SECURITY

Isolate

Isolate the cluster perimeter with strong authentication & authorization across all interfaces

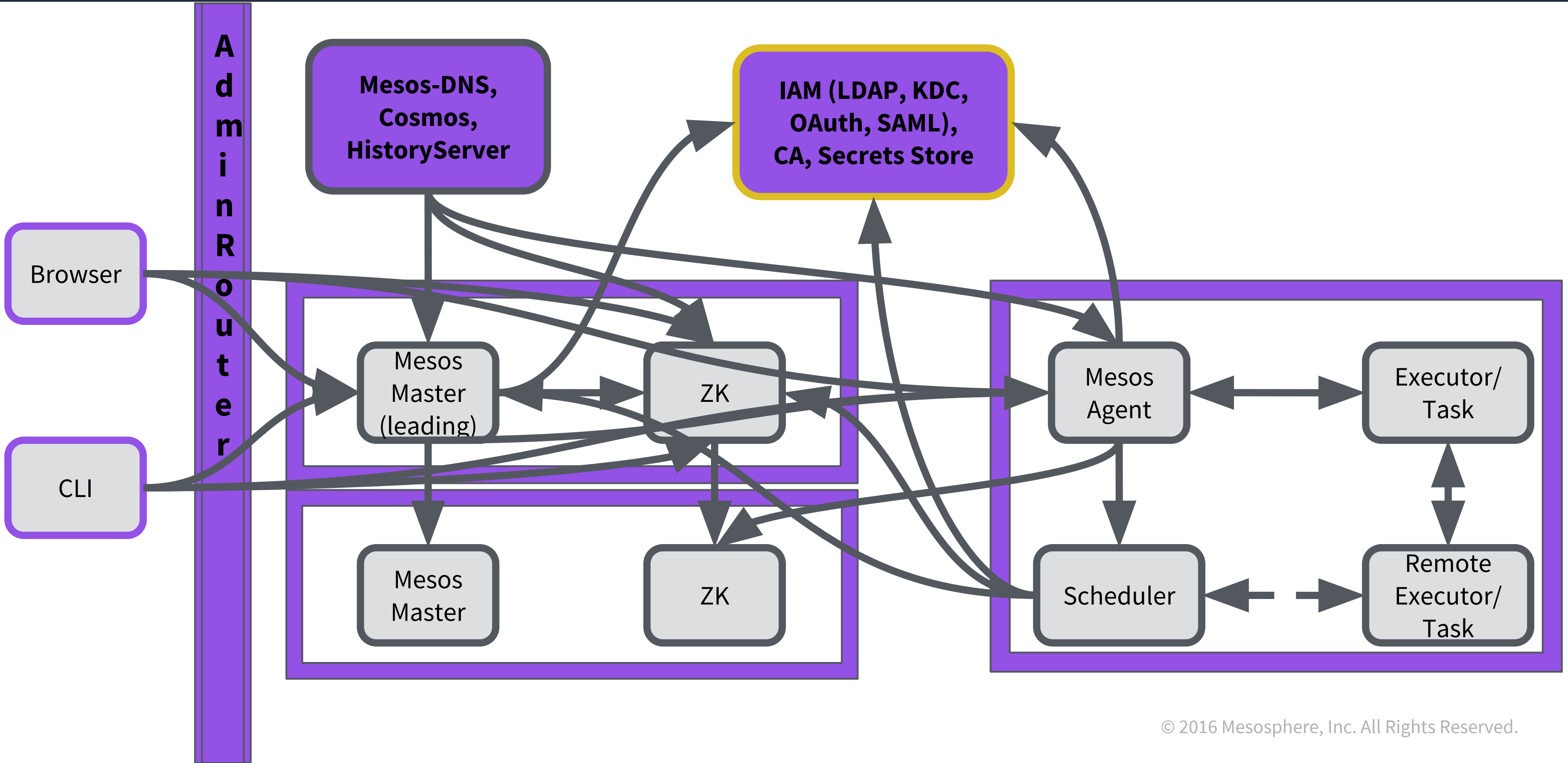
Protect

Secure & protect cluster internal communication

Enhance

Enhance cluster security with advanced 3rd Party Security Service integrations

DC/OS SECURITY



Security

DC/OS CLUSTER SECURITY

SECURITY MODE - DISABLED

- Connections from outside cluster
  - HTTP connections are not redirected to HTTPS. However, both HTTP and HTTPS connections will be served.
- systemd-started service communications
  - unencrypted
- User service communications
  - unencrypted

SECURITY MODE - PERMISSIVE (DEFAULT)

- Connections from outside cluster
    - HTTP connections to the root cluster URL are redirected to HTTPS. HTTP connections that include a path (i.e., http://cluster-url.com/path/) are not redirected to HTTPS.
- systemd-started service communications
    - encrypted
- User service communications
    - Encryption optional for Cassandra, Confluent, DSE, HDFS, Kafka, and Spark.

