Dissecting Schnorr's Digital Signature Scheme

Scott Baker

ABSTRACT

I was always curious about how personal information was stored and delivered through the Internet. In the real world we sign a receipt to verify that we are the people using our credit cards, but in the digital world, how can our signature become valid? I searched for the information and discovered that digital signatures exist via the Internet, and data encryption standard, known as the DES, is the standard the United States employ. The DES is the combination of El Gamal and Schnorr's digital signature. Through this research paper I explored the Schnorr's digital signature mainly, by using the MATHLAB and the math model tools. The bases of the key generated by the digital signature have its principal in number theory, so I also learned about the modular rules.

INTRODUCTION

What is cryptography?

Cryptography dates back to the ancient times when simple methods were used to encrypt messages. Scytale and Polybius were simple devices used to encrypt messages for military services to deliver secret messages between troops. The most known encryption method is the Caesar cipher by Julius Caesar in 100 BC, which uses a substitution cipher. In a substitution cipher, each letter is shifted to substitute another letter. For instance, shifted by 5 places implies that 'A' is replaced with 'F'. However, due to the frequency of letters used in the language, this method was easily broken.

Later in the 19th century, electric encryption key was introduced by Hebern which was called the Hebern rotor machine. Using a single rotor, the key was in the rotating disk and the key encoded a substitution table. Therefore, pressing the keyboard resulted the output of the cipher text. However, as the Caesar cipher, the first electronic contraption was broken by letter frequencies. As a result, in order to spread out the letter frequency, the Germans used the Enigma machine through the end of World War I and heavily during World War II. There were 3 to 4 motors for each machine and the rate the motor rotates resulted the cipher text. Therefore, the initial setting of the keyboard was required to decode the encrypted message.

However, Enigma machine was decrypted by Poland's cipher bureau. Throughout history, encryption was developed mainly to secure military information and deliver war tactics. Broken encryption methods resulted in failure of certain battles.

Therefore, it was important to not only secure their messages but also to decrypt the enemies messages. The field of cryptography was enlarged by the two world wars and major breakthroughs were made during this time.

Why is Cryptography Important?

In the later 1900s, the use of cryptography became widespread with commercial usage. Businesses were trying to secure their information from competitors and customers were demanding some kind of encryption methods. One of the innovative company, IBM formed a crypto group and designed a cipher, Lucifer. Now IBM possessed a good crypto support. Later, known for a block cipher, Lucifer was adopted as the DES(Data Encryption Standard). However, the computing power increased geometrically that eventually broke the key through brute force attack. The brute force attack led to increase the complexity of the secret key to slow the time it takes to obtain a possible plain message.

Most of the cryptography remains partly hidden. Since the value of encryption method is determined through how secure the key remains, major discoveries are not exposed publicly. As a result, the discoveries of new methods were not revealed to public and were primarily operated in encryption related institutions.

Digital signatures can be seen as handwritten signatures in electronic format. It refers to any electronic data that carries

the purpose of a signature, but not all electronic signatures use digital signatures. It demonstrates the authenticity of the message or document. A valid digital signature will provide the recipient with the certainty that the message was sent by the person he or she wanted. It is commonly used in software distribution, financial transactions, and places to detect forgery or tampering.

EL GAMAL DIGITAL SIGNATURE

Above diverse digital signature mechanisms, Schnorr's Signature is one of the modern digital signatures using asymmetric cryptography. It also provides a non- repudiation, which means that the signer cannot successfully claim that they did not signed a signature. Non-repudiation schemes have time stamps. Therefore, even when the private key is exposed, the time stamp makes the digital signature valid. Also, Schnorr's signature is based on discrete logarithm problems which base comes from the El Gamal Digital Signature.

El Gamal digital signature scheme is one of the famous digital signatures based on difficulty of computing discrete logarithms. Taher ElGamal described it. It employs asymmetric key encryption and its algorithms are used in the free GNU Privacy Guard software. Its security depends upon the difficulty of a certain problem.

In order to generate a key pair (public key - a secret key), a first chosen large prime integer P and large integer G, where G <R. The sender of the signed document (Alice) and receiver (Bob) use in the calculations similar large integers P (~ 10308 or ~ 21024) and G (~ 10154 or ~ 2512), which are not secret.

