

Join Today For Free And Earn crypto while learning about Crypto For Beginners
 (https://courses.blockgeeks.com/reports/)



LOGIN (HTTPS://BLOCKGEEKS.COM/LOGIN/)

Ask

Blockchain (https://blockgeeks.com/abg)
 Community



Guides (https://blockgeeks.com/guides/?
 view=sections)



Favorite (/my-
 Guides favorites)

Blockchain

For (/guides/?
 Businessstagfilter=true&filter=Blockchain%20for%20Business)

For (/guides/?
 Investorstagfilter=true&filter=Blockchain%20for%20Investors)

For (/guides/?
 Intermediatetagfilter=true&filter=Blockchain%20for%20intermediate)

For (/guides/?
 Developerstagfilter=true&filter=Blockchain%20for%20Developers)

Startups (/guides/?
 tagfilter=true&filter=Blockchain%20startups)

Cryptocurrency

Security (/guides/?
 tagfilter=true&filter=security)

For (/guides/?
 Investorstagfilter=true&filter=Crypto%20for%20investors)

Blockchain Community

RSK (/guides/?
 tagfilter=true&filter=rsk)

Aion (/guides/?
 tagfilter=true&filter=aion)


EARN CRYPTO (/REGISTER/)


Network [tagfilter=true&filter=aion-network](#))

Cosmos [\(/guides/?tagfilter=true&filter=cosmos\)](#)

Maker [\(/guides/?tagfilter=true&filter=maker\)](#)

Malta [\(/guides/?tagfilter=true&filter=malta\)](#)

 Articles [\(/articles/?view=sections\)](#)

 Videos [\(/guides/?tagfilter=true&filter=videos\)](#)

CONTENT

EARN CRYPTO ([/REGISTER/](#))

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

What are zkSNARKs:
Spooky Moon Math

Mid-Tech Solution

Low-Tech Solution

Proof vs Proof Of
Statements

The Schnorr
Identification Protocol

How to make zero
knowledge proofs non-
interactive?

What is the use of Zk-
Snarks?

How do ZkSnarks work?

Functionality of
ZkSnarks

The use of ZkSnarks in
cryptocurrency

Looking Ahead

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

EARN CRYPTO (/REGISTER/)

Navigation

#Blockchain 101 (/guides/?tagfilter=true&filter=blockchain-101)

#Blockchain for Business (/guides/?tagfilter=true&filter=blockchain-for-business)

#Blockchain for Developers (/guides/?tagfilter=true&filter=blockchain-for-developers)

#Zcash (/guides/?tagfilter=true&filter=zcash)

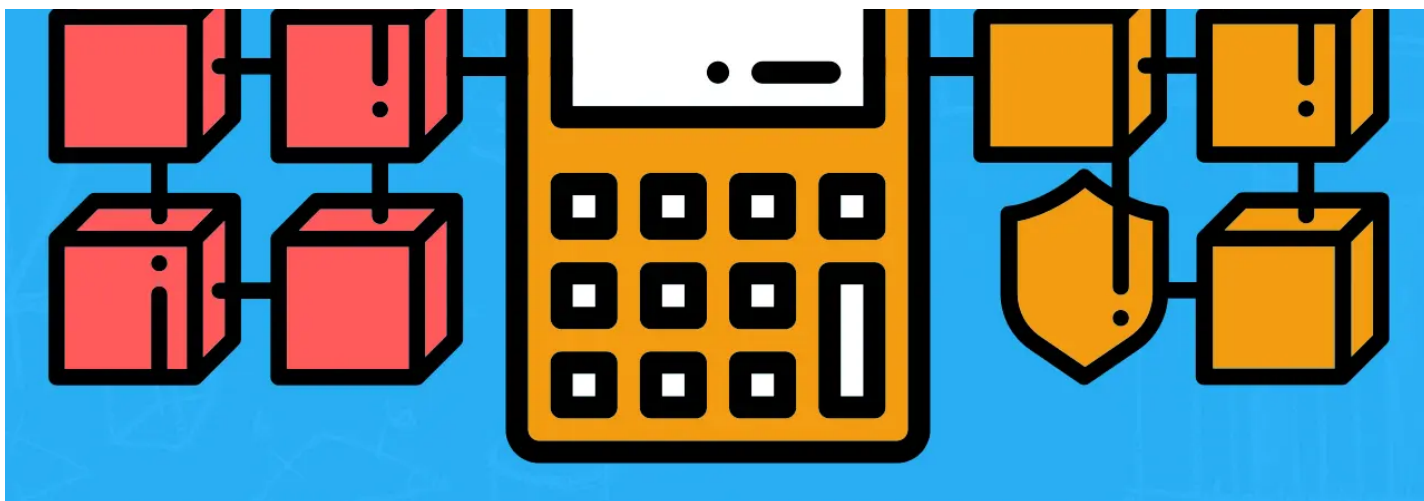
What are zkSNARKs? The Comprehensive Spooky Moon Math Guide

blockgeeks.com/author/robert

3 years ago

6 COMMENTS ([HTTPS://BLOCKGEEKS.COM/GUIDES/WHAT-IS-ZKSNARKS/#COMMENTS](https://blockgeeks.com/guides/what-is-zksnarks/#comments))





What are zkSNARKS?