SECURITY MODE - STRICT

- Connections from outside cluster
  - All HTTP connections to the root cluster URL are redirected to HTTPS
- systemd-started service communications
  - encrypted
- User service communications
  - Encryption required to access any DC/OS or Mesos API. Of the packages in the default Universe available to install, only the following support encryption: Cassandra, Confluent, DSE, HDFS, Kafka, and Spark

SECURITY MODE COMPARISON

config.yaml flag	Encryption	Scheduler service authentication	Mesos and Marathon resource authentication	Mesos master/agent permissions	Default Linux user
`disabled`	Disabled, except for HTTPS requests from outside of the cluster	Disabled	Disabled	Disabled	`root`
`permissive`	Optional	Optional	Required	Disabled	`root`
`strict`	Required	Required	Required	Enabled	`nobody`

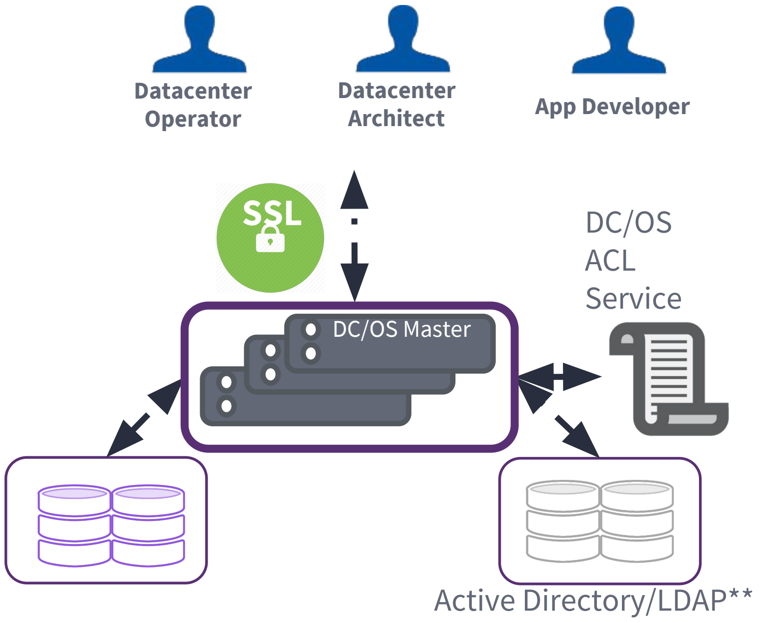
Security

IDENTITY & AUTHORIZATION MANAGEMENT

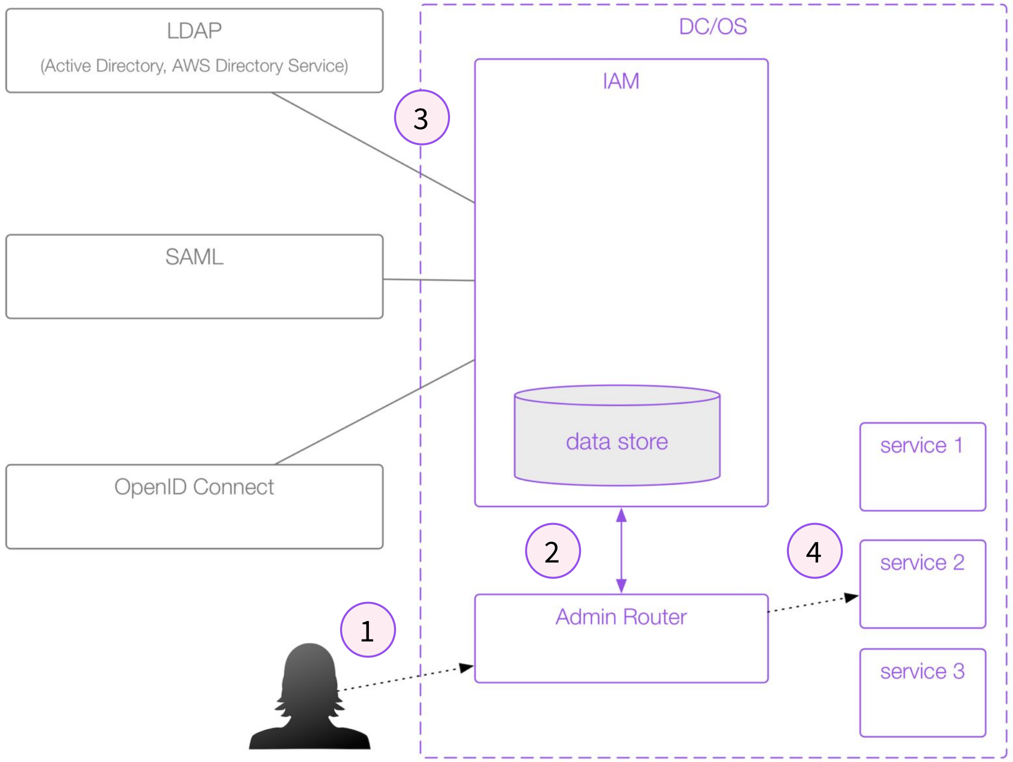
DC/OS IAM

User identity management with fine-grained access control

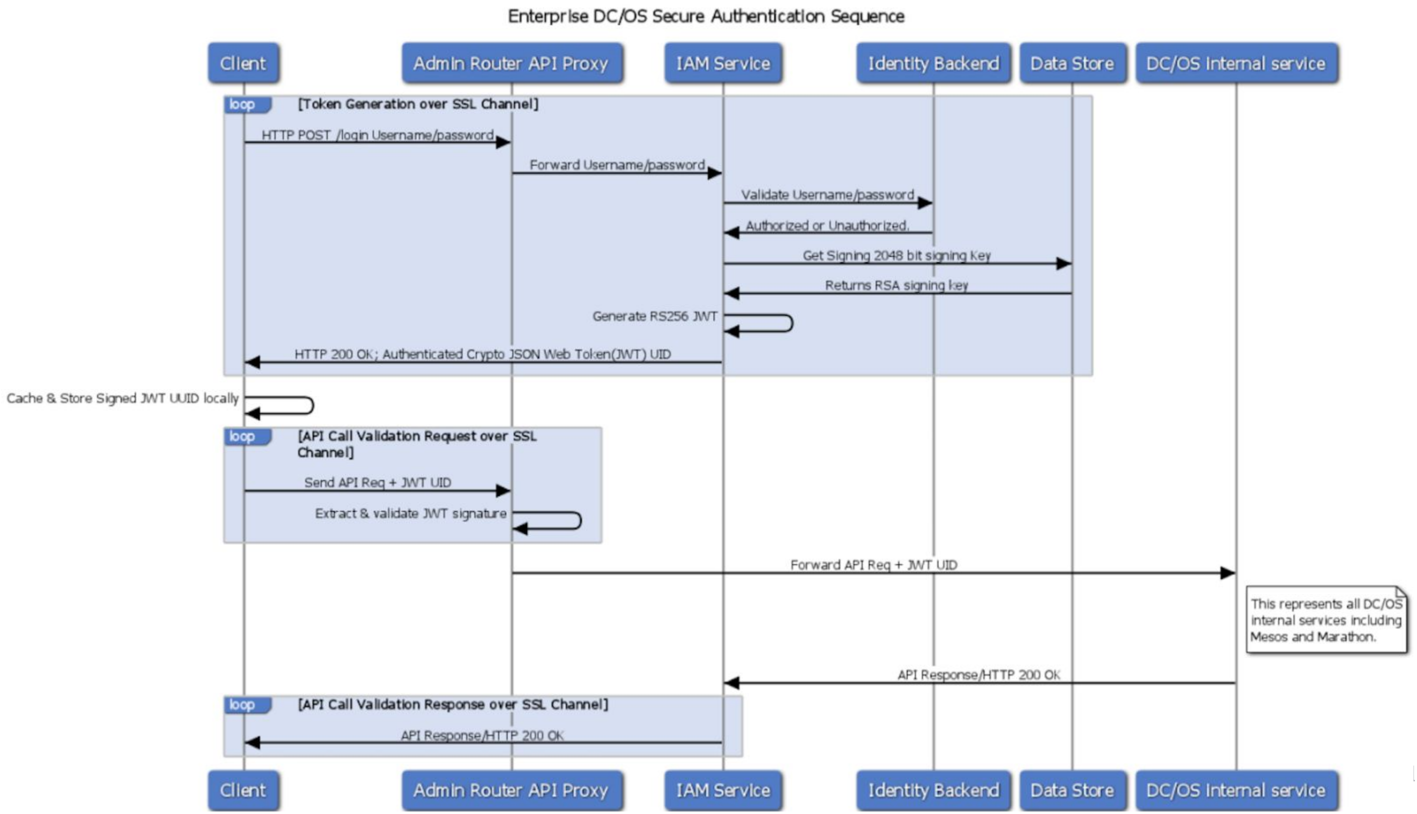
- Centralized access and authorization management
- Access logs to enable compliance with audit requirements
- Unified User & User Groups management across DC/OS API, UI & CLI.
- Secure SSL-based authentication, service level RBAC
- Integration with Corporate LDAP, SAML, OpenID, O-Auth & Kerberos



