03-项目部署：如何快速部署IAM系统？

你好，我是孔令飞。

上一讲，我们一起安装和配置了一个基本的 Go 开发环境。这一讲，我就来教你怎么在它的基础上，快速部署好 IAM 系统。

因为我们要通过一个 IAM 项目来讲解怎么开发企业级 Go 项目，所以你要对 IAM 项目有比较好的了解，了解 IAM 项目一个最直接有效的方式就是去部署和使用它。

这不仅能让你了解到 IAM 系统中各个组件功能之间的联系，加深你对 IAM 系统的理解，还可以协助你排障，尤其是跟部署相关的故障。此外，部署好 IAM 系统也能给你后面的学习准备好实验环境，边学、边练，从而提高你的学习效率。

所以，今天我们专门花一讲的时间来说说怎么部署和使用 IAM 系统。同时，因为 IAM 系统是一个企业级的项目，有一定的复杂度，我建议你严格按照我说的步骤去操作，否则可能会安装失败。

总的来说，我把部署过程分成 2 大步。

1. 安装和配置数据库：我们需要安装和配置 MariaDB、Redis和MongoDB。
2. 安装和配置 IAM 服务：我们需要安装和配置 iam-apiserver、iam-authz-server、iam-pump、iamctl和 man 文件。

当然啦，如果你实在不想这么麻烦地去安装，我也在这一讲的最后给出了一键部署 IAM 系统的方法，但我还是希望你能按照我今天讲的详细步骤来操作。

那话不多说，我们直接开始操作吧！

下载 iam 项目代码。

因为 IAM 的安装脚本存放在 iam 代码仓库中，安装需要的二进制文件也需要通过 iam 代码构建，所以我们需要先下载 iam 代码：

因为 IAM 的安装脚本存放在 iam 代码仓库中，安装需要的二进制文件也需要通过 iam 代码构建，所以在安装之前，我们需要先下载 iam 代码：

```
$ mkdir -p $WORKSPACE/golang/src/github.com/marmotedu
$ cd $WORKSPACE/golang/src/github.com/marmotedu
$ git clone --depth=1 https://github.com/marmotedu/iam
```

其中，marmotedu 和 marmotedu/iam 目录存放了本实战项目的代码，在学习过程中，你需要频繁访问这 2 个目录，为了访问方便，我们可以追加如下 2 个环境变量和 2 个 alias 到`$HOME/.bashrc`文件中：

```
$ tee -a $HOME/.bashrc << 'EOF'
# Alias for quick access
export GOWORK="$WORKSPACE/golang/src"
export IAM_ROOT="$GOWORK/github.com/marmotedu/iam"
alias mm="cd $GOWORK/github.com/marmotedu"
alias i="cd $GOWORK/github.com/marmotedu/iam"
EOF
$ bash
```

之后，我们可以先通过执行 alias 命令 mm 访问 $GOWORK/github.com/marmotedu 目录，再通过执行 alias 命令 i 访问 $GOWORK/github.com/marmotedu/iam 目录。

这里我也建议你善用 alias，将常用操作配置成 alias，方便以后操作。

在安装配置之前需要执行以下命令export going用户的密码，这里假设密码是 iam59!z$：

```
export LINUX_PASSWORD='iam59!z$'
```

安装和配置数据库

因为 IAM 系统用到了 MariaDB、Redis、MongoDB 数据库来存储数据，而 IAM 服务在启动时会先尝试连接这些数据库，所以为了避免启动时连接数据库失败，这里我们先来安装需要的数据库。

安装和配置 MariaDB

IAM 会把 REST 资源的定义信息存储在关系型数据库中，关系型数据库我选择了 MariaDB。为啥选择 MariaDB，而不是 MySQL呢？。选择 MariaDB 一方面是因为它是发展最快的 MySQL 分支，相比 MySQL，它加入了很多新的特性，并且它能够完全兼容 MySQL，包括 API 和命令行。另一方面是因为 MariaDB 是开源的，而且迭代速度很快。

首先，我们可以通过以下命令安装和配置 MariaDB，并将 Root 密码设置为 iam59!z$：

```
$ cd $IAM_ROOT
$ ./scripts/install/mariadb.sh iam::mariadb::install
```

然后，我们可以通过以下命令，来测试 MariaDB 是否安装成功：

```
$ mysql -h127.0.0.1 -uroot -p'iam59!z$'
MariaDB [(none)]>
```

安装和配置 Redis

在 IAM 系统中，由于 iam-authz-server 是从 iam-apiserver 拉取并缓存用户的密钥/策略信息的，因此同一份密钥/策略数据会分别存在 2 个服务中，这可能会出现数据不一致的情况。数据不一致会带来一些问题，例如当我们通过 iam-apiserver 创建了一对密钥，但是这对密钥还没有被 iam-authz-server 缓存，这时候通过这对密钥访问 iam-authz-server 就会访问失败。

为了保证数据的一致性，我们可以使用 Redis 的发布订阅(pub/sub)功能进行消息通知。同时，iam-authz-server 也会将授权审计日志缓存到 Redis 中，所以也需要安装 Redis key-value 数据库。我们可以通过以下命令来安装和配置 Redis，并将 Redis 的初始密码设置为 `iam59!z$`：

```
$ cd $IAM_ROOT
$ ./scripts/install/redis.sh iam::redis::install
```

这里我们要注意，scripts/install/redis.sh 脚本中 iam::redis::install 函数对 Redis 做了一些配置，例如修改 Redis 使其以守护进程的方式运行、修改 Redis 的密码为 `iam59!z$`等，详细配置可参考函数 iam::redis::install 函数。

安装完成后，我们可以通过以下命令，来测试 Redis 是否安装成功：

```
$ redis-cli -h 127.0.0.1 -p 6379 -a 'iam59!z$' # 连接 Redis，-h 指定主机，-p 指定监听端口，-a 指定登录密码
```

