Strong, secure & human-free database credentials in Amazon Web Services

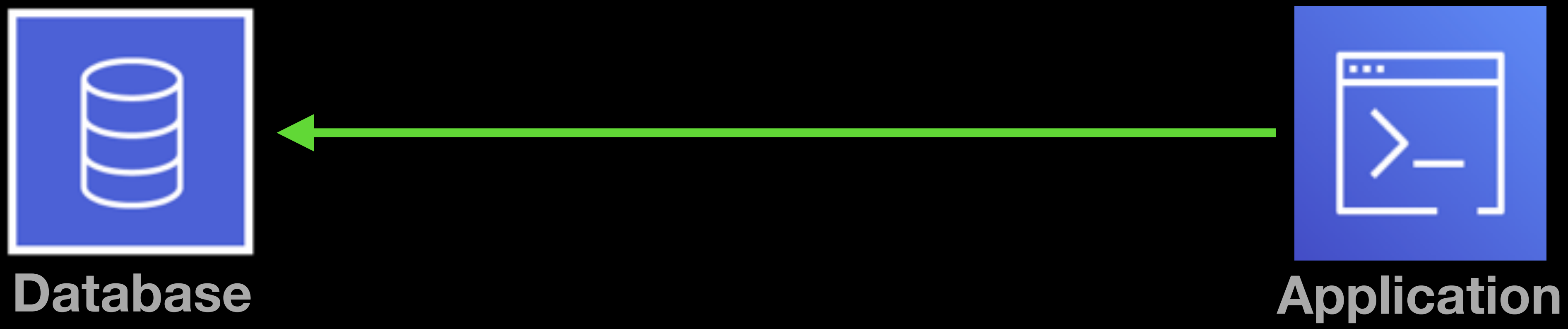
@cariadeccleston / #TechExeter19

~~The Architecture~~

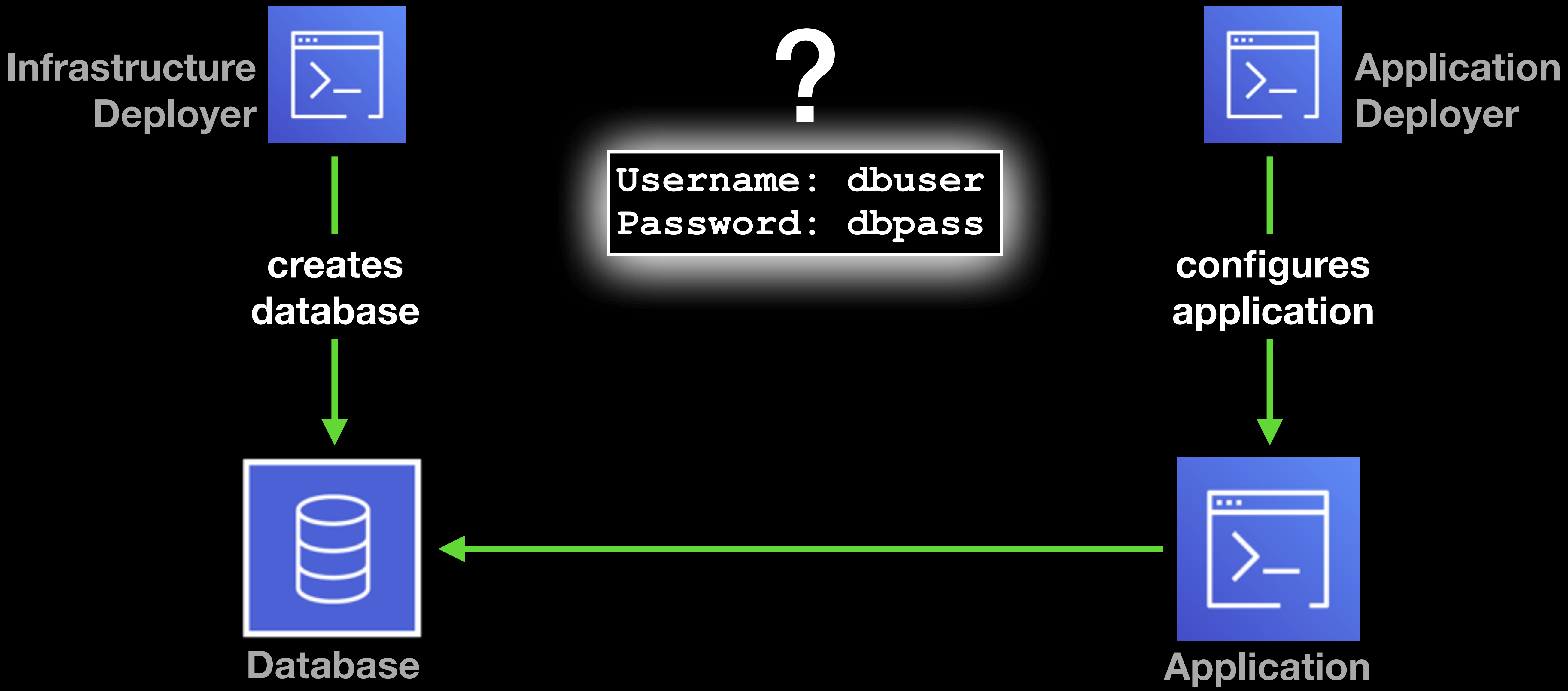
The Challenge

When a database is created…

…the application needs to know the credentials.



Some Kind of Store?



Infrastructure Deployer

creates database

Database

?

```
Username: dbuser
Password: dbpass
```

Application Deployer

configures application

Application

Amazon Web Services is a collection of services.

Which service can help us manage secrets?

Secrets Manager.

What’s a Secret?

Set the value of “my-password” to “trustno1”.

Set the value of “my-credentials” to “{
  "user": "cariad",
  "pass": "trustno1"
}”.

What's the value of “my-password”?

“trustno1”

What's the value of “my-credentials”?

“{
  "user": "cariad",
  "pass": "trustno1"
}”

What makes it “secret”?

