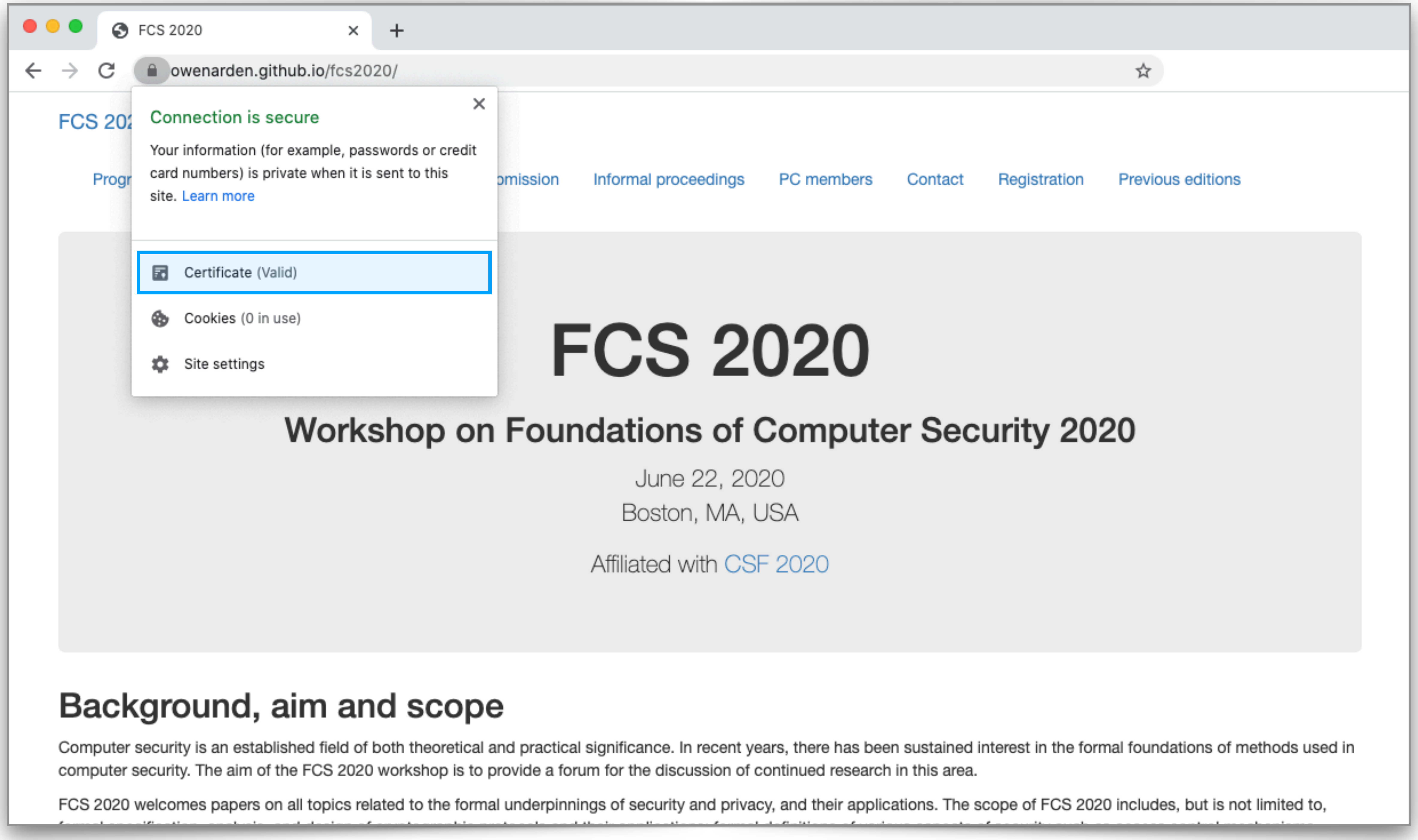
Assertion-Carrying Certificates

Waqar Aqeel, Zachary Hanif, James Larisch, Olamide Omolola,
Taejoong Chung, Dave Levin, Bruce Maggs, Alan Mislove, Bryan Parno, Christo Wilson

Duke University · University of Maryland · Harvard University · TU Graz, Graz University of Technology · Northeastern University · Virginia Tech · Carnegie Mellon University

The Public Key Infrastructure is how users know with whom they are communicating online



Connection is secure

Your information (for example, passwords or credit card numbers) is private when it is sent to this site. Learn more

Certificate (Valid)

Cookies (0 in use)

