Applied Cryptography and Network Security

MD5 Considered Harmful Today

Alexander Sotirov, Marc Stevens, Jacob Appelbaum, Arjen Lenstra, David Molnar, Dag Arne Osvik, and Benne de Weger

Adam J. Lee

March 18, 2014

[Many figures borrowed from the authors' web page http://www.win.tue.nl/hashclash/rogue-ca/]

University of Pittsburgh

What is a hash function?

Recall: A hash function is a function that maps a variable-length input to a fixed-length code

Hash functions should possess the following 3 properties:

- Preimage resistance: Given a hash output value *z*, it should be infeasible to calculate a message *x* such that *H(x)* = *z*
- Collision resistance: It is infeasible to find two messages *x* and *y* such that *H(x)* = *H(y)*
- Second preimage resistance: Given a message *x*, it is infeasible to calculate a second message *y* such that *H(x)* = *H(y)*

Question: Why are cryptographic hash functions important to PKIs?

Digital signature operations are expensive to compute! Instead of signing a certificate *c*, we actually sign H(*c*).

What happens if we don't have these properties?

Attack 1: No preimage resistance

- One attack is being able to recover a password from H(<password>)
- Not critical to the security of PKIs, but would cause issues in general

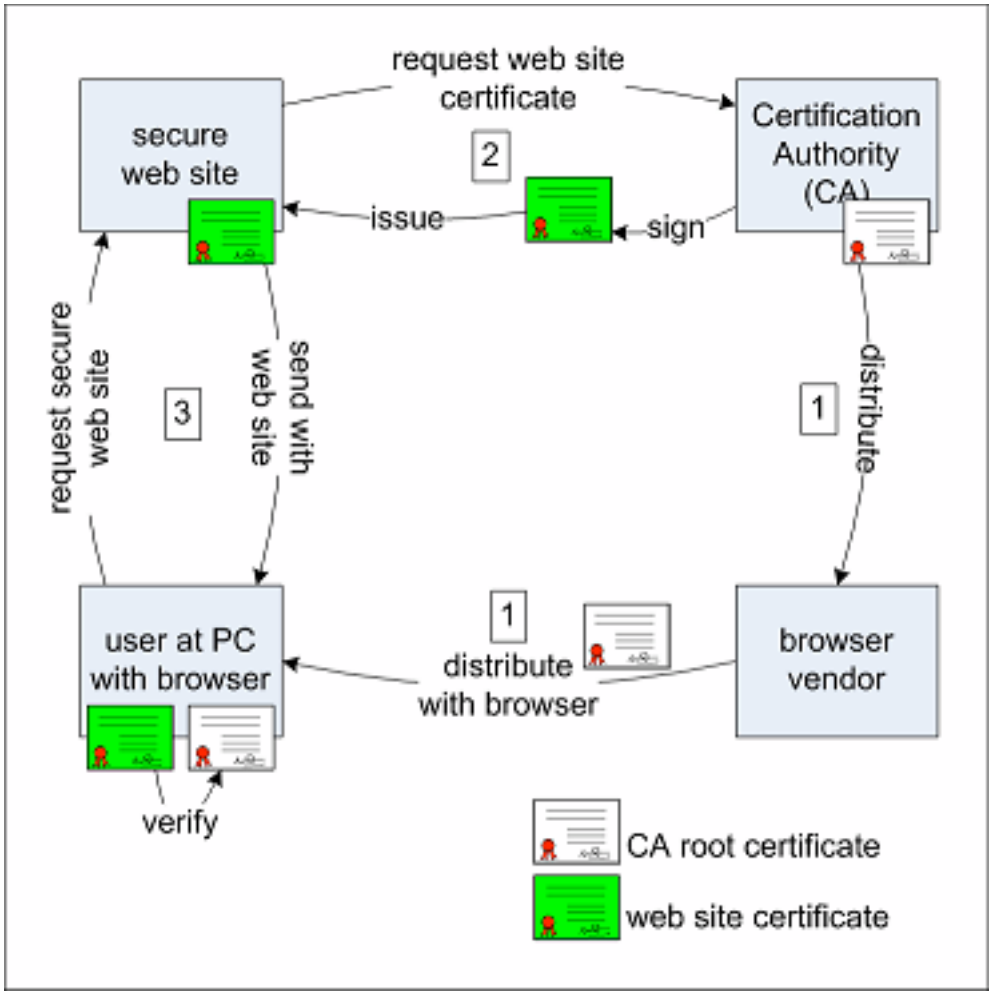
Attack 2: No second preimage resistance?

- Assume that we have a message m_1 with a signature computed over H(m_1)
- If we don't have second preimage resistance, we can find a message m_2 such that H(m_1) = H(m_2)
- The result: It looks like the signer signed m_2!