0:00 / 0:00

What is zkSNARKs: Spooky Moon Math. With Ethereum entering the Metropolis phase, it is going to introduce various changes which are going to make it more abstraction and privacy friendly. One of those changes is the introduction of “Zero-Knowledge Succinct Non-Interactive Argument of Knowledge” aka Zk-Snarks. Zk-Snarks runs on the idea of zero knowledge proofs.

In this article, we are going to go through the idea of zero knowledge proofs and its application in the blockchain technology.

What are zkSNARKs: Spooky Moon Math

Zero Knowledge proofs came about in 1980's thanks to the work of MIT researchers [Shafi Goldwasser](https://en.wikipedia.org/wiki/Shafi_Goldwasser) (https://en.wikipedia.org/wiki/Shafi_Goldwasser), [Silvio Micali](https://en.wikipedia.org/wiki/Silvio_Micali) (https://en.wikipedia.org/wiki/Silvio_Micali) and [Charles Rackoff](https://en.wikipedia.org/wiki/Charles_Rackoff). (https://en.wikipedia.org/wiki/Charles_Rackoff) They were working on problems related to interactive proof systems, where a Prover exchanges messages with a Verifier (more on provers and verifiers later) to convince them that they have a knowledge of a certain proof without declaring what that knowledge is.

Before they made their landmark discovery, most proof systems were based on the “soundness” properties of the proof system. It was always assumed that the “prover” could be the malicious one in any scenario wherein they will try to fool the verifier. These 3 researchers flipped the idea on its head by questioning the morality of the verifier instead of the prover. The question they asked was, how can anyone know for sure that the verifier won’t leak the knowledge and there were also concerns raised as to the amount of knowledge about the prover that the verifier will get to know during the process of verification.

There are various real world consequences of this conundrum and one of the most famous ones have to do with password protection. Suppose you want to login to a website using a password. The standard protocol is that the client (you) will write in their password and send it to the server, the server will then hash the password and equate it to the hash that they have stored in their system. If the values match up, then you can enter the system.

Can you see the huge flaw in this system right?

The server has the plaintext version of your password, and your privacy is at the mercy of the server (the verifier in this scenario). If the server gets compromised or attacked, then your password will be with the malicious party and the consequences could be dire. In order to counter these scenarios, zero knowledge proofs are absolutely essential and path breaking in every sense.

There are two parties when it comes to a zero knowledge proof (as stated above), the prover and the verifier. Zero knowledge states that a prover can prove to the verifier that they possess a certain knowledge without telling them what that knowledge actually is

Properties of a zero knowledge proof

For a ZKP to work it needs to satisfy certain parameters:

- **Completeness:** If the statement is true then an honest verifier can be

EARN CRYPTO (/REGISTER/)

convinced of it by an honest prover.

- **Soundness:** If the prover is dishonest, they can't convince the verifier of the soundness of the statement by lying.
- **Zero-Knowledge:** If the statement is true, the verifier will have no idea what the statement actually is.

So now that we have a basic idea of what a zero-knowledge proof is, let's check out some examples of it before we dive deep into zk-snarks and its application in [the blockchain](https://blockgeeks.com/guides/what-is-blockchain-technology/) (<https://blockgeeks.com/guides/what-is-blockchain-technology/>).

Case #1 Alibaba's Cave

In this example, the prover (P) is saying to the verifier(V) that they know the password of the secret door at the back of the cave and they want to prove it to the verifier without actually telling them the password.

So this is what it looks like:

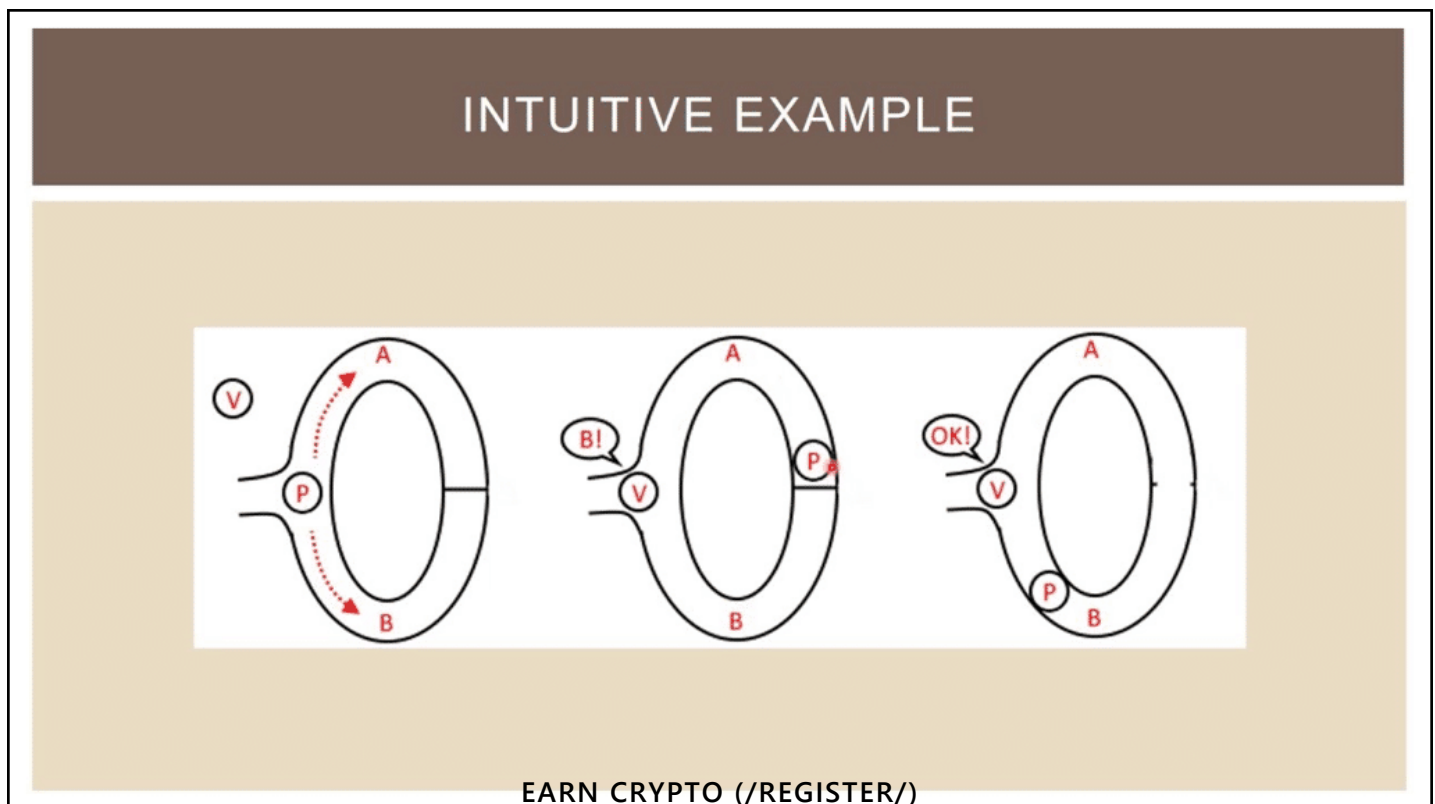


Image courtesy: Scott Twombly (YouTube channel)

The Prover goes down any of the paths A and B, suppose they initially decide to go through path A and reach the secret door at the back. When they do so, the verifier V comes in at the entrance, with no knowledge of which path the prover actually took and declares that they want to see the prover appear from path B.

In the diagram, as you can see, the prover does indeed appear in path B. But what if this was dumb luck? What if the prover didn't know the pass code, and took the path B, was stuck at the door and by sheer fortune, the verifier told him to appear from path B, the one they were originally on anyway?

So, to test the validity, the experiment is done multiple times. If the prover can appear at the correct path every single time, it proves to the verifier that the prover indeed knows the password even though the verifier doesn't know what the password actually is.

Let's see how the three properties of zero knowledge are satisfied in this example:

- **Completeness:** Since the statement was true, the honest prover convinced the honest verifier.
- **Soundness:** If the prover was dishonest, they couldn't have fooled the verifier because the test was done multiple times. Eventually, the prover's luck had to run out.
- **Zero-Knowledge:** The verifier never knew what the password was, but was convinced that the prover had possession of it.

Case #2 Finding Waldo

Remember finding Waldo?

Of course, you do, you must have seen it somewhere, either in real life or online.

For those who don't know, Finding Waldo is a game where you have to find

"Waldo" among a sea of people. It is a simple "Spot the guy" game. Just to give you a basic idea, this is what the game looks like:



Image courtesy: Youtube (IntoConnection)

And the idea is to find Waldo who looks like this:



EARN CRYPTO (/REGISTER/)

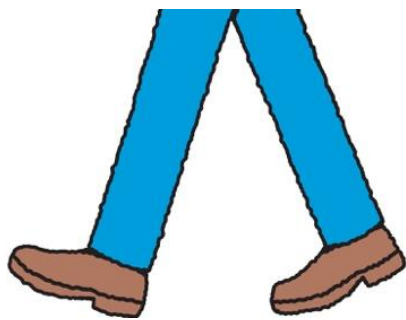


Image courtesy: Pinterest

Seems pretty straightforward right? Find this guy among the sea of other people that you see in the game. Ok, so where does the concept of Zero Knowledge come in here? Imagine there are two people Anna and Carl. Anna tells Carl that she knows where Wally is but she doesn't want to show him where exactly he is. So, how can she prove to him that she has found Wally without showing his exact position?

There was, an interesting paper by Naor, Naor and Reingold which shows two Zero Knowledge solutions to this problem. There is a "Mid-Tech Solution" and a "Low-Tech Solution". Let's discuss both of them.

Mid-Tech Solution

The reason why this solution is "mid-tech" is because our prover and verifier need access to a photocopy machine to make this work. So this is how it goes. First, Anna and Carl would make a photocopy of the original game. Then Anna, whilst making sure that Carl isn't looking, will cut out Waldo from the photocopy and then destroy the leftovers. After that she can show the Waldo cut out to Carl and prove that she did know where Waldo was after all without pinpointing his exact location to Carl.

There are problems with this solution. While it does fulfill the "Zero Knowledge" criteria, it doesn't fulfill the "Soundness" criteria. There are many ways that Anna could have cheated here. She could have had a random Waldo cut out with her

EARN CRYPTO (/REGISTER/)

from the very beginning and could have just shown it to Carl without actually knowing where Waldo was. So what is the solution to this?

The solution to this is meticulous and careful testing. Firstly, Anna and Carl will take a photocopy of the game. Then Carl will draw a distinctive pattern at the back of the photocopy. After that, Carl will escort Anna to a room where she will be isolated and have no chance of cheating whatsoever. If Anna comes out with a cutout of Waldo, then Carl can be convinced that she actually knew where Waldo was without revealing the solution. They can repeat this experiment multiple times and Carl can compare the different cutouts of Waldo to be even further sure about the validity of Anna's claim.