The algorithm is as follows:

1. Alice chooses random integer X, 1< X ≤ (P-1), and compute

$$Y = G^x \bmod P.$$

The number Y is the public key used to verify the signature of the sender. The number of Y is open to all potential recipients of transferred documents. The number X is the sender's private key for signing documents and should be kept secret.

2. Alice hashes a message M using the hash function *h*:

$$m=h(M),\ 1<m<(P-1),$$

and generates random integer K, 1<K<(P-1) such that *K* and *(P-1)* are coprime.

3. Alice computes an integer *a* by the formula

$$a = G^K \bmod P.$$

4. Alice computes integer *b*, solving the equation (see e.g.[9]):

$$m=X*a+K*b \pmod{(P-1)}$$

with help of extended Euclid's algorithm.

The pair of numbers (*a*,*b*) form digital signature S

$$S=(a,b)$$

affixed to the document M.

A triple (M, a, b) is sent to the recipient, while the pair (X, K) is kept secret.

5. After receiving the signed message (M, a, b) Bob should verify is the signature

S = (a, b) corresponding to the message M.

He first calculates the hash-value of the received message M:

$$m = h(M).$$

6. Then he calculates the value

$$A = Y^a a^b \pmod P$$

and recognizes the message authentic if and only if

$$A = G^m \pmod P.$$

In other words, a receiver checks the equity of ratio

$$Y^a a^b \pmod P = G^m \pmod P.$$

The last equation is satisfied if the signature S = (a, b)

We can demonstrate that the last equation is satisfied if and only the signature is S= (a,b) under the document M that is obtained through using a secret key of X, from which the public key Y was obtained. Therefore, without disclosing the key itself, one can make sure the sender of the message M was the holder of the private key, which is X, and the sender signed the document M.

However, the problem of ElGamal digital signature scheme is in the large size of *P.*

SCHNORR'S DIGITAL SIGNATURE

To decrease the size of the signature Schnorr created new scheme, based on ElGamal. It includes 3 big steps

i. Generating of keys.

ii. Generating of signature.

iii. Verification of signature.

Generating of Keys

Let's consider the 1st step:

Generating of keys for Schnorr signature is performed similar to DSA [3]:

1. Alice selects prime number *p*, which length (as usual) is equal to 1024 bits.
2. Alice selects another prime number *q* such that $\boldsymbol{p}$-1=0 *(mod q)*. It is accepted to choose the size of *q* equals to 160 bits.
3. Alice selects *e1*≠1, such that $\boldsymbol{e_1^q=1}$ *(mod p)* . It can be realized by organizing simple iteration procedure.
4. Alice selects integer number *d*<*q*, as her secret key.
5. Alice calculates $\boldsymbol{e_2 \cdot e_1^{q-d} \bmod P}$.

   So, public Alice's key is *(e1,e2,p,q)* and her secret key is *d*.

Example 1.

Suppose p=88667. Let's factor *p-1* with help of MatLab [5] :

```
>> factor(88666)
ans =        2        43        1031
```

So, we can select q=1031.

To do the next step we should use special function, which has been generated in previous paper (see [3]):

```
function R = modulopower (X, N, M)
X = rem(X, M);
R = 1;
while N > 0
   if rem(N, 2) == 0
      X = rem(X * X, M);
      N = N / 2;
   else
      R = rem(R * X, M);
      N = N - 1;
   end
end
end
```

Remark, that there are several elements that are satisfy equation $\boldsymbol{e_1^q=1}$ *(mod p).*

They can be found by the cycle:

```
>>p=88677; q=1031;i=i0;
>> while modulopower(i,q,p)~=1
i=i+1
   end
```

E.g. if i_0=70300 then e_i=70322.

Let's check it

```
>> modulopower(70322,1031,88667)
ans =  1
```

I.e. $\boldsymbol{e_1^{1031}=1}$ *(mod 88667).*

Let's Alice select *d*=755, so $\boldsymbol{e_2 - 70322^{1031-755} \bmod 88667}$ =13136:

```
>> modulopower(70322,1031-755,88667)
ans =  13136
```

Thus, Alice's set of keys consists of (70322,13136,88667,1031).