Site settings

FCS 2020

Workshop on Foundations of Computer Security 2020

June 22, 2020
Boston, MA, USA

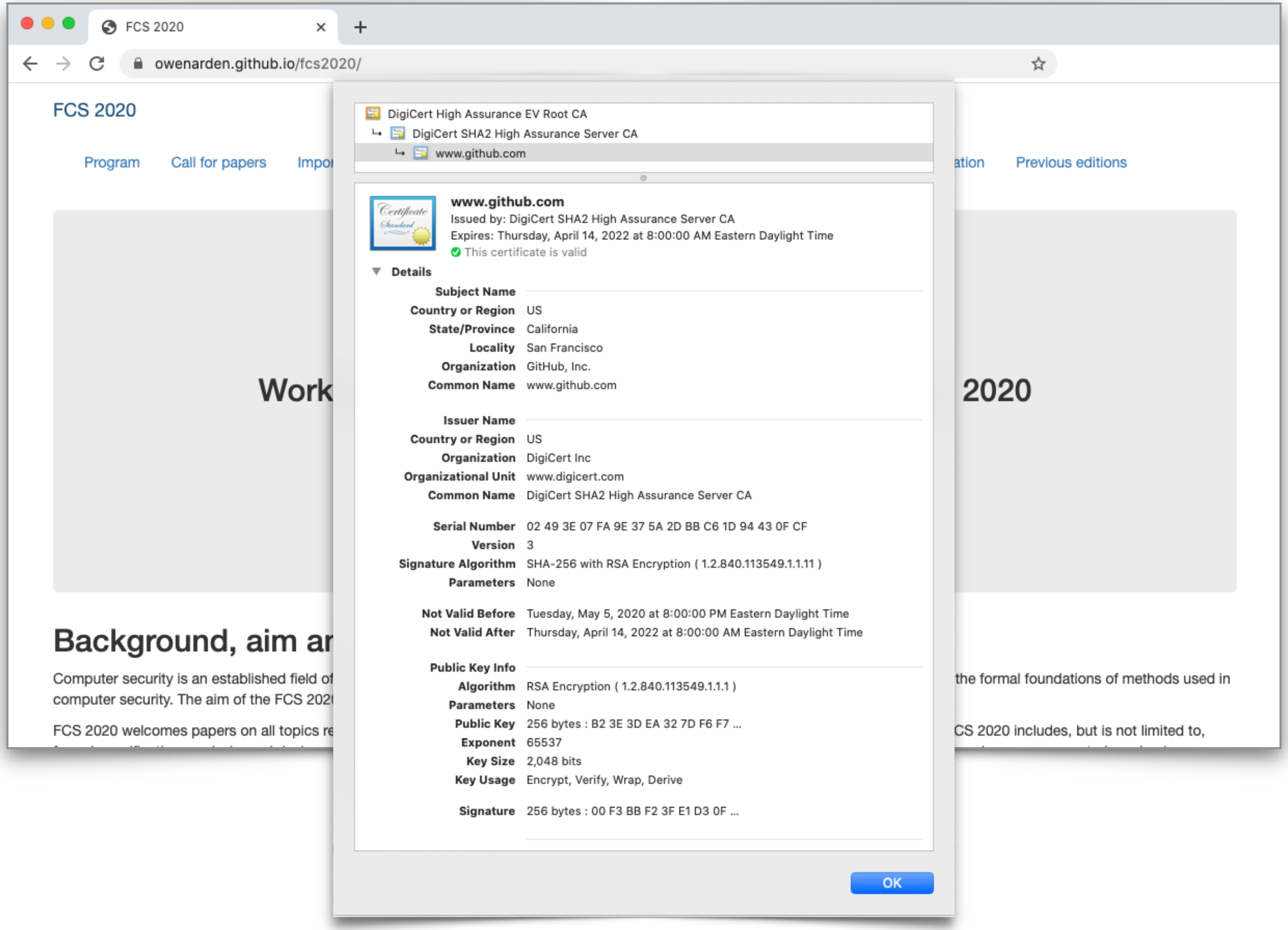
Affiliated with CSF 2020

Background, aim and scope

Computer security is an established field of both theoretical and practical significance. In recent years, there has been sustained interest in the formal foundations of methods used in computer security. The aim of the FCS 2020 workshop is to provide a forum for the discussion of continued research in this area.

FCS 2020 welcomes papers on all topics related to the formal underpinnings of security and privacy, and their applications. The scope of FCS 2020 includes, but is not limited to,

Certificates encapsulate identity (who hosts are) and capability (what they can do)



DigiCert High Assurance EV Root CA
↳ DigiCert SHA2 High Assurance Server CA
↳ www.github.com

www.github.com
Issued by: DigiCert SHA2 High Assurance Server CA
Expires: Thursday, April 14, 2022 at 8:00:00 AM Eastern Daylight Time
✓ This certificate is valid

▼ **Details**

Subject Name	
Country or Region	US
State/Province	California
Locality	San Francisco
Organization	GitHub, Inc.
Common Name	www.github.com
Issuer Name	
Country or Region	US
Organization	DigiCert Inc
Organizational Unit	www.digicert.com
Common Name	DigiCert SHA2 High Assurance Server CA
Serial Number	02 49 3E 07 FA 9E 37 5A 2D BB C6 1D 94 43 0F CF
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	None
Not Valid Before	Tuesday, May 5, 2020 at 8:00:00 PM Eastern Daylight Time
Not Valid After	Thursday, April 14, 2022 at 8:00:00 AM Eastern Daylight Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	None
Public Key	256 bytes : B2 3E 3D EA 32 7D F6 F7 ...
Exponent	65537
Key Size	2,048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	256 bytes : 00 F3 BB F2 3F E1 D3 0F ...

Background, aim ar

Computer security is an established field of … the formal foundations of methods used in computer security. The aim of the FCS 202

FCS 2020 welcomes papers on all topics re … CS 2020 includes, but is not limited to,

Certificates encapsulate identity (who hosts are) and capability (what they can do)

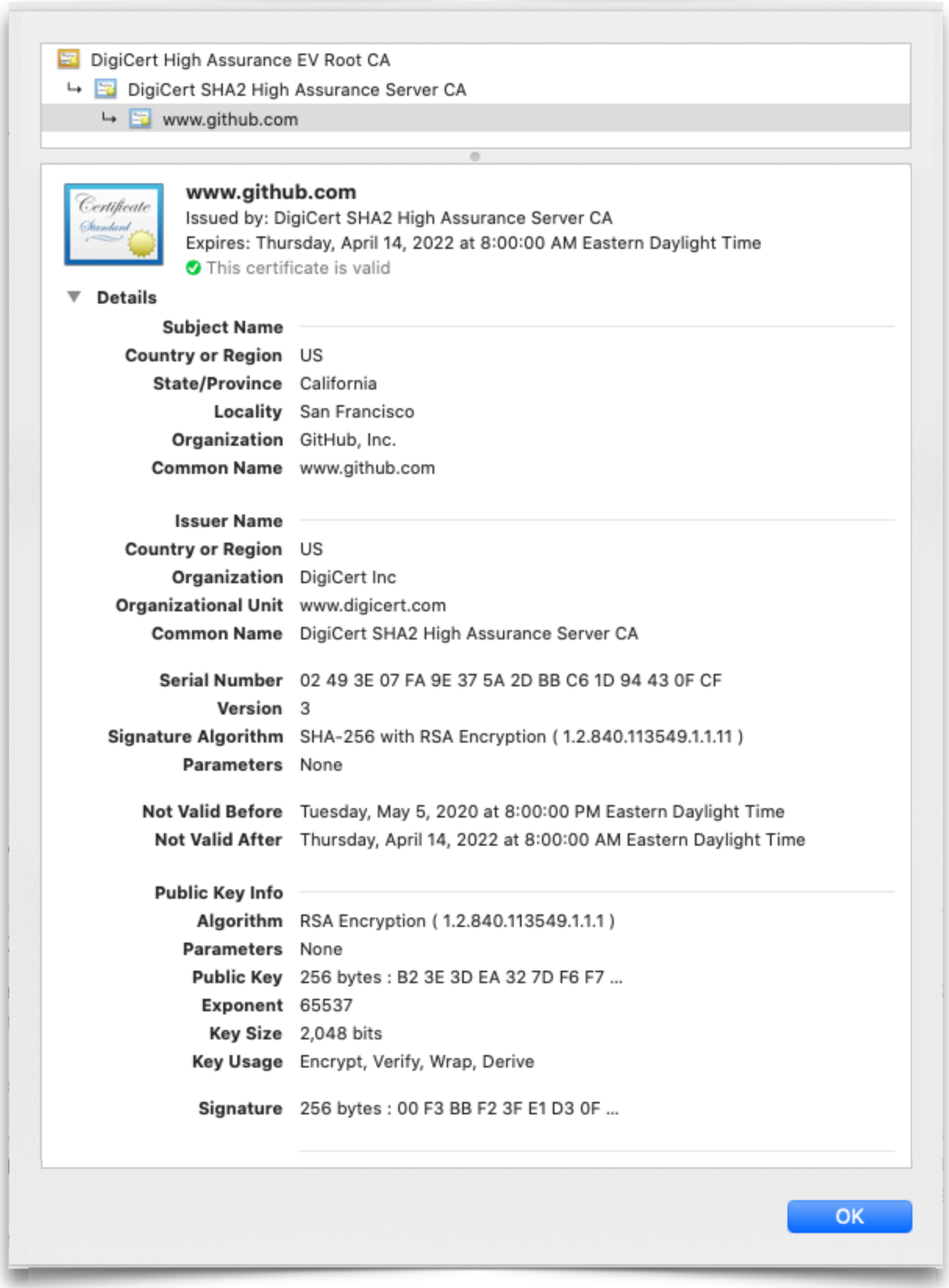
Traditional PKI roles

Subject Name
Who the cert is about

Issuer Name
Who vetted the subject's identity

Expiration Dates
When is the certificate no longer valid

Public key and signature
Attestation of cryptographic identity



DigiCert High Assurance EV Root CA
↳ DigiCert SHA2 High Assurance Server CA
↳ www.github.com

www.github.com
Issued by: DigiCert SHA2 High Assurance Server CA
Expires: Thursday, April 14, 2022 at 8:00:00 AM Eastern Daylight Time
✓ This certificate is valid