安装和配置 MongoDB

因为 iam-pump 会将 iam-authz-server 产生的数据处理后存储在 MongoDB 中，所以我们也需要安装 MongoDB 数据库。主要分两步安装：首先安装 MongoDB，然后再创建 MongoDB 账号。

第 1 步，安装 MongoDB

首先，我们可以通过以下 4 步来安装 MongoDB。

1. 配置 MongoDB yum 源，并安装 MongoDB。

CentOS8.x 系统默认没有配置安装 MongoDB 需要的 yum 源，所以我们需要先配置好 yum 源再安装：

```
$ sudo tee /etc/yum.repos.d/mongodb-org-4.4.repo<<'EOF'
[mongodb-org-4.4]
name=MongoDB Repository
baseurl=https://repo.mongodb.org/yum/redhat/$releasever/mongodb-org/4.4/x86_64/
gpgcheck=1
enabled=1
gpgkey=https://www.mongodb.org/static/pgp/server-4.4.asc
EOF
```

```
$ sudo yum install -y mongodb-org
```

2. 关闭 SELinux。

在安装的过程中，SELinux 有可能会阻止 MongoDB 访问/sys/fs/cgroup，所以我们还需要关闭 SELinux：

```
$ sudo setenforce 0
$ sudo sed -i 's/^SELINUX=.*$/SELINUX=disabled/' /etc/selinux/config # 永久关闭 SELINUX
```

3. 开启外网访问权限和登录验证。

MongoDB 安装完之后，默认情况下是不会开启外网访问权限和登录验证，为了方便使用，我建议你先开启这些功能，执行如下命令开启：

```
$ sudo sed -i '/bindIp/{s/127.0.0.1/0.0.0.0/}' /etc/mongod.conf
$ sudo sed -i '/^#security/a\security:\n  authorization: enabled' /etc/mongod.conf
```

4. 启动 MongoDB。

配置完 MongoDB 之后，我们就可以启动它了，具体的命令如下：

```
$ sudo systemctl start mongod
$ sudo systemctl enable mongod # 设置开机启动
$ sudo systemctl status mongod # 查看 mongod 运行状态，如果输出中包含 active (running)字样说明 mongod 成功启动
```

安装完 MongoDB 后，我们就可以通过 mongo 命令登录 MongoDB Shell。如果没有报错，就说明 MongoDB 被成功安装了。

```
$ mongo --quiet "mongodb://127.0.0.1:27017"
>
```

第 2 步，创建 MongoDB 账号

安装完 MongoDB 之后，默认是没有用户账号的，为了方便 IAM 服务使用，我们需要先创建好管理员账号，通过管理员账户登录 MongoDB，我们可以执行创建普通用户、数据库等操作。

1. 创建管理员账户。

首先，我们通过 `use admin` 指令切换到 admin 数据库，再通过 `db.auth("用户名", "用户密码")` 验证用户登录权限。如果返回 1 表示验证成功；如果返回 0 表示验证失败。具体的命令如下：

```
$ mongo --quiet "mongodb://127.0.0.1:27017"
> use admin
switched to db admin
> db.createUser({user:"root",pwd:"iam59!z$",roles:["root"]})
Successfully added user: { "user" : "root", "roles" : [ "root" ] }
> db.auth("root", "iam59!z$")
1
```

此外，如果想删除用户，可以使用 `db.dropUser("用户名")` 命令。

`db.createUser` 用到了以下 3 个参数。

- user: 用户名。
- pwd: 用户密码。
- roles: 用来设置用户的权限，比如读、读写、写等。

因为 admin 用户具有 MongoDB 的 Root 权限，权限过大安全性会降低。为了提高安全性，我们还需要创建一个 iam 普通用户来连接和操作 MongoDB。

2. 创建 iam 用户，命令如下：

```
$ mongo --quiet mongodb://root:'iam59!z$'@127.0.0.1:27017/tyk_analytics?authSource=admin # 用管理员账户连接 Mo
> use iam_analytics
switched to db iam_analytics
> db.createUser({user:"iam",pwd:"iam59!z$",roles:["dbOwner"]})
Successfully added user: { "user" : "iam", "roles" : [ "dbOwner" ] }
> db.auth("iam", "iam59!z$")
1
```

创建完 iam 普通用户后，我们就可以通过 iam 用户登录 MongoDB 了：

```
$ mongo --quiet mongodb://iam:'iam59!z$'@127.0.0.1:27017/iam_analytics?authSource=iam_analytics
```

至此，我们成功安装了 IAM 系统需要的数据库 MariaDB、Redis 和 MongoDB。

安装和配置 IAM 系统

要想完成 IAM 系统的安装，我们还需要安装和配置 iam-apiserver、iam-authz-server、iam-pump 和 iamctl。这些组件的功能我们在第1讲详细讲过，如果不记得你可以翻回去看看。

> 提示：IAM 项目我会长期维护、定期更新，欢迎你 Star & Contributing。

准备工作

在开始安装之前，我们需要先做一些准备工作，主要有 5 步。

1. 初始化 MariaDB 数据库，创建 iam 数据库。
2. 配置 scripts/install/environment.sh。
3. 创建需要的目录。
4. 创建 CA 根证书和密钥。
5. 配置 hosts。

第 1 步，初始化 MariaDB 数据库，创建 iam 数据库。

安装完 MariaDB 数据库之后，我们需要在 MariaDB 数据库中创建 IAM 系统需要的数据库、表和存储过程，以及创建 SQL 语句保存在 IAM 代码仓库中的 configs/iam.sql 文件中。具体的创建步骤如下。

1. 登录数据库并创建 iam 用户。

```
$ cd $IAM_ROOT
$ mysql -h127.0.0.1 -P3306 -uroot -p'iam59!z$' # 连接 MariaDB，-h 指定主机，-P 指定监听端口，-u 指定登录用户，-p 指
MariaDB [(none)]> grant all on iam.* TO iam@127.0.0.1 identified by 'iam59!z$';
Query OK, 0 rows affected (0.000 sec)
MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.000 sec)
```

