Cryptography & Encryption:6G7Z1011 : Digital Signatures, and Introduction to Collision Algorithms

Keith Yates

March 8, 2019

Four Key Goals

Recall the four key goals :

1. Confidentiality : information is kept secret except from the authorised parties;
2. Integrity: data tampering is easily detectable;
3. Authentication: it is possible to determine the sender of the message.
4. Non-repudiation: the sender of a message can not deny the contents of a message

Note authentication and non-repudiation are not the same, the distinction is subtle.

Digital Signatures

Today we discuss Digital Signatures, they make use of ideas that are very similar to the RSA algorithm and the ElGamal algorithm (and we have already coded them).

Why bother with Signatures?

To date we have discussed algorithms that resolve the following problems:

1. the encryption and decryption of data;
2. showed it is possible to establish a shared secret between two people which they could then use to encrypt data.

All of this scenarios have involved Bob and Alice communicating securely and Eve has played the role of the trouble maker.

Why bother with signatures?

The above techniques allowed Alice and Bob to establish a shared key and thus communicate securely. The problem is that in many transactions whilst Alice and Bob need to communicate securely not only do they not trust Eve they do not trust each other. As such there is an interest in a method that would verify (to say a judge) that Alice did send a message to Bob and Bob cannot deny its contents. This can be achieved by a digital signature.

What is a Digital Signature?

As the term signature might suggest, a digital signature indicates that the person who signs a document agrees with the contacts of the document. We introduce two new characters:

1. Sam : signs a document.
2. Victor: verifies the document has been signed by Sam.

Obvious Criteria

We require the two obvious criteria:

1. Only Sam can create Sam's signature.
2. It is easy for anybody to prove that the document is signed by Sam.

Notation

1. K_{Pr} Sam's private signing key.
2. K_{Pu} Sam's public verification key.
3. Sign : an algorithm that takes a document D and a private key K_{Pr} and produce a signed document
4. Verify: an algorithm that takes the signed document and a verification key K_{Pu} and returns true if Sam signed the document or false if Sam did not sign the document.

The Algorithm

The algorithm described is from the original RSA paper, and is termed the RSA digital signature method. Sam does the following, he picks two large primes p and q he forms $N = pq$ and picks a v a verification exponent. Sam solves for s in

$$sv = 1 \mod (p-1)(q-1). \tag{0.1}$$

Note N and v are public knowledge.

Euler's Formula

Recall for distinct primes p and q then if

$$g = \gcd(p-1, q-1) \tag{0.2}$$

then

$$a^{(p-1)(q-1)/g} = 1 \mod pq \tag{0.3}$$

for all a satisfying $\gcd(a, pq) = 1$.

Signing and Verifying

1. Denote by D ($1 < D < N$) the document to be signed
2. The signed document is

$$S = D^s \mod N. \tag{0.4}$$

3. Victor has knowledge of S, N and v so he can evaluate

$$S^v \mod N \tag{0.5}$$

and this is D because

$$S^v = D^{sv} = D \mod N. \tag{0.6}$$

Covered the theory of this last week, page 170 of the text by Paar.

Solving Equations in Finite Fields

In Cryptography we are interested in solving equations of the form

$$a^x = b \mod c \tag{0.7}$$

for x, where all other terms known in eqn. 0.7 are known. We have already solved these equation by brute force, we now look at a new solution technique *collision algorithms* and *iterative algorithms*. To discuss collision algorithms we need to introduce some now ideas and terminology.

Probability

Formally a probability space is a function

$$\mathbb{P} : \Omega \to [0,1], \quad A \mapsto \mathbb{P}(A). \tag{0.8}$$

1. Ω is the sample space;
2. For $A \in \Omega$ we read $\mathbb{P}(A)$ as the probability event A occurred.

Probability

The formal definition takes some getting used to, a simple example will help. Consider the following sample space

$$\omega = \{(m, n) \mid 1 \leq m, n \leq 6\}\,; \tag{0.9}$$

it has a natural representation as the events generated when you roll two dice, so $(2, 3)$ is the first row is 2 and the second throw is a 3 and the probability of this is

$$\mathbb{P}((2, 3)) = \frac{1}{36}. \tag{0.10}$$

Another example this time of a compound event

$$\mathbb{P}(\text{second throw } = \text{first throw}) = \frac{1}{6}. \tag{0.11}$$

problem

A fair coin is tossed 10 times. What are the probabilities of the following events:

1. $E_1 = \{\text{the first 5 tosses are heads}\}$
2. $E_2 = \{\text{the first 5 tosses are heads and the rest are tails}\}$
3. $E_3 = \{\text{exactly 5 tosses are heads}\}$

collision algorithms

There are forty people in a room.