▼ **Details**

Subject Name	
Country or Region	US
State/Province	California
Locality	San Francisco
Organization	GitHub, Inc.
Common Name	www.github.com
Issuer Name	
Country or Region	US
Organization	DigiCert Inc
Organizational Unit	www.digicert.com
Common Name	DigiCert SHA2 High Assurance Server CA
Serial Number	02 49 3E 07 FA 9E 37 5A 2D BB C6 1D 94 43 0F CF
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	None
Not Valid Before	Tuesday, May 5, 2020 at 8:00:00 PM Eastern Daylight Time
Not Valid After	Thursday, April 14, 2022 at 8:00:00 AM Eastern Daylight Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	None
Public Key	256 bytes : B2 3E 3D EA 32 7D F6 F7 ...
Exponent	65537
Key Size	2,048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	256 bytes : 00 F3 BB F2 3F E1 D3 0F ...

The PKI has had to evolve to meet new threats, deployments, and opportunities

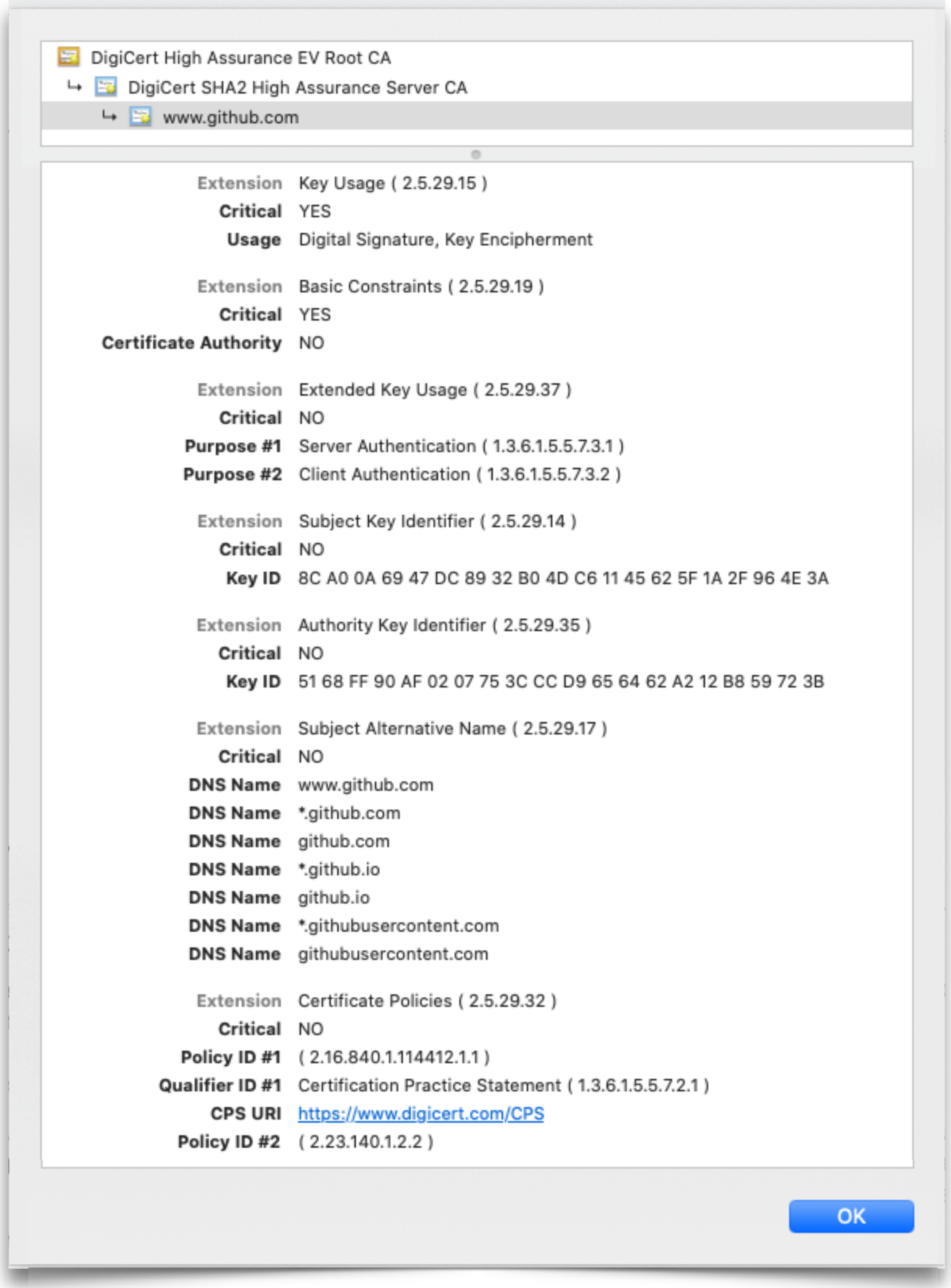
Traditional PKI roles

Subject Name
Who the cert is about

Issuer Name
Who vetted the subject’s identity

Expiration Dates
When is the certificate no longer valid

Public key and signature
Attestation of cryptographic identity



```
DigiCert High Assurance EV Root CA
 ↳ DigiCert SHA2 High Assurance Server CA
    ↳ www.github.com

           Extension  Key Usage ( 2.5.29.15 )
            Critical  YES
               Usage  Digital Signature, Key Encipherment

           Extension  Basic Constraints ( 2.5.29.19 )
            Critical  YES
Certificate Authority  NO

           Extension  Extended Key Usage ( 2.5.29.37 )
            Critical  NO
          Purpose #1  Server Authentication ( 1.3.6.1.5.5.7.3.1 )
          Purpose #2  Client Authentication ( 1.3.6.1.5.5.7.3.2 )

           Extension  Subject Key Identifier ( 2.5.29.14 )
            Critical  NO
              Key ID  8C A0 0A 69 47 DC 89 32 B0 4D C6 11 45 62 5F 1A 2F 96 4E 3A

           Extension  Authority Key Identifier ( 2.5.29.35 )
            Critical  NO
              Key ID  51 68 FF 90 AF 02 07 75 3C CC D9 65 64 62 A2 12 B8 59 72 3B

           Extension  Subject Alternative Name ( 2.5.29.17 )
            Critical  NO
            DNS Name  www.github.com
            DNS Name  *.github.com
            DNS Name  github.com
            DNS Name  *.github.io
            DNS Name  github.io
            DNS Name  *.githubusercontent.com
            DNS Name  githubusercontent.com

           Extension  Certificate Policies ( 2.5.29.32 )
            Critical  NO
         Policy ID #1  ( 2.16.840.1.114412.1.1 )
      Qualifier ID #1  Certification Practice Statement ( 1.3.6.1.5.5.7.2.1 )
             CPS URI  https://www.digicert.com/CPS
         Policy ID #2  ( 2.23.140.1.2.2 )
```