Low-Tech Solution

This solution required very basic equipment. The idea is simple. Get a huge cardboard, one that is twice the size of the game and cut out a small rectangle on it. Now, when Carl isn't looking, Anna can move the cardboard on the game in such a way that the rectangle is directly on top of Waldo. Now, she can tell Carl to have a look and this is what he will see:



Figure 1: Bob's View

Image Courtesy: Applied Kid Cryptography by Naor And Reingold

So, while Carl may get a very basic idea of where Waldo actually can be, he doesn't know the exact location. Anna has hence proved to Carl that she knows where Waldo is without pinpointing his exact location.

Case #3: Sudoku

Another great application of zero knowledge is in Sudoku. For those who don't know, Sudoku is a Japanese puzzle where you get a 9X9 table which looks something like this:

	9			8		4		
		2		4	1			5
3							6	
	1							
7	6			2			1	9
							8	
	2							8
5			2	9		3		
		4		5			2	

Image courtesy: Computational Complexity Blog.

The idea is to fill up every row, every column and every 3X3 block with numbers from 1-9 and no number should repeat itself. So, the solution for the puzzle above looks like this:

1	9	7	6	8	5	4	3	2
6	8	2	3	4	1	7	9	5
3	4	5	9	7	2	8	6	1
4	1	8	5	6	9	2	7	3
7	6	3	8	2	4	5	1	9
2	5	9	7	1	3	6	8	4
9	2	6	4	3	7	1	5	8
5	7	1	2	9	8	3	4	6
8	3	4	1	5	6	9	2	7

EARN CRYPTO (/REGISTER/)

Image courtesy: Computational Complexity Blog.

As you can see, every row, column, and 3X3 block are unique and not a single number has been repeated. Let's go back to our old friends Anna and Carl. Anna has found the solution to the Sudoku puzzle and Carl, skeptic that he is doesn't believe her and wants Anna to prove that she does indeed know the solution. Anna wants to prove her honesty, but at the same time, she doesn't want Carl to know the exact solution of the puzzle. How will she go about it? Anna is going to use Zero Knowledge to prove the validity of her claim.

Firstly, Carl will run the Sudoku solution through a computer program which has been verified, to be honest and the program will run the numbers through a randomly chosen substitution cipher. Say, for this particular problem the cipher that the program has chosen is this:

Original Number	Substituted number
1	2
2	8
3	6
4	5
5	4
6	9
7	1
8	7
9	3

The chosen program and cipher is such that each *digit* has the same chance of being transmuted into its substitution as any other number. Basically, 1 has as much chance of being transmuted as 3 and 4 have as much chance of being transmuted as 9 and so on and so forth. So, using this cipher gives us the following solution to the puzzle above:

2	3	1	9	7	4	5	6	8
9	7	8	6	5	2	1	3	4
6	5	4	3	1	8	7	9	2

EARN CRYPTO (/REGISTER/)

5	2	7	4	9	3	8	1	6
1	9	6	7	8	5	4	2	3
8	4	3	1	2	6	9	7	5
3	8	9	5	6	1	2	4	7
4	1	2	8	3	7	6	5	9
7	6	5	2	4	9	3	8	1

Image courtesy: Computational Complexity Blog.

Anna gets the transmuted solution now, keep in mind that Carl still doesn't know what the original solution was and he doesn't possess the transmuted solution either. So, what Anna does now is that she hides all the numbers in the puzzle by using a "lockbox mechanism", basically Carl won't be able to see any of the numbers and will see an empty 9X9 grid in front of him.

Carl now has 28 choices in front of him:

- Reveal a row.
- Reveal a column.
- Reveal a 3X3 box.
- Reveal the transmuted version of the original puzzle.

Suppose Carl wants to know what the third row looks like:

6	5	4	3	1	8	7	9	2

EARN CRYPTO (/REGISTER/)



Image courtesy: Computational Complexity Blog.

This is what he will see. Carl will see that every number in the row is unique and since every possible number in the original solution had the same probability of being transmuted via the cipher, Carl will have no clue as to what the original solution is

Now suppose, Carl decides to take the last option and wants to see what the original puzzle in looks like when transmuted:

	3			7		5		
		8		5	2			4
6							9	
	2							
1	9			8			2	3
							7	
	8							7
4			8	3		6		
		5		4			8	

Image courtesy: Computational Complexity Blog.

Once again, since the cipher was chosen at random and all the numbers have the same probability of being transmuted, Carl will have no idea what the original solution is. Carl can now go through all 28 of his choices and eventually he will be satisfied with the validity of Anna's statement.

Why?

Because, if Anna was indeed cheating, there is no way that she could have found a cipher to give unique solutions for all 28 of Carl's choices. If Carl just chose one option, Anna's chances of getting away with cheating are $27/28$. BUT if Carl chose to do random test multiple times, suppose he chooses to test it 150 times, Anna's choice of getting away with cheating drops down to $(27/28)^{150}$ which is $< 0.5\%$.

EARN CRYPTO (/REGISTER/)

So, let's check out the zero knowledge properties of this scenario:

- **Completeness:** The cipher program being used has been verified, to be honest, and both Anna and Carl are following protocol.
- **Soundness:** If Carl does random tests 150 times, Anna's chances of getting away with cheating is $< 0.5\%$.
- **Zero-Knowledge:** Anna never had to reveal to Carl what the original solution was.

