Creating a signing certificate

Let's create a production ready certificate that the Identity microservice can use for signing tokens.

In Postman

1. Do a GET for the discovery endpoint

2. Copy the value of jwks_uri
   https://playeconomy.eastus.cloudapp.azure.com/identity-svc/.well-known/openid-configuration/jwks

3. Do a GET for jwks_uri

4. Explain this is still a temporary sign in credential stored in disk. Only good for a single machine.

5. Copy the sign in credential somewhere

6. Get an access token

7. Decode the generated token

8. Show that kid in the access token matches the one in the jwks_uri

9. Do a GET on /users. It should work.

10. Destroy the identity pod

11. Once new pod starts, do a GET on jwks_uri again

12. Compare new value with old value. Notice kid changed

13. Do a GET on /users. It should not work.

14. Explain that if value changes the tokens won't work anymore. We need a permanent signing credential.

In Play.Identity

15. Add signing-cer.yaml:

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: signing-cert
spec:
```

```yaml
  secretName: signing-cert
  issuerRef:
    name: letsencrypt-prod
    kind: ClusterIssuer
  dnsNames:
    - playeconomy.eastus.cloudapp.azure.com
```

16. Update README:

~~~
## Create the signing certificate
```powershell
kubectl apply -f .\kubernetes\signing-cert.yaml -n $namespace
```
~~~

17. Run the command

18. Explore the generated secret:

```
kubectl get secret signing-cert -n identity -o yaml
```

19. Update identity.yaml:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  …
spec:
  …
  template:
    …
    spec:
      containers:
        - name: identity
          …
          env:
            - name: ServiceSettings__MessageBroker
              value: SERVICEBUS
            - name: ServiceSettings__KeyVaultName
              value: playeconomy
            - name: IdentitySettings__PathBase
              value: /identity-svc
            - name: IdentitySettings__CertificateCerFilePath
              value: "/certificates/certificate.crt"
            - name: IdentitySettings__CertificateKeyFilePath
              value: "/certificates/certificate.key"
```

```
     …
     readinessProbe:
     …
     volumeMounts:
      - name: certificate-volume
       mountPath: /certificates
   volumes:
    - name: certificate-volume
     secret:
      secretName: signing-cert
      items:
       - key: tls.key
        path: certificate.key
       - key: tls.crt
        path: certificate.crt
…
```

20. Commit and push

In the next lesson you will update the Identity service to be able to use the new signing certificate.