2. 用 iam 用户登录 MariaDB，执行 iam.sql 文件，创建 iam 数据库。

```
$ mysql -h127.0.0.1 -P3306 -uiam -p'iam59!z$'
MariaDB [(none)]> source configs/iam.sql;
MariaDB [iam]> show databases;
+--------------------+
| Database           |
+--------------------+
| iam                |
| information_schema |
+--------------------+
2 rows in set (0.000 sec)
```

上面的命令会创建 iam 数据库，并创建以下数据库资源。

- 表：user 是用户表，用来存放用户信息；secret 是密钥表，用来存放密钥信息；policy 是策略表，用来存放授权策略信息；policy_audit 是策略历史表，被删除的策略会被转存到该表。

- admin 用户：在 user 表中，我们需要创建一个管理员用户，用户名是 admin，密码是 Admin@2021。
- 存储过程：删除用户时会自动删除该用户所属的密钥和策略信息。

第 2 步，配置 scripts/install/environment.sh。

IAM 组件的安装配置都是通过环境变量文件 scripts/install/environment.sh 进行配置的，所以我们要先配置好 scripts/install/environment.sh 文件。这里，你可以直接使用默认值，提高你的安装效率。

第 3 步，创建需要的目录。

在安装和运行 IAM 系统的时候，我们需要将配置、二进制文件和数据文件存放到指定的目录。所以我们需要先创建好这些目录，创建步骤如下。

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ sudo mkdir -p ${IAM_DATA_DIR}/{iam-apiserver,iam-authz-server,iam-pump} # 创建 Systemd WorkingDirectory 目
$ sudo mkdir -p ${IAM_INSTALL_DIR}/bin #创建 IAM 系统安装目录
$ sudo mkdir -p ${IAM_CONFIG_DIR}/cert # 创建 IAM 系统配置文件存放目录
$ sudo mkdir -p ${IAM_LOG_DIR} # 创建 IAM 日志文件存放目录
```

第 4 步， 创建 CA 根证书和密钥。

为了确保安全，IAM 系统各组件需要使用 x509 证书对通信进行加密和认证。所以，这里我们需要先创建 CA 证书。CA 根证书是所有组件共享的，只需要创建一个 CA 证书，后续创建的所有证书都由它签名。

我们可以使用 CloudFlare 的 PKI 工具集 cfssl 来创建所有的证书。

1. 安装 cfssl 工具集。

我们可以直接安装 cfssl 已经编译好的二进制文件，cfssl 工具集中包含很多工具，这里我们需要安装 cfssl、cfssljson、cfssl-certinfo，功能如下。

- cfssl：证书签发工具。
- cfssljson：将 cfssl 生成的证书（json 格式）变为文件承载式证书。

这两个工具的安装方法如下：

```
$ cd $IAM_ROOT
$ ./scripts/install/install.sh iam::install::install_cfssl
```

2. 创建配置文件。

CA 配置文件是用来配置根证书的使用场景 (profile) 和具体参数 (usage、过期时间、服务端认证、客户端认证、加密等)，可以在签名其它证书时用来指定特定场景：

```
$ cd $IAM_ROOT
$ tee ca-config.json << EOF
{
  "signing": {
    "default": {
      "expiry": "87600h"
    },
    "profiles": {
      "iam": {
        "usages": [
          "signing",
          "key encipherment",
          "server auth",
          "client auth"
        ],
        "expiry": "876000h"
      }
    }
  }
}
EOF
```

上面的 JSON 配置中，有一些字段解释如下。

- signing：表示该证书可用于签名其它证书（生成的 ca.pem 证书中 CA=TRUE）。
- server auth：表示 client 可以用该证书对 server 提供的证书进行验证。
- client auth：表示 server 可以用该证书对 client 提供的证书进行验证。
- expiry：876000h，证书有效期设置为 100 年。

3. 创建证书签名请求文件。

我们创建用来生成 CA 证书签名请求（CSR）的 JSON 配置文件：

```
$ cd $IAM_ROOT
$ tee ca-csr.json << EOF
{
  "CN": "iam-ca",
  "key": {
    "algo": "rsa",
    "size": 2048
  },
  "names": [
    {
      "C": "CN",
      "ST": "BeiJing",
      "L": "BeiJing",
      "O": "iam",
      "OU": "marmotedu"
    }
```

```
  ],
  "ca": {
    "expiry": "876000h"
  }
}
EOF
```

上面的 JSON 配置中，有一些字段解释如下。

- CN：Common Name，iam-apiserver 从证书中提取该字段作为请求的用户名 (User Name)，浏览器使用该字段验证网站是否合法。
- O：Organization，iam-apiserver 从证书中提取该字段作为请求用户所属的组 (Group)。

除此之外，还有两点需要我们注意。

- 不同证书 csr 文件的 CN、C、ST、L、O、OU 组合必须不同，否则可能出现 `PEER'S CERTIFICATE HAS AN INVALID SIGNATURE` 错误。
- 后续创建证书的 csr 文件时，CN 都不相同（C、ST、L、O、OU 相同），以达到区分的目的。

4. 创建 CA 证书和私钥

首先，我们通过 `cfssl gencert` 命令来创建：

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ cfssl gencert -initca ca-csr.json | cfssljson -bare ca
$ ls ca*
ca-config.json  ca.csr  ca-csr.json  ca-key.pem  ca.pem
$ sudo mv ca* ${IAM_CONFIG_DIR}/cert # 需要将证书文件拷贝到指定文件夹下（分发证书），方便各组件引用
```

