

Introduction to Cryptography

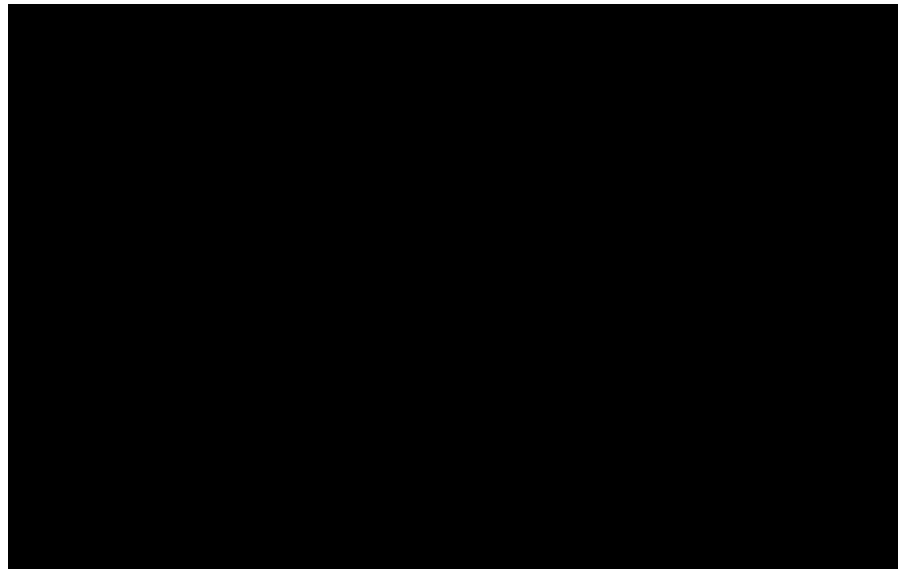
Lecture 9

Public Key Infrastructure (PKI),
hash chains, hash trees

Benny Pinkas

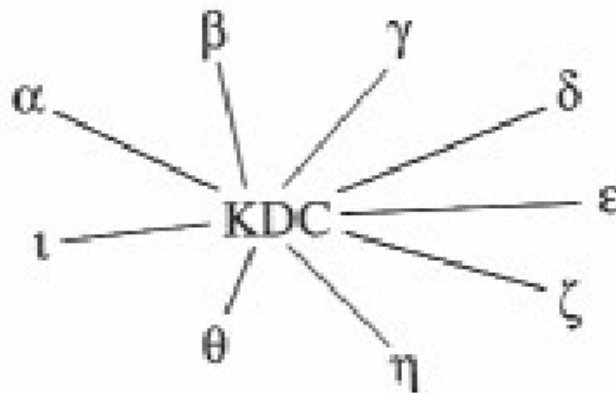
Key Infrastructure for symmetric key encryption

- Each user has a shared key with each other user
 - A total of $n(n-1)/2$ keys
 - Each user stores $n-1$ keys



Key Distribution Center (KDC)

- The KDC shares a symmetric key K_u with every user u
- Using this key they can establish a trusted channel
- When u wants to communicate with v
 - u sends a request to the KDC
 - The KDC
 - *authenticates* u
 - *generates* a key K_{uv} to be used by u and v
 - *sends* $Enc(K_u, K_{uv})$ to u , and $Enc(K_v, K_{uv})$ to v



Key Distribution Center (KDC)

- Advantages:
 - A total of n keys, one key per user.
 - easier management of joining and leaving users.
- Disadvantages:
 - The KDC can impersonate anyone
 - The KDC is a single point for failure, for both
 - security,
 - and quality of service
- Multiple copies of the KDC
 - More security risks
 - But better availability

Certification Authorities (CA)

- Public key technology requires every user to remember its private key, and to have access to other users' public key
- How can the user verify that a public key PK_v corresponds to user v ?
 - What can go wrong otherwise?
- A simple solution:
 - A trusted public repository of public keys and corresponding identities
 - Doesn't scale up
 - Requires online access per usage of a new public key

Certification Authorities (CA)

- The Certificate Authority (CA) is trusted party.
- All users have a copy of the public key of the CA
- The CA signs Alice's digital certificate. A simplified certificate is of the form *(Alice, Alice's public key)*.

- When we get Alice's certificate, we
 - Examine the identity in the certificate
 - Verify the signature
 - Use the public key given in the certificate to
 - Encrypt messages to Alice
 - Or, verify signatures of Alice
- The certificate can be sent by Alice without any interaction with the CA.