Attack 3: No collision resistance

- This means that we can find two messages that have the same hash
- The authors use a clever variant of this type of attack to construct two certificates that have the same MD5 hash!
- This is bad news…

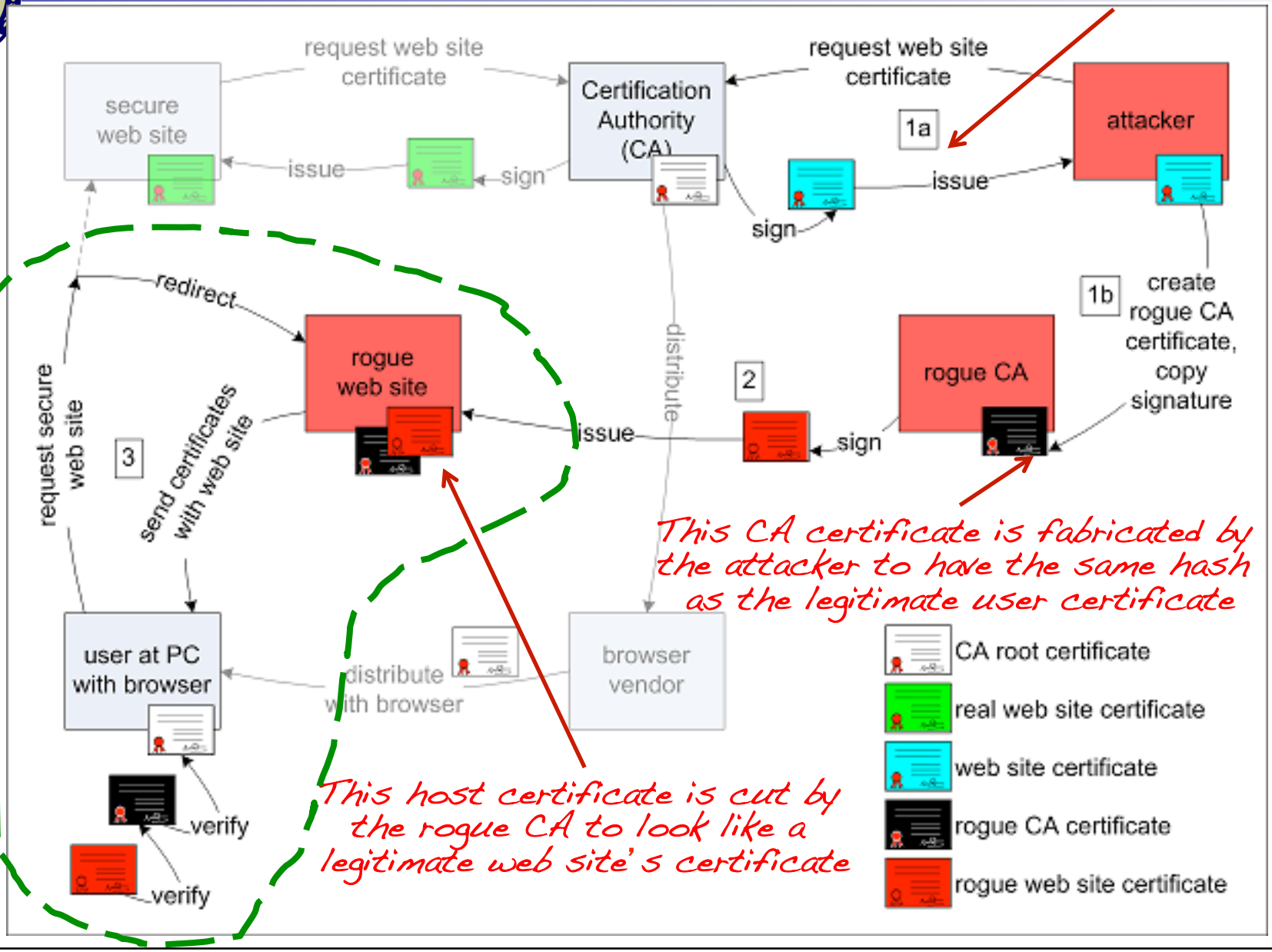
How SSL/TLS should work…



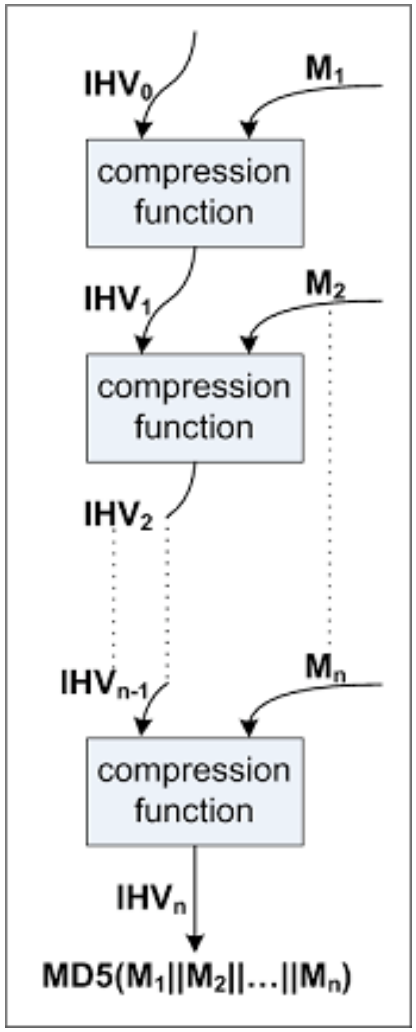
What's the big deal?

