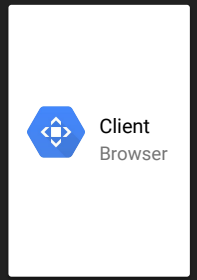
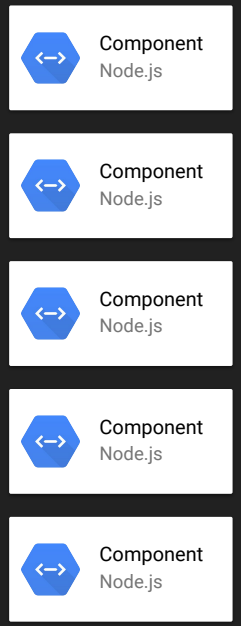
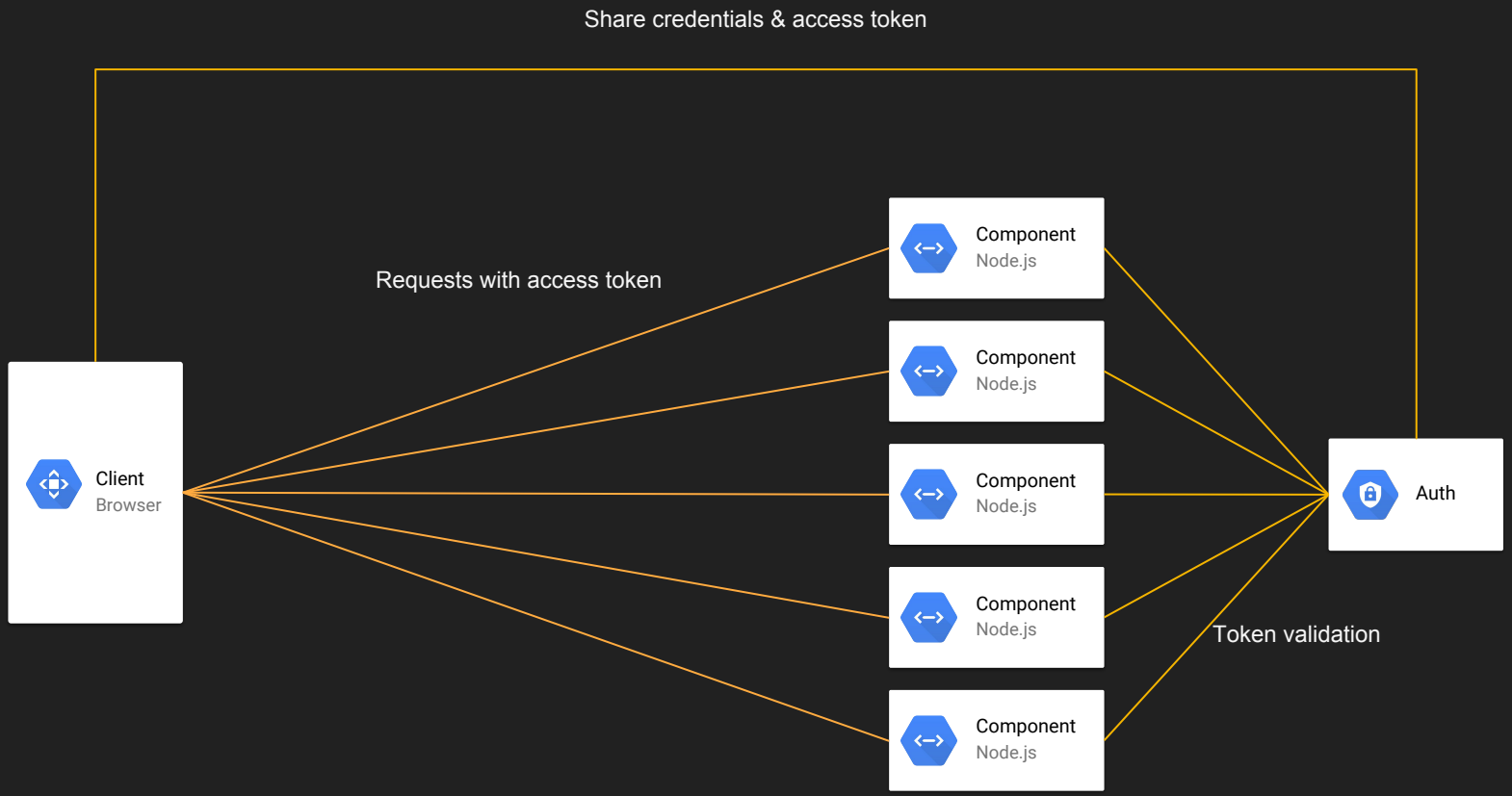