Proof vs Proof Of Statements

Now that we know the theoretical aspects of zero knowledge proofs and its application in various examples, what is its practical application in blockchain? Why is everyone raving about [Zcash](https://blockgeeks.com/guides/zcash/) (<https://blockgeeks.com/guides/zcash/>) for implementing **ZKP** (zero knowledge proofs) and why is everyone excited about Ethereum doing the same? Before we expand on that, it is important to know one more important theoretical concept.

What exactly are we proving by using ZKP? In a broad spectrum, there are two statements that you can prove by using ZKP. Proofs aka facts and proof of knowledge.

- **Proofs:** These are the intrinsic truths about the universe that you may want to prove via ZKP. Eg. "number X belongs to a group Y".
- **Proof of knowledge:** You may also want to prove that you have knowledge of a particular idea without revealing what that particular knowledge is. As can be seen in the examples of Sudoku, Waldo and Alibaba's cave given above.

It is important to note the difference between these two because they are completely different. In the cryptocurrency world, we are mostly focused around "proof of knowledge". One of the most important breakthroughs in proving

EARN CRYPTO (REGISTER)

proof of knowledge via zero knowledge proof came when Claus-Peter Schnorr in the 1980s came up with the Schnorr identification protocol. This protocol lays the basics of modern key signature cryptography and displays how Zero-knowledge can be seamlessly integrated into modern cryptographical practices.

The Schnorr Identification Protocol

To understand what the Schnorr Identification is about let's bring back our old friends Anna and Carl. Anna has announced to the world that she has a public key and can accept and receive information through it. Carl, always the skeptic, thinks that Anna is lying. The only way that Anna can prove her honesty is by showing her private key to Carl, but she doesn't want to reveal her private key.

So, how will Anna reveal her knowledge of her private key without revealing it? This is where the Schnorr protocol comes in. Before we even begin to understand how the protocol works, there are certain parameters that you need to know:

- p = Any prime number.
- q = factor of $p-1$.
- " a " such that $a^q = 1 \pmod p$.

Now keep in mind, in the Schnorr protocol, these 3 variables are global. Meaning anyone has knowledge of what these 3 variables for a particular scenario are.

Now we come to the two keys, the secret private key that we will call " s " and the public key that we will call " v ".

s can be any value as long as $0 < s < q$.

$v = a^s \pmod q$.

The public key " v " will be global and public knowledge along with p, q and a .

EARN CRYPTO (/REGISTER/)

However, ONLY Anna will have the knowledge of what "s" is, because that is her private key.

So, now that we have defined the variable, let's see how the information exchange and the validity of Anna's statement can work WITHOUT her revealing what the private key is.

Anna signs and sends an encrypted message

Suppose Anna wants to send a message "M" to Carl encoded with her private key. How will she do it if she were to follow Schnorr's protocol?

Firstly, she will choose a random number "r" such that $0 < r < q$.

Now she will compute a value x such that:

$$X = a^r \bmod p.$$

Now that she has computed the value of X, she is going to concatenate this with the original message. What is concatenation? Suppose we have two strings "hello" and "world". If we concatenate these two, then we will get "hello world". Concatenation basically means adding two strings and making it one.

So, she is going to concatenate M and X to get $M||X$. and she is going to store the hash of this value in e.

Basically, $e = H(M||X)$ where $H()$ is the hash function.

Finally, when all this is done, she will do one final computation. She is going to get a value "y" such that:

$$y = (r + s \cdot e) \bmod q$$

Now that all the computations are over, she is going to send the following pieces of information to Carl:

- The message "M".

EARN CRYPTO (/REGISTER/)

- The signatures e and y.

- The signatures e and y .

Carl receives the message and verifies Anna's proof of knowledge

Now Carl has received the following pieces of information from Anna: The message (M) and the signatures (e and y).

Along with that, he has the following pieces of information that is known publicly to everyone:

- Anna's public key " v ".
- The prime number that Anna chose " p ".
- " q " which is the factor of " $p-1$ " which Anna chose.
- And the " a " such that $a^q = 1 \pmod p$, this also Anna chose.

Now, Carl will have to compute X' such that:

$$X' = a^y * v^e \pmod p.$$

Now let's do some simple substitution:

We know that $v = a^{-s}$, let's substitute that in the equation above and we get:

- $X' = a^y * a^{-se} = a^{(y-se)}$.
- Now we also know that $y = r + s^*e$.
- Which means: $r = y - s^*e$.

Let's substitute this value in the equation above:

- We get: $X' = a^r$.
- As we have already seen above: $X = a^r$.
- So technically: $X = X'$.

But Carl doesn't know the value of " X " because he never received that value. All

EARN CRYPTO (/REGISTER/)

that he received are the following: The message M , the signatures (e and y) and the host of public variables (public key " v ", p , q , and a).

He never received " X " but he knows that if Anna is speaking the truth then X' has to be equal to X . But, he does know the value of e and the message M .

So he is going to solve for e by doing the following:

$$e = H(M || X').$$

Note that earlier we solved for e by doing: $H(M || X)$.

So, by that logic, if the two values of e come up to be the same then that means $X = X'$.

This also means that Anna did indeed have the private key all along and she was not lying.