上述命令会创建运行 CA 所必需的文件 ca-key.pem（私钥）和 ca.pem（证书），还会生成 ca.csr（证书签名请求），用于交叉签名或重新签名。

创建完之后，我们可以通过 `cfssl certinfo` 命名查看 cert 和 csr 信息：

```
$ cfssl certinfo -cert ${IAM_CONFIG_DIR}/cert/ca.pem # 查看 cert(证书信息)
$ cfssl certinfo -csr ${IAM_CONFIG_DIR}/cert/ca.csr # 查看 CSR(证书签名请求)信息
```

第 5 步，配置 hosts。

iam 通过域名访问 API 接口，因为这些域名没有注册过，还不能在互联网上解析，所以需要配置 hosts，具体的操作如下：

```
$ sudo tee -a /etc/hosts <<EOF
127.0.0.1 iam.api.marmotedu.com
127.0.0.1 iam.authz.marmotedu.com
EOF
```

安装和配置 iam-apiserver

完成了准备工作之后，我们就可以安装 IAM 系统的各个组件了。首先我们通过以下 3 步来安装 iam-apiserver 服务。

第 1 步，创建 iam-apiserver 证书和私钥。

其它服务为了安全都是通过 HTTPS 协议访问 iam-apiserver，所以我们要先创建 iam-apiserver 证书和私钥。

1. 创建证书签名请求：

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ tee iam-apiserver-csr.json <<EOF
{
  "CN": "iam-apiserver",
  "key": {
    "algo": "rsa",
    "size": 2048
  },
  "names": [
    {
      "C": "CN",
      "ST": "BeiJing",
      "L": "BeiJing",
      "O": "iam",
      "OU": "marmotedu"
    }
  ],
  "hosts": [
    "127.0.0.1",
    "localhost",
    "iam.api.marmotedu.com"
  ]
}
EOF
```

代码中的 hosts 字段是用来指定授权使用该证书的 IP 和域名列表，上面的 hosts 列出了 iam-apiserver 服务的 IP 和域名。

2. 生成证书和私钥：

```
$ cfssl gencert -ca=${IAM_CONFIG_DIR}/cert/ca.pem \
```

```
  -ca-key=${IAM_CONFIG_DIR}/cert/ca-key.pem \
  -config=${IAM_CONFIG_DIR}/cert/ca-config.json \
  -profile=iam iam-apiserver-csr.json | cfssljson -bare iam-apiserver
$ sudo mv iam-apiserver*pem ${IAM_CONFIG_DIR}/cert # 将生成的证书和私钥文件拷贝到配置文件目录
```

第 2 步，安装并运行 iam-apiserver。

iam-apiserver 作为 iam 系统的核心组件，需要第一个安装。

1. 安装 iam-apiserver 可执行程序：

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ make build BINS=iam-apiserver
$ sudo cp _output/platforms/linux/amd64/iam-apiserver ${IAM_INSTALL_DIR}/bin
```

2. 生成并安装 iam-apiserver 的配置文件（iam-apiserver.yaml）：

```
$ ./scripts/genconfig.sh scripts/install/environment.sh configs/iam-apiserver.yaml > iam-apiserver.yaml
$ sudo mv iam-apiserver.yaml ${IAM_CONFIG_DIR}
```

3. 创建并安装 iam-apiserver systemd unit 文件：

```
$ ./scripts/genconfig.sh scripts/install/environment.sh init/iam-apiserver.service > iam-apiserver.service
$ sudo mv iam-apiserver.service /etc/systemd/system/
```

4. 启动 iam-apiserver 服务：

```
$ sudo systemctl daemon-reload
$ sudo systemctl enable iam-apiserver
$ sudo systemctl restart iam-apiserver
$ systemctl status iam-apiserver # 查看 iam-apiserver 运行状态，如果输出中包含 active (running)字样说明 iam-apise
```

第 3 步，测试 iam-apiserver 是否成功安装。

测试 iam-apiserver 主要是测试 RESTful 资源的 CURD：用户 CURD、密钥 CURD、授权策略 CURD。

首先，我们需要获取访问 iam-apiserver 的 Token，请求如下 API 访问：

```
$ curl -s -XPOST -H'Content-Type: application/json' -d'{"username":"admin","password":"Admin@2021"}' http:/
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm1vdGVkdS5jb20iLCJleHAiOjE2MTc5MjI4OTQsImlkZW
```

代码中下面的 HTTP 请求通过`-H'Authorization: Bearer <Token>'`指定认证头信息，将上面请求的 Token 替换 <Token> 。

用户 CURD

创建用户、列出用户、获取用户详细信息、修改用户、删除单个用户、批量删除用户，请求方法如下：

```
# 创建用户
$ curl -s -XPOST -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6Ik

# 列出用户
$ curl -s -XGET -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm1vd

# 获取 colin 用户的详细信息
$ curl -s -XGET -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm1vd

# 修改 colin 用户
$ curl -s -XPUT -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp

# 删除 colin 用户
$ curl -s -XDELETE -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm

# 批量删除用户
$ curl -s -XDELETE -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm
```

密钥 CURD

创建密钥、列出密钥、获取密钥详细信息、修改密钥、删除密钥请求方法如下：

```
# 创建 secret0 密钥
$ curl -s -XPOST -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6Ik

# 列出所有密钥
$ curl -s -XGET -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm1vd

# 获取 secret0 密钥的详细信息
$ curl -s -XGET -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm1vd

# 修改 secret0 密钥
$ curl -s -XPUT -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp

# 删除 secret0 密钥
$ curl -s -XDELETE -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm
```

这里我们要注意，因为密钥属于重要资源，被删除会导致所有的访问请求失败，所以密钥不支持批量删除。

授权策略 CURD

创建策略、列出策略、获取策略详细信息、修改策略、删除策略请求方法如下：

```
# 创建策略
$ curl -s -XPOST -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6Ik

# 列出所有策略
$ curl -s -XGET -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm1vd

# 获取 policy0 策略的详细信息
$ curl -s -XGET -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm1vd

# 修改 policy 策略
$ curl -s -XPUT -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp

# 删除 policy0 策略
$ curl -s -XDELETE -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJpYW0uYXBpLm1hcm
```