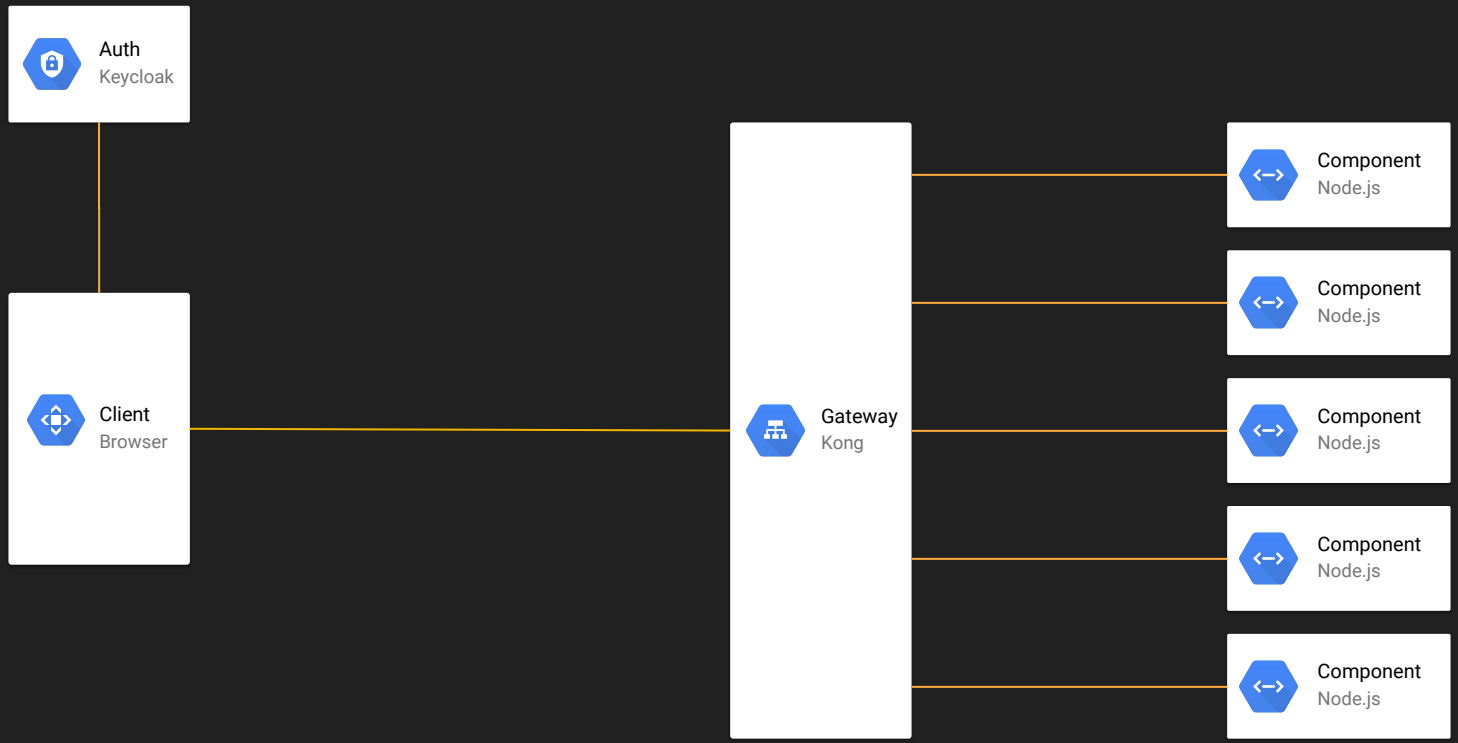
# Securing Microservices with Kong and Keycloak


