Q1

Diagram the certificate chain your browser uses to authenticate https://web.cs.dartmouth.edu/Links to an external site.

DigiCert Global Root CA

- Who is the issuer?
  - The issue is itself, it is the root.
  - Organization: DigiCert
- What's the validity period?
  - Thursday, November 9, 2006 at 4:00:00 PM Pacific Standard Time - Sunday, November 9, 2031 at 4:00:00 PM Pacific Standard Time
- What algorithm was used to sign it?
  - SHA-1 with RSA Encryption (1.2.840.113549.1.1.5)

DigiCert TLS RSA SHA256 2020 CA1

- Who is the issuer
  - DigiCert
- What's the validity period?
  - Tuesday, April 13, 2021 at 5:00:00 PM Pacific Daylight Time - Sunday, April 13, 2031 at 4:59:59 PM Pacific Daylight Time
- What algorithm was used to sign it?
  - SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)

*.dartmouth.edu

- Who is the issuer
  - DigiCert
- What's the validity period?
  - Monday, February 14, 2022 at 4:00:00 PM Pacific Standard Time - Saturday, March 18, 2023 at 4:59:59 PM Pacific Daylight Time
- Algorithm
  - SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)

Q2

Go on a treasure hunt! In the wild, find two strange/interesting examples of server certificate trust paths. Any self-signed "Snake Oil" certs? Anything revoked or expired? Any crazy-long path? Any bad crypto?

Note:

sites that are intentionally configured this way (like badssl) in order to test browsers don't count yes... in class, we did visit an expired site...

I found all of these using www.ssllabs.com/ssltest

- https://affashionhouse.com/

  - This has a self-signed certificate
- http://www.techats.com/
  - Doesn't appear to have a certificate (or just isn't trusted by Chrome)

Q3 10.5

For each of the following, give an example (with justification) of an application of cryptography where it might make sense to deploy.

1. A flat key hierarchy with just one level.
    1. A simple chatroom with not too many people active at any given time. Having an HSM, or another tier might not be worth the investment for a developer. Especially if the developer wants the chatroom network to be ad-hoc.
2. A two-level key hierarchy
    1. An example of a good two-level hierarchy might be a larger network chat-system such as whatsapp that also wants strict peer to peer communication, but doesn't want to generate keys on each device for performance issues. There might be a KC (key center) that generates and distributes keys, or handles message translation (I believe whatsapp probably goes with the former.)
    2. Would be interesting to read more into their actual encryption process here
3. A three-level key hierarchy
    1. Potentially in a military setting, when keys are incredible sensitive, and an HSM can hold master keys in a very secret setting. I.e. where the system has the means to hold a specialized machine in a controlled environment, and then wants to distribute keys from there. Even if the system is large, the size if justified by the need for security, backups, and protection of keys.

Q4 10.9

Key backup is an important part of the cryptographic key lifecycle

1. Why is it important to backup cryptographic keys?

It is important to backup keys for several reasons. First, because keys can get lost, or destroyed by adversaries. If this happens, and there is no backup of the key, then all data encrypted under that key can longer be decrypted, and has been lost. Another reason might be to audit current keys. By having a backup of the key in another system, it's possible for an organization to run internal audits of their own cryptographic keys/ciphertexts to determine whether their system is secure or not, without impacting the system that actually uses those keys.

1. In what ways might backup of cryptographic keys differ from backup of more general data on a computer system?

While cryptographic keys are just special data, their backup needs to be kept private to the same degree that their in-use copies are. Otherwise the privacy of both instances of the key is at stake. So therefore, security measures that are taken to keep the original key safe must also be taken to keep the backup safe. Backups therefore should ideally be kept in controlled environments, and on specialized machines. Other backups might simply be stored on a hard-drive, or in cold-storage on a cloud server, encrypted if it's important data, or encrypted it isn't. Be careful though, when we encrypt the backup, we need the key we are encrypting it also to be backed up somehow...

1. As a system admin of a small org deploying symmetric crypto for protection of all traffic on the local intranet, suggest what techniques and procedures you will use for the backup (and subsequent management of backed-up) crypto keys.

If I were a system admin of a small org deploying symmetric crypto for protection of all traffic on the local intranet, for each key that I deployed I would encrypt that key under a key that I can generate with knowledge of who the key is intended for, and my master key. That way, if the person the key is intended for loses their key, they can request the backup, but someone else who doesn't know who the key is intended for (even if they find my master key) doesn't necessarily know how to decrypt that key. This way the backups are secured in my control, but only at the request(and obviously identity confirmation) of the intended user.

Q5 10.11

Give an example of a real cryptographic application which.

1. "enforces" the principle of key separation (explain why)

   Session Key Types • Root Key – A 32-byte value that is used to create Chain Keys . • Chain Key – A 32-byte value that is used to create Message Keys . • Message Key – An 80-byte value that is used to encrypt message contents . 32 bytes are used for an AES-256 key, 32 bytes for a HMACSHA256 key, and 16 bytes for an IV.