Encryption.

What’s a Key?

Encrypt
Decrypt
“gehfgab1”.

“gehfgab1”

If everyone can decrypt,
is it really secret?

Not everyone can.

What's a Policy?

Alice Bob Charlie

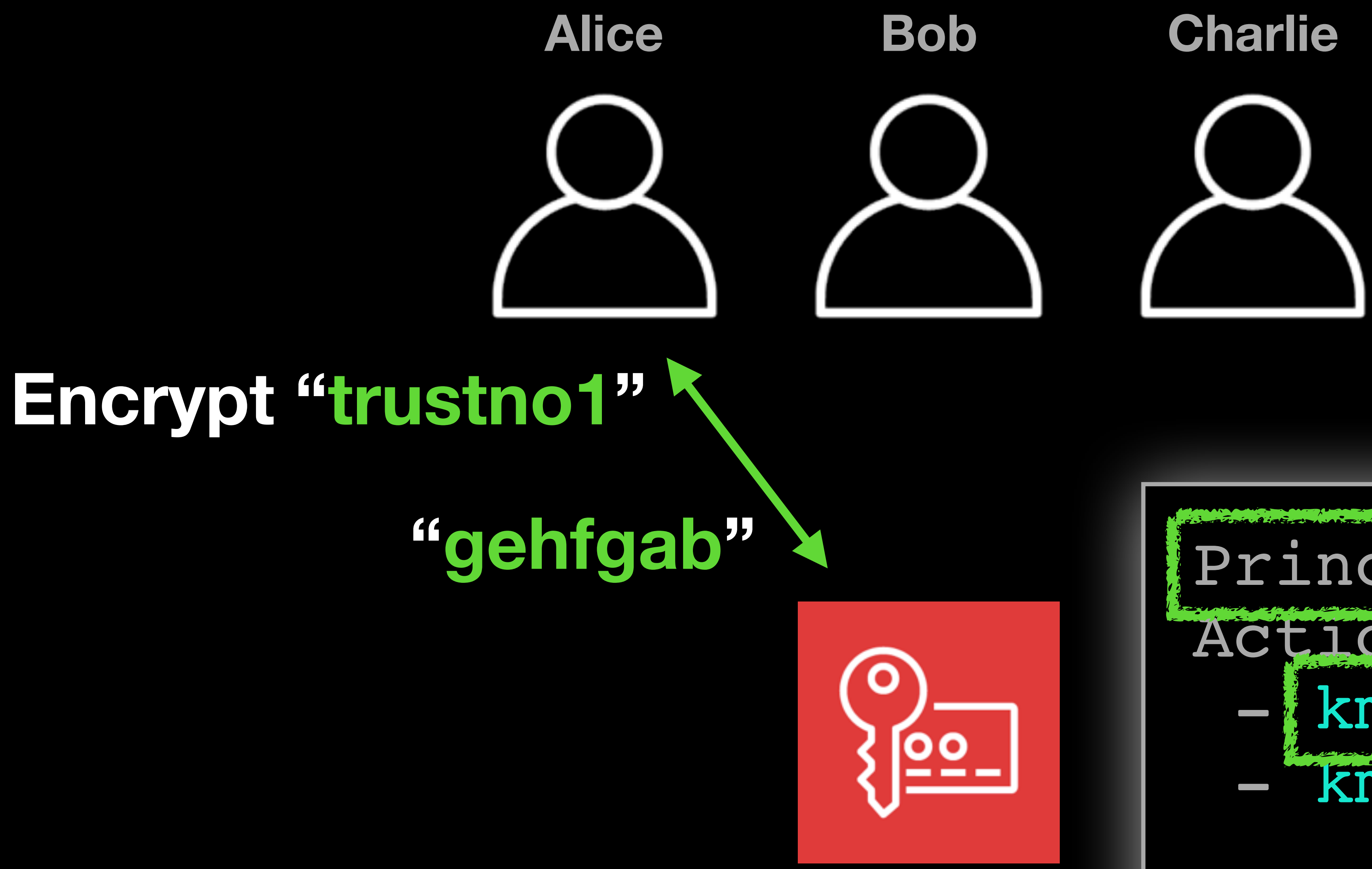
```
Principal: Alice
Action:
  - kms:Encrypt
  - kms:Decrypt

Principal: Bob
Action:
  - kms:Decrypt
```