Client
Browser


Component
Node.js
Component
Node.js
Component
Node.js
Component
Node.js
Component
Node.js


Auth


Share credentials & access token
Requests with access token
Client
Browser
Component
Node.js
Component
Node.js
Component
Node.js
Component
Node.js
Component
Node.js
Auth
Token validation


Auth
Keycloak
Client
Browser
Gateway
Kong
Component
Node.js
Component
Node.js
Component
Node.js
Component
Node.js
Component
Node.js

# Keycloak

*Authentication suite*

Open source
Admin UI & RESTful API
Customizable login views
Email validation, password resets, one-time passwords, etc
Social Login
LDAP & Active Directory
OpenID Connect, OAuth v2
Adapter & Client libraries

Core Concepts:

- Realms
- Clients
- Users

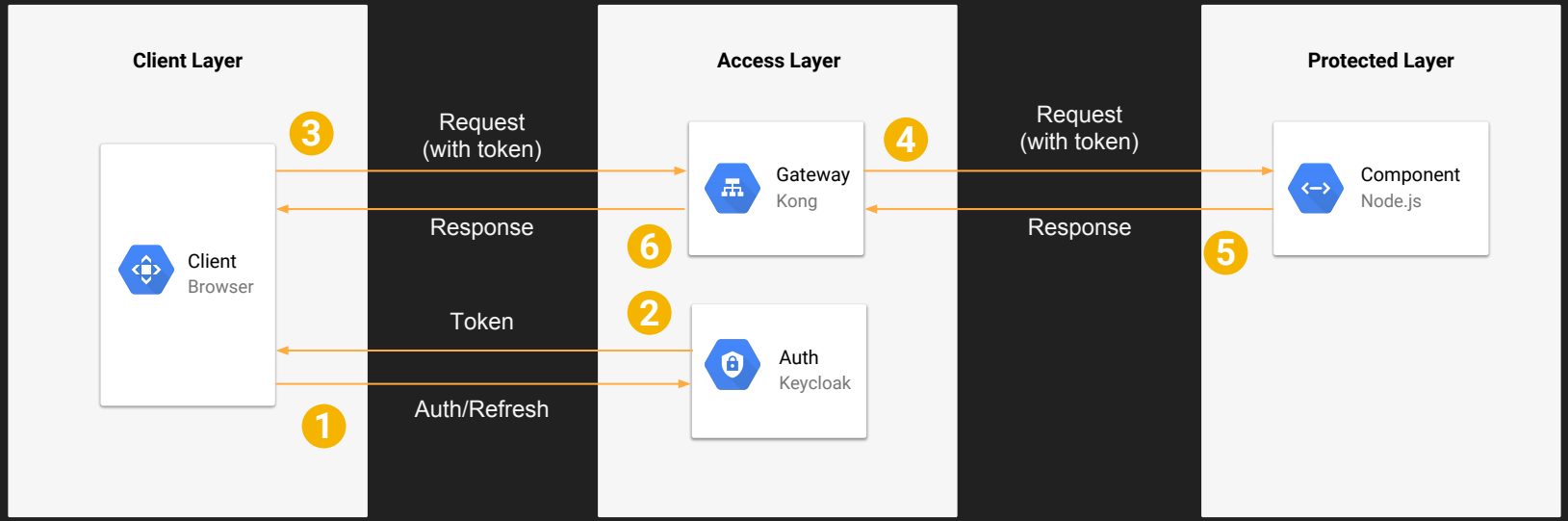
# Kong

*API Gateway*

Open source
Scalable
RESTful API
Plugins
Security
Traffic control
Logging
Transformations

Core Concepts:

- APIs
- Consumers
- Plugins


Client Layer
Access Layer
Protected Layer
3
Request
(with token)
4
Request
(with token)
Client
Browser
Gateway
Kong
Component
Node.js
Response
6
Response
5
2
Token
Auth
Keycloak
1
Auth/Refresh

# The Realm


HOUSE STARK
HOUSE GREYJOY
HOUSE ARRYN
HOUSE TULLY
HOUSE LANNISTER
HOUSE TARGARYEN
HOUSE TYRELL
HOUSE BARATHEON
HOUSE MARTELL