安装 iamctl

上面，我们安装了 iam 系统的 API 服务。但是想要访问 iam 服务，我们还需要安装客户端工具 iamctl。具体来说，我们可以通过 3 步完成 iamctl 的安装和配置。

第 1 步，创建 iamctl 证书和私钥。

iamctl 使用 https 协议与 iam-apiserver 进行安全通信，iam-apiserver 对 iamctl 请求包含的证书进行认证和授权。iamctl 后续用于 iam 系统访问和管理，所以这里创建具有最高权限的 admin 证书。

1. 创建证书签名请求。

下面创建的证书只会被 iamctl 当作 client 证书使用，所以 hosts 字段为空。代码如下：

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ cat > admin-csr.json <<EOF
{
  "CN": "admin",
  "key": {
    "algo": "rsa",
    "size": 2048
  },
  "names": [
    {
      "C": "CN",
      "ST": "BeiJing",
      "L": "BeiJing",
      "O": "iamctl",
      "OU": "marmotedu"
```

```
    }
  ],
  "hosts": []
}
EOF
```

2. 生成证书和私钥：

```
$ cfssl gencert -ca=${IAM_CONFIG_DIR}/cert/ca.pem \
  -ca-key=${IAM_CONFIG_DIR}/cert/ca-key.pem \
  -config=${IAM_CONFIG_DIR}/cert/ca-config.json \
  -profile=iam admin-csr.json | cfssljson -bare admin
$ mkdir -p $(dirname ${CONFIG_USER_CLIENT_CERTIFICATE}) $(dirname ${CONFIG_USER_CLIENT_KEY}) # 创建客户端证书
$ mv admin.pem ${CONFIG_USER_CLIENT_CERTIFICATE} # 安装 TLS 的客户端证书
$ mv admin-key.pem ${CONFIG_USER_CLIENT_KEY} # 安装 TLS 的客户端私钥文件
```

第 2 步，安装 iamctl。

iamctl 是 IAM 系统的客户端工具，其安装位置和 iam-apiserver、iam-authz-server、iam-pump 位置不同，为了能够在 shell 下直接运行 iamctl 命令，我们需要将 iamctl 安装到`$HOME/bin`下，同时将 iamctl 的配置存放在默认加载的目录下：`$HOME/.iam`。主要分 2 步进行。

1. 安装 iamctl 可执行程序：

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ make build BINS=iamctl
$ cp _output/platforms/linux/amd64/iamctl $HOME/bin
```

2. 生成并安装 iamctl 的配置文件（config）：

```
$ ./scripts/genconfig.sh scripts/install/environment.sh configs/config > config
$ mkdir -p $HOME/.iam
$ mv config $HOME/.iam
```

因为 iamctl 是一个客户端工具，可能会在多台机器上运行。为了简化部署 iamctl 工具的复杂度，我们可以把 config 配置文件中跟 CA 认证相关的 CA 文件内容用 base64 加密后，放置在 config 配置文件中。具体的思路就是把 config 文件中的配置项 client-certificate、client-key、certificate-authority 分别用如下配置项替换 client-certificate-data、client-key-data、certificate-authority-data。这些配置项的值可以通过对 CA 文件使用 base64 加密获得。

假如，certificate-authority 值为/etc/iam/cert/ca.pem，则 certificate-authority-data 的值为 cat "/etc/iam/cert/ca.pem" | base64 | tr -d '\r\n'，其它-data 变量的值类似。这样当我们再部署 iamctl 工具时，只需要拷贝 iamctl 和配置文件，而不用再拷贝 CA 文件了。

第 3 步，测试 iamctl 是否成功安装。

执行 iamctl user list 可以列出预创建的 admin 用户，如下图所示：

```
[going@dev ~]$ iamctl user list
NAME    NICKNAME        EMAIL                   PHONE           CREATED                 UPDATED
admin   admin           admin@foxmail.com       1812884xxxx     2021-05-06 05:13:14     2021-05-06 05:13:14
[going@dev ~]$
```

安装和配置 iam-authz-server

接下来，我们需要安装另外一个核心组件：iam-authz-server，可以通过以下 3 步来安装。

第 1 步，创建 iam-authz-server 证书和私钥。

1. 创建证书签名请求：

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ tee iam-authz-server-csr.json <<EOF
{
  "CN": "iam-authz-server",
  "key": {
    "algo": "rsa",
    "size": 2048
  },
  "names": [
    {
      "C": "CN",
      "ST": "BeiJing",
      "L": "BeiJing",
      "O": "iam-authz-server",
      "OU": "marmotedu"
    }
  ],
  "hosts": [
    "127.0.0.1",
    "localhost",
    "iam.authz.marmotedu.com"
  ]
}
EOF
```

代码中的 hosts 字段指定授权使用该证书的 IP 和域名列表，上面的hosts列出了 iam-authz-server 服务的 IP 和域名。

2. 生成证书和私钥：

```
$ cfssl gencert -ca=${IAM_CONFIG_DIR}/cert/ca.pem \
  -ca-key=${IAM_CONFIG_DIR}/cert/ca-key.pem \
  -config=${IAM_CONFIG_DIR}/cert/ca-config.json \
  -profile=iam iam-authz-server-csr.json | cfssljson -bare iam-authz-server
$ sudo mv iam-authz-server*pem ${IAM_CONFIG_DIR}/cert # 将生成的证书和私钥文件拷贝到配置文件目录
```

第 2 步，安装并运行 iam-authz-server。

安装 iam-authz-server 步骤和安装 iam-apiserver 步骤基本一样，也需要 4 步。

1. 安装 iam-authz-server 可执行程序：

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ make build BINS=iam-authz-server
$ sudo cp _output/platforms/linux/amd64/iam-authz-server ${IAM_INSTALL_DIR}/bin
```