Question: Can you describe a scenario in which having two certificates with the same hash would be problematic?

Attack Overview



How does MD5 work?



Given a variable length input string, MD5 produces a 128-bit hash value

MD5 uses a Merkle-Damgård iterative construction

- 128-bit intermediate hash value (IHV) and a 512-bit message chunk are fed into a compression function that generates a 128-bit output
- This output is concatenated to the next 512-bit message chunk and the process is repeated
- The final IHV is the hash value for the message

Note that:

- The initial message must be padded out to a multiple of 512 bits
- IHV_0 is a publicly-known fixed value (0x67452301 0xEFCDAB89 0x98BADCFE 0x10325476)

Early attacks against MD5

Early partial attacks on MD5 were suggestive of larger troubles

- [1993] Den Boer and Bosselaers found a partial collision of the MD5 compression function (two different IHVs lead to the same output value)
- [1996] Dobbertin found a full collision of the MD5 compression function

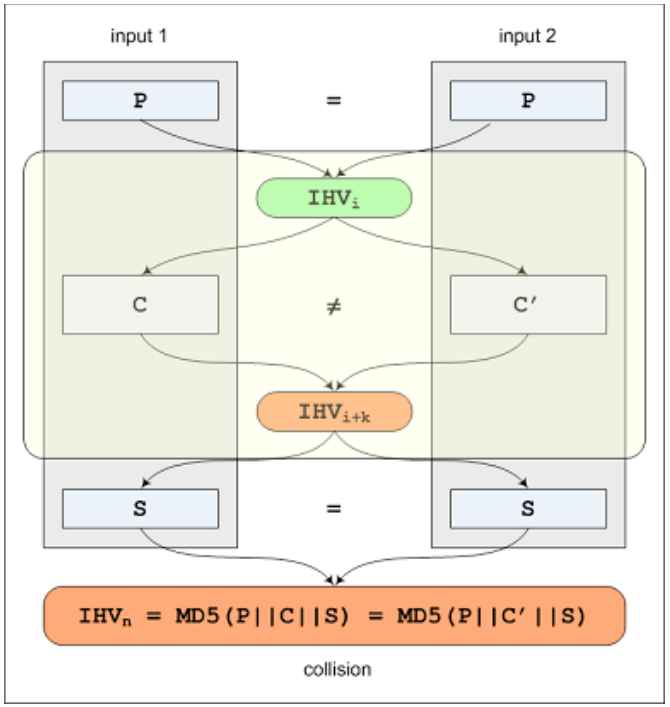
---

The first real attack on MD5 was found by Wang and Yu in 2004

Given any IHV, they showed how to compute two pairs $\{M_1, M_2\}$ and $\{M_1', M_2'\}$ such that:

- $IHV_1 = CF(IHV, M_1) \neq IHV_1' = CF(IHV, M_1')$
- $IHV_2 = CF(IHV_1, M_2) = CF(IHV_1', M_2')$

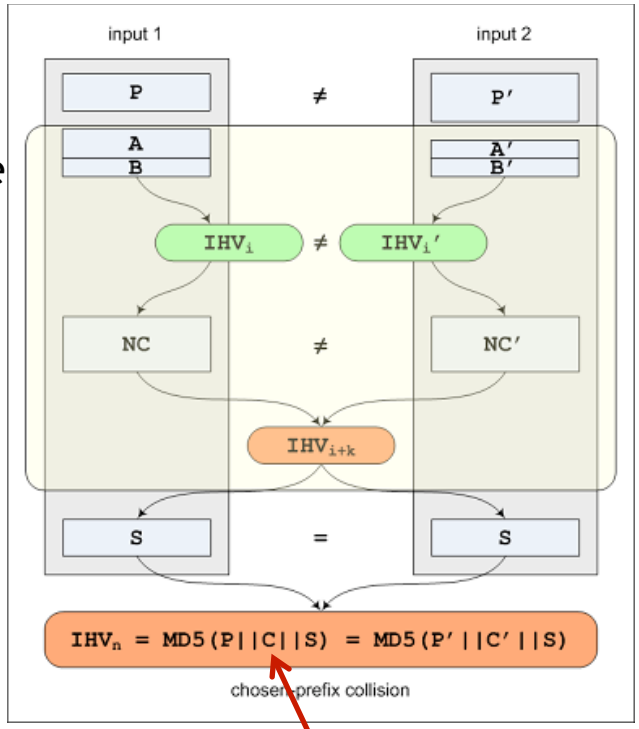
This yields a fairly scary generalization



In 2007 Stevens refined the Wang and Yu attack to allow differing message prefixes

How does this process work?