New additions to the PKI

Key Usage
Certificate signing, authentication

Subject Alternate Names
Support deployments in CDNs

The PKI has had to evolve to meet new threats, deployments, and opportunities

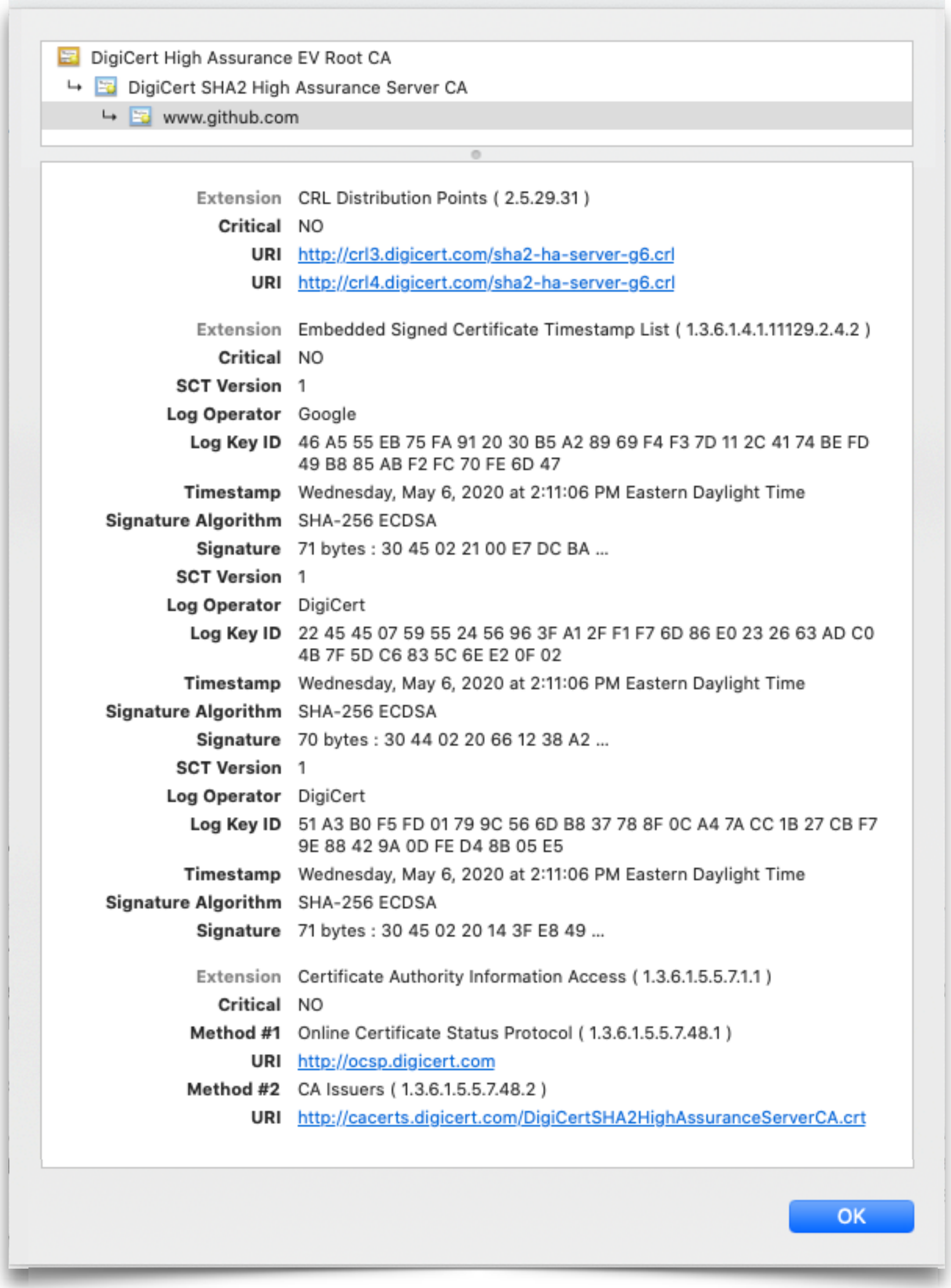
Traditional PKI roles

Subject Name
Who the cert is about

Issuer Name
Who vetted the subject’s identity

Expiration Dates
When is the certificate no longer valid

Public key and signature
Attestation of cryptographic identity



```
DigiCert High Assurance EV Root CA
  ↳ DigiCert SHA2 High Assurance Server CA
      ↳ www.github.com

         Extension  CRL Distribution Points ( 2.5.29.31 )
          Critical  NO
               URI  http://crl3.digicert.com/sha2-ha-server-g6.crl
               URI  http://crl4.digicert.com/sha2-ha-server-g6.crl

         Extension  Embedded Signed Certificate Timestamp List ( 1.3.6.1.4.1.11129.2.4.2 )
          Critical  NO
       SCT Version  1
      Log Operator  Google
        Log Key ID  46 A5 55 EB 75 FA 91 20 30 B5 A2 89 69 F4 F3 7D 11 2C 41 74 BE FD
                    49 B8 85 AB F2 FC 70 FE 6D 47
         Timestamp  Wednesday, May 6, 2020 at 2:11:06 PM Eastern Daylight Time
Signature Algorithm  SHA-256 ECDSA
         Signature  71 bytes : 30 45 02 21 00 E7 DC BA ...
       SCT Version  1
      Log Operator  DigiCert
        Log Key ID  22 45 45 07 59 55 24 56 96 3F A1 2F F1 F7 6D 86 E0 23 26 63 AD C0
                    4B 7F 5D C6 83 5C 6E E2 0F 02
         Timestamp  Wednesday, May 6, 2020 at 2:11:06 PM Eastern Daylight Time
Signature Algorithm  SHA-256 ECDSA
         Signature  70 bytes : 30 44 02 20 66 12 38 A2 ...
       SCT Version  1
      Log Operator  DigiCert
        Log Key ID  51 A3 B0 F5 FD 01 79 9C 56 6D B8 37 78 8F 0C A4 7A CC 1B 27 CB F7
                    9E 88 42 9A 0D FE D4 8B 05 E5
         Timestamp  Wednesday, May 6, 2020 at 2:11:06 PM Eastern Daylight Time
Signature Algorithm  SHA-256 ECDSA
         Signature  71 bytes : 30 45 02 20 14 3F E8 49 ...

         Extension  Certificate Authority Information Access ( 1.3.6.1.5.5.7.1.1 )
          Critical  NO
         Method #1  Online Certificate Status Protocol ( 1.3.6.1.5.5.7.48.1 )
               URI  http://ocsp.digicert.com
         Method #2  CA Issuers ( 1.3.6.1.5.5.7.48.2 )
               URI  http://cacerts.digicert.com/DigiCertSHA2HighAssuranceServerCA.crt
```

New additions to the PKI

Key Usage
Certificate signing, authentication

Subject Alternate Names
Support deployments in CDNs

Revocation Information
New ways to deliver revocations

Certificate Transparency
Allows greater insight into CA (mis)behavior

The PKI must continue to evolve
but adding new features is *slow* and *laborious*

Traditional PKI roles

Subject Name
Who the cert is about

Issuer Name
Who vetted the subject's identity

Expiration Dates
When is the certificate no longer valid

Public key and signature
Attestation of cryptographic identity

New additions to the PKI

Key Usage
Certificate signing, authentication

Subject Alternate Names
Support deployments in CDNs

Revocation Information
New ways to deliver revocations

Certificate Transparency
Allows greater insight into CA (mis)behavior

Future additions

Naming constraints
Let non-CAs issue their own certs, limited to domains they control

Signed exchanges
Sign-over the hosting of some resources to a third party

Multi-rooted certificates
Minimize the reliance on a small set of trusted certificate authorities

And many more!