Exercise 1.

Suppose p=48731, q=443. It is required:

1. check the condition $\boldsymbol{P-1=0}$ *(mod q);*
2. find arbitrary e_1 near 10000 with help of MATLAB, organizing iteration procedure;
3. select d<443 and e_2 with help of MATLAB.

Solution

1. Let's factor number p using embedded MatLab function *factor:*

   ```
   >> factor(48730)
   ans =       2       5       11 443
   ```

2. Acting as in Example 1, we get

   ```
   p=48731;q=443;i=10000; while modulopower(i,q,p)~=1
   i=i+1
      end
   …
   ```

```
i = 10228
```

So, e_1=10228.

3. Selecting $d=357$ we can calculate $e_2 = e_1^{q-d} mmmmmm\ pp$ with help of MatLab

```
>> e1=10228;e2=modulopower(e1,q-d,p) e2 = 42300
```

Thus, Alice's set of public keys consists of (10228,42300,48731,443).

Generating the Signature

Let's consider the 2nd step

1. Alice select arbitrary number $r \epsilon (1,q)$. It should be performed every time when she is going to transfer a new message.
2. Alice calculates and generates first signature $S1$ with help of hash function–:

   $S_1=h(M|g_1)$ where M is message; a|b denotes concatenation ofexpressions.
3. Alice calculates second signature $S2=(r+d*S1) mod\ q$.
4. Alice transfers $M, S1$ and $S2$.

Example 2

Calculate S1 and S2 for the test message M=1000 using Alice's keys, generating in Example 1.

Solution.

1. Let's use MATLAB embedded function *randi* to find *r*

   >> q=1031;r=randi(q,1,1) ans = 987
2. Then we find g1

   >> r=987;e1=70322;p=88667;g1= modulopower(e1,r,p)

   g1 = 85882
3. To define concatenation of numbers we write simple function for calculating the number of digits of g1:

```
function f=numdigits(x)
n=x; k=1;
if n<10;
   s=0;
else
   s=mod(n,10);
   while n>10;
      h=mod(n,10);
      g=h/10;
      n=(n/10)-g;
      k=k+1;
   end    1
end
k
end
```

Applying this function gives

```
>> numdigits(g1)
k = 5
```

So, concatenation will be done as

```
>> M=1000;M1=M*10^k+g1
M1 = 100085882
```

4. To end this block we use hash function $H(x)= x^2 \ mod\ Z$, where $Z<p$ is integer which has the same digits as p:

```
>> M1=100085882; Z=76543;S1=mod(M1^2,Z)
S1 = 37718
```

5. Calculating S_2

```
>> q=1031;d=755;r=987;S2=mod(r+d*S1,q)
S2 = 826
```

Thus, expression to transfer are *M=1000; S1=37718; S2=826.*

Exercise 2.

Calculate S1 and S2 for the test message M=2000 using Alice's keys, generating in Exercise 1 with the same hash-function as in Example 2.

Solution

1. Let's use MATLAB embedded function *randi* to find *r*

```
>> q1=443;r1=randi(q1,1,1)
r1=63
```

2. Then we find *g1*

```
>> e1=10228;p1=48731;g1= modulopower(e1,r1,p1)
g1 = 15963
```

3. To define concatenation of numbers we write simple function for calculating the number of digits of g1:

```
>> g1=15963; numdigits(g1)
k=5
```

So, concatenations will be done as

```
>> M=2000;M1=M*10^k+g1
```

```
M1= 200015963
```

4. To end this block we use hash function H(x)= x2 mod Z as in Example 2

```
>> M1=200015963; Z=43210;S1=mod(M1^2,Z)
S1 = 12668
```

5. Calculating *S2*

```
>> q=443;d=357;r=63;S2=mod(r+d*S1,q)
S2 = 395
```

Thus, expression to transfer are *M=2000; S1=12668; S2=404.*

Verification of the Signature

Let's consider the last step.

After receiving pocket *(M,S_1,S_2)* from Alice, Bob do the next 1)

1. Calculates V=h(M| $e_1^{S_2} e_2^{S_1} \bmod p$).
2. Checks the validity of $S_2 \bmod p = V \bmod p$. If true then a message is adopted if false then a message is rejected.