WORKFLOW


LDAP
(Active Directory, AWS Directory Service)
SAML
OpenID Connect
DC/OS
IAM
data store
Admin Router
service 1
service 2
service 3
1
2
3
4

AUTHENTICATION


Enterprise DC/OS Secure Authentication Sequence
Client
Admin Router API Proxy
IAM Service
Identity Backend
Data Store
DC/OS Internal service
loop
[Token Generation over SSL Channel]
HTTP POST /login Username/password
Forward Username/password
Validate Username/password
Authorized or Unauthorized.
Get Signing 2048 bit signing Key
Returns RSA signing key
Generate RS256 JWT
HTTP 200 OK; Authenticated Crypto JSON Web Token(JWT) UID
Cache & Store Signed JWT UUID locally
loop
[API Call Validation Request over SSL Channel]
Send API Req + JWT UID
Extract & validate JWT signature
Forward API Req + JWT UID
This represents all DC/OS internal services including Mesos and Marathon.
API Response/HTTP 200 OK
loop
[API Call Validation Response over SSL Channel]
API Response/HTTP 200 OK
Client
Admin Router API Proxy
IAM Service
Identity Backend
Data Store
DC/OS Internal service

USERS

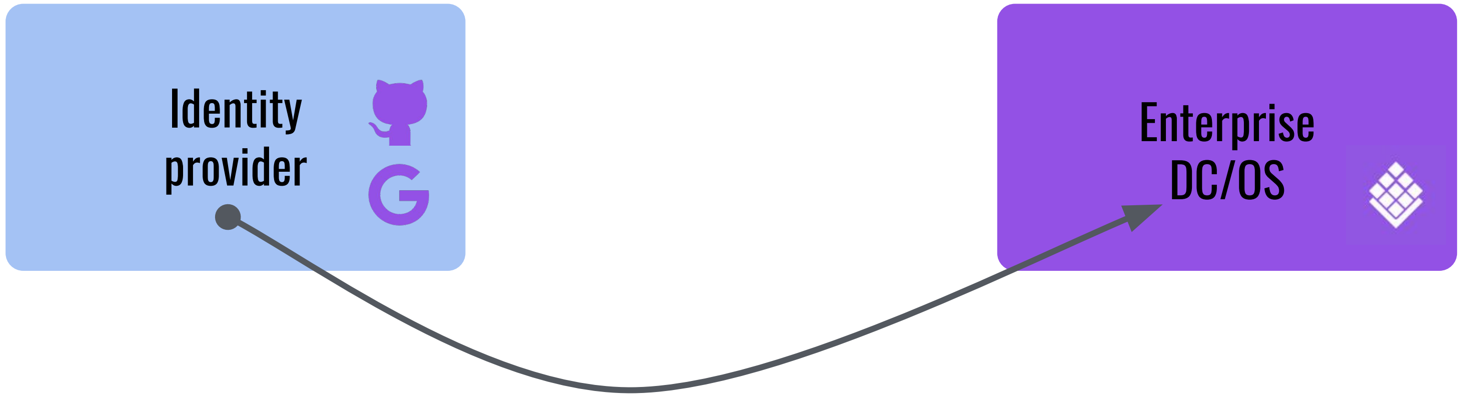
There are two types of users

- Local users - exist only in DC/OS
- External users - delegate verification of credentials to LDAP, SAML, or OpenID Connect

Create and import groups to simplify the addition of external users

SSO W/ SAML & OPEN ID CONNECT

- Integrate with existing identity providers such as Github, Google, Azure, Onelogin, OKTA, etc.
- Support SAML & OAuth2 support
- Easily import existing SAML users into DC/OS user groups



PERMISSIONS

Permissions can be assigned on users or groups. The types of permissions that can be assigned are:

- Systemd service permissions
  - `dcos:[enforcer]:[interface]:[component|service|package]`
- User service permissions
  - `dcos:[enforcer]:[enforcer]:[interface]`
- Master/Agent permissions
  - `dcos:[enforcer]:[master/agent]:[interface]`
- Superuser permissions
  - `dcos:superuser`

Actions: `Create`, `Read`, `Update`, `Delete`, `Full` (depending on object)