Who

Service API

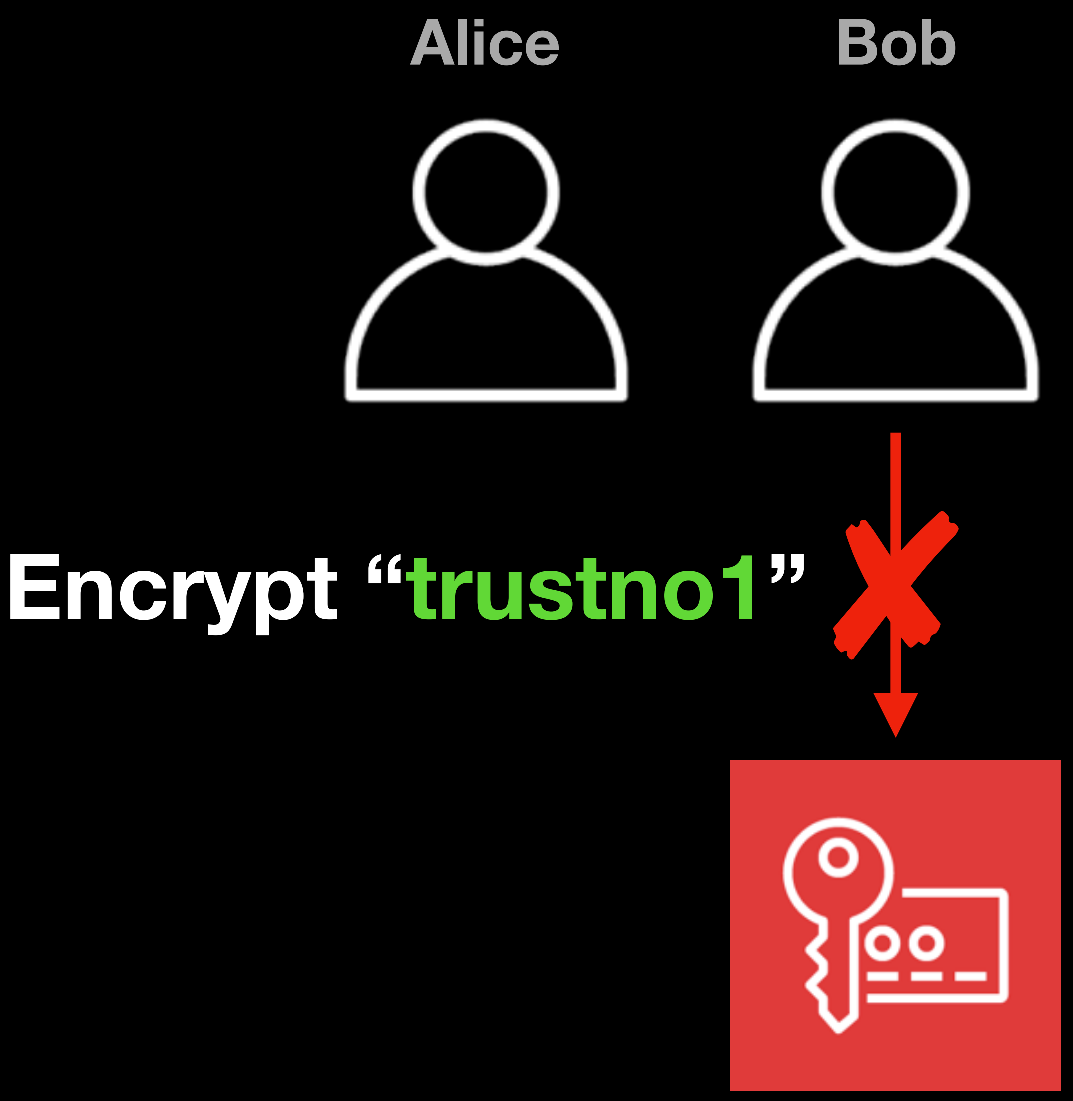
What's a Policy?



```
Principal: Alice
Action:
  - kms:Encrypt
  - kms:Decrypt

Principal: Bob
Action:
  - kms:Decrypt
```

What's a Policy?



Alice

Bob

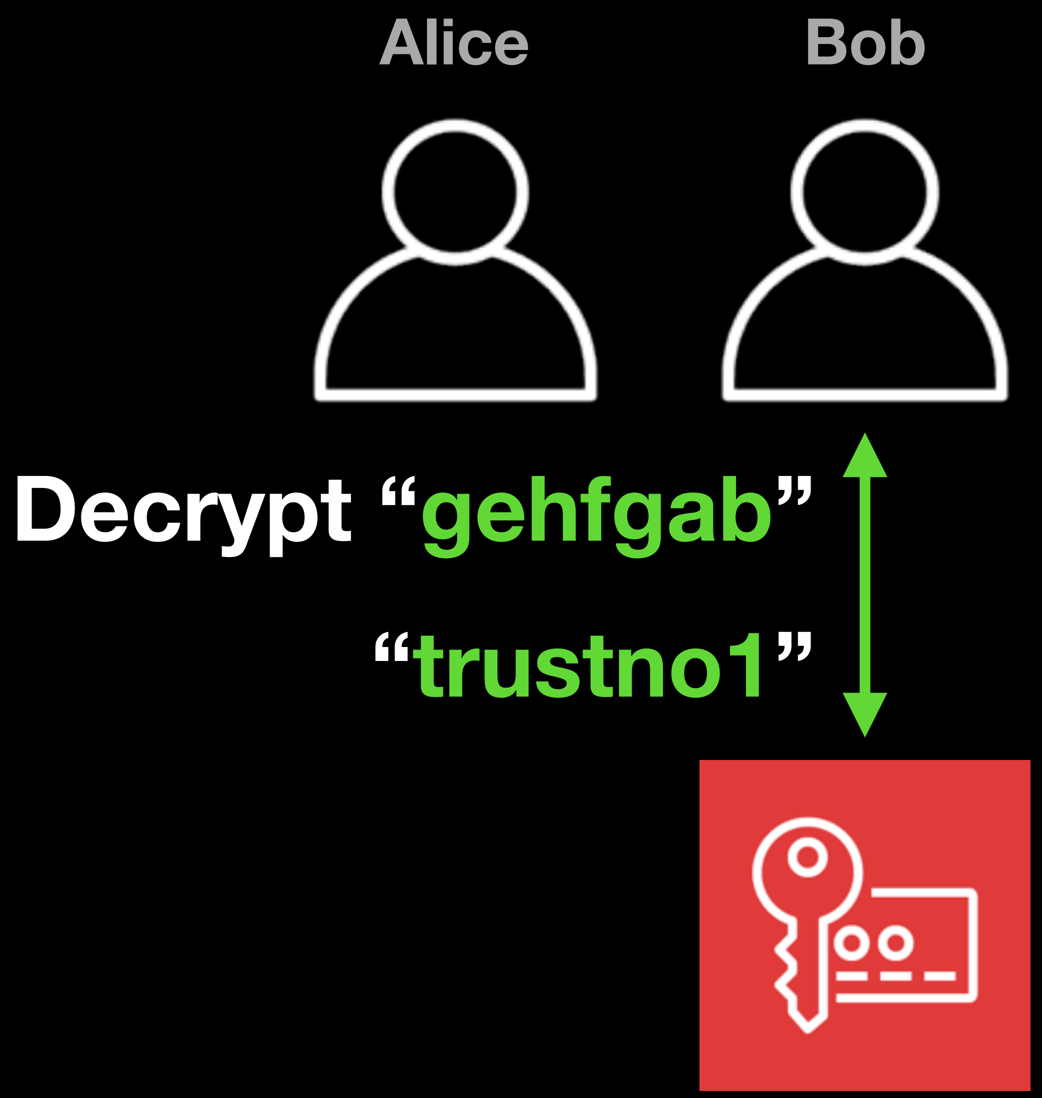
Charlie

Encrypt “trustno1”

```
Principal: Alice
Action:
  - kms:Encrypt
  - kms:Decrypt

Principal: Bob
Action:
  - kms:Decrypt
```

What's a Policy?