2. 生成并安装 iam-authz-server 的配置文件（iam-authz-server.yaml）：

```
$ ./scripts/genconfig.sh scripts/install/environment.sh configs/iam-authz-server.yaml > iam-authz-server.ya
$ sudo mv iam-authz-server.yaml ${IAM_CONFIG_DIR}
```

3. 创建并安装 iam-authz-server systemd unit 文件：

```
$ ./scripts/genconfig.sh scripts/install/environment.sh init/iam-authz-server.service > iam-authz-server.se
$ sudo mv iam-authz-server.service /etc/systemd/system/
```

4. 启动 iam-authz-server 服务：

```
$ sudo systemctl daemon-reload
$ sudo systemctl enable iam-authz-server
$ sudo systemctl restart iam-authz-server
$ systemctl status iam-authz-server # 查看 iam-authz-server 运行状态，如果输出中包含 active (running)字样说明 iam
```

第 3 步，测试 iam-authz-server 是否成功安装。

1. 创建密钥，并从代码的输出中提取secretID 和 secretKey。

```
$ curl -s -XPOST -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6Ik
{"metadata":{"id":23,"name":"authztest","createdAt":"2021-04-08T07:24:50.071671422+08:00","updatedAt":"2021
```

2. 生成访问 iam-authz-server 的 Token。

iamctl 提供了 `jwt sigin` 命令，可以根据 secretID 和 secretKey 签发 Token，方便你使用。

```
$ iamctl jwt sign ZuxvXNfG08BdEMqkTaP41L2DLArlE6Jpqoox 7Sfa5EfAPIwcTLGCfSvqLf0zZGCjF3l8 # iamctl jwt sign $
eyJhbGciOiJIUzI1NiIsImtpZCI6Ilp1eHZYTmZHMDhCZEVNcWtUYVA0MUwyRExBcmxFNkpwcW9veCIsInR5cCI6IkpXVCJ9.eyJhdWQiOi
```

如果你的开发过程中有些重复性的操作，为了方便使用，也可以将这些操作以iamctl子命令的方式集成到iamctl命令行中。

3. 测试资源授权是否通过。

我们可以通过请求 `/v1/authz` 来完成资源授权：

```
$ curl -s -XPOST -H'Content-Type: application/json' -H'Authorization: Bearer eyJhbGciOiJIUzI1NiIsImtpZCI6Il
{"allowed":true}
```

如果授权通过会返回：`{"allowed":true}`。

安装和配置 iam-pump

安装 iam-pump 步骤和安装 iam-apiserver、iam-authz-server 步骤基本一样，具体步骤如下。

第 1 步，安装 iam-pump 可执行程序。

```
$ cd $IAM_ROOT
$ source scripts/install/environment.sh
$ make build BINS=iam-pump
$ sudo cp _output/platforms/linux/amd64/iam-pump ${IAM_INSTALL_DIR}/bin
```

第 2 步，生成并安装 iam-pump 的配置文件（iam-pump.yaml）。

```
$ ./scripts/genconfig.sh scripts/install/environment.sh configs/iam-pump.yaml > iam-pump.yaml
$ sudo mv iam-pump.yaml ${IAM_CONFIG_DIR}
```

第 3 步，创建并安装 iam-pump systemd unit 文件。

```
$ ./scripts/genconfig.sh scripts/install/environment.sh init/iam-pump.service > iam-pump.service
$ sudo mv iam-pump.service /etc/systemd/system/
```

第 4 步，启动 iam-pump 服务。

```
$ sudo systemctl daemon-reload
$ sudo systemctl enable iam-pump
$ sudo systemctl restart iam-pump
$ systemctl status iam-pump # 查看 iam-pump 运行状态，如果输出中包含 active (running)字样说明 iam-pump 成功启动。
```

第 5 步，测试 iam-pump 是否成功安装。

```
$ curl http://127.0.0.1:7070/healthz
{"status": "ok"}
```

经过上面这 5 个步骤，如果返回 **{"status": "ok"}** 就说明 iam-pump 服务健康。

安装 man 文件

IAM 系统通过组合调用包：`github.com/cpuguy83/go-md2man/v2/md2man` 和 `github.com/spf13/cobra` 的相关函数生成了各个组件的 man1 文件，主要分 3 步实现。

第 1 步，生成各个组件的 man1 文件。

```
$ cd $IAM_ROOT
$ ./scripts/update-generated-docs.sh
```

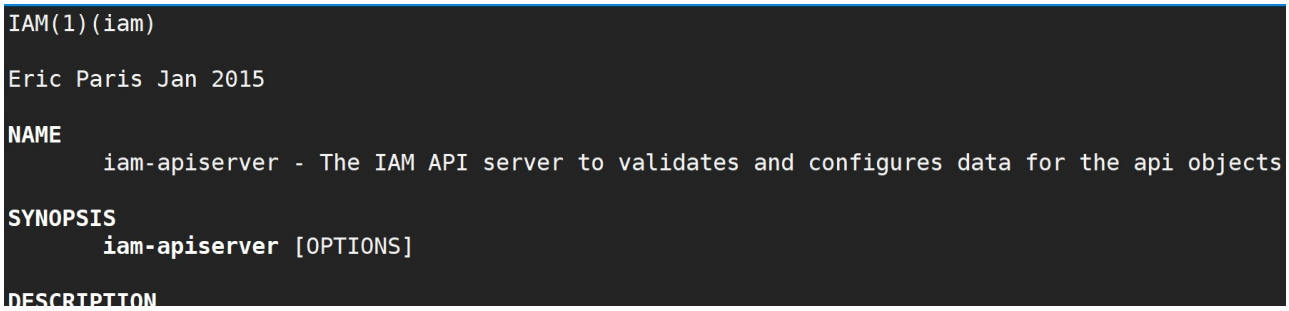
第 2 步，安装生成的 man1 文件。

```
$ sudo cp docs/man/man1/* /usr/share/man/man1/
```