So, let's run this entire scenario through the three properties of zero knowledge proofs:

- **Completeness:** Carl was convinced of Anna's honesty because at the end $X = X'$.
- **Soundness:** The plan was sound because the only way Anna could have proved her honesty was by using her private key. She couldn't have lied about having the private key.
- **Zero Knowledge:** Carl never found out what Anna's private key was.

Schnorr's protocol gives a very real world cryptographical application of zero knowledge proofs.

How to make zero knowledge proofs non-interactive?

With earlier zero-knowledge verification systems there was one big problem. For

EARN.CRYPTO (/REGISTER/)

it to work, the prover and the verifier had to be online at the same time. In other words, the process was “interactive”. This made the entire system inefficient and almost impossible to scale up. The verifiers couldn’t possibly be online at the same time as provers all the time? There needed to be a system to make this more efficient.

In 1986, Fiat and Shamir invented the Fiat-Shamir heuristic and successfully changed the interactive zero-knowledge proof to non-interactive zero knowledge proof. This helped the entire protocol work without any interaction. The procedure behind it is very simple.

So, to give you an example, this is how zero knowledge proofs used to work before Fiat and Shamir.

Let’s prove this using simple discrete logarithms.

- Anna wants to prove to Carl that she knows a value x such that $y = g^x$ to a base g .
- Anna picks a random value v from a set of values Z , and computes $t = g^v$ and sends t to Carl.
- Carl picks a random value c from the set Z and sends it to Anna.
- Anna computes $r = v - c \cdot x$ and returns r to Carl.
- Carl checks if $t = g^r \cdot y^c$ holds or not (since $r = v - c \cdot x$, $y = g^x$ and by simple substitution, $g^{(v - c \cdot x)} \cdot g^{c \cdot x} = g^v = t$).
- Carl doesn’t know the value of x , by merely checking if $t = g^r \cdot y^c$ he can verify that Anna does indeed know the value of x .

Now while the above interaction is zero-knowledge, the problem with this is that Anna and Carl need to be online and exchanging values for it to work.

EARN CRYPTO (/REGISTER/)

How can Anna prove to Carl that she has knowledge of something without Carl

How can Anna prove to Carl that she has knowledge of something without Carl being online? She can do so by using a simple cryptographic hash function, as Fiat and Shamir theorized.

Let's look how the example above would work in a non-interactive way:

- Anna wants to prove to Carl that she knows a value x such that $y = g^x$ to a base g .
- Anna picks a random value v from a set of values Z , and computes $t = g^v$.
- Anna computes $c = H(g, y, t)$ where $H()$ is a hash function.
- Anna computes $r = v - c \cdot x$.
- Carl or anyone can then check if $t = g^r \cdot y^c$.

So, as you can see, zero knowledge proofs were made noninteractive. And this was what laid the foundations for Zk-Snarks.

What is the use of Zk-Snarks?

Zk-Snarks stands for "Zero-Knowledge Succinct Non-Interactive Argument of Knowledge". Its use in modern blockchain technology is immense. To understand its application, it is important to know how a smart contract works. [A smart contract](https://blockgeeks.com/guides/smart-contracts/) (https://blockgeeks.com/guides/smart-contracts/) is basically an escrow of funds which gets activated once a particular function is done.

Eg. Anna puts 100 ETH in a smart contract that she gets into with Carl. Carl has to do a particular task, on the completion of which, Carl will get the 100 ETH from the smart contract.

This gets complicated when the tasks that Carl has to do are multi layered and confidential. Suppose you have entered a smart contract with Anna. Now, you will only get the payment if you do A, B and C. What if you don't want to reveal the details of A, B, and C because they are confidential to your company and you

EARN CRYPTO (/REGISTER/)

don't want any competitors to know what you have to do?

What Zk-Snarks does is that it proves that those steps have been taken in the smart contract without revealing what those steps actually are. It is very useful in protecting you and your company's privacy. It can just reveal part of the process without showing the whole process itself and prove that you are being honest about your claims.

How do ZkSnarks work?

A Zk-Snark consists of 3 algorithms: G, P and V.

G is a key generator that takes an input "lambda" (which must be kept confidential and shouldn't be revealed under any circumstances) and a program C. It then proceeds to generate two publicly available keys, a proving key pk, and a verification key vk. These keys are both public and available to any of the concerned parties.

P is the prover who is going to use 3 items as input. The proving key pk, the random input x, which is publicly available, and the private statement that they want to prove the knowledge of without revealing what it actually is. Let's call that private statement "w". The P algorithm generates a proof prf such that: $\text{prf} = P(\text{pk}, x, w)$.

The verifier algorithm V has basically returned a boolean variable. A Boolean variable has only two choices, it can be TRUE or it can be FALSE. So, the verifier takes in the verifying key, public input x and proof prf as input such as:

$V(\text{vk}, x, \text{prf})$

..and returns TRUE if the prover is correct and false otherwise.

Now, about the parameter lambda. The value of the "Lambda" must be kept confidential because then anyone can use it to generate fake proofs. These fake proofs will return a value of TRUE regardless of whether the prover actually has

prover will return a value of true regardless of whether the prover actually has knowledge of private statement "w" or not.

Functionality of ZkSnarks

For showing the functionality of a Zk-Snark we are going to use the same example function that Christian Lundkvist used in his article for Consensys. This is what the example program looks like:

```
function C(x, w)
{
return ( sha256(w) == x );
}
```

Basically, the function C takes in 2 values as input, a public hash value "x" and the secret statement that needs to be verified "w". If the SHA-256 hash value of w equals "x" then the function returns TRUE otherwise it returns FALSE. (SHA-256 is the hash function that is used in Bitcoin).