Alice

Bob

Charlie

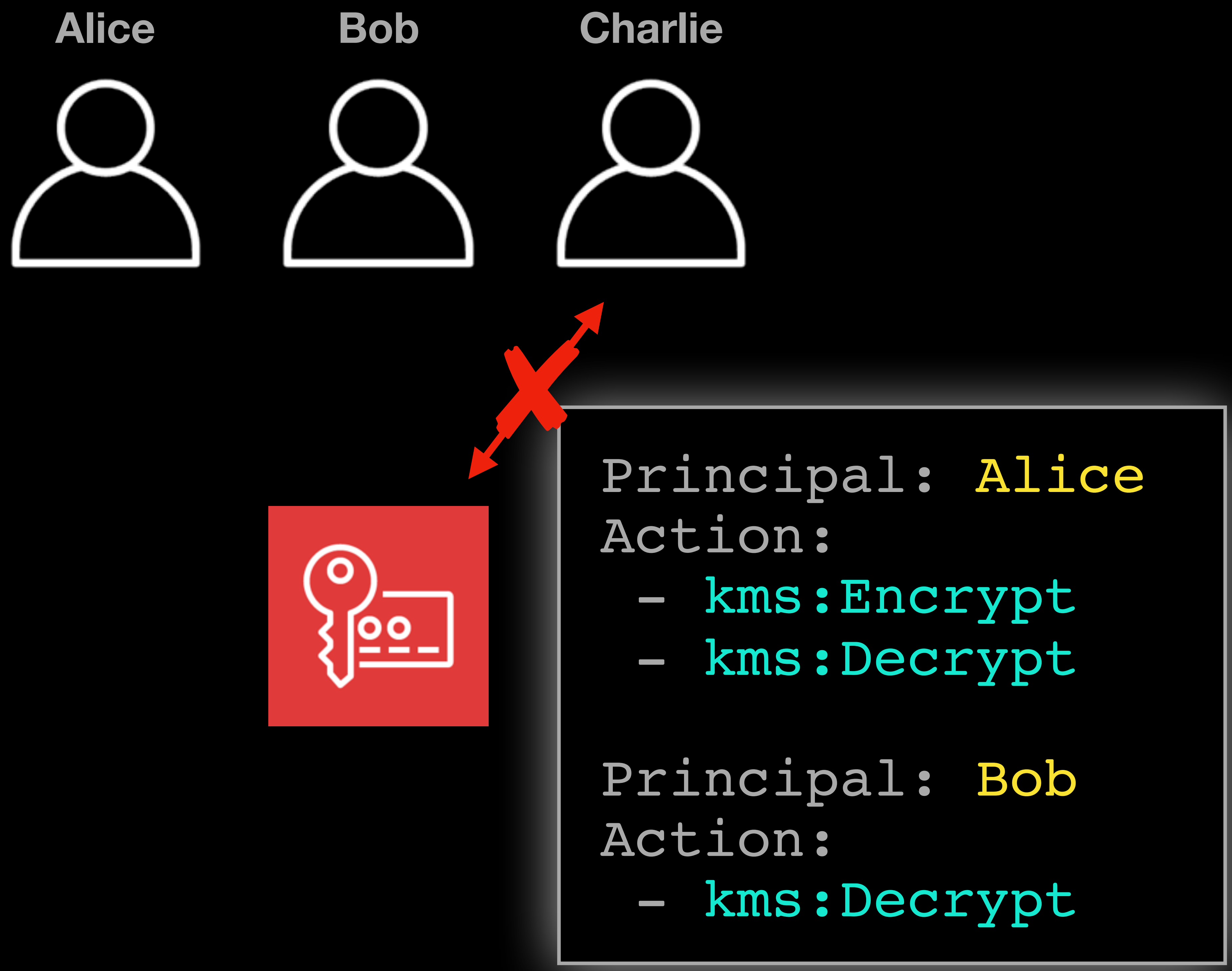
Decrypt “gehfgab”

“trustno1”

```
Principal: Alice
Action:
  - kms:Encrypt
  - kms:Decrypt

Principal: Bob
Action:
  - kms:Decrypt
```

What's a Policy?



