HyperSpace Security, Inc.'s Blog Post #3:
Polynomial Based Secret Sharing (and Why it is Broken)

Welcome Again

Your interest in HyperSpace Security, Inc.'s blog is appreciated!

Some background information follows:

HyperSpace offers a radical breakthrough in data security by eliminating the need to store a cryptographic master key, which is vulnerable to today's computers and social engineering. With HyperSpace's technology, master keys are never persistently stored and therefore cannot be hacked or stolen resulting in a "keyless" system. The technology not only protects against attacks based on quantum computing, it also provides unique benefits by securing payment systems, enhancing enterprise key management, bolstering multi-factor authentication, and securing blockchains and private key infrastructure.

In this forum, we will discuss issues related to data security, advances in mathematics and quantum computing, vulnerabilities of blockchains and other security technologies to attack, hacks that have harmed real people in the real world, and more. Some of these posts will get a little deep into some math. That said, we will try to make the content understandable to anyone interested in these fields.

Please feel free to provide comments and feedback. We value your input!

Today's Topic

The topic of today's blog post is polynomial based secret sharing, for example using Shamir's algorithm (https://en.wikipedia.org/wiki/Shamir%27s_Secret_Sharing), and why such approaches are not cryptographically secure. This post is going to be more math-heavy than the prior ones, but that cannot be avoided when dealing with this topic.

What is Polynomial Based Secret Sharing?

Per Wikipedia:

> Shamir's Secret Sharing is an algorithm in cryptography created by Adi Shamir. It is a form of secret sharing, where a secret is divided into parts, giving each participant its own unique part, where some of the parts or all of them are needed in order to reconstruct the secret.

> Counting on all participants to combine the secret might be impractical, and therefore sometimes the threshold scheme is used where any k of the parts are sufficient to reconstruct the original secret.

…

Example:

> Suppose that our secret is 1234 *(S = 1234)*.
>
> We wish to divide the secret into 6 parts, where any subset of 3 parts is sufficient to reconstruct the secret. At random we obtain two numbers: 166 and 94.
>
> Our polynomial to produce secret shares (points) is therefore:
>
> f(x) = 1234 +166x +94x^2.
>
> We construct 6 points $D_{x-1} = (x,f(x))$ from the polynomial:
>
> $D_0 = (1,1494); D_1 = (2, 1942); D_2 = (3, 2578); D_3 = (4, 3402); D_4 = (5, 4414); D_5 = (6, 5614)$
>
> We give each participant a different single point (both *x and f(x)*). Because we use D_{x-1} instead of D_x the points start from *(1, f(1))* and not *(0, f(0))*. This is necessary because *f(0)* is the secret.
>
> In order to reconstruct the secret any 3 points will be enough.

OK, that probably makes as much sense to most people as if it was written in the original Klingon (that is a Star Trek reference for you non-Trekkies). Let's simplify:

(1) Pick your secret;
(2) Decide how many people or devices will have to come together to get back your secret – call that number *k*;
(3) Construct a polynomial of order *k-1;*
(4) Pick a bunch of points on the polynomial – call the number of points *n*;
(5) Issue those points to *n* people/devices; and
(6) Any *k* of those *n* people/devices can recreate the secret.

Wow, still looks like the original Klingon. Let's really break it down:

(1) You have a secret;
(2) You have a lot of friends;
(3) You do not want to bug all of them every time you need the secret back;
(4) Do this whole polynomial math thing; and
(5) Now you only have to bug a few of them to get back your secret, but none of them by themselves has the secret.

Sounds great, but there is a problem.

Geometric Attack

Turns out that any information from one of your friends gives up some information about the secret because it tells them something about the polynomial. One approach to extracting that information is called a "geometric attack." Back to Wikipedia:

> [T]his attack exploits the fact that we know the order of the polynomial and so gain insight into the paths it may take between known points this reduces possible values of unknown points since it must lie on a smooth curve.

The existence of the geometric attack was why I abandoned a polynomial based approach to secret sharing decades ago.

Other attacks exist including variations of the geometric attack, but those are beyond the scope of this blog post.

The Defense

Professor Adi Shamir, PhD, is pretty smart. He is probably one of the top hundred mathematicians who has ever lived. After all, Professor Shamir is the "S" in "RSA" (Rivest, Sharmir, and Adleman – the creators of the RSA protocol used to protect pretty much every so-called secure online transaction). So of course he came up with a defense: Use finite field arithmetic. Oh boy, we're back to Klingon. What the heck is that?

Imagine you are only allowed to use numbers from 1 to 11. What happens if you need a 17? You just subtract 11 from it, resulting in 6. Then you use the 6. How about needing a 25? Subtract 11 twice, resulting in 3. The period of this field is 11.

Polynomial based secret sharing works just fine with finite field arithmetic. That approach also was believed to block the geometric attack.

The Counter

Countering this defense involves math not dissimilar from that used to factor large composite numbers in a finite field. Certain applications of prime number spirals can be used to do so. See our blog post #1 for some information about those spirals. I won't get into the details, but application of such math can counter the defense described above.

Another approach also is coming to the forefront: quantum computing. See our blog post #2 for some information. Quantum computing is particularly good at dealing with finite field arithmetic. For example, the key to Shor's algorithm is the ability to extract a periodic portion from a problem using a quantum Fourier transform. Recall my note of a "period" when

explaining a finite field. See https://arxiv.org/pdf/quant-ph/9906059.pdf and http://www-bcf.usc.edu/~tbrun/Course/lecture13.pdf for some explanation.

Applying either of these approaches shatters the "defense," leaving polynomial based secret sharing subject to the geometric attack.

We Are Immune

Our Shadowing technology achieves the goals of polynomial based secret sharing described above and other goals as well. In addition, none of the attacks discussed here works on our patented Shadowing technology. We designed it that way.

About the Author

Dane C. Butzer is the inventor of HyperSpace Security, Inc.'s Key Shadowing technology. He holds a Bachelor of Science in Electrical Engineering, a Master of Science, and a Juris Doctor, all from the Ohio State University. Mr. Butzer also has been published in Applied Optics and in the Appendix of the first edition of Bruce Schneier's seminal work *Applied Cryptography*. He has held several patents including U.S. Patent No. 9,634,836 for "Key Shadowing."

For More Information

Questions can be posted on this blog. We will try to answer those questions. You can also follow us on Twitter @keyshadowing or contact the author at dbutzer@keyshadowing.com.