Trường Đại học Bách Khoa
TPHCM - ĐHQG TPHCM

Cryptography

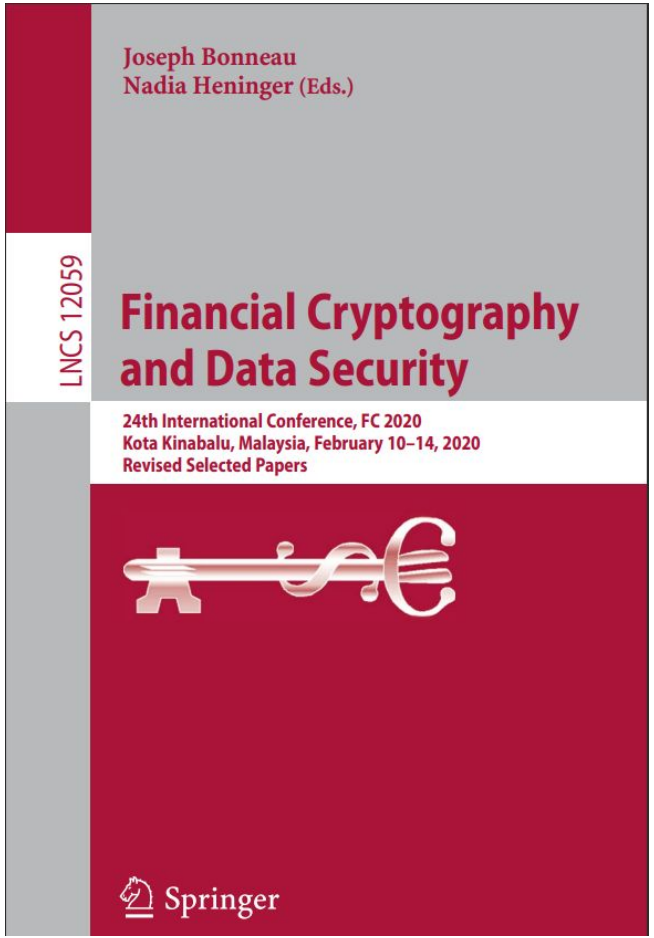
Financial Cryptography and Data Security

24th International Conference, FC 2020 Kota Kinabalu, Malaysia, February 10–14, 2020 Revised Selected Papers

Breaking the Encryption Scheme of the Moscow Internet Voting System

Đinh Thanh Phong - 2270243

Introduction



Check for updates

Breaking the Encryption Scheme of the Moscow Internet Voting System

Pierrick Gaudry[1(✉)] and Alexander Golovnev[2]

[1] CNRS, Inria, Université de Lorraine, Nancy, France
pierrick.gaudry@loria.fr
[2] Harvard University, Cambridge, USA

Abstract. In September 2019, voters for the election at the Parliament of the city of Moscow were allowed to use an Internet voting system. The source code of it had been made available for public testing. In this paper we show two successful attacks on the encryption scheme implemented in the voting system. Both attacks were sent to the developers of the system, and both issues had been fixed after that.

The encryption used in this system is a variant of ElGamal over finite fields. In the first attack we show that the used key sizes are too small. We explain how to retrieve the private keys from the public keys in a matter of minutes with easily available resources.

When this issue had been fixed and the new system had become available for testing, we discovered that the new implementation was not semantically secure. We demonstrate how this newly found security vulnerability can be used for counting the number of votes cast for a candidate.

Introduction

Abstract. In September 2019, voters for the election at the Parliament of the city of Moscow were allowed to use an Internet voting system. The source code of it had been made available for public testing. In this paper we show two successful attacks on the encryption scheme implemented in the voting system. Both attacks were sent to the developers of the system, and both issues had been fixed after that.

The encryption used in this system is a variant of ElGamal over finite fields. In the first attack we show that the used key sizes are too small. We explain how to retrieve the private keys from the public keys in a matter of minutes with easily available resources.

When this issue had been fixed and the new system had become available for testing, we discovered that the new implementation was not semantically secure. We demonstrate how this newly found security vulnerability can be used for counting the number of votes cast for a candidate.

Vào tháng 9 năm 2019, cử tri tham gia cuộc bầu cử tại Quốc hội thành phố Mátxcơva đã được phép sử dụng hệ thống bỏ phiếu qua Internet.

Mã nguồn được cung cấp để thử nghiệm công khai.

Trong bài báo này, trình bày hai cuộc tấn công thành công vào sơ đồ mã hóa được triển khai trong hệ thống bỏ phiếu.

Cả hai cuộc tấn công đều được gửi đến các nhà phát triển hệ thống và cả hai vấn đề đều đã được khắc phục sau đó.

Mã hóa được sử dụng trong hệ thống này là một biến thể của ElGamal.