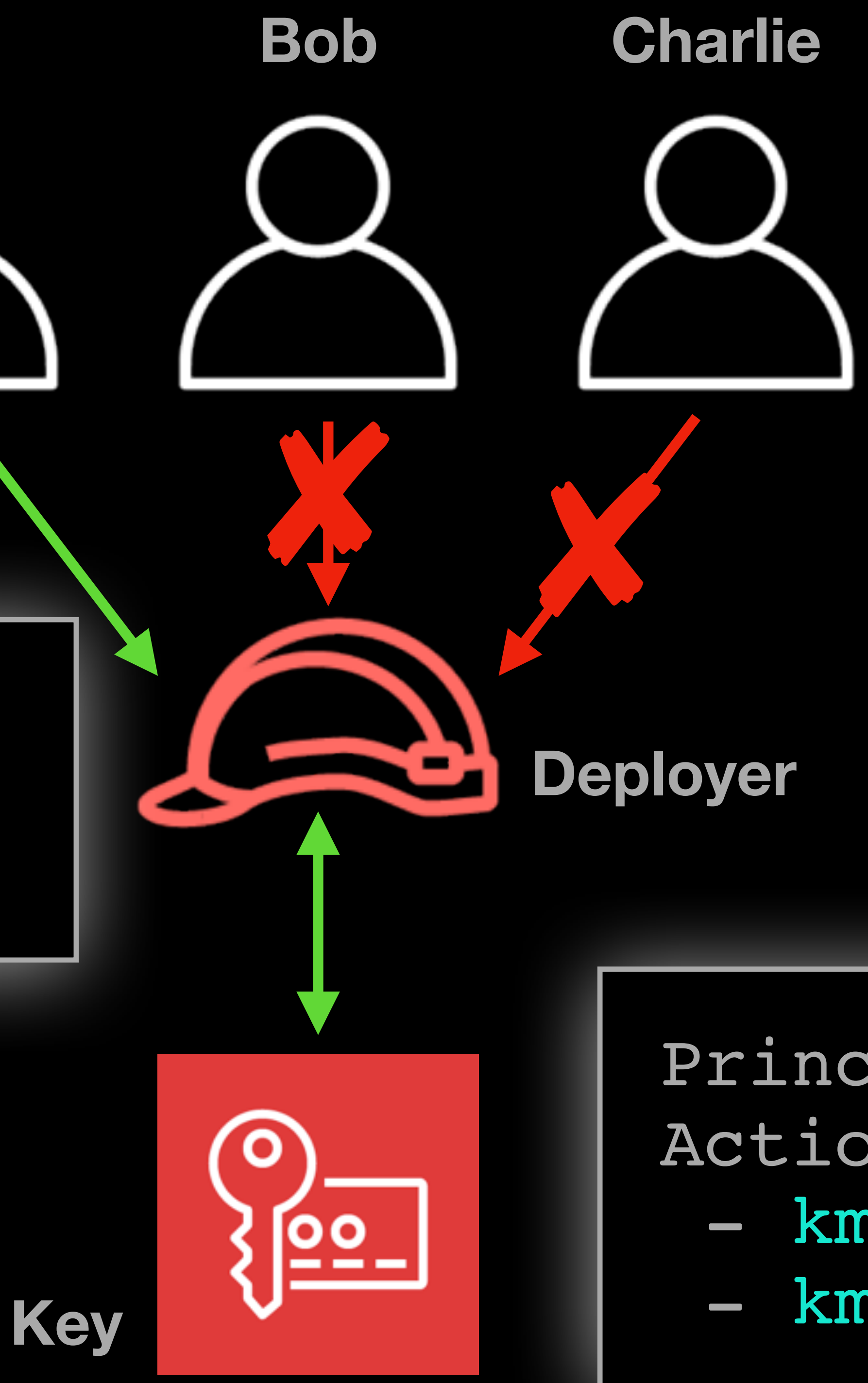
```
Principal: Alice
Action:
  - kms:Encrypt
  - kms:Decrypt

Principal: Bob
Action:
  - kms:Decrypt
```

But Alice doesn't need these permissions all the time.

Only when she's assuming the role of a deployer.

What's a Role?



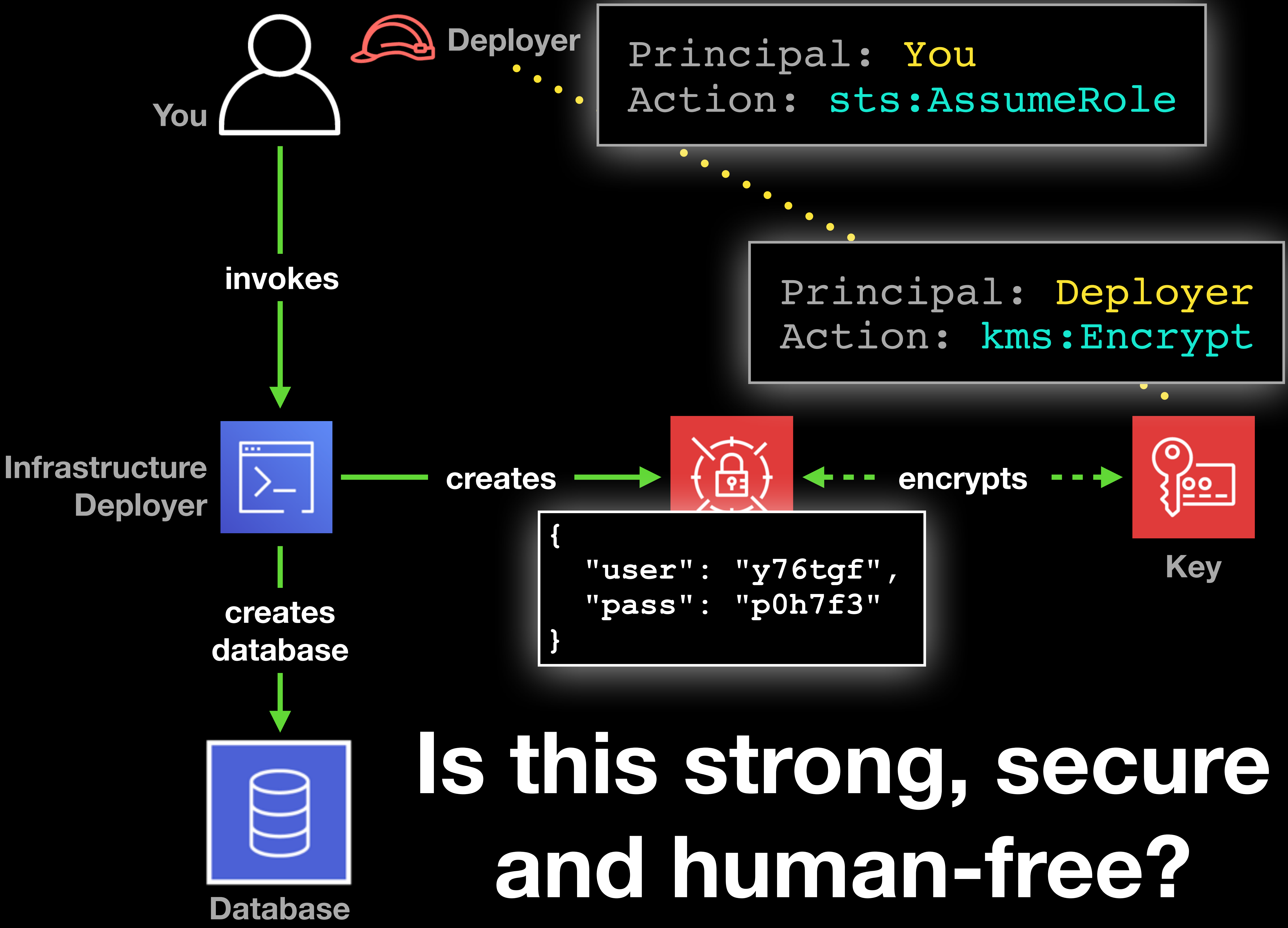
```
Principal: Alice
Action:
 - sts:AssumeRole
```

```
Principal: Deployer
Action:
 - kms:Encrypt
 - kms:Decrypt
```