SYSTEMD SERVICES

Permissions for the API and web interfaces of DC/OS systemd services

- `dcos:adminrouter:ops:ca:ro`
- `dcos:adminrouter:ops:ca:rw`
- `dcos:adminrouter:ops:exhibitor`
- `dcos:adminrouter:ops:historyservice`
- `dcos:adminrouter:ops:mesos`
- `dcos:adminrouter:ops:mesos-dns`
- `dcos:adminrouter:ops:mesos-dns`
- `dcos:adminrouter:ops:networking`
- `dcos:adminrouter:package`
- `dcos:adminrouter:ops:slave`
- `dcos:adminrouter:ops:system-health`

USER SERVICES

To access a user service, two permissions are needed:

1. Admin Router permission to service or job
2. Marathon or Metronome permission

- `dcos:adminrouter:service:marathon`
- `dcos:service:marathon:marathon:services[:/group]`
- `dcos:service:marathon:marathon:admin:config`
- `dcos:service:marathon:marathon:admin:leader`
- `dcos:service:marathon:marathon:admin:events`

MASTER / AGENT SERVICES

Control access to mesos tasks / endpoints on master or agent nodes.
These are disabled on all modes other than `security:strict`

- `dcos:mesos:master:framework:role[:role_name]`
- `dcos:mesos:master:framework:principal[:service_account_id]`
- `dcos:mesos:master:executor:app_id[:namespace]`
- `dcos:mesos:master:quota:role[:role_name]`
- `dcos:mesos:master:reservation:role[:role_name]`
- `dcos:mesos:master:reservation:principal[:service_account_id]`
- `dcos:mesos:master:task:user[:linux_user_name]`
- `dcos:mesos:master:task:app_id[:namespace]`
- `dcos:mesos:master:volume:principal[:service_account_id]`
- `dcos:mesos:master:volume:role[:role_name]`
- `dcos:mesos:master:weight:role[:role_name]`
- `dcos:mesos:master:flags`
- `dcos:mesos:master:log`
- `dcos:mesos:master:endpoint:path[:path]`

IAM API

Manage users, user groups, permissions, and LDAP configuration

- Include application/json as the Content-Type in the HTTP header
- Requires `dcos:superuser` permission and auth token
- `https://<master-ip>/acs/api/v1`
  - `/login`
  - `/jwks`
  - `/oidc`
  - `/saml`
  - `/ldap`
  - `/users`
  - `/groups`
  - `/permissions`

Lab 7a

DC/OS IAM

LAB 7A - DC/OS IAM

1. Login to the DC/OS UI as a superuser
2. Go to **System** -> **Services** and create a new service group `dev`
3. Go to **System** -> **Organization** and create a new user `alice`
4. Click on **Groups** and create a new user group `developers`
5. Click on the group and click on **Add Permission**
6. Add five permissions:

```
dcos:service:marathon:marathon:services:/dev #actions: Allow All
dcos:adminrouter:service:marathon #actions: Full
dcos:adminrouter:ops:slave #actions: Full
dcos:adminrouter:ops:mesos #actions: Full
dcos:secrets:default:dev #actions: Allow All
```

7. Click on the user group `developers` and add user `alice`
8. Logout and login as `alice`

LAB 7A - DC/OS IAM API

1. Use the DCOS CLI to login to the cluster

```
dcos auth login
```

2. View all users

```
curl -H "Authorization: token=$(dcos config show core.dcos_acs_token)" $(dcos config show core.dcos_url)/acs/api/v1/users | jq .
```

3. View all groups

```
curl -H "Authorization: token=$(dcos config show core.dcos_acs_token)" $(dcos config show core.dcos_url)/acs/api/v1/groups | jq .
```

4. View all permissions

```
curl -H "Authorization: token=$(dcos config show core.dcos_acs_token)" $(dcos config show core.dcos_url)/acs/api/v1/acls | jq .
```

SYSTEMD SERVICE AUTHENTICATION

- In strict or permissive mode, DC/OS services on the master node are automatically provisioned with credentials and permissions during bootstrap sequence.
- The DC/OS Certificate Authority does not appear in any lists of trusted certificate authorities.
- Requests coming in from outside the cluster, such as from a browser or cURL, will result in warning messages.