Trong lần tấn công đầu tiên, chỉ ra rằng kích thước khóa được sử dụng quá nhỏ. Cách lấy khóa riêng từ khóa chung chỉ trong vài phút với các tài nguyên sẵn có.

Khi sự cố này đã được khắc phục và hệ thống mới đã có sẵn để thử nghiệm, tác giả phát hiện ra rằng việc triển khai mới không an toàn về mặt ngữ nghĩa. Tác giả chứng minh cách lỗ hổng bảo mật mới được tìm thấy này có thể được sử dụng để đếm số phiếu bầu cho một ứng cử viên.

Introduction

Bỏ phiếu điện tử ngày càng được sử dụng rộng rãi cho các cuộc bầu cử có mức độ quan trọng thấp, với các hệ thống có chất lượng khác nhau.

Một số quốc gia đã cấm hoàn toàn việc sử dụng bỏ phiếu điện tử trong trường hợp đó (ví dụ Đức năm 2009, Hà Lan năm 2008, hay Na Uy năm 2013),

Trong khi các quốc gia khác sử dụng nó một cách thường xuyên hoặc tổ chức thử nghiệm với các cuộc bầu cử ngày càng cao hơn (Thụy Sĩ, Estonia, Canada)

Thuật ngữ bỏ phiếu điện tử có thể bao gồm các tình huống khác nhau và trong tác phẩm này, tác giả quan tâm đến bỏ phiếu qua Internet chứ không phải bỏ phiếu có sự hỗ trợ của máy diễn ra tại các điểm bỏ phiếu.

1 Introduction

Electronic voting is more and more widely used for low-stakes elections, with systems of various qualities. The situation for important politically binding elections is more contrasted. Some countries have completely banned the use of e-voting in that case (for instance, Germany in 2009, the Netherlands in 2008, or Norway [11] in 2013), while other countries use it on a regular basis or organize experiments with higher and higher stakes elections (Switzerland [9,15,26], Estonia [16], Canada [12]).

The term electronic voting can cover different situations, and in this work, we are interested in Internet voting, not machine-assisted voting that takes place in polling stations. This increases the difficulty to guarantee properties like authentication or coercion-resistance that are easier to obtain at a polling station, where an officer can check classical identity cards and where the voters can go to a polling booth to isolate themselves and choose freely.

But even more basic properties like vote secrecy and verifiability are not easy to obtain if one wants to keep things simple and without advanced cryptographic tools like zero-knowledge proofs, proof of equivalence of plaintexts, oblivious transfer, etc.

Introduction

Đối với các cuộc bầu cử quan trọng, ban tổ chức sẽ yêu cầu một sản phẩm có thể được kiểm tra bởi các chuyên gia độc lập và để khuyến khích có nhiều phản hồi hơn, có thể tổ chức thử nghiệm công khai với chương trình tiền thưởng lỗi liên quan.
Ví dụ, ở Thụy Sĩ, một quốc gia có lịch sử lâu dài về việc thử nghiệm bỏ phiếu qua Internet, trong lịch sử đã từng phát hiện vấn đề bảo mật trong quá khứ.

Ở Nga, ngày 8 tháng 9 năm 2019 là ngày bầu cử địa phương, nơi phải bầu ra các thống đốc và đại diện cho nghị viện địa phương.
Tại Moscow, nhân dịp cuộc bầu cử Quốc hội Thành phố (Duma Moscow), người ta đã quyết định thử nghiệm việc sử dụng bỏ phiếu qua Internet.
Một hệ thống bỏ phiếu được thiết kế đặc biệt cho cuộc bầu cử này.
Việc triển khai nó là trách nhiệm của một cơ quan dịch vụ của Thành phố có tên là Sở Công nghệ Thông tin.
Vào tháng 7, hệ thống đã được mở để thử nghiệm công khai.

For high-stakes elections, a bad practice that tends to become less accepted by the population is to have some designated experts that study the security of the product, but how it really works remains secret to voters. Therefore, in more and more cases, the organization will ask for a product that can be audited by independent experts, and as an incentive to have more feedback, public testing with an associated bug bounty program can be organized. For instance, this has been recently the case in Switzerland, which is a country with a long history of experiments with Internet voting. A security problem was actually discovered at this occasion [14,22].

In Russia, September 8, 2019 was a day of local elections, where governors and representatives for local parliaments must be elected. In Moscow, at the occasion of this election for the City Parliament (Moscow Duma), it was decided to test the use of Internet voting. Voters from 3 electoral districts (among a total of 45 districts) were allowed to register for using Internet voting instead of using classical paper voting at polling stations.

A voting system was designed specifically for this election. For lack of a proper name, we will call it the Moscow Internet voting system. Its deployment is the responsibility of a service of the City called the Department of Information Technology. In July, the system was opened for public testing.