Technology Recap

- **Secret:** Key/value pair.
- **Key:** Encrypts and decrypts.
- **Role:** Privilege that can be assumed.
- **Policy:** Describes privilege.

I need to deploy a new database for a new app…

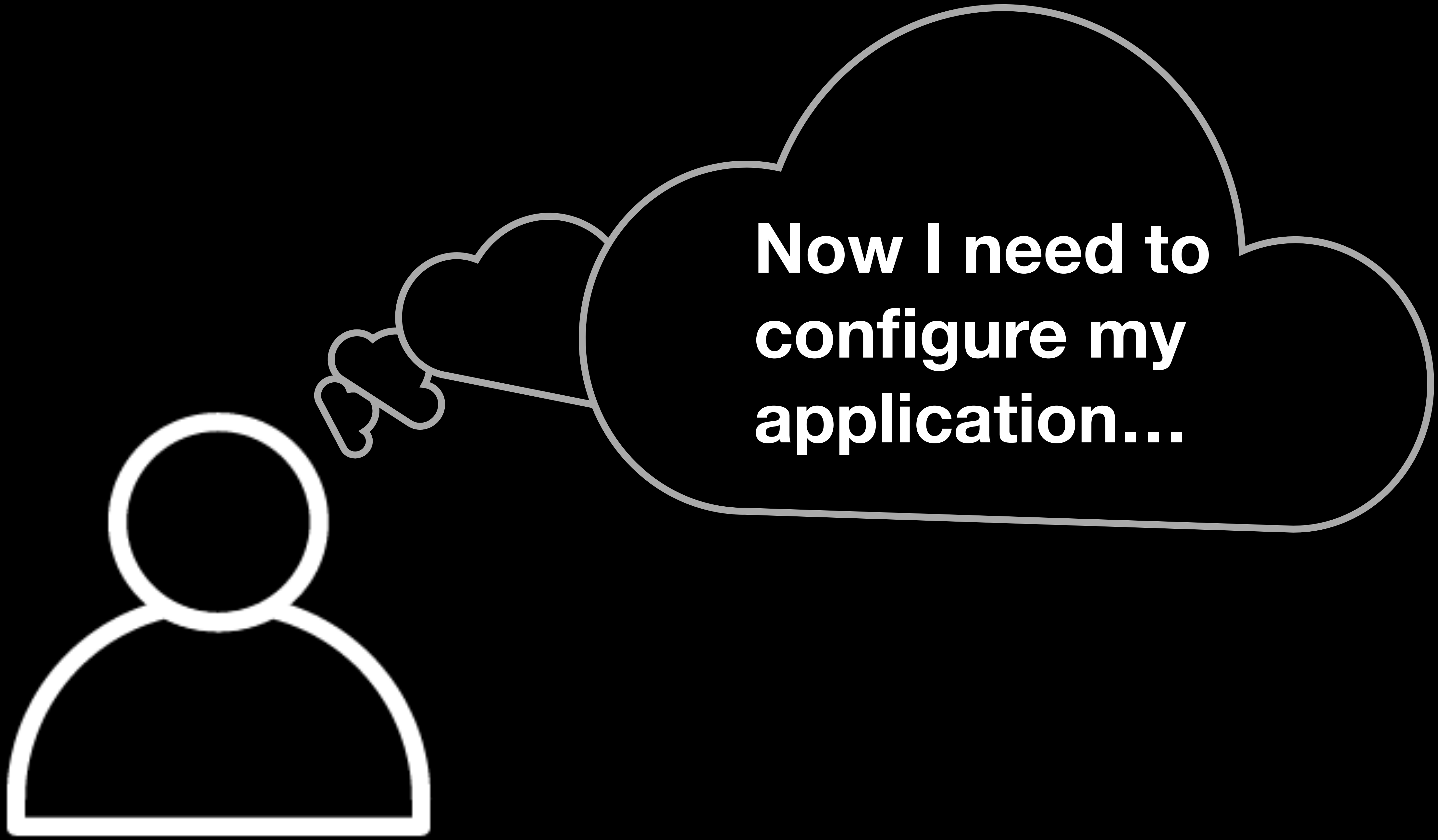


```
Principal: You
Action: sts:AssumeRole
```