Certification Authorities (CA)

- Unlike KDCs, the CA does not have to be online to provide keys to users
 - It can therefore be better secured than a KDC
 - The CA does not have to be available all the time
- Users only keep a single public key – of the CA
- The certificates are not secret. They can be stored in a public place.
- When a user wants to communicate with Alice, it can get her certificate from either her, the CA, or a public repository.
- A compromised CA
 - can mount active attacks (certifying keys as being Alice's)
 - but it cannot decrypt conversations.

Certification Authorities (CA)

- For example.
 - To connect to a secure web site using SSL or TLS, we send an `https://` command
 - The web site sends back a public key⁽¹⁾, and a certificate.
 - Our browser
 - Checks that the certificate belongs to the url we're visiting
 - Checks the expiration date
 - Checks that the certificate is signed by a CA whose public key is known to the browser
 - Checks the signature
 - If everything is fine, it chooses a session key and sends it to the server encrypted with RSA using the server's public key

⁽¹⁾ This is a very simplified version of the actual protocol.

Certificate Viewer: "www.bankpoalim.co.il"

General Details

This certificate has been verified for the following uses:
SSL Server Certificate

Issued To

Common Name (CN) www.bankpoalim.co.il
 Organization (O) Bank Hapoalim Ltd.
 Organizational Unit (OU) Internet departement
 Serial Number 6C:F8:30:09:B9:46:C5:FA:11:8A:40:CD:14:6A:EB:A3

Issued By

Common Name (CN) <Not Part Of Certificate>
 Organization (O) VeriSign Trust Network
 Organizational Unit (OU) VeriSign, Inc.

Validity

Issued On 7/12/2004
 Expires On 7/13/2005

Fingerprints

SHA1 Fingerprint 11:E2:F6:A4:E3:05:F9:96:7F:E6:09:40:17:47:A9:20:1F:C8:96:9F
 MD5 Fingerprint 6C:E9:C5:CD:40:E1:28:3A:9F:49:5D:D8:5A:F4:94:EB

Help Close

gon&dt=924&nls=HE

Gizmodo Educated Guesswork The New York Times ... The Register: Sci/T

בנק הפועלים

ברוכים הבאים

לצורך כניסה לשירות יש להקל
"כניסה לחשבונך".

קוד משתמש : ?
 ת.ז. : ?
 סיסמא : ?

כניסה

זה מאובטח בשיטות

לחצו כאן לפרטים נוספים.

כל הזכויות שמורות לבנק

מפקידים לקופ"ג
ונהנים ממענק מיוחד

הפקידו עכשיו לקופ"ג

דרך פועלים באינטרנט

ותיהנו מהחזר דמי ניהול

בשיעור של 0.25%

מסכום ההפקדה.

לפרטים נוספים

www.bankpoal

Welcome to Gmail - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://gmail.google.com/?dest=http%3A%2F%2Fgmail.google.com%2Fgmail

Gmail - Inbox (5) Latest Headlines Furl It CNET News.com -- T... Slashdot: News for n... Gizmodo Educated Guesswork The New York Times ... The Register: Sci/

Certificate Viewer: "gmail.google.com"

General Details

This certificate has been verified for the following uses:
SSL Server Certificate

Issued To
Common Name (CN) **gmail.google.com**
Organization (O) **Google Inc.**
Organizational Unit (OU) <Not Part Of Certificate>
Serial Number 04:E1:7F

Issued By
Common Name (CN) <Not Part Of Certificate>
Organization (O) Equifax
Organizational Unit (OU) Equifax Secure Certificate Authority

Validity
Issued On 3/30/2004
Expires On 3/31/2006

Fingerprints
SHA1 Fingerprint D0:D5:54:CD:CE:59:5E:6C:32:63:0F:91:C1:CC:E2:B0:23:C0:F8:70
MD5 Fingerprint D4:A1:6F:0D:E2:0E:8A:1F:F4:A2:00:56:54:84:C0:56

Help Close

delete mail and you
received.

Gmail Sign In

Username:

Password:

Don't ask for my password for 2 weeks.

[Forgot your password?](#)

Learn more [about Gmail](#).