1. What is the probability that someone has the same birthday as you?
2. What is the probability that at least two people in the room have the same birthday?

The answers are different, this question is termed ‘the birthday paradox’

Wrong Answer

- Q: There are 40 people in a room, what is the probability that someone has the same birthday as you?
- A: the answer is not $\frac{40}{365}$, if you think it is then what would your answer be if there were 500 people in the room?

Answer Part 1

$$\mathbf{P}(\text{some one has your birthday}) = \mathbf{1} - \mathbf{P}(\text{no one has your birthday})$$
$$= 1 - \prod_{i=1}^{40} \mathbf{P}(\text{person } \mathbf{i} \text{ does not have your birthday})$$
$$= 1 - \left(\frac{364}{365}\right)^{40} = 10.4\% \tag{0.12}$$

Answer Part II

The second question does not 'fix' a birthday

$$\begin{aligned}&\mathbf{P}(\text{two people share a birthday}) = \mathbf{1} - \mathbf{P}(\text{all 40 have different b'days})\\ &= 1 - \prod_{i=1}^{40} \mathbf{P}(\text{person } \mathbf{i} \text{ birthday different from } \mathbf{1}, \ldots \mathbf{i} - \mathbf{1})\\ &= 1 - \left(\tfrac{365}{365}\ \tfrac{364}{365}\ \tfrac{363}{365} .. \tfrac{326}{365}\right) = 89\%\end{aligned} \tag{0.13}$$

Summary

1. It requires 23 people to have a better than 50 % chance of a matched birthday
2. It requires 253 people to have a better than 50 % chance of someone having your birthday.

In a collision algorithm we (typically) generate two lists, a match (for example, a shared birthday) across two lists corresponds to a solution to our problem, and while we can not say for definite that we will have reached a solution by list length j we can say

1. If the list length is greater than m there is a x% probability of their being a solution in the list

This technique is popular in cryptography

Relevance to Cryptography

We can turn the argument around, suppose we are given a number m and ask if it is prime then we could do the following:

1. pick a random number $x < m$ if $x \mid m$ then the number is not prime and the process terminates
2. go back to step 1.

Suppose the algorithm had not terminated after x-steps then could we say something about its probability of being prime?

Searching Lists

We proceed in a abstract manner. How the lists are generated is obviously dependent upon the algorithm under consideration, but we can say something about matches in general lists. We wish a statement of the type:

- After completing n iterations of this process there is a x% probability you will have achieved y.

or to keep you focused

- After completing n iterations of this process there is a x% probability at least one of the iterates is a prime number.

collision theorem

⌜A box contains N balls, n are red and $N - n$ are blue. Bob

1. Removes a ball, notes the colour;
2. Replaces ball in box.

The above process is repeated m times then

a) The probability that Bob selected at least one red ball is

$$\mathbb{P}(\text{at least one red}) = 1 - \left(1 - \frac{n}{N}\right)^m \qquad (0.14)$$

b) A lower bound for

$$\mathbb{P}(\text{at least one red}) = 1 - \exp\left(\frac{-mn}{N}\right) \qquad (0.15)$$

Note m does not have to be be particularly large to ensure $\mathbb{P}$(at least one red) is near 1, eqn. 0.15 is important it tells us how long we need to keep picking balls to ensure there is a probability greater that (say) 0.99999 that we have found an answer

⌟

proof of collision theorem

The proof is straight forward

$$\begin{aligned}\mathbb{P}(\text{at least one red in } m \text{ attempts}) &= 1 - \mathbb{P}(\text{all } m \text{ choices are blue}) \\ &= 1 - \prod_{i=1}^{m} \mathbb{P}(\text{choice } i \text{ is blue}) \\ &= 1 - \prod_{i=1}^{m} \left(\frac{N-n}{N}\right) \\ &= 1 - \left(1 - \frac{n}{N}\right)^{m}\end{aligned} \tag{0.16}$$

proof of second part of collision theorem

The following is true. For all $x \in \mathbb{R}$

$$\exp(-x) \geq 1 - x \tag{0.17}$$

Thus setting $x = \frac{n}{N}$

$$\exp(\tfrac{-n}{N}) \geq 1 - \tfrac{n}{N} \quad \text{and} \quad \exp(\tfrac{-nm}{N}) \geq \left(1 - \tfrac{n}{N}\right)^m . \tag{0.18}$$

$$\begin{aligned} \mathbb{P}(\text{at least one red in } m \text{ attempts}) &= 1 - \left(1 - \tfrac{n}{N}\right)^m \\ &\geq 1 - \exp(\tfrac{-nm}{N}). \end{aligned} \tag{0.19}$$

Note this bound is <u>independent</u> of any algorithm, that is the balls are drawn are random.