Is there *one extension* we could add that would make the PKI:

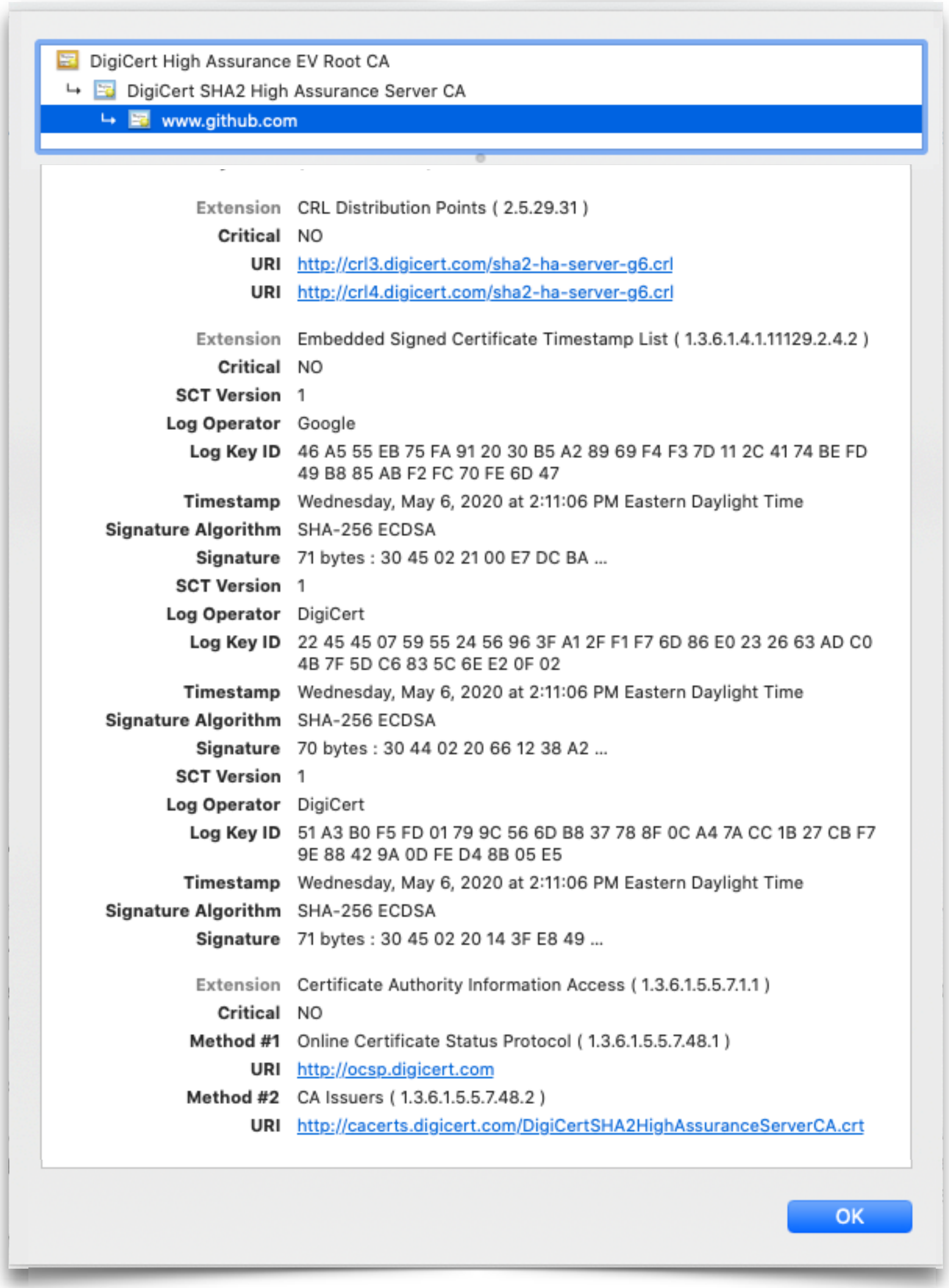
- More evolvable
- More customizable to new deployments
- Easier to formally verify

Insight: A certificate is a set of constraints

Name *Validity period* *Allowed usages*

Why not encode constraints in small programs in the certificate?

Assertion-Carrying Certificates (ACCs)