SERVICE AUTHENTICATION

- Not all services in the Universe support authentication
- For Cassandra, Confluent, DSE, HDFS, and Kafka services to use encryption, set them up to authenticate with the Mesos master
- To configure one of the Universe services that supports encryption to use it in `security:permissive` or `security: strict` modes, create a config.json file and perform a custom install.

```
{
  "service": {
    "principal":
"service-account-id",
    "secret_name": "secret-name"
  }
}
```

CERTIFICATES MANAGEMENT

- The DC/OS Certificate Authority signs the certificates and provisions them to systemd-started services during the bootstrap sequence, accomplishing encrypted communications with no manual intervention.
- The DC/OS Certificate Authority does not appear in any lists of trusted certificate authorities.
- Requests coming in from outside the cluster, such as from a browser or cURL, will result in warning messages.
- API for getting and listing certs, creating and signing CSRs

CERTIFICATE AUTHORITY API

Manage TLS certifications used by Enterprise DC/OS

- Include `application/json` as the Content-Type in the HTTP header
- Requires `dcos:superuser` permission and auth token
- `https://<master-ip>/ca/api/v2`
    - `/info`
    - `/certificates`
    - `/sign`
    - `/newkey`

CONFIGURING DEFAULT LINUX USER

Container type	**Security mode: disabled**	**Security mode: permissive**	**Security mode: strict**
Mesos	Task runs under `root` Fetched/created files owned by `root`	Task runs under `root` Fetched/created files owned by `root`	Task runs under `nobody` Fetched/created files owned by `nobody`
Docker	Task runs under `root` Fetched/created files owned by `root`	Task runs under `root` Fetched/created files owned by `root`	Task runs under `root` Fetched/created files owned by `nobody`

CONFIGURING DEFAULT LINUX USER

```
{
     "id": "linux-user-override",
     "cmd": "whoami && tee file && sleep 1000",
     "user": "<your-test-user-account>",
     "uris": [
          "https://dcos.io/assets/images/logos/mesosphe
re.svg"
     ]
}
```

Security

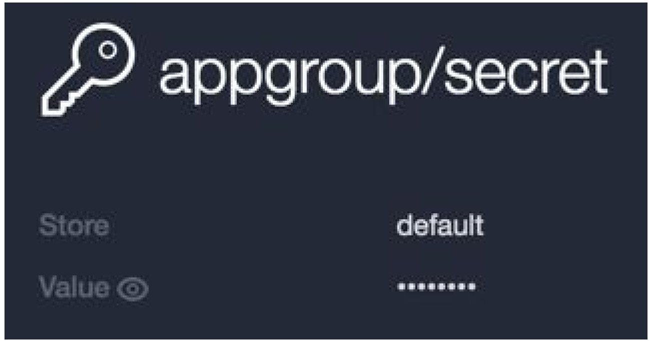
SECRETS

SECRETS MANAGEMENT

- Integrated Secrets Management Service
- Built-in encrypted secret store based on Vault
- ACL based secret authorization controls
- Dynamic secret injection into containers

CREATING SECRETS

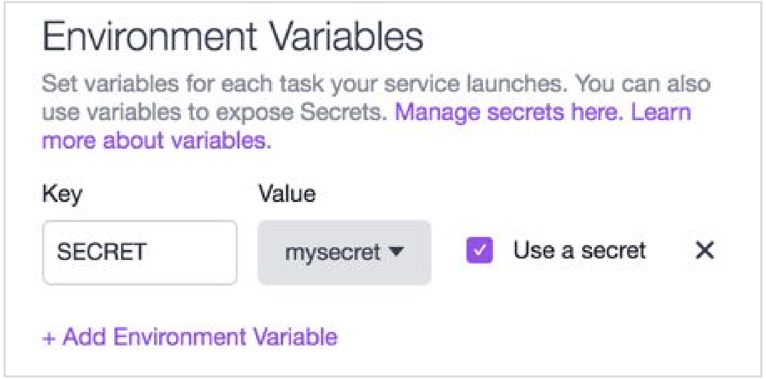
1. Log into the DC/OS UI
2. Navigate to **System -> Security -> Secrets**
3. Click New Secret
4. In the ID field, provide path to secret (i.e. appgroup/secret)


appgroup/secret

Store	default
Value	•••••••••

CONFIGURING SERVICE

1. Log into the DC/OS UI as superuser
2. Click **Services > Deploy**
3. For the app id, match the path that the secret is stored under (i.e. /**appgroup**/<app-id>)
4. Click on **Environment Variables**
5. Check **Use a Select**
6. Select a secret and provide a key
7. Deploy the service


Environment Variables

Set variables for each task your service launches. You can also use variables to expose Secrets. Manage secrets here. Learn more about variables.