Red v Blue

I kept the discussion to red and blue balls to keep it abstract. The important point here is that you can split the set you are searching in into two disjoint sets: Examples:

1. Red balls = prime numbers, Blue ball = non prime numbers;
2. Red balls = divisible by 11, 17 and 19 , blue balls = not divisible by at least one of 11, 17 or 19.

And note again, the bound in eqn. 0.19 is just randomly drawn balls.

nomenclature

1. Collision algorithms;
2. Meet-in-the-middle algorithm/attack;
3. Square root algorithm/attack.

The above terms are all used to describe collision algorithms.

Iteration

An iterative process is a function such that the state of the system at stage $n+1$ is dependent on the system at stage n. Given a start point x_1 we generate $\{x_i \mid i \in \mathbb{N}\}$. For example, let $x_1 = 1$

$$f : \mathbb{N} \to \mathbb{R}, \quad x \mapsto x^2 + 1. \tag{0.20}$$

then

$$\begin{aligned} x_1 &= 1 \\ x_2 &= (x_1)^2 + 1 = 1^2 + 1 = 2 \\ x_3 &= (x_2)^2 + 1 = 2^2 + 1 = 5 \\ x_4 &= (x_3)^2 + 1 = 5^2 + 1 = 26 \end{aligned} \tag{0.21}$$

Pollard's Rho Method

Let $f : X \to X$, $x \mapsto f(x)$ with X a finite set. For any $x_0 \in X$ define

$$x_1 = f(x_0), \quad x_2 = f(x_1), \quad x_3 = f(x_2), x_4 = f(x_3), \ldots \tag{0.22}$$

that is $x_n = f \circ f \circ f \ldots f(x_0)$, where $\circ$ denotes function composition. We seek

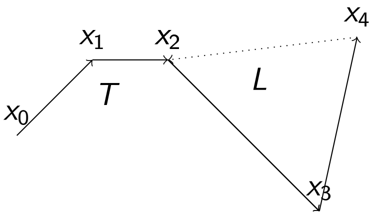

Figure: Iterative maps in finite sets need to 'return' to an earlier point, the loop L is the closed circle, and the tail T consists of those points not in the loop M

Pollard's ρ method

Let X be a finite set $f : X \to X$, $x \mapsto f(x)$ a function (any function!) and x_0 a start point (as we often work mod n then X is finite).

1. If the orbit $\{x_0, x_2, \ldots\}$ of x_0 has a tail of length T and a loop of length L then $x_{2i} = x_i$ for some $1 \leq i < T + L$.
2. If the map f is sufficiently random (I will expand on this) then the expected length E of $L + M$ before we find a solution is
$$E(T + L) \approx 1.2\sqrt{|X|}. \tag{0.23}$$

Iterative algorithms

1. Start algorithm with a ‘guess’ x_1.
2. Update to $x_{i+1} = f(x_i)$ (the choice of f dependent on the problem).
3. Check some criteria : if criteria is satisfied we have an answer and quit.
4. Increment i.
5. Return to 2.

Solving Equations

We illustrate iterative equation solving with its simplest usage: *the fixed point method*. Suppose we wished to solve

$$x^3 - x - 1 = 0 \quad \text{then we are interested in} \quad f(x) = (1+x)^{1/3}. \tag{0.24}$$

Why? Because a fixed point of f corresponds to a solution to $x^3 - x - 1$. There are certain mathematical criteria that have to be in place (and they are) and the sequence

$$x_1, f(x_1), f^2(x_1), f^3(x_1) \tag{0.25}$$

will converge to a solution with a ‘reasonable’ choice of x_0.

Reading Exercise

It is a reading exercise/Homework to read up on Pollard's Rho method. I will illustrate the iterative mechanism with a simple example that you will code in labs.