   WhatsApp White Paper

The key separation is enforced here by the 80-byte value who's components are used as keys for different purposes. It enforces this separation by identifying the keys are distinct, and specifying (likely in the programming state machine) that they are to be used only for heir intended purposes.

1. "abuses" the principle of key separation (justify why, if possible)

> Note at first I thought this mean "abuses" the presense of key separation...

More keys might make it easier to find one, if the probability of finding a key is $1/p$, and you have n keys, the probability of finding at least one key is $\frac{n}{k}$. Maybe...

Similar to the example with the HSM, if an attacker could compromise the labeling method that labels keys for their individual uses, then the attacker, without knowing the key, could get it to be used for something other than it was intended for. This might provide the attacker with more knowledge of the key, or use a strong key for algorithm X on algorithm Y for which it is very weak.

> Then I read piazza and it said it's actually "abuses" the lack of key separation...

If keys are not separated, then more information regarding a key could be leaked through its repeated use in different cryptographic primitives. For example, say we use an RSA private key (which we know is a prime), also as a DSA key. While knowing K is prime doesn't help us in RSA, it might cut down the search space, or allow for more precise cryptanalysis methods on DSA.

Q6 11.8

Visit the website of a well known commercial CA to establish

Looking at DigitCert

1. what levels of public-key certificate they issue
    1. Basic SSL
    2. Secure Site SSL
    3. Secure Site Pro
2. What credentials they require for registration of these different levels of public-key cert.
    1. They ask for a CSR (Certificate Signing Request)
    2. common name (e.g., www.example.com),
    3. organization name and location (country, state/province, city/town),
    4. key type (typically RSA),
    5. and key size (2048-bit minimum).
    6. (This is the same stuff that openssl asks for when generating & signing keys)
    7. website URL
        1. wildcard URLs
    8. "Server app details for the host you install the certificate on"
3. what liability they accept for these different levels of public-key cert.
    1. $1.25M, $1.75M, $2M of warranty on the website (they don't really say what this warranty is for though...) I imagine it's damages if there are security breaches.

4. how often they publish cert revocation lists
    1. daily
5. how clients access CRLs
    1. Received by a client over the internet
6. what advice they provide clients concerning the importance of checking CRLS
    1. They suggest you recache it daily
    2. "Traditionally, the issuing CA signs a CRL file daily, which is retrieved by the client over the internet. But in situations where the device can't easily or regularly connect to the internet, the CRL can be stored and cached."

Q7 11.11

provide an example of a real security incident arising due to a failure in some phase of the public-key certificate lifecycle

https://www.mail-archive.com/dev-security-policy@lists.mozilla.org/msg04654.html

1. explain what went wrong
    - There was an issue in the identity registration process that allowed emails (which they used for verification) to be mangled in some way such that the email used to then verify an identity was not the same as intended by the applicant.
2. explain why this was allowed to happen
    - This was allowed to happen because the registration process was automated, and used Optical Character Recognition technology I guess to transcribe written text into characters. This process of applicant registration is an incredible sensitive component of a CAs job, and usually cannot be automated (maybe for this purpose.)
3. explain how such an incident could have been prevented
    - Various ways, instead of using Optical Character Recognition, typing into a textbook (which also might cause error but can be reverted). Or having an individual do the transcription instead. Making the automation better, or removing it altogether would fix this. But honestly, this seems like a simple bug, not a big fundamental problem with the system.

Q8 11.15

alice wishes to securely send an important message to bob, who has a public encryption key he generated himself. neither alice or bob are part of any formal public-key certificate management system, and bob does not intend to acquire a public-key certificate for his public key. Suggest how alice might obtain bob's public key and acquire sufficient assurance of purpose of this key if:

1. alice and bob are friends

Bob could hand Alice his public key when they meet next. Since they know and trust each other Bob would tell Alice the intended purpose of the public key, and then Alice could, until the key is retired, send bob secure messages over a public channel.

2. alice and bob are remote business partners

Since Alice and Bob are business partners, they likely trust the same businesses. Bob could utilize this business connection to share the public key. The business could act almost as a certificate management system. If we trust messages that are sent securely through the business channels, then Bob could easily share his public key through that business channel (secure business email), the assumption is that the business channel is secure to the business, but not secure to the individuals. By sharing the public key through this channel it is almost certainly coming from Bob (since we trust the business and it's actors), then Alice and Bob can use the public key on another channel to communicate.

1. alice and bob are strangers

Sounds like craigslist...

This is really a CA-free certification model (since in the other cases they at least have some turst already between the two parties). In this case though there is no initial trust. In this case, Bob can self-sign the certificate, and maybe send along some other information that might help alice make an informed decision to trust bob (linkedIn link, resume, note from a co-worker, etc.) Then alice can make an informed decision to trust this public key (and it's purpose). To check the public key (and it's purpose) before sending anything too secure. Alice could test Bob by sending a nonce to him encrypted under his public key. If he properly decrypts it, and shares in response the nonce, then he has access to the private key for the key he gave to Alice. Then they can continue.