1. Choose P and P' at will
2. Choose A and A' s.t. (P || A) and (P' || A') are of the same bit length
3. In a "birthday step", choose B and B' such that (P || A || B) and (P' || A' || B') are a multiple of 512 bits and the output IHVs have a special structure
4. This special structure allows the attacker to find two near collision blocks NC and NC' such that the resulting MD5 function collides!



Note: C = A || B || NC

One use of this attack is generating different documents (plus some hidden content) that end up with the same hash value/signature!

Clearly, this opens the door to forged certificates…

How can we launch this attack against X.509?

An X.509 certificate contains quite a bit of information

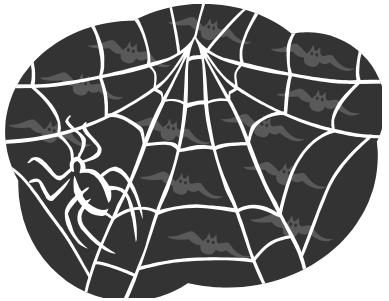
- Version
- Serial number: Must be unique amongst all certificates issued by the same CA
- Signature algorithm ID: What algorithm was used to sign this certificate?
- Issuer's distinguished name (DN): Who signed this certificate
- Validity interval: Start and end of certificate validity
- Subject's DN: Who is this certificate for?
- Subject's public key information: The public key of the subject
- Issuer's unique ID: Used to disambiguate issuers with the same DN
- Subjects unique ID: Used to disambiguate subjects with the same DN
- Extensions: Typically used for key and policy information
- Signature: A digital signature of (a hash of) all other fields

---

Insight 1: Stevens's chosen prefix attack means that it might be possible to generate two certificates with different subjects, but the same signature!

Insight 2: Collision blocks can potentially be hidden is (part) of the public key block and/or in extension fields!

Successfully launching this attack means finding a CA that will grant a certificate that has a signature over an MD5 hash



To find such a server, the authors wrote a web crawler

- Crawler ran for about a week
- Found 100,000 SSL certificates
- 30,000 certificates signed by "trusted" CAs
- Of these, 6 issued certificates with MD5-based signatures signed in 2008

Of the certificates found with signatures over MD5 hashes, 97% were issued by RapidSSL (http://www.rapidssl.com/)

RapidSSL issues certificates in an online manner, so they actually made an ideal target for launching this attack

- Predictable timing
- Not human-based, so multiple requests would not seem strange

Question: Why attack a production server instead of a test server?

Unfortunately, we have no control over the certificate serial number or validity fields!

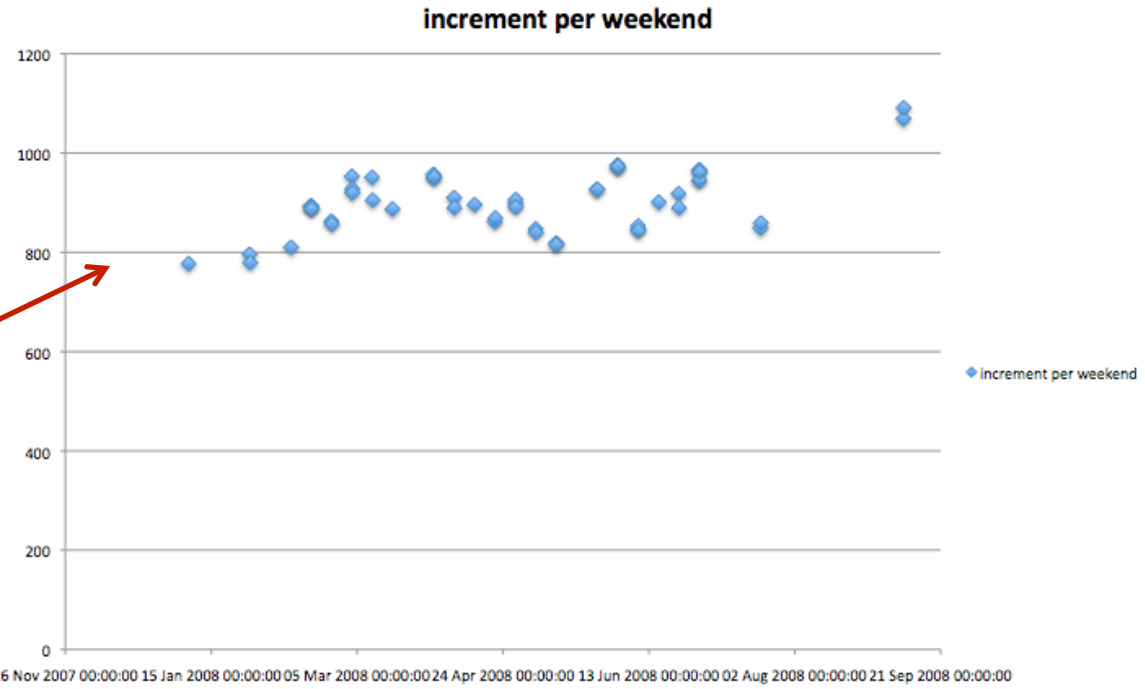
Question: Why might this pose a problem?