Keycloak
=
Protector of the Realm


The Wall
=
Kong
Gateway

*Step 0:*
Setup example component & client

# setup component

```
app.get('/free', function (req, res) {
  res.json(['cat', 'dog', 'cow'])
})

app.get('/paid', function (req, res) {
  if (!req.headers['authorization']) return res.end()
  let roles = getRoles(req)
  if (roles.includes('subscriber'))
    res.json(['super cat', 'super dog', 'super cow'])
  else
    res.json({ message: 'Nope, pay first' })
})

app.listen(3001)
```

# # setup client

```
const keycloak = Keycloak({
  url: 'http://localhost:8080/auth',
  realm: 'demo-realm',
  clientId: 'demo-client'
})

keycloak.init({ onLoad: 'login-required' })
  .error(function () { alert('error') })
  .success(function (authenticated) {
    if (authenticated) alert('Authenticated')
  })

function getFree () {
  get('http://localhost:8000/component/free')
}

function getPaid () {
  get('http://localhost:8000/component/paid')
}
```

# # setup client

```
function get (route) {
  let req = new XMLHttpRequest()
  req.open('GET', route, true)
  req.setRequestHeader('Accept', 'application/json')
  req.setRequestHeader('Authorization', 'Bearer ' + keycloak.token)
  req.onreadystatechange = function () {
    if (req.readyState === 4) {
      if (req.status === 200) {
        alert('Response: ' + req.responseText)
      } else {
        alert('Request returned: ' + req.status)
      }
    }
  }
  req.send()
}
```

*Step 1:*
Initialize Kong + Keycloak… Kongcloak? :)

# init postgres

```
docker run -d --name kong-database \
  -p 5432:5432 \
  -e "POSTGRES_USER=kong" \
  -e "POSTGRES_DB=kong" \
  postgres:9.4
```

# # run db migrations

```
docker run --rm \

  --link kong-database:kong-database \

  -e "KONG_DATABASE=postgres" \

  -e "KONG_PG_HOST=kong-database" \

  kong:latest kong migrations up
```

# init kong

```
docker run -d --name kong \
  --link kong-database:kong-database \
  -e "KONG_DATABASE=postgres" \
  -e "KONG_PG_HOST=kong-database" \
  -p 8000:8000 -p 8443:8443 -p 8001:8001 -p 8444:8444 \
  kong
```

# init keycloak

```
docker run \
  -e KEYCLOAK_USER=$KEYCLOAK_USERNAME \
  -e KEYCLOAK_PASSWORD=$KEYCLOAK_PASSWORD \
  --name keycloak \
  -p 8080:8080 \
  jboss/keycloak
```

*Step 2:*
Configure Kongcloak

# keycloak - specify realm

```
"realm": "demo-realm",
"enabled": true,
"registrationAllowed": false,
"sslRequired": "external",
"requiredCredentials": ["password"],
```

# keycloak - specify clients

```
"clients": [
  {
    "clientId": "demo-client",
    "enabled": true,
    "publicClient": true,
    "redirectUris": ["http://localhost:3000/*"],
    "webOrigins": ["http://localhost:3000"]
  }
],
```

# keycloak - specify roles

```
"roles": {
  "client": {
    "demo-client": [
      {
        "name": "subscriber",
        "description": "Someone who pays subscription fees"
      }
    ]
  }
}
```

# keycloak - specify users

```
"users": [
  {
    "username": "jdoe",
    "enabled": true,
    "email": "jdoe@example.com",
    "credentials": [{ "type": "password", "value": "password" }],
    "clientRoles": {
      "account": ["view-profile", "manage-account"]
    }
  },
  {
    "username": "bgates",
    "enabled": true,
    "email": "bgates@example.com",
    "credentials": [{ "type": "password", "value": "password" }],
    "clientRoles": {
      "account": ["view-profile", "manage-account"],
      "demo-client": ["subscriber"]
    }
  }
],
```

# kong - declare endpoints

```
{
  "name": "component",
  "upstream_url": "http://192.168.1.132:3001",
  "uris": ["/component"],
  "strip_uri": true,
  "plugins": [
    {
      "name": "jwt"
    },
    {
      "name": "cors",
      "config": {
        "origins": ["http://localhost:3000/*"],
        "methods": ["GET"],
        "credentials": true,
        "max_age": 3600
      }
    }
  ]
}
```

# kong - declare consumers

```
{
  "username": "demo-consumer",
  "plugins": [
    {
      "name": "jwt"
    }
  ]
}
```

*Demo!*