Check out our [new features!](#)

A few words about [privacy and Gmail](#).

es - [Terms of Use](#)

Transferring data from gmail.google.com... gmail.goo

Certificates Other People's Web Sites Authorities

You have certificates on file that identify these certificate authorities:

Certificate Name	Security Device
Unizeto Sp. z o.o.	
--Certum CA	Builtin Object Token
VISA	
--GP Root 2	Builtin Object Token
--Visa eCommerce Root	Builtin Object Token
ValiCert, Inc.	
--http://www.valicert.com/	Builtin Object Token
--http://www.valicert.com/	Builtin Object Token
--http://www.valicert.com/	Builtin Object Token
VeriSign, Inc.	
--Verisign Class 3 Public Primary Certification Authority	Builtin Object Token
--Verisign Class 1 Public Primary Certification Authority	Builtin Object Token
--Verisign Class 2 Public Primary Certification Authority	Builtin Object Token
--Verisign Class 1 Public Primary Certification Authority - G2	Builtin Object Token
--Verisign Class 2 Public Primary Certification Authority - G2	Builtin Object Token
--Verisign Class 3 Public Primary Certification Authority - G2	Builtin Object Token
--Verisign Class 4 Public Primary Certification Authority - G2	Builtin Object Token
--VeriSign Class 1 Public Primary Certification Authority - G3	Builtin Object Token
--VeriSign Class 2 Public Primary Certification Authority - G3	Builtin Object Token
--VeriSign Class 3 Public Primary Certification Authority - G3	Builtin Object Token
--VeriSign Class 4 Public Primary Certification Authority - G3	Builtin Object Token
--Class 1 Public Primary OCSP Responder	Builtin Object Token
--Class 2 Public Primary OCSP Responder	Builtin Object Token
--Class 3 Public Primary OCSP Responder	Builtin Object Token
--VeriSign Time Stamping Authority CA	Builtin Object Token
beTRUSTed	
--beTRUSTed Root CAs	Builtin Object Token
--beTRUSTed Root CA-Baltimore Implementation	Builtin Object Token
--beTRUSTed Root CA - Entrust Implementation	Builtin Object Token
--beTRUSTed Root CA - RSA Implementation	Builtin Object Token

View Edit Import Delete

OK Help

Certificates

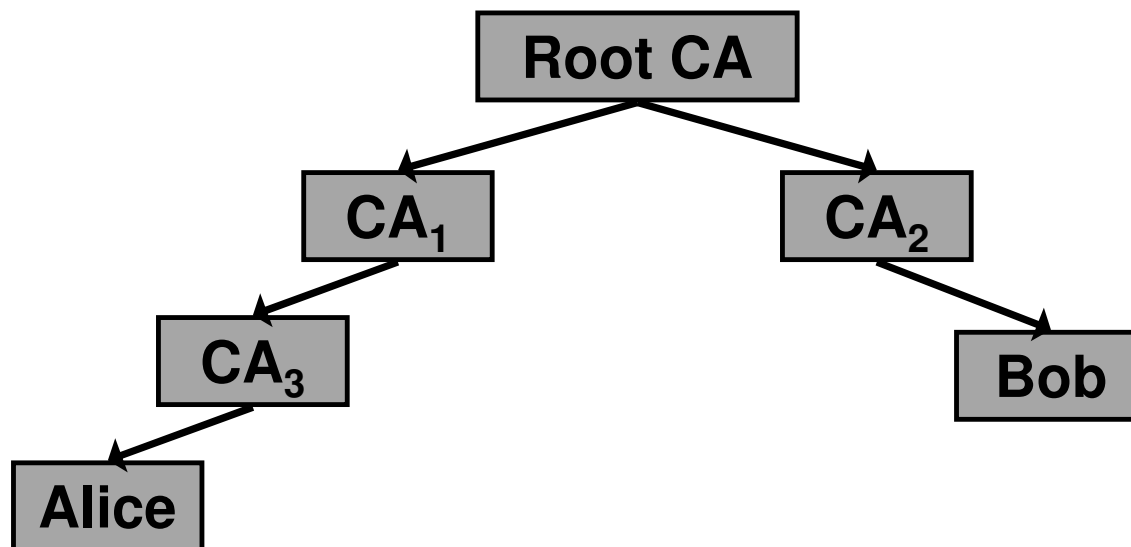
- A certificate usually contains the following information
 - Owner's name
 - Owner's public key
 - Encryption/signature algorithm
 - Name of the CA
 - Serial number of the certificate
 - Expiry date of the certificate
 - ...
- Your web browser contains the public keys of some CAs
- A web site identifies itself by presenting a certificate which is signed by a chain starting at one of these CAs

Public Key Infrastructure (PKI)