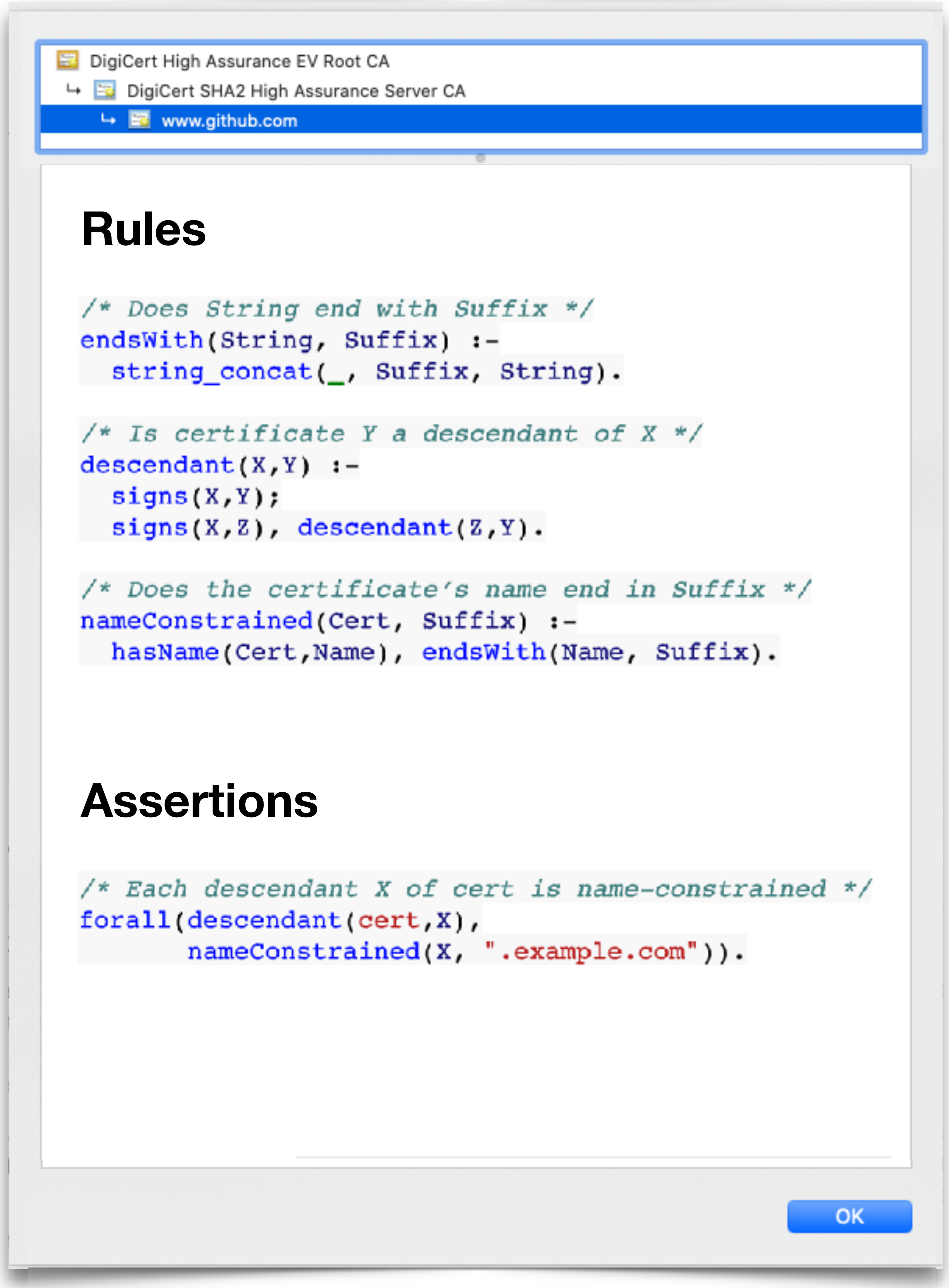


DigiCert High Assurance EV Root CA
↳ DigiCert SHA2 High Assurance Server CA
↳ www.github.com

Extension	CRL Distribution Points (2.5.29.31)
Critical	NO
URI	http://crl3.digicert.com/sha2-ha-server-g6.crl
URI	http://crl4.digicert.com/sha2-ha-server-g6.crl
Extension	Embedded Signed Certificate Timestamp List (1.3.6.1.4.1.11129.2.4.2)
Critical	NO
SCT Version	1
Log Operator	Google
Log Key ID	46 A5 55 EB 75 FA 91 20 30 B5 A2 89 69 F4 F3 7D 11 2C 41 74 BE FD 49 B8 85 AB F2 FC 70 FE 6D 47
Timestamp	Wednesday, May 6, 2020 at 2:11:06 PM Eastern Daylight Time
Signature Algorithm	SHA-256 ECDSA
Signature	71 bytes : 30 45 02 21 00 E7 DC BA ...
SCT Version	1
Log Operator	DigiCert
Log Key ID	22 45 45 07 59 55 24 56 96 3F A1 2F F1 F7 6D 86 E0 23 26 63 AD C0 4B 7F 5D C6 83 5C 6E E2 0F 02
Timestamp	Wednesday, May 6, 2020 at 2:11:06 PM Eastern Daylight Time
Signature Algorithm	SHA-256 ECDSA
Signature	70 bytes : 30 44 02 20 66 12 38 A2 ...
SCT Version	1
Log Operator	DigiCert
Log Key ID	51 A3 B0 F5 FD 01 79 9C 56 6D B8 37 78 8F 0C A4 7A CC 1B 27 CB F7 9E 88 42 9A 0D FE D4 8B 05 E5
Timestamp	Wednesday, May 6, 2020 at 2:11:06 PM Eastern Daylight Time
Signature Algorithm	SHA-256 ECDSA
Signature	71 bytes : 30 45 02 20 14 3F E8 49 ...
Extension	Certificate Authority Information Access (1.3.6.1.5.5.7.1.1)
Critical	NO
Method #1	Online Certificate Status Protocol (1.3.6.1.5.5.7.48.1)
URI	http://ocsp.digicert.com
Method #2	CA Issuers (1.3.6.1.5.5.7.48.2)
URI	http://cacerts.digicert.com/DigiCertSHA2HighAssuranceServerCA.crt

Assertion-Carrying Certificates (ACCs)

Add small programs that must be run as part of the certificate's validation



DigiCert High Assurance EV Root CA
↳ DigiCert SHA2 High Assurance Server CA
↳ www.github.com
OK

Rules

```
/* Does String end with Suffix */
endsWith(String, Suffix) :-
  string_concat(_, Suffix, String).

/* Is certificate Y a descendant of X */
descendant(X,Y) :-
  signs(X,Y);
  signs(X,Z), descendant(Z,Y).

/* Does the certificate's name end in Suffix */
nameConstrained(Cert, Suffix) :-
  hasName(Cert,Name), endsWith(Name, Suffix).
```

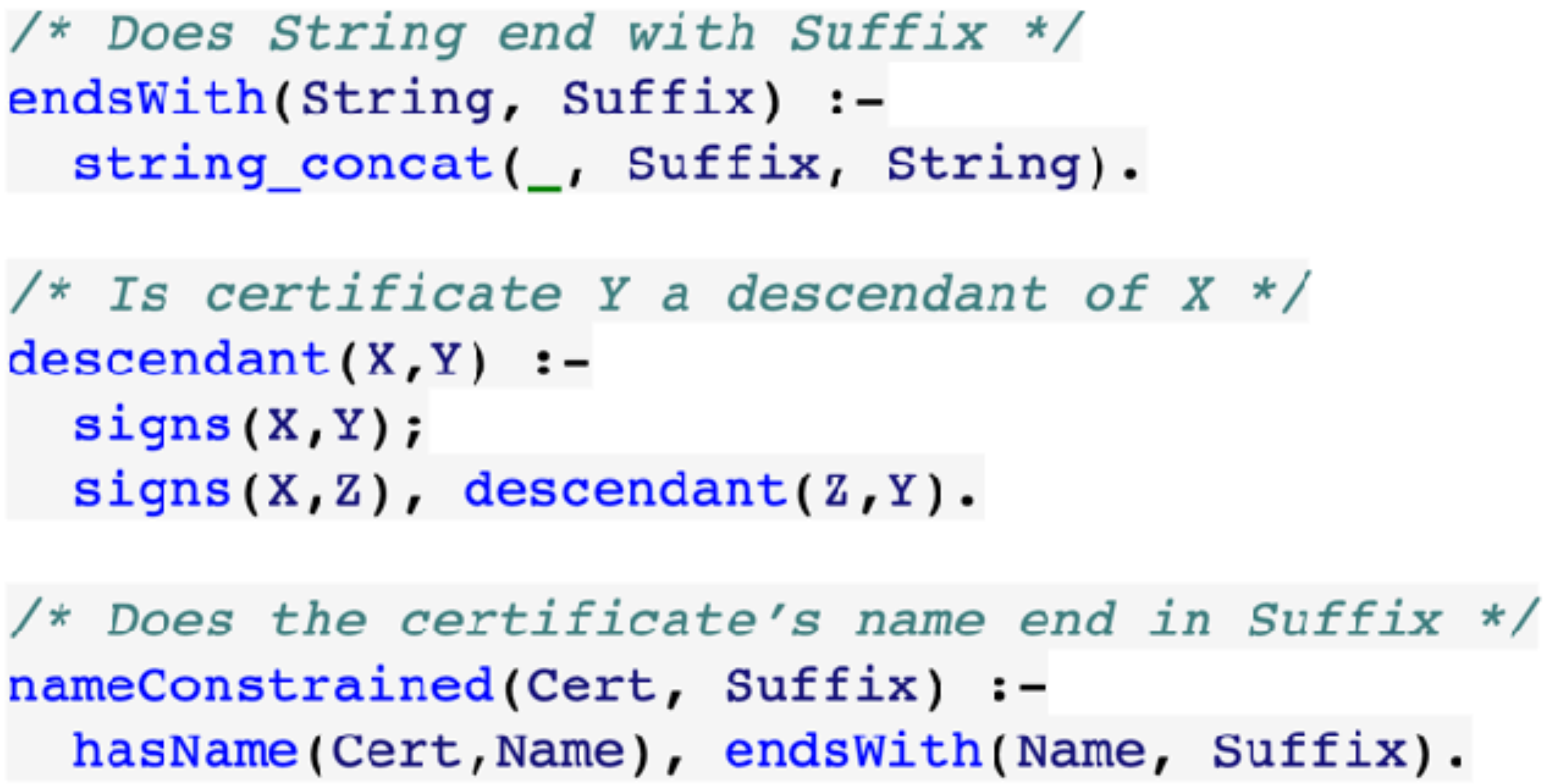
Assertions

```
/* Each descendant X of cert is name-constrained */
forall(descendant(cert,X),
       nameConstrained(X, ".example.com")).
```