第 3 步，检查是否成功安装 man1 文件。

```
$ man iam-apiserver
```

执行 `man iam-apiserver` 命令后，会弹出 man 文档界面，如下图所示：

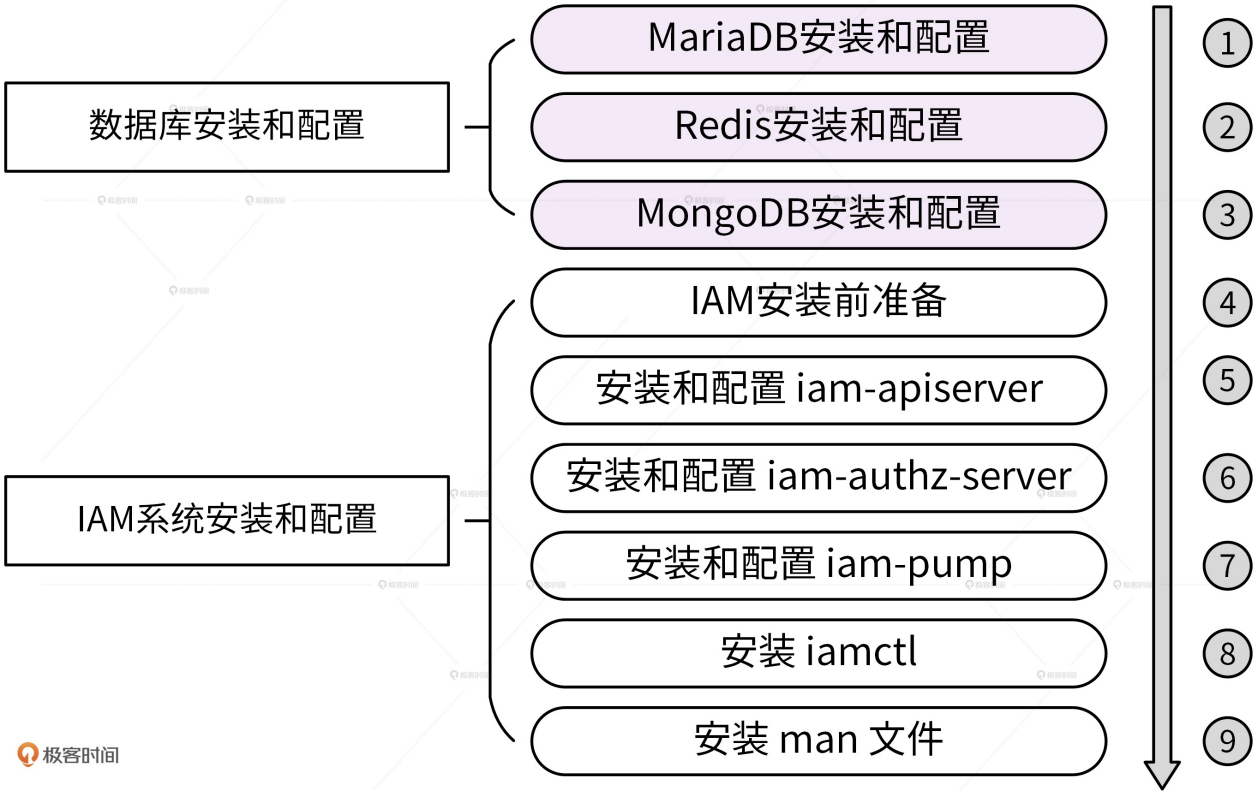


至此，IAM 系统所有组件都已经安装成功了，你可以通过 `iamctl version` 查看客户端和服务端版本，代码如下：

```
$ iamctl version -o yaml
clientVersion:
  buildDate: "2021-04-08T01:56:20Z"
  compiler: gc
  gitCommit: 1d682b0317396347b568a3ef366c1c54b3b0186b
  gitTreeState: dirty
  gitVersion: v0.6.1-5-g1d682b0
  goVersion: go1.16.2
  platform: linux/amd64
serverVersion:
  buildDate: "2021-04-07T22:30:53Z"
  compiler: gc
  gitCommit: bde163964b8c004ebb20ca4abd8a2ac0cd1f71ad
  gitTreeState: dirty
  gitVersion: bde1639
  goVersion: go1.16.2
  platform: linux/amd64
```

总结

这一讲，我带你一步一步安装了 IAM 应用，完成安装的同时，也希望能加深你对 IAM 应用的理解，并为后面的实战准备好环境。为了更清晰地展示安装流程，这里我把整个安装步骤梳理成了一张脑图，你可以看看。



此外，我还有一点想提醒你，我们今天讲到的所有组件设置的密码都是 **iam59!z$**，你一定要记住啦。

课后练习

请你试着调用 iam-apiserver 提供的 API 接口创建一个用户：xuezhang，并在该用户下创建 policy 和 secret 资源。最后调用 iam-authz-server 提供的/v1/authz 接口进行资源鉴权。如果有什么有趣的发现，记得分享出来。

期待在留言区看到你的尝试，我们下一讲见！

彩蛋：一键安装

如果学完了**第02讲**，你可以直接执行如下脚本，来完成 IAM 系统的安装：

```
$ export LINUX_PASSWORD='iam59!z$' # 重要：这里要 export going 用户的密码
$ version=v1.0.0 && curl https://marmotedu-1254073058.cos.ap-beijing.myqcloud.com/iam-release/${version}/ia
$ cd /tmp/iam/ && ./scripts/install/install.sh iam::install::install
```