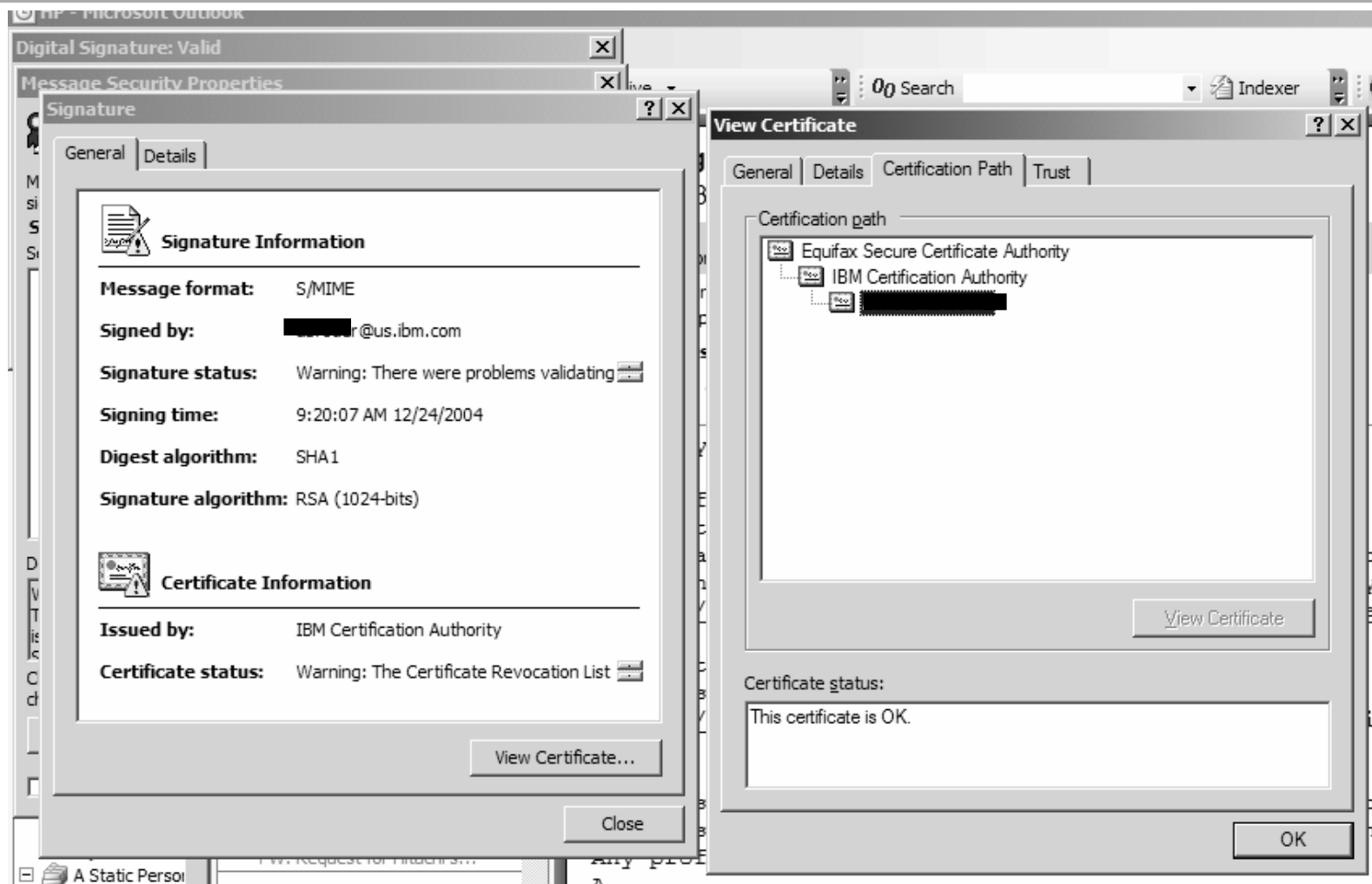
- The goal: build trust on a global level
- Running a CA:
 - If people trust you to vouch for other parties, everyone needs you.
 - A license to print money
 - But,
 - The CA should limit its responsibilities, buy insurance...
 - It should maintain a high level of security
 - Bootstrapping: how would everyone get the CA's public key?