Assertion-Carrying Certificates (ACCs)

Add small programs that must be run as part of the certificate's validation

Rules

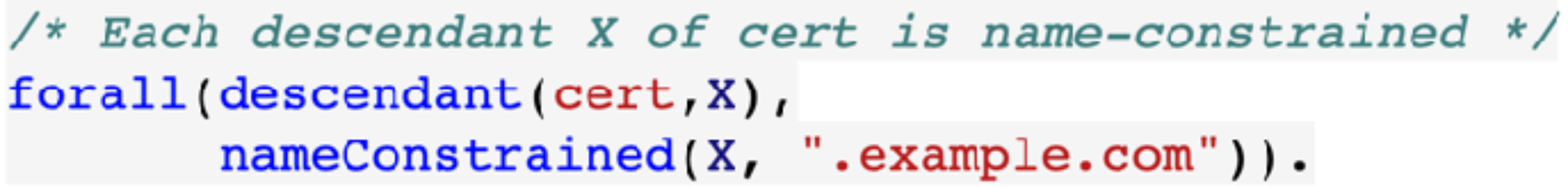


```
/* Does String end with Suffix */
endsWith(String, Suffix) :-
  string_concat(_, Suffix, String).

/* Is certificate Y a descendant of X */
descendant(X,Y) :-
  signs(X,Y);
  signs(X,Z), descendant(Z,Y).

/* Does the certificate's name end in Suffix */
nameConstrained(Cert, Suffix) :-
  hasName(Cert,Name), endsWith(Name, Suffix).
```

Define new capabilities

What it means to be name-constrained

Assertions



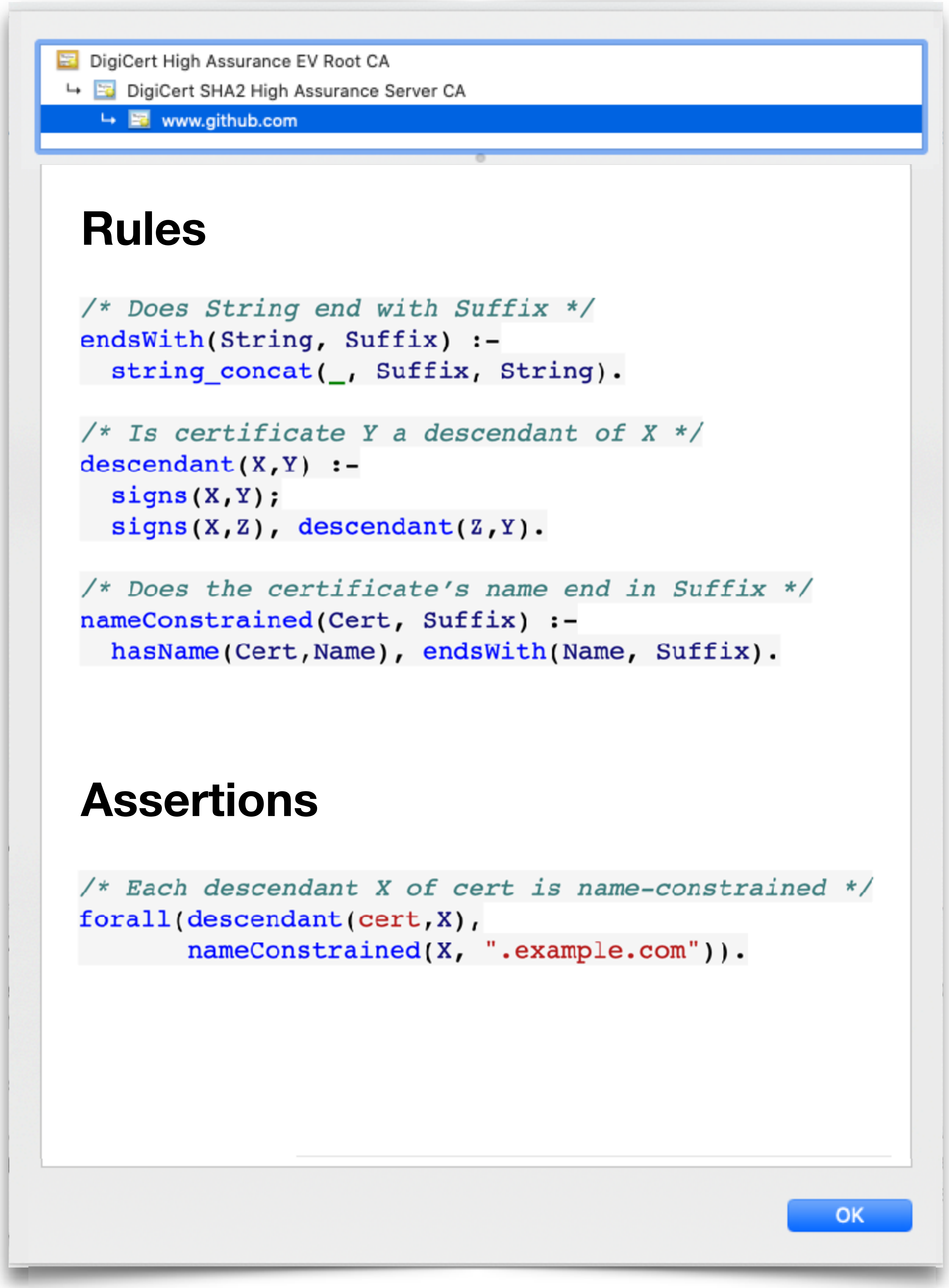
```
/* Each descendant X of cert is name-constrained */
forall(descendant(cert,X),
       nameConstrained(X, ".example.com")).
```

