Tr

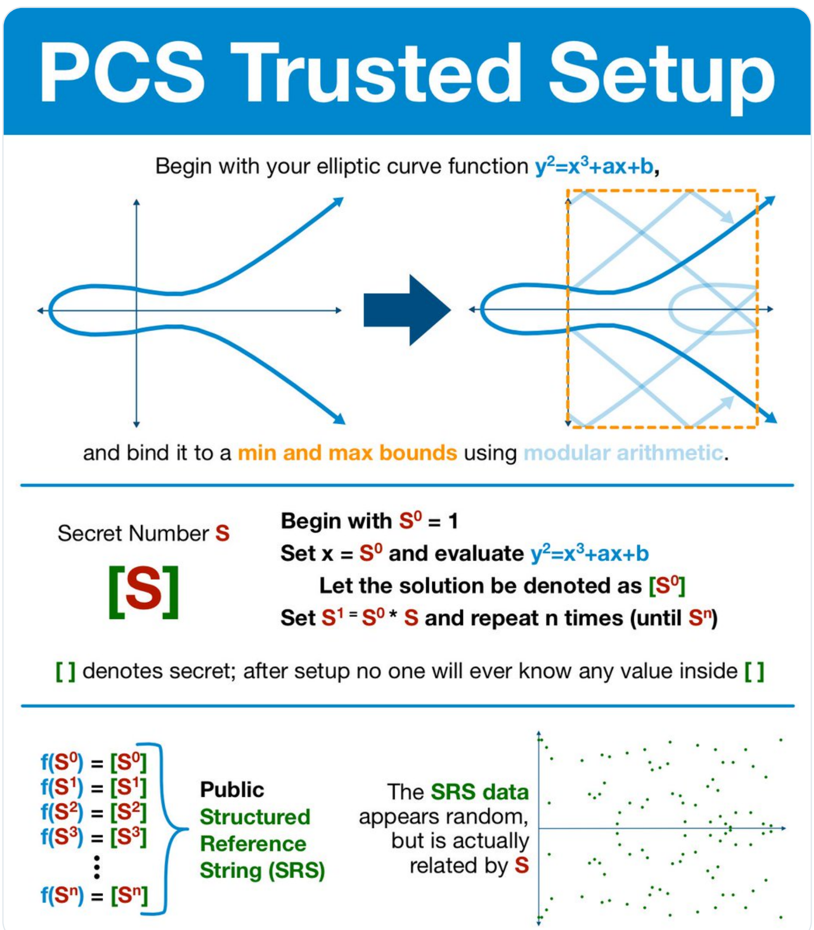
Haym @SalomonCrypto

Oct 17 • 20 tweets • SalomonCrypto/status/1581864402076151809

(1/19) Elliptic Curve Cryptography: Trusted Setups

Successful cryptography is cryptography that transforms legible data into digital-static. Before we go big, let's wrap our mind around something simple.

An instruction manual for creating a permanently secret number.



(2/19) As previously mentioned, the purpose of this series is to give you enough knowledge to gain some perspective on some of the most complicated and bleeding edge technology of 2022.

But don't worry, you will be fine with high-school level math.

> **Haym**
> @SalomonCrypto · **Follow**
>
> Replying to @SalomonCrypto
>
> (2/18) Preface:
>
> This is part of a series on elliptic curve cryptography and its applications for @ethereum. This is simplified to a MINIMAL level, aiming at ~high-school math.
>
> @VitalikButerin @dankrad @danboneh @chaseklvk, if you read this, I'm sorry for what I did to the math.
>
> 4:03 PM · Oct 15, 2022
>
> 19 Reply Copy link
>
> **Read 2 replies**

(3/19) Any (decent) cryptographic system has the same two part goal:

1) transform data into digital-nonsense, indistinguishable from random noise
2) allow specific individuals (and only those individuals) to reverse the process and recover the original data

(4/19) The core of elliptic curve cryptography in one tweet:

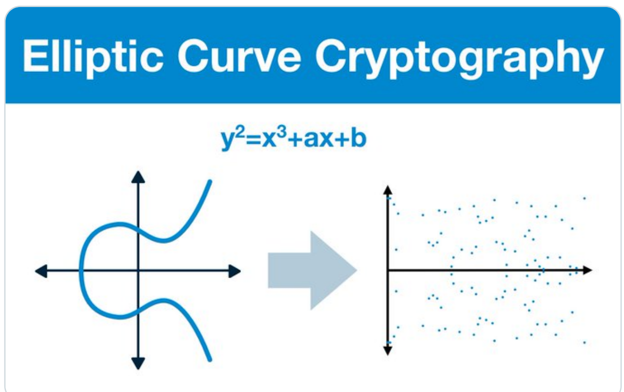
It is very difficult to solve the elliptic curve discrete logarithm problem (that is, if you add two points over and over again using modular arithmetic, it takes A LOT of work to find out how many times you did it).