Public Key Infrastructure (PKI)

- Monopoly: a single CA vouches for all public keys
- Monopoly + delegated CAs:
 - top level CA can issue certificates for other CAs
 - Certificates of the form
 - $[(\text{Alice}, \text{PK}_A)_{\text{CA}_3}, (\text{CA}_3, \text{PK}_{\text{CA}_3})_{\text{CA}_1}, (\text{CA}_1, \text{PK}_{\text{CA}_1})_{\text{TOP-CA}}]$



Certificate chain



Public Key Infrastructure

- Oligarchy
 - Multiple trust anchors (top level CAs)
 - Pre-configured in software
 - User can add/remove CAs
- Top-down with name constraints
 - Like monopoly + delegated CAs
 - But every delegated CA has a predefined portion of the name space (il, ac.il, haifa.ac.il, cs.haifa.ac.il)
 - More trustworthy

Revocation

- Revocation is a key component of PKI
 - Each certificate has an expiry date
 - But certificates might get stolen, employees might leave companies, etc.
 - Certificates might therefore need to be revoked before their expiry date
 - Before using a certificate we must verify that it has not been revoked

Certificate Revocation Lists (CRLs)

- A revocation agency (RA) issues a list of revoked certificates (i.e., “bad” certificates)
 - The list is updated and published regularly (e.g. daily)
 - Users must consult the most recent CRL in addition to checking the expiry date
- Advantages: simple.
- Drawbacks:
 - Scalability. CRLs can be huge. There is no short proof that a certificate is valid. (The proof is global, rather than individual.)
 - There is a vulnerability windows between a compromise of certificate and the next publication of a CRL.
 - Need a reliable way of distributing CRLs.
- Improving scalability using “delta CRLs”

Explicit revocation: OCSP

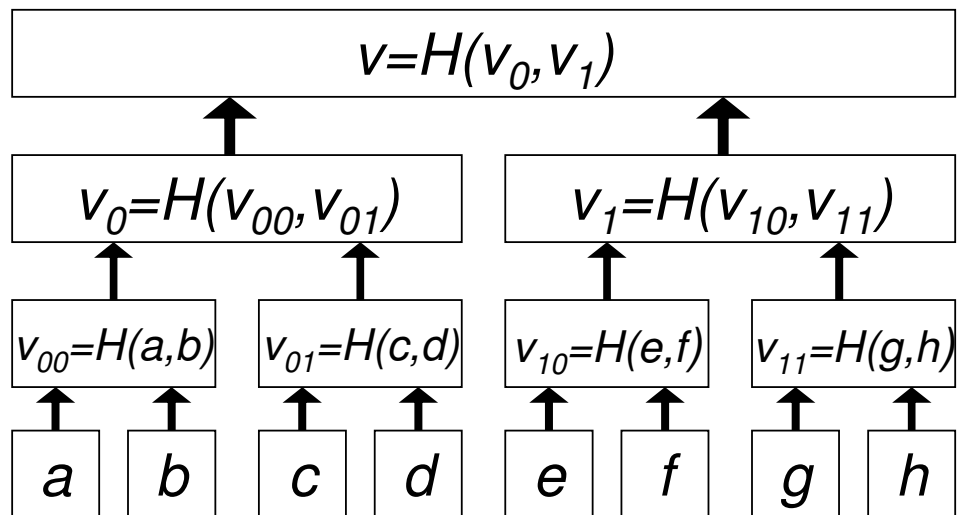
- OCSP (Online Certificate Status Protocol)
 - RFC 2560, June 1999.
- OCSP can be used in place, or in addition, to CRLs
- OCSP sends a request for certificate status information.
 - The server sends back a response of "current", "expired," or "unknown".
 - The response is signed (by the CA, or a Trusted Responder, or An Authorized Responder certified by the CA).
- Provides instantaneous status of certificates
 - Overcomes the chief limitation of CRL: the fact that updates must be frequently downloaded to keep the list current

Certificate Revocation System (CRS)

- Certificate Revocation System (Micali'96)
- Uses a hash chain
 - The certificate includes $Y_{365} = f^{365}(Y_0)$, and $N = f(N_0)$. f is one-way.
 - On day d ,
 - If the certificate is valid, then $Y_{365-d} = f^{365-d}(Y_0)$ is sent by the CA to the certificate holder or to a directory.
 - If the certificate is invalid, the value N_0 is published.
 - The certificate receiver uses the daily value ($f^{365-d}(Y_0)$) to verify that the certificate is still valid. (how?)
- Disadvantage:
 - Daily overhead, even when a cert is not revoked.