- Because these fields occur in the "chosen" prefix of the certificate and thus must be known before the collision blocks can be computed!

Predicting the validity period turned out to be trivial

- Certificate always issued 6 seconds after it was requested
- Valid for exactly one year

Furthermore, it turns out that RapidSSL uses a counter to populate the serial number field!



800-1000 certificates issued per weekend

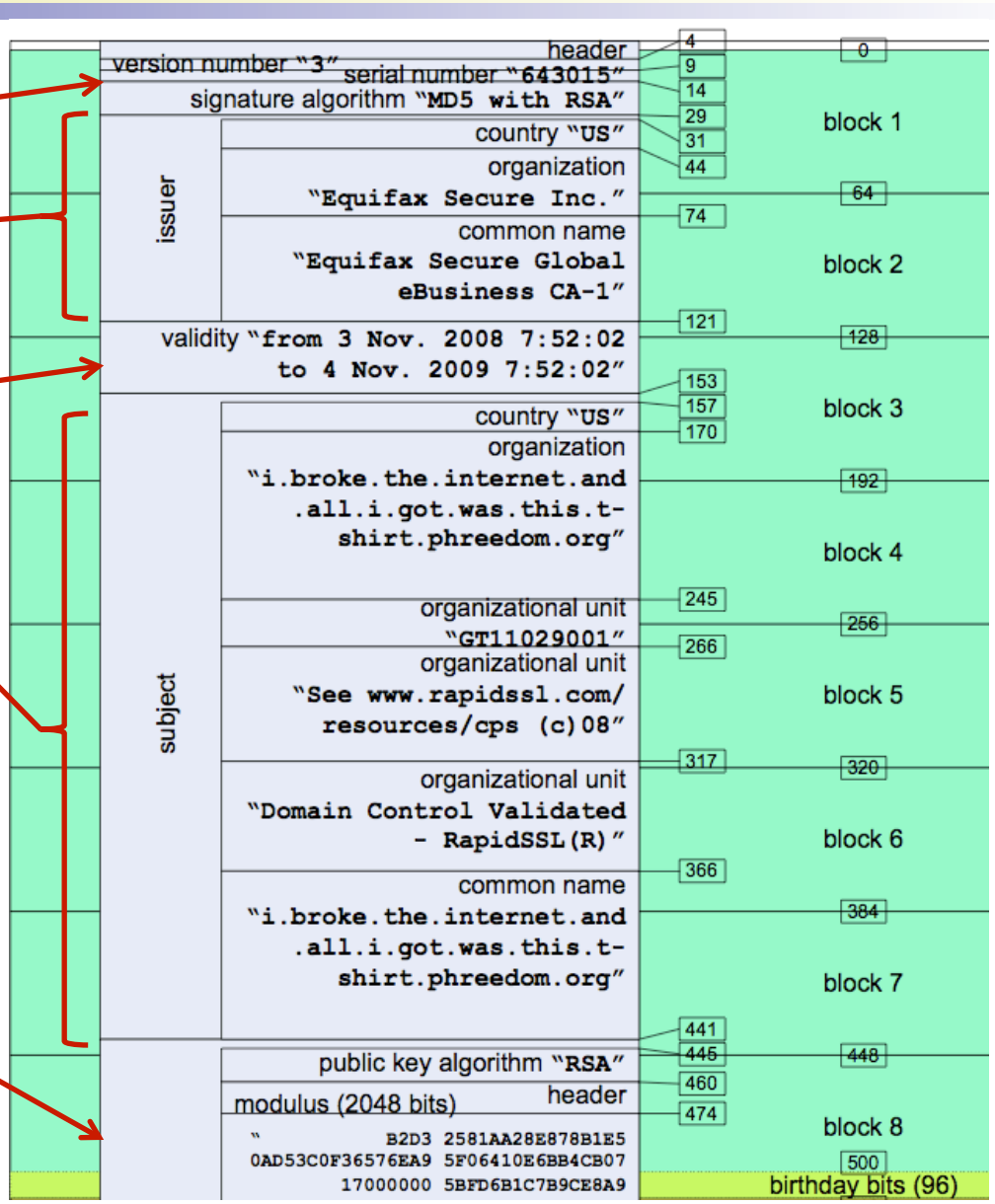
Setting up a legitimate certificate request

Predicted serial number

Info pulled from CA certificate

Derived validity period

Subject information completely chosen by the attacker



Question: How do you choose a public key (2048-bit modulus and an exponent) when part of it is the collision blocks needed to make the attack work?!?!