Haym
@SalomonCrypto · **Follow**

(1/25) Degen's Handbook: A Practical Guide to Elliptic Curve Cryptography

Elliptic Curve Cryptography is TOUGH; sometimes you need to walk away from the details and reframe on the bigger picture.

Let's start over and level set on the basics; I think this will help it all click.



5:17 PM · Oct 16, 2022

Read the full conversation on Twitter

454 Reply Copy link

Read 12 replies

(5/19) Quick math reminder - modular arithmetic.

tl;dr if you divide x by y the remainder is z, which we write as x mod y = z

Haym
@SalomonCrypto · **Follow**

Replying to @SalomonCrypto

(9/21) The modulo operator (written as "mod") is the operation that finds the remainder.

10 / 3 = 3 with a remainder of 1 = 10 mod 3 = 1
14 / 6 = 2 with a remainder of 2 = 14 mod 6 = 2
3 / 10 = 0 with a remainder of 3 = 3 mod 10 = 3

Imagine a clock being mod 12.

2:43 AM · Oct 13, 2022

6 Reply Copy link

Read 1 reply

(6/19) If a clock is mod 12, then consider the following question: "Alice left at 4 and arrived at 6. How many hours did she spend traveling?"

2 hours? 14 hours? 26 hours? The only way to figure it out is to start guessing.

This is the discrete logarithm problem.

(7/19) Again, it is very hard to solve the elliptic curve discrete logarithm problem. Like "1000s of gigabrains have been working for 30+ years/there are trillions of $s at stake and we still just guessing" hard.

This is the property we are going to build our system out of.

(8/19) We are going to start off by hiding a secret number inside an elliptic curve.

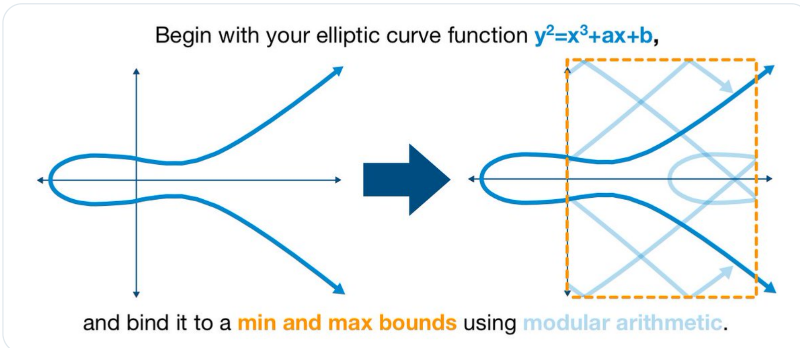
The number must be 100% random and secret. At the end of this process this number will be thrown away; it will never be known directly, it will just exist hidden in an elliptic curve.

(9/19) For now, just use god mode: we are just going to assert that a single computer generates a random number S and permanently discards it at the end of this process.

In practice, we use methods based around secure multiparty computation... but we'll get to that part later.

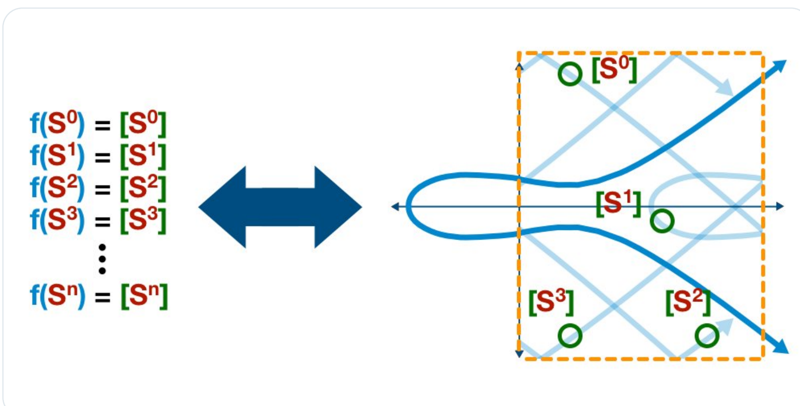
(10/19) Now that we have our secret S, it's time to prepare our elliptic curve.