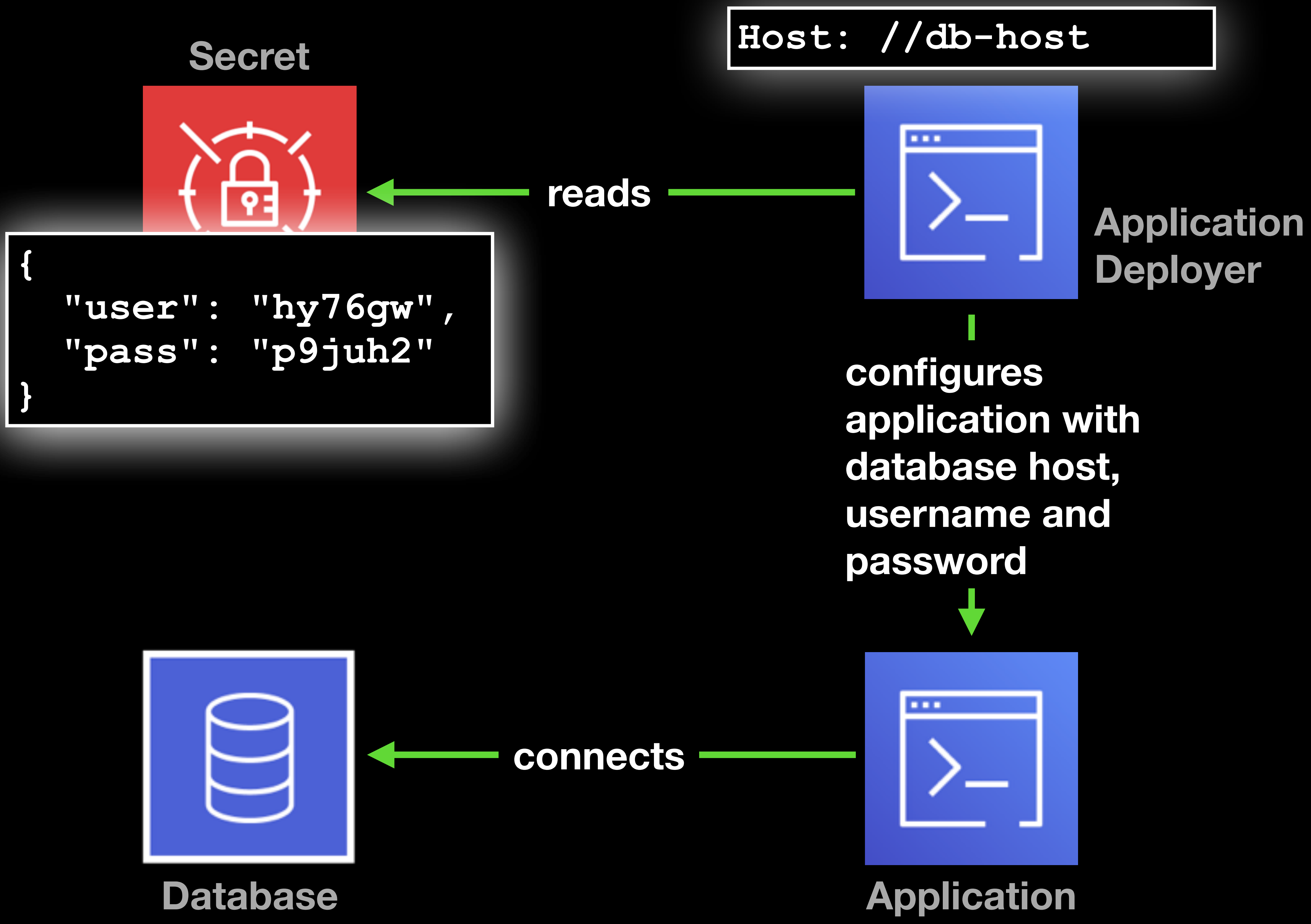
```
Principal: Deployer
Action: kms:Encrypt
```

```
{
  "user": "y76tgf",
  "pass": "p0h7f3"
}
```

Is this strong, secure and human-free?



Now I need to configure my application…



```
Host: //db-host
```

Secret

```
{
   "user": "hy76gw",
   "pass": "p9juh2"
}
```

reads

Application Deployer

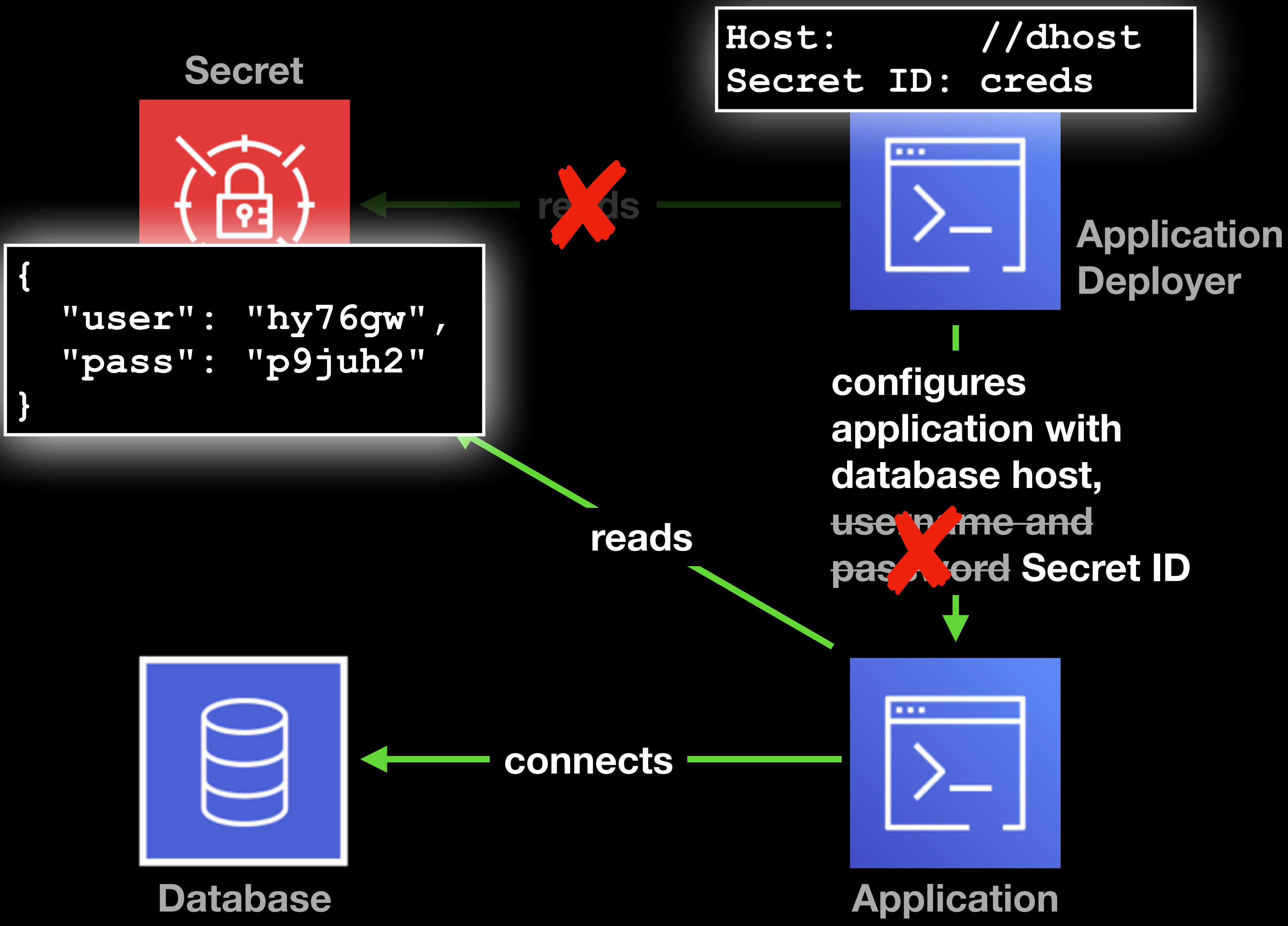
configures application with database host, username and password