Key	Value		
SECRET	mysecret	✓ Use a secret	✕

SECRETS API

Manage secrets and perform admin operations

- Include `application/json` as the Content-Type in the HTTP header
- Requires `dcos:superuser` permission and auth token
- `https://<master-ip>/secrets/v1`
  - `/init/{store}`
  - `/seal-status/{store}`
  - `/unseal/{store}`
  - `/secret/{store}/{path/to/secret}`
  - `/store`
  - `/revoke/{store}/{path/to/secret}`
  - `/renew/{store}/{path/to/secret}`

Security and Authentication

FILE-BASED SECRETS

Overview:

- Available in application containers
- Encryption support for Kerberos `keytab` files
- Configuration with the DC/OS CLI
- Maximum file size of 1 MB

Security and Authentication

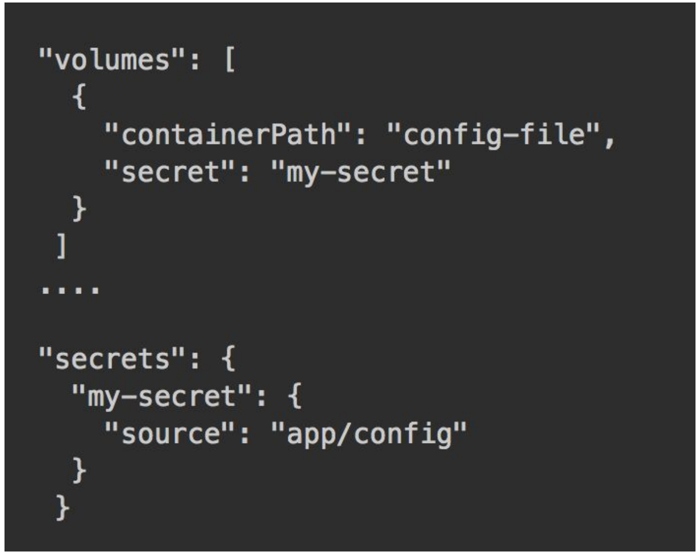
FILE-BASED SECRETS

Configuration Parameters in config.json

- Volumes and Secrets

Keytab Configuration in config.json:

- secrets_enabled=“true”
- keytabs_secrets_path=/path/to/secret



```
"volumes": [
  {
    "containerPath": "config-file",
    "secret": "my-secret"
  }
 ]
....

"secrets": {
  "my-secret": {
    "source": "app/config"
  }
 }
```

Lab 7b

DC/OS SECRETS

LAB 7B - DC/OS SECRETS

1. Login to the DC/OS UI as a superuser
2. Go to **System** -> **Security** -> **Secrets**
3. Click on **New Secret** and create a secret with id: `dev/`**`<secret-name>`**
4. Create a new marathon service /dev/<app-id> and apply the secret:

```
"env": {
        "MYSECRET": {
        "secret": "secret0"
        }
       },
  …

 "secrets": {
            "secret0": {
            "source": "dev/<secret-name>"
            }
           },
```

LAB 7A - DC/OS SECRETS API

1. Use the DCOS CLI to login to the cluster

```
dcos auth login
```

2. View secrets store

```
curl -H "Authorization: token=$(dcos config show core.dcos_acs_token)" $(dcos config show core.dcos_url)/secrets/v1/store | jq .
```

3. View seal status of store “default”

```
curl -H "Authorization: token=$(dcos config show core.dcos_acs_token)" $(dcos config show core.dcos_url)/secrets/v1/seal-status/default | jq .
```

Security

CLI

DCOS ENTERPRISE CLI

```
dcos package install --cli dcos-enterprise-cli

Installing CLI subcommand for package [dcos-enterprise-cli]
version [1.0.0]
New command available: dcos security

dcos security --help

Commands:
  cluster  Cluster management commands.
  org      Account management commands.
  secrets  Secrets management commands.
```

SECRETS CLI

```
dcos security secrets --help

Usage: dcos-security security secrets [OPTIONS] COMMAND
[ARGS]...

Commands:
  create             Create a secret.
  create-sa-secret  Create a service account secret.
  delete             Delete a secret.
  get                Get a secret from the store by its path.
  list               List secret keys in a given path.
  update             Update a secret.
```

Lab 7b

SECURITY CLI

LAB 7C - DC/OS SECURITY CLI

1. Use the DCOS CLI or Marathon UI to install the security cli

   ```
   dcos package install dcos-enterprise-cli --yes
   ```

2. View secrets store status

   ```
   dcos security cluster secret-store show
   ```

3. View all users

   ```
   dcos security org users show
   ```

4. View all groups

   ```
   dcos security org groups show
   ```

5. View secrets in path dev

   ```
   dcos security secrets list dev
   ```

Enterprise DC/OS Fundamentals

Security Auditing

DC/OS Security Auditing

- Access information for common DC/OS services running on Master Nodes including mesos-master, history-service, mesos-dns, etc
- Provide audit information that can be used to review API access and provide an audit trail.
- Identified by `type=audit`
- Can be used for filtering at Log Collection system (ELK, Splunk)