Let's bring back our old friends Anna and Carl for this example. Anna being the prover and Carl the skeptic is the verifier.

The first thing that Carl, as the verifier, has to do is to generate the proving and verifying key using the generator G. For this, Carl needs to generate the random value "lambda". As stated above, however, he needs to be super careful with Lambda because he can't let Anna know its value to stop her from creating fake proofs.

Anyway, this is what that will look like:

$G(C, \text{lambda}) = (\text{pk}, \text{vk}).$

Now that the two keys are generated, Anna needs to prove the validity of the

EARN CRYPTO & REGISTER

statement by generating the proof. She is going to generate the proof using the proving algorithm P . She is going to prove that she knows the secret value “ w ” which hashes (on parsing through SHA-256) to give the output x . So, the proving algorithm for proof generation looks like this:

$$\text{prf} = P(\text{pk}, x, w).$$

Now that she has generated the proof “ prf ”, she is going to give the value to Carl who is finally going to run the verification algorithm of Zk-Snarks:

This is what that will look like:

$$V(\text{vk}, x, \text{prf}).$$

Here, vk is the verifying key and x is the known hash value and prf is the proof that he has gotten from Anna. If this algorithm returns TRUE then this means that Anna was honest and she indeed had the secret value “ w ”. If it returns FALSE then this means that Anna was lying about knowing what “ w ” is.

The use of ZkSnarks in cryptocurrency

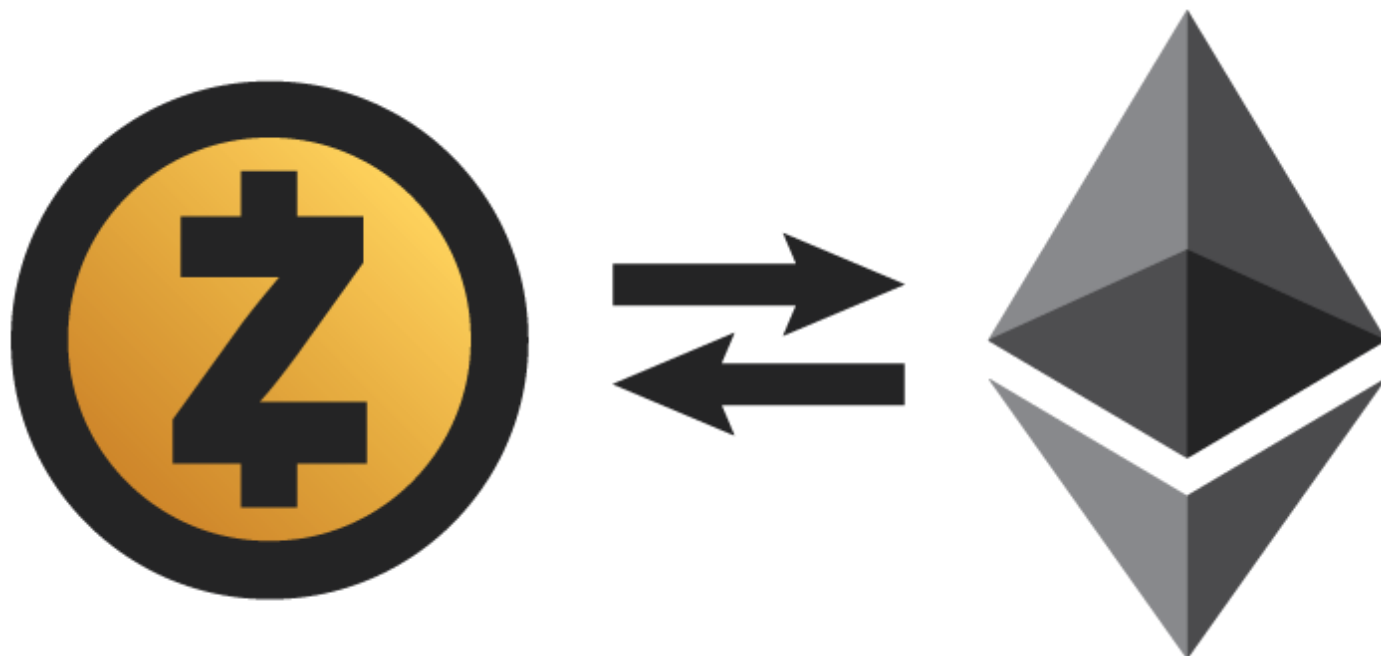


Image Courtesy: Zcash

EARN CRYPTO (/REGISTER/)

Zcash (<https://blockgeeks.com/guides/zcash/>) is a **cryptocurrency** (<https://blockgeeks.com/guides/what-is-cryptocurrency/>) launched by Zerocoin Electric Coin Company on 9th September 2016 and is the first example a cryptocurrency marrying the concepts of blockchain technology with ZkSnarks. It aims to provide completely safe and shielded transaction spaces for its users without revealing details (such as their addresses) to anyone.

Ethereum (<https://blockgeeks.com/guides/what-is-ethereum/>) wants to integrate ZkSnarks as it enters its Metropolis phase and the way that they are planning to do so is by creating an alliance with **Zcash** (<https://blockgeeks.com/guides/zcash/>) which will include the mutual exchange of value. The chief developer of **Zcash**, (<https://blockgeeks.com/guides/zcash/>) Zooko Wilcox, gave a presentation in DevCon2 in Shanghai which explored the future of such an alliance. According to him, there are 3 ways that Z-Cash and by extension, zk-snarks could be integrated with Ethereum.