First, we must bind our curve to a minimum and maximum bounds using modular arithmetic. As previously discussed, this will (intentionally) introduce the elliptic curve discrete logarithm problem.



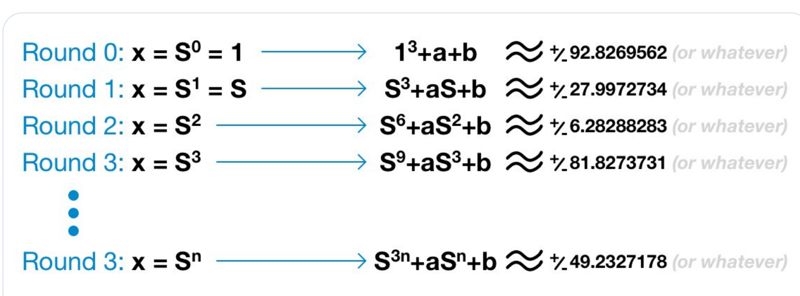
(11/19) Next, we are going to use our (modular) elliptic curve and our secret (number) S to generate a series of values.

We want to start with S^0 and end with S^n. Each time we feed it into our elliptic curve formula and generate a new value, representing a point on the curve.



(12/19) Take a look at the example below. Don't pay attention to the specific numbers (I literally bashed my keyboard like an ape).

The point is 1) to illustrate how S progresses with each round and 2) to understand the outputs are real values/numbers, not formulas.

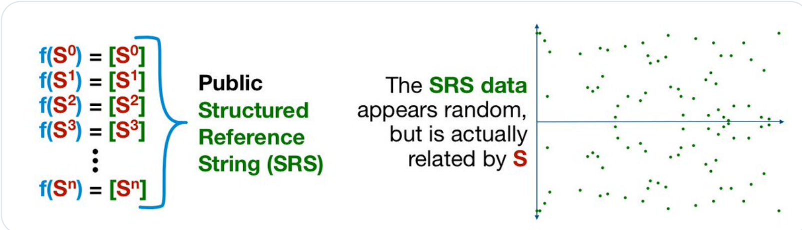


(13/19) In fact, both parts components of the elliptic curve point (x and y) are provided - which includes S, S^2, S^3, etc. But this is when the discrete logarithm problem comes into play.

The difficult problem: discovering S in precisely this situation.

(14/19) Without access to S, the list of points look like noise. Yes, we can see the clear horizontal symmetry, but otherwise it seems near random (for the record, this randomness is at the mathematical level, no need to rely on my graphic)

But that's the point! It's NOT random!



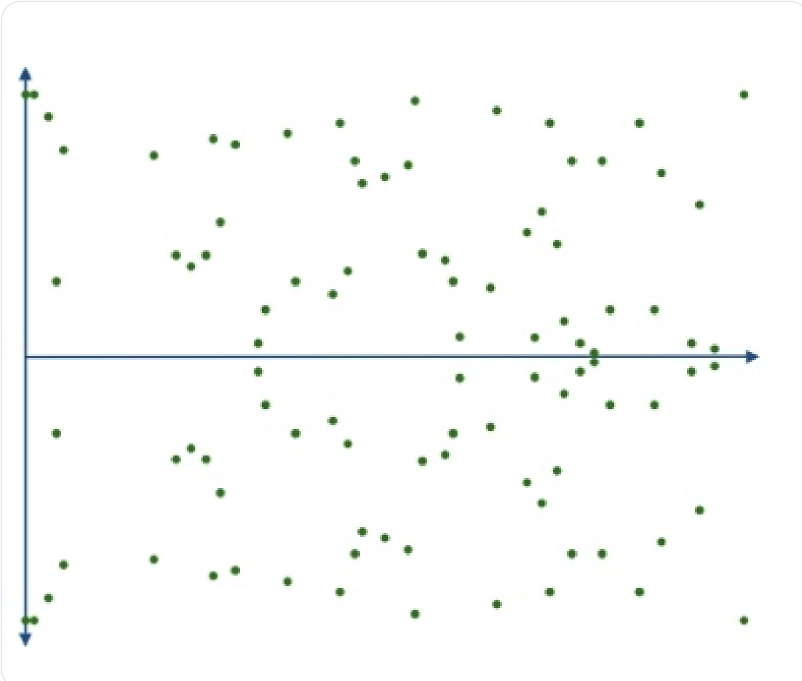
(15/19) We use this process to generate n points - n is as many as we desire, we'll understand how many we need a little later in the series.