Example 3.

Let's check the Alice's authenticity of the signature, generating in Example 2.

1. To calculate we will use the rule $a \cdot b \bmod p = ((a \bmod p) \cdot (b \bmod p)) \bmod p$ [9]. I.e. .

Applying function modulopower, for known values of e_1,e_2,S_1,S_2,p we get

```
>> e1=70322;e2=13136;S1=37718;S2=826;p=88667;
G=mod(modulopower(e1,S2,p)*modulopower(e2,S1,p),p)
G = 85882
```

2. To calculate h(M|G) we act analogously to the Example 2

```
>> numdigits(G); M1=M*10^k+G
k = 5
M1 = 100085882
>> Z=76543;V=mod(M1^2,Z)
V =     37718
```

Since *V=S1 mod p,* so the message is adopted.

Exercise 3.

Check the Alice's authenticity of the signature, generating in Exercise 2.

Solution

1. Applying function modulopower, for known values of *e1,e2,S1,S2,p* we get

```
>> e1=10228;p=48731;e2=42300;S1=12668;S2=395;
G=mod(modulopower(e1,S2,p)* modulopower(e2,S1,p),p)
G = 15963
```

2. To calculate h(M|G) we act analogously to the Example 2

```
>> numdigits(G); M1=M*10^k+G
k = 5
M1 = 200015963
>> Z=43210; V=mod(M1^2,Z)
V = 12668
```

Since *V=S1 mod p,* so the message is adopted.

SCHNORR AUTHENTICATION PROTOCOL

The algorithm of the protocol is as follows:

1. **Pretreatment**.

   Alice chooses a random number *r, (r< q)* and calculates . These computations are preliminary and may be made long before Bob advents.

2. **Initiation**. Alice sends *x* to Bob.
3. Bob chooses a random number *R* between 0 and *2t-1* and sends it to Alice.
4. Alice computes *S= r + dR (mod q)* and sends *S* to Bob.
5. **Confirmation**. Bob computes *(mod p)* and identifies Alice if z=x.

The security of this algorithm depends on the parameter t. The autopsy of the algorithm is approximately equal to 2^t. This algorithm can be also realized in MatLab without any difficulties.

CONCLUSION

The research paper about Schnorr's Digital Signature gave me an idea of how my personal information was secured throughout the internet. Generating the key and authentication process was simple in form but complicated in decoding. Also, this process of exploration taught me how to use MATLAB and format a research paper. Exploring the way that digital signatures developed from the El Gamal digital signature to the Schnorr's signature and how the United States adopted their DES to combine the two ways was intriguing in how the development took place.

Throughout the history of cryptography, it always has been a matter of how the information can be secured secretly to prevent the hackers from hacking into the information and causing disruption. Authentication is an important matter since it proves the accuracy of the information. The fast developing hacking industry has induced developers to come up with a more secure and complicated method. Since nowadays, brute force attack can decrypt any message that is encrypted and it is the matter of time that causes problem, the key has to be longer and harder to find out. Especially, the quantum computers directly employ the quantum mechanical phenomena to perform the operation of data. They differ from digital computers and foster the speed of decryption. Coming up with the way to prevent the brute force attack can be a future object for people in the field but also decoding the key will always be parallel to the development.

Therefore, I would like to further explore how to generate and decode the key in a more efficient way. How to factorize the prime number has always been enigma to the developers in the field, so I want to explore how they came up with the thought and how efficient factorizing can take place using certain methods.

REFERENCES

1. *A. Menezes, P.van Oorschot, S. Vanstone.* Handbook of Applied Cryptography. — CR C Press, 1996.
2. *Douglas R. Stinson* Cryptography: Theory and Practice, Third Edition. - CRC Press, 2005.
3. *B. Schneier*, Applied Cryptography. - John Wiley & Sons 1996.
4. Digital signature URL: http://en.wikipedia.org/wiki/Digital_signature
5. ElGamal signature scheme https://en.wikipedia.org/wiki/ElGamal_signature_scheme
6. Schnorr signature URL: https://en.wikipedia.org/wiki/Schnorr_signature.
7. MATLAB URL: http://en.wikipedia.org/wiki/MATLAB