Answer: By being extremely clever

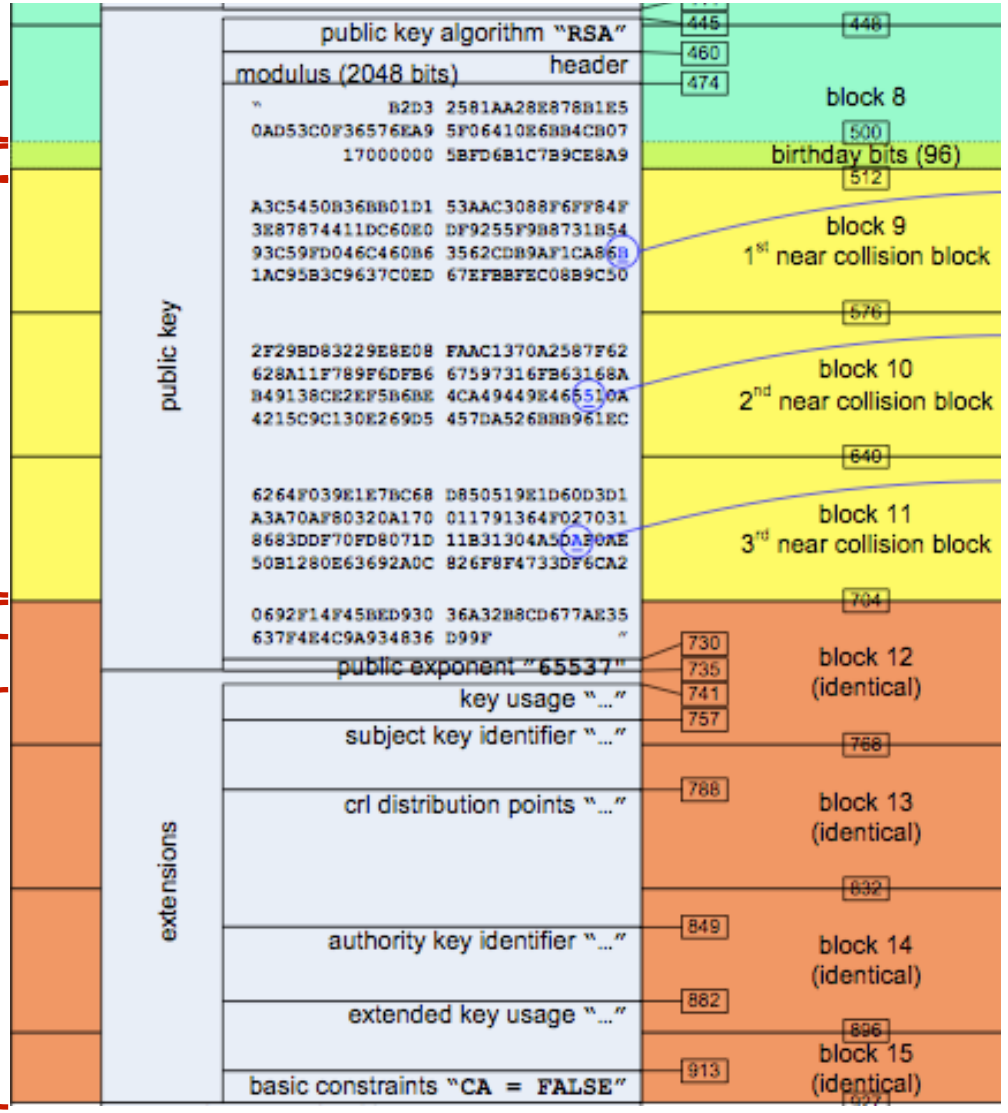
(a) chosen uniformly at random

(b) chosen to set up relationship between this certificate and the rogue CA cert to be generated next