此外，你也可以参考 IAM 部署指南 教程进行安装，这个安装手册可以让你在创建完普通用户后，一键部署整个 IAM 系统，包括实战环境和 IAM 服务。

精选留言：

- Q 2021-05-27 15:43:39
  ---用户名和密码有错---
  $ curl -s -XPOST -H'Content-Type: application/json' -d '{"username":"admin","password":"Admin@2

021"}' http://127.0.0.1:8080/login
{"message":"incorrect Username or Password"}

2021-05-27 15:36:32.340 INFO gorm@v1.21.4/callbacks.go:124 mysql/user.go:69 ReadMapCB: expect { or n, but found , error found in #0 byte of ...||..., bigger context ...||...[1.701ms] [rows:1] SELECT * FROM `user` WHERE name = 'admin' ORDER BY `user`.`id` LIMIT 1
2021-05-27 15:36:32.340 ERROR apiserver/auth.go:146 get user information failed: ReadMapCB: expect { or n, but found , error found in #0 byte of ...||..., bigger context ...||...
2021-05-27 15:36:32.341 INFO middleware/logger.go:135 401 - [127.0.0.1] "2.055136ms POST /login"
{"requestID": "c4bdae71-6fb4-4a74-9730-06102f5e4e0e", "username": ""}

作者回复2021-05-27 19:47:53
感谢Q哥的反馈。

这个报错最新的master分支和v1.0.0版本已经修复了。

已经安装的同学可以通过以下操作来修复下：
$ git clone --depth=1 https://github.com/marmotedu/iam
$ mysql -h127.0.0.1 -uiam -p

登陆mysql之后执行：
> drop database iam;
> source iam/config/iam.sql

通过以上步骤就可以。
这里失败的原因是，user, secret, policy表中少了一个字段：instanceID。

- Derek 2021-05-27 23:17:31
cd $IAM_ROOT 这个路径是哪个啊？我咋没看到，我的环境变量里没这个值啊 [1赞]

作者回复2021-05-29 10:04:19
文档顺序有问题，文档已经更新了

- CK1.0 2021-05-27 17:51:27
感谢pedro的反馈。

在最新的master分支和v1.0.0 2个问题均已修复。

如果遇到这类问题，可通过以下方式继续安装：

$ cd ${IAM_ROOT}
$ cd ../
$ rm -rf iam
$ git clone --depth=1 https://github.com/marmotedu/iam
$ cd iam
$ ./scripts/install/iam-apiserver.sh iam::apiserver::install
$ ./scripts/install/iam-authz-server.sh iam::authzserver::install
$ ./scripts/install/iam-pump.sh iam::pump::install
$ ./scripts/install/iamctl.sh iam::iamctl::install
$ ./scripts/install/man.sh iam::man::install

$ ./scripts/install/test.sh iam::test::test # 全量测试 [1赞]

- Tiandh 2021-05-28 17:08:52
一键部署 -C 后面应该是 /tmp/
| tar -xz -C / tmp/

作者回复2021-05-29 09:58:11
感谢Tiandh哥反馈，我联系编辑修改下

- Geek_sky 2021-05-28 13:56:31
老师，我在创建mongodb用户db.createUser({user:"root",pwd:"iam59!z$",roles:["root"]})时，报错：
uncaught exception: Error: couldn't add user: command createUser requires authentication :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DB.prototype.createUser@src/mongo/shell/db.js:1386:11
@(shell):1:1
这个需要怎么处理?

作者回复2021-05-29 10:05:39
是不是新创建的mongodb呢，新建的第一次登陆时不用认证的，可以把mongodb卸载重新安装试试

- Iron Man 2021-05-28 11:29:53
文章的顺序有点问题，创建$IAM_ROOT和从github clone代码的描述应该放到数据库安装之前，不然第一个cd $IAM_ROOT命令就报错了

作者回复2021-05-29 10:00:02
已经修复了，感谢反馈！

- kkgo 2021-05-27 23:43:59
已看完，坐等更新

作者回复2021-05-29 10:03:59
大佬，优秀！

- pedro 2021-05-27 11:17:13
老师的脚本玩的真的非常溜，我几乎无阻碍的到了 iam-apiserver 安装这个地方了，但是遇到了两个问题，第一个文中关于 git clone --depth 的命令有错误，depth 没有指定参数，导致 clone 失败；第二个问题很棘手，google 尝试了几个办法都没有解决，在运行：
```sh
make build BINS=iam-apiserver
```
脚本的时候，会报错：
```sh
==========> Building binary iam-apiserver f96a5c8 for linux amd64
verifying github.com/marmotedu/marmotedu-sdk-go@v1.0.0/go.mod: checksum mismatch
downloaded: h1:QAuHe4YwnwlHYcktAFodwYyzxp2lqRDIi0yh1WbLtOM=
go.sum: h1:314QsW/6+tVtngSxPzipgFJNCQMPFtDQQiXC7O66BwM=

SECURITY ERROR
This download does NOT match an earlier download recorded in go.sum.
The bits may have been replaced on the origin server, or an attacker may
have intercepted the download attempt.
```

```

我尝试用 go clean --modcache 和 go mod tidy 都没有解决，还是报校验错误，可能 sdk-go 这个包本身就有问题，这个包又是老师维护的，麻烦老师答疑解惑，多谢了～

作者回复2021-05-27 19:50:22
感谢pedro的反馈。

最新的master分支和v1.0.0版本已经修复了。

这里可以通过下面的方法来修复：
$ rm go.sum
$ go mod tidy

- 赵新星 2021-05-26 20:23:07
为什么不云原生化呢

作者回复2021-05-26 23:30:09
因为云原声化会涉及到很多内容，比如日志，监控，容器化部署等，内容很多，不太适合，在前面加进来，增加上手难度。刚开始也只是准备一个开发联调环境，直接虚拟机部署反而更高效

- dobby 2021-05-26 19:37:51
已看完，等更新咯。

作者回复2021-05-26 23:30:29
优秀!
```