Database

connects

Application

Is this secure?



Secret
reads
Application Deployer
configures application with database host, ~~username and password~~ Secret ID
reads
Database
connects
Application

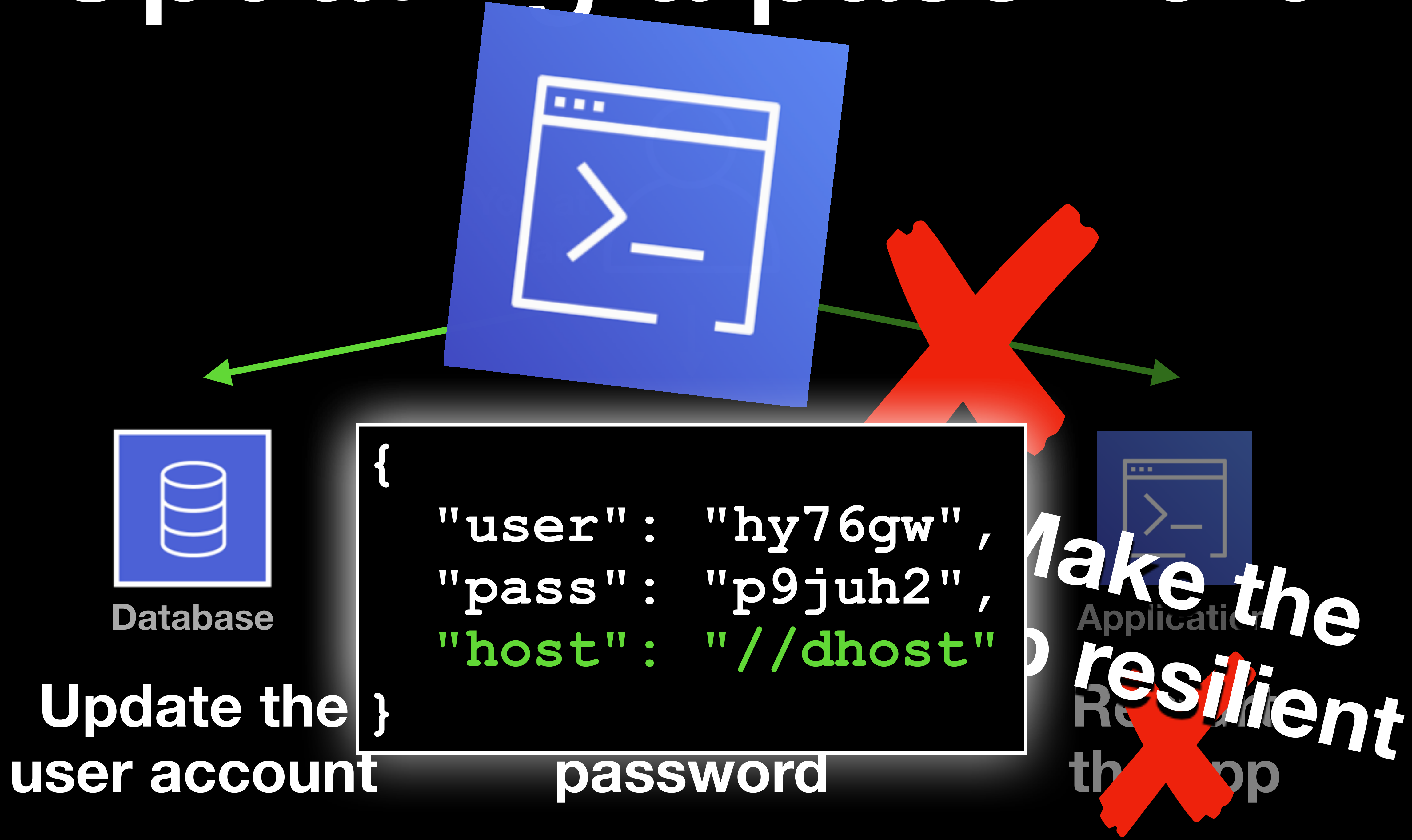
```
Host:       //dhost
Secret ID:  creds
```

```
{
   "user":  "hy76gw",
   "pass":  "p9juh2"
}
```

Prove it.

How could we react to leaked credentials?

Updating a password



```
{
   "user": "hy76gw",
   "pass": "p9juh2",
   "host": "//dhost"
}
```

Update the user account

password

~~Restart the app~~

Make the app resilient

Prove it.

- **Strong:** Maximum **length.**
  Full **complexity** rules.
  No pattern **bias**.

- **Secure:** **Invisible** by default.
  No credentials **on-machine**.
  Blind password rotation.

- **Human-free:** Don't **make-up** credentials.
  Don't **know** credentials.
  Push-button operations.

@cariadeccleston
#TechExeter19

http://cariad.me

http://github.com/cariad/
aws-postgresql-secrets-
environment-demo