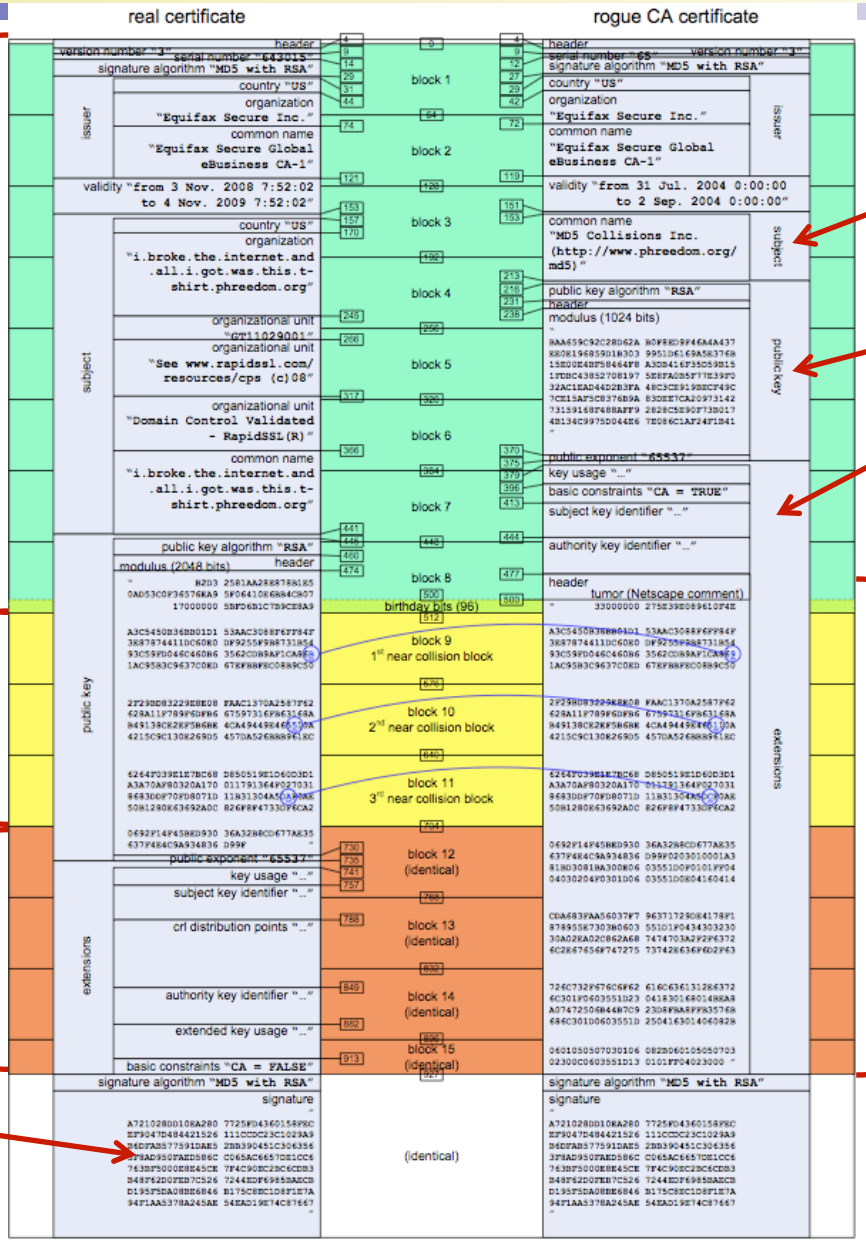
(c) output of the authors' collision-finding algorithm

(d) chosen by the attacker such that when the bits comprising parts (a)-(d) are interpreted as an integer, the result is a number "n" such that n = pq

This is just standard PKI junk…



Creating the rogue CA certificate



Chosen prefix

Collision blocks (different but related)

Identical trailers

Signature of CA

Shorter subject info

Small public key

Standard public key stuff (Note that this cert is for a CA!)

(Typically ignored) certificate extension to hold collision blocks and trailer from original certificate

The Nitty-Gritty

The attack used by the authors had two stages:

1. Finding the 96 “birthday bits”
2. Finding the three near collision blocks

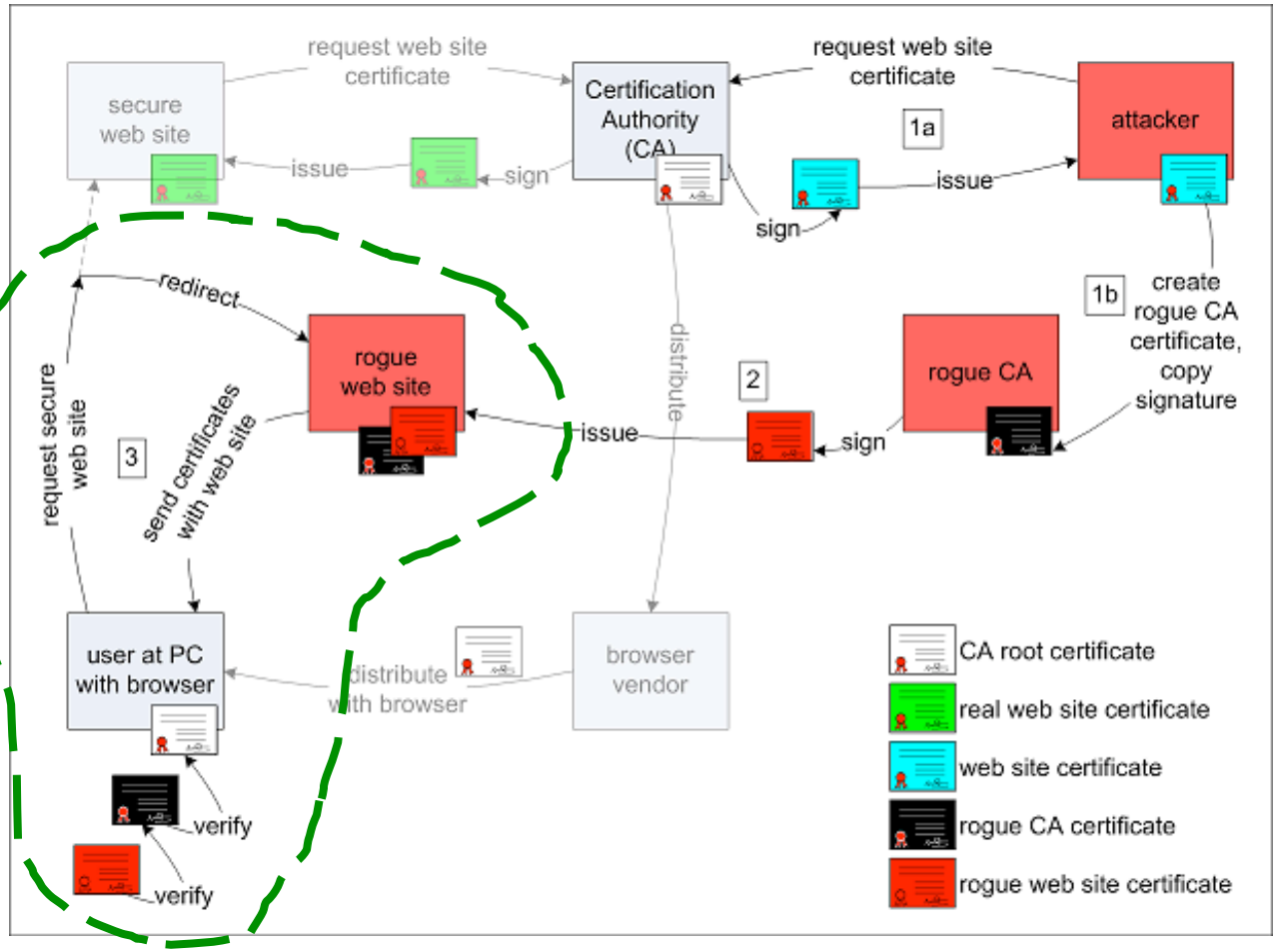
Stage 1 is computationally expensive, but well-suited for execution on the processors used by PlayStation3 game consoles. This took about 18 hours on a cluster of 200 PS3s.

Stage 2 is not terribly expensive, and is better suited to run on commodity PCs. This takes 3-10 hours on a single quad-core PC.

Attacking only on weekends (due to reduced CA load) the authors were able to carry out their attack in 4 weeks for a cost of $657.

This attack has very real implications…

In short, it is possible for a malicious principal to get a cheap X.509 host certificate, and leverage this to create a trusted CA certificate!



The Bottom Line: Consider MD5 broken!

Broken? What do you mean by broken?

Since MD5 does not have all three properties required of cryptographic hash functions, it is not considered to be "cryptographically strong"

While you could still use MD5 for non-cryptographic applications (such as?), it is better to avoid it altogether

---

SHA-1 is the cryptographic hash function that is now most often used

Attacks against SHA-1 have been announced, but still require a large amount of computing effort to mount

- E.g., Collision finding in 2^{63} steps, rather than 2^{80}

NIST recently ran a competition to design a replacement for the SHA family of hash functions

How did vendors react to the MD5 break?

December 30th: This work was presented at the Chaos Computing Congress

December 31st

- Verisign issues a statement, stops using MD5
- Microsoft issues Security Advisory (961509): "Research proves feasibility of collision attacks against MD5"
- Mozilla has a short item in the Mozilla Security Blog: "MD5 Weaknesses Could Lead to Certificate Forgery"

January 2nd

- RSA has an entry in the Speaking of Security blog: "A Real New Year's Hash"
- US-CERT, the US Department of Homeland Security's Computer Emergency Readiness Team, published Vulnerability Note VU#836068: "MD5 vulnerable to collision attacks"

January 15th

- Cisco published "Cisco Security Response: MD5 Hashes May Allow for Certificate Spoofing"

Discussion

Question 1: People have had evidence that MD5 was weak since the mid 1990s. Why did it take this long to finally convince vendors to stop using MD5? Do you think that this will change the response to future cryptographic vulnerabilities?

Question 2: In the 1980s, Adi Shamir and Eli Biham "discovered" differential cryptanalysis, which is a general means of attacking block ciphers. It was later revealed that NSA and IBM actually discovered this technique first, but kept it a secret. What if this MD5 attack was known about before it was discovered in the public domain? What would the implications be?