DC/OS Security Auditing

1. From a DC/OS Master node TAIL all Audit events from all DCOS Services

```
journalctl -f | grep type=audit
```

1. View Mesos Master Service Audit Logs from Newest to Oldest

```
journalctl -rlu dcos-mesos-master | grep type=audit
```

1. View Admin Router Audit Logs from Newest to Oldest

```
journalctl -rlu dcos-adminrouter | grep type=audit
```

Security Auditing

DC/OS Security Auditing Sample Output

```
core@ip-10-0-4-90 ~ $ journalctl -f | grep type=audit
Oct 21 14:03:04 ip-10-0-4-90.ec2.internal bouncer.sh[1876]: [161021-14:03:04.303] [1897:Thread-7]
[bouncer.app.internal.PolicyQuery] INFO: type=audit timestamp=2017-10-21T14:03:04.303652Z srcip=127.0.0.1 authorizer=bouncer
uid=dcos_history_service action=full object=dcos:adminrouter:ops:mesos result=allow reason="User is superuser (user group
`superusers` in ACL, action ignored)"
Oct 21 14:03:04 ip-10-0-4-90.ec2.internal nginx[2145]: 2017/10/21 14:03:04 [notice] 10023#0: *26189 [lua] auth.lua:83:
auditlog(): type=audit timestamp=2017-10-21T14:03:04Z authorizer=adminrouter object=dcos:adminrouter:ops:mesos action=full
result=allow reason="Bouncer PQ response" srcip=10.0.4.90 srcport=39012 request_uri=/mesos/state-summary uid=dcos_history_service
while sending to client, client: 10.0.4.90, server: master.mesos, request: "GET /mesos/state-summary HTTP/1.1", host:
"leader.mesos"
Oct 21 14:03:04 ip-10-0-4-90.ec2.internal mesos-master[2440]: I1021 14:03:04.310232  2447 logfmt.cpp:164] dstip=10.0.4.90
type=audit timestamp=2017-10-21 14:03:04.310183936+00:00 reason="Valid authorization token" uid="dcos_history_service"
object="/master/state-summary" authorizer="mesos-master" action="GET" result=allow srcip=10.0.4.90 dstport=5050 srcport=41478
Oct 21 14:03:05 ip-10-0-4-90.ec2.internal bouncer.sh[1876]: [161021-14:03:05.446] [1897:Thread-13]
[bouncer.app.internal.PolicyQuery] INFO: type=audit timestamp=2017-10-21T14:03:05.446008Z srcip=127.0.0.1 authorizer=bouncer
uid=bootstrapuser action=full object=dcos:adminrouter:ops:historyservice result=allow reason="User is superuser (user group
`superusers` in ACL, action ignored)"
Oct 21 14:03:05 ip-10-0-4-90.ec2.internal nginx[2145]: 2017/10/21 14:03:05 [notice] 10023#0: *26013 [lua] auth.lua:83:
auditlog(): type=audit timestamp=2017-10-21T14:03:05Z authorizer=adminrouter object=dcos:adminrouter:ops:historyservice
action=full result=allow reason="Bouncer PQ response" srcip=10.0.6.92 srcport=22050
request_uri=/dcos-history-service/history/last?_timestamp=1477058585412 uid=bootstrapuser while sending to client, client:
10.0.6.92, server: master.mesos, request: "GET /dcos-history-service/history/last?_timestamp=1477058585412 HTTP/1.1", host:
"thomaskra-elasticl-nff3iqoph6vh-963846176.us-east-1.elb.amazonaws.com", referrer:
"https://thomaskra-elasticl-nff3iqoph6vh-963846176.us-east-1.elb.amazonaws.com/"
Oct 21 14:03:06 ip-10-0-4-90.ec2.internal bouncer.sh[1876]: [161021-14:03:06.046] [1897:Thread-6]
[bouncer.app.internal.PolicyQuery] INFO: type=audit timestamp=2017-10-21T14:03:06.046257Z srcip=127.0.0.1 authorizer=bouncer
uid=bootstrapuser action=full object=dcos:adminrouter:service:metronome result=allow reason="User is superuser (user group
`superusers` in ACL, action ignored)"
```

SUMMARY

In this module we looked at DC/OS security

- DC/OS IAM
- Cluster security
- Secrets

MESOSPHERE