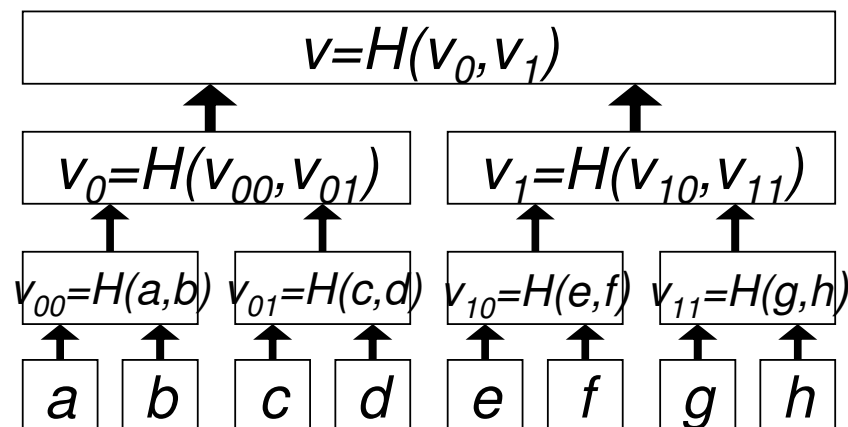
Merkle Hash Tree

- A method of hashing together n values, x_1, \dots, x_n , such that
 - The result is a single hash value
 - For any x_i , it is possible to prove that it appeared in the original list, using a proof of length $O(\log n)$.



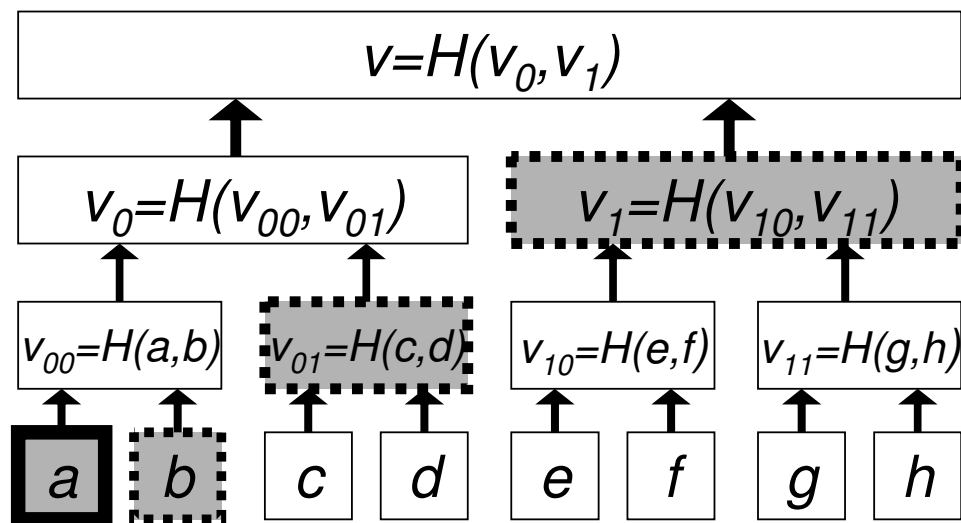
Merkle Hash Tree

- H is a collision intractable hash function
- Any change to a leaf results in a change to the root
- To sign the set of values it is sufficient to sign the root (a single signature instead of n).
- How do we verify that an element appeared in the signed set?



Verifying that a appears in the signed set

- Provide a 's leaf, and the siblings of the nodes in the path from a to the root. ($O(\log n)$ values)
- The verifier can use H to compute the values of the nodes in the path from the leaf to the root.
- It then compares the computed root to the signed value



Improving the overhead of CRS

- Originally (for a year long certificate)
 - the certificate includes $f^{365}(Y_0)$
 - On day d , certificate holder obtains $f^{365-d}(Y_0)$
 - The certificate receiver computes $f^{365}(Y_0)$ from $f^{365-d}(Y_0)$ by invoking $f()$ d times.
- Slight improvement:
 - The CA assigns a different leaf for every day, constructs a hash tree, and signs the root.
 - On day d , it releases node d and the siblings of the path from it to the root.
 - This is the proof that the certificate is valid on day d
 - The overhead of verification is $O(\log 365)$.

Certificate Revocation Tree (CRT) [Kocher]

- A CRT is a hash tree with leaves corresponding to statements about ranges of certificates
 - Statements describe regions of certificate ids, in which only the smallest id is revoked.
 - For example: “if $100 \leq \text{id} < 234$, then cert is revoked iff $\text{id}=100$ ”.
 - Each certificate matches exactly one statement.
 - The statements are the leaves of a signed hash tree.
 - To examine the state of a certificate we retrieve the statement for the corresponding region.
 - A single hash tree is used for all certs.
 - Advantage: a short proof for the status of a certificate.
 - Drawback: the entire hash tree must be updated daily.