First method is called Baby Zoe (Zoe = Zcash on Ethereum). It adds a zk-snark pre-compiler on Ethereum and makes a mini **Zcash** (<https://blockgeeks.com/guides/zcash/>) smart contract on Ethereum. The idea is to see whether the Ethereum system can create a zk-snark enabled **DAPP** (<https://blockgeeks.com/guides/dapps-the-decentralized-future/>) on top of its **blockchain** (<https://blockgeeks.com/guides/what-is-blockchain-technology/>).

The Second method is to integrate the Ethereum computability inside the **Zcash blockchain** (<https://blockgeeks.com/guides/zcash/>). As Wilcox puts it, the greatest asset of Ethereum is its computability and people want to see whether they can integrate it on a zk-snark based blockchain like Zcash. Can people create DAPPS on a blockchain made on zero knowledge proofs? That is something that they are waiting to see.

The third and the most exciting aspect is **Project Alchemy**. This is basically the connection and integration of the two blockchains such that one can

connection and interoperation of the two blockchains such that one can seamlessly move between the two. The way that [Zcash plans](#) (<https://blockgeeks.com/guides/zcash/>) to do that is by cloning the BTC Relay. It is an Ethereum script which was written to create a [Bitcoin](#) (<https://blockgeeks.com/guides/what-is-bitcoin-a-step-by-step-guide/>) light client inside Ethereum. The [Zcash](#) (<https://blockgeeks.com/guides/zcash/>) clone will use the same concept to create a [Zcash](#) (<https://blockgeeks.com/guides/zcash/>) light client inside Ethereum.

If this works then we will have the first, decentralized currency system in the world which facilitates the creation of [DAPPS](#) (<https://blockgeeks.com/guides/dapps-the-decentralized-future/>) with zero knowledge ingrained in it.

Looking Ahead

There is no doubt that the introduction of zero knowledge proofs is going to be a huge game changer for [Ethereum](#) (<https://blockgeeks.com/guides/what-is-ethereum/>). In an increasingly open, connected and supervised world, any sort of privacy is welcome. How the integration happens remains to be seen, but going by the theoretical concepts itself, one can't help but get excited.

Like what you read? Give us one like or share it to your friends

624

6

Like what you're reading?

EARN CRYPTO (/REGISTER/)

Join our community and get access to over 50 free video lessons, workshops, and guides like

<https://blockgeeks.com/guides/what-is-zksnarks/>

47/50

this! No credit card needed!

GET STARTED (/REGISTER/)



RELATED GUIDES

Virtual Financial Assets And The Regulation of Blockchain Micro-Loans

(<https://blockgeeks.com/guides/virtual-financial-assets-and-the-regulation-of-blockchain-micro-loans/>)

Sidechains on Top of Bitcoin's Network: RSK & Liquid

(<https://blockgeeks.com/guides/sidechains-on-top-of-bitcoins-network-rsk-liquid/>)

DAML - An open-source ecosystem for building smart contract based distributed applications

(<https://blockgeeks.com/guides/daml-an-open-source-ecosystem-for-building-smart-contract-based-distributed-applications/>)

Working in Blockchain For Non Developers [No Coding Needed]

(<https://blockgeeks.com/guides/working-in-blockchain-for-non-developers-no-coding-needed/>)

What is a Blockchain Operating System? A Deep Dive Guide

(<https://blockgeeks.com/guides/what-is-a-blockchain-operating-system/>)

[\(https://blockgeeks.com/abg/\)](https://blockgeeks.com/abg/)

Have questions?

We have built an incredible community of blockchain enthusiasts from every corner of the industry. If you have questions, we have answers!

ASK COMMUNITY

Course library

[\(https://courses.blockgeeks.com/course-library/\)](https://courses.blockgeeks.com/course-library/)

Become a Mentor

[\(https://blockgeeks.com/mentor/\)](https://blockgeeks.com/mentor/)

Advertise with us

Guides

[\(https://blockgeeks.com/guides/\)](https://blockgeeks.com/guides/)

[\(https://blockgeeks.com/blockchain-partnership/\)](https://blockgeeks.com/blockchain-partnership/)

EARN CRYPTO (/REGISTER/)

[Write for us](#)[Infographics](#)[\(https://blockgeeks.com/guest-post-submissions/\)](https://blockgeeks.com/guest-post-submissions/)[\(https://blockgeeks.com/blockchain-infographics/\)](https://blockgeeks.com/blockchain-infographics/)[FAQ](#)[Support](#)[\(https://courses.blockgeeks.com/faq/\)](https://courses.blockgeeks.com/faq/)<https://blockgeeks.freshdesk.com/support/home>[Privacy](#)[Terms](#)[\(https://blockgeeks.com/policy/rms/\)](https://blockgeeks.com/policy/rms/)[\(https://blockgeeks.com/terms/\)](https://blockgeeks.com/terms/)[Submit a Course](#)[\(https://blockgeeks.com/contact-us/\)](https://blockgeeks.com/contact-us/)[\(https://blockgeeks.com/what-is-zk-snarks/\)](https://blockgeeks.com/what-is-zk-snarks/)**Blockgeeks**<https://blockgeeks.com>

© 2020 Blockgeeks

EARN CRYPTO (/REGISTER/)