Enforce them as constraints

All certificates following this one must be name-constrained

Assertion-Carrying Certificates (ACCs)

Language goals


DigiCert High Assurance EV Root CA
DigiCert SHA2 High Assurance Server CA
www.github.com
OK

Rules

```
/* Does String end with Suffix */
endsWith(String, Suffix) :-
  string_concat(_, Suffix, String).

/* Is certificate Y a descendant of X */
descendant(X,Y) :-
  signs(X,Y);
  signs(X,Z), descendant(Z,Y).

/* Does the certificate’s name end in Suffix */
nameConstrained(Cert, Suffix) :-
  hasName(Cert,Name), endsWith(Name, Suffix).
```

Assertions

```
/* Each descendant X of cert is name-constrained */
forall(descendant(cert,X),
       nameConstrained(X, ".example.com")).
```

All constraints across *all* certs in the chain must hold

Certs can never relax constraints further up the chain

Browsers can add their own constraints, as well

The language should be concise and expressive

Does not need to be Turing-complete

Should be formally verifiable

Must not broaden the attack surface

A logic-based programming language is a natural fit

Assertion-Carrying Certificates (ACCs)

What is the appropriate constraint language?

Prolog		Datalog
X	Non-Turing-complete	✅
X	Declarative	✅
X	Termination guaranteed	✅
½	Amenable to static analysis	✅
✅	Fully expressive	½
✅	Negation	½
✅	Unbounded lists, numbers, strings	X

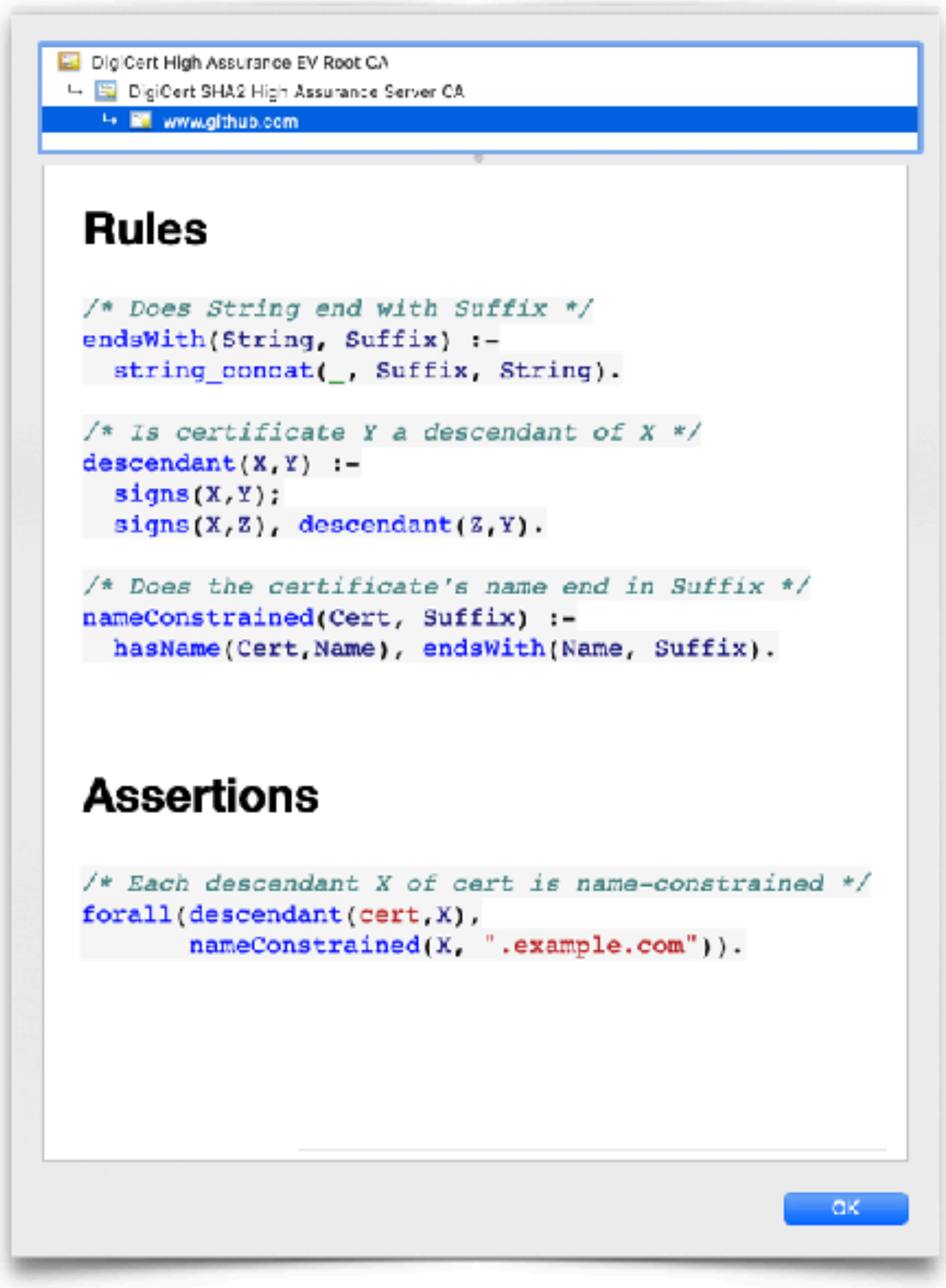
We might not need these

Assertion-Carrying Certificates (ACCs)

Allow for a far more agile PKI

Today's PKI is slow to evolve

ACCs add small programs that must be run as part of the certificate's validation


DigiCert High Assurance EV Root CA
DigiCert SHA2 High Assurance Server CA
www.github.com
OK

Rules

```
/* Does String end with Suffix */
endsWith(String, Suffix) :-
  string_concat(_, Suffix, String).

/* Is certificate Y a descendant of X */
descendant(X,Y) :-
  signs(X,Y);
  signs(X,Z), descendant(Z,Y).

/* Does the certificate's name end in Suffix */
nameConstrained(Cert, Suffix) :-
  hasName(Cert,Name), endsWith(Name, Suffix).
```

Assertions

```
/* Each descendant X of cert is name-constrained */
forall(descendant(cert,X),
       nameConstrained(X, ".example.com")).
```

Ongoing and Future Efforts

Implementing long-desired features

Naming constraints, signed exchanges, and more

Re-implementing various browsers' validation logic in Prolog/Datalog

Chrome, Firefox, mbedTLS — in far fewer lines of code

Exploring ways to verify correctness:

- Static analysis
- Certificate fuzzing
- Using the languages' imputation

Is there any certificate that is valid but where constraint X does not hold?