At the end of the process we permanently discard S. After this step, the true value of S becomes forever lost.

(16/19) This is all that remains of S. A secret number lost in a scattered mess of points.

An arbitrary splattering of points, indistinguishable from random data.

Exactly what we are looking for in a robust cryptographic system!



(17/19) Unfortunately, dear reader, it’s time to draw this chapter to a close. We need to keep these threads succinct!

But let’s sum up the key takeaways on our way out.

If nothing else, here's what you should walk away with:

(18/19)
- we want to hide a secret number S
- we generate points by repeatedly multiplying S with itself and applying our elliptic curve function
- the specific points - related by S - are indistinguishable from random data
-the true value of S is destroyed and lost, forever

(19/19) And for those who just can't wait, here's a peek around the corner:

First, go back to the first tweet. Did you notice the title? PCS = Polynomial Commitment Scheme

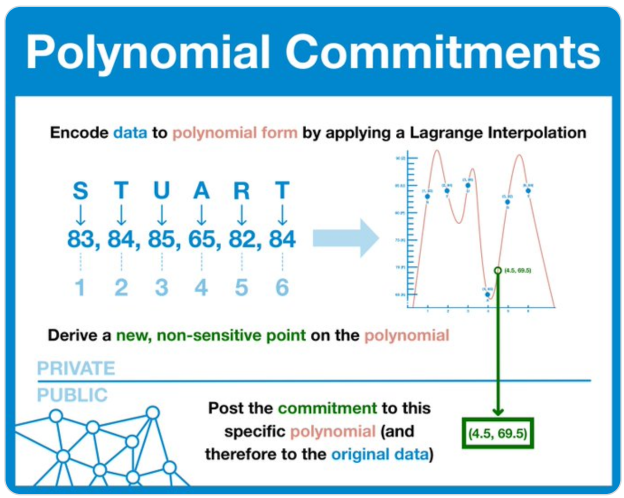
Second, take a look at the thread below. You'll be happy you got ahead of the reading!

Haym
@SalomonCrypto · **Follow**

(1/17) Cryptography Basics: Polynomial Commitments

Creating unique digital fingerprints is core to cryptography. We use tools like hashing to provide cryptographic-certainty without revealing any info.

But hashing is so destructive, is there a better way to commit to data?



1:50 AM · Oct 16, 2022

Read the full conversation on Twitter

370 Reply Copy link

Read 15 replies

Like what you read? Help me spread the word by retweeting the thread (linked below).

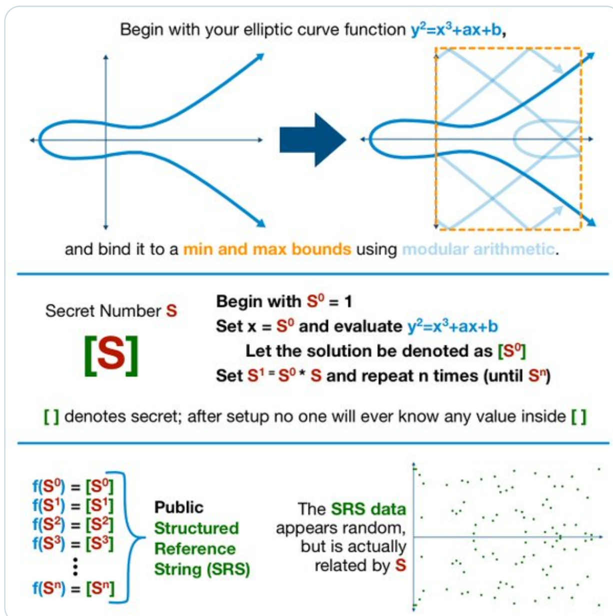
Follow me for more explainers and as much alpha as I can possibly serve.

Haym
@SalomonCrypto · **Follow**

(1/19) Elliptic Curve Cryptography: Trusted Setups

Successful cryptography is cryptography that transforms legible data into digital-static. Before we go big, let's wrap our mind around something simple.

An instruction manual for creating a permanently secret number.



4:27 AM · Oct 17, 2022

Read the full conversation on Twitter

246 Reply Copy link

Read 7 replies

• • •