Introduction

Mô tả về Thử thách công cộng:
17/7/2019, một số mã của hệ thống đã được đăng lên mạng và ban tổ chức đã yêu cầu công chúng thử nghiệm một số kịch bản tấn công.

Tiền thưởng lên tới 2 triệu rúp (khoảng 30.000 USD). Hệ thống này không được ghi chép đầy đủ, nhưng từ mã nguồn và mô tả ngắn gọn về hệ thống, tác giả biết rằng nó sử dụng chuỗi khối Ethereum và mã hóa ElGamal.

Trong một trong các kịch bản tấn công, ban tổ chức đưa ra một thử thách bao gồm “public key” và một số tin nhắn được mã hóa.

Cuộc tấn công được coi là thành công nếu các tin nhắn được giải mã trong vòng 12 giờ (khoảng thời gian trong tương lai, cuộc bầu cử thực sự), trước khi ban tổ chức tiết lộ khóa riêng và các thông điệp ban đầu .

Description of the Public Challenge

On July 17, 2019, some of the system's code was posted online [10], and the organizers asked the public to test several attack scenarios [24]. A bounty program of up to 2 millions rubles (approx. $30,000) was associated to it. We believed that the fact that most of the information is in Russian and that almost no description of the system (in any language) is available apart from the source code was a reason for having a low advertisement of this challenge at the international level, even among the e-voting community.

The system is poorly documented, but from the source code and brief descriptions of the system [20], we know that it uses the Ethereum blockchain [3] and ElGamal encryption. No advanced cryptographic tools are present in the source code (no verifiable mixnets [13], for instance, while they are quite frequent in modern systems).

In one of the attack scenarios, the organizers publish a challenge consisting of the public key and some encrypted messages. The attack was considered successful if the messages got decrypted within 12 h (the duration of the future, real election), before the organizers reveal the private key and the original messages. All of these cryptographic challenges (keys and encrypted data) were put in the public repository of the source code, in a special sub-directory called `encryption-keys`.

Introduction

Đóng góp trong bài viết này:
Tác giả mô tả hai cuộc tấn công mà chúng tôi đã thực hiện trên hệ thống, theo kịch bản tấn công này.
Cuộc tấn công đầu tiên lợi dụng thực tế là kích thước khóa rất nhỏ nên với phần mềm chuyên dụng, có thể tính toán logarit rời rạc và suy ra các khóa riêng trong thời gian ít hơn nhiều so với 12 giờ cho phép đối với nhiệm vụ này.
Sau đó, mã nguồn đã được sửa đổi. Cuộc tấn công thứ hai là chống lại phiên bản mới này và dựa vào một cuộc tấn công nhóm con để lộ một bit thông tin liên quan đến tin nhắn gốc.
Trong bối cảnh bỏ phiếu điện tử, điều này có thể đủ để thu được nhiều thông tin về sự lựa chọn của cử tri, và thực sự, trong hệ thống Moscow, sự rò rỉ thực sự rất mạnh mẽ. Trong tháng 8, một số thử nghiệm công khai đã được thực hiện với các tình nguyện viên, sau khi hệ thống được vá lỗi chống lại các cuộc tấn công như trên. Đối với công việc này, tác giả đã sử dụng các nguồn thông tin về hệ thống bỏ phiếu qua Internet ở Moscow:

- Mã nguồn công khai.
- Điều này bao gồm mã Javascript để chạy ở phía máy khách, mã PHP cho phía máy chủ và mã Solidity để chạy dưới dạng hợp đồng thông minh trong chuỗi khối Ethereum.

Contributions

In this paper, we describe two attacks that we mounted on the system, following this attack scenario. The first attack uses the fact that the key sizes are so small that, with specialized software, it is possible to compute discrete logarithms and deduce the private keys in far less than the 12 h allowed for this task. After this, the source code was modified. Our second attack is against this new version and relies on a subgroup attack that reveals one bit of information related to the original message. In an e-voting context, this can be enough to get a lot of information about the voter's choice, and indeed, in the Moscow system, the leakage was really strong. During August, several public tests were done, with volunteers, after the system was patched against our attacks. In this work, after describing our attacks, we will discuss the general protocol, which is some kind of moving target, since there is no proper specification, no clear security claims and on top of that, deep changes were made until very late before the real election.

For this work, we used the following different sources of information about the Moscow Internet voting system:

- The public source code, of course. This includes Javascript code to be run on the client side, PHP code for the server side, and Solidity code to be run as smart contracts in an Ethereum blockchain.
- The articles published in the press, sometimes quoting the designers of the system. This includes various sources, with different opinions about the use of Internet voting in this context. We considered some of these sources as non-reliable.
- Private discussions with the designers and with journalists investigating the current situation.

In the following, we will refer to different versions of the source code. In order to make our terminology precise, we give the exact revision numbers of these versions, corresponding to `git` commits in the public repository [10]:

- The "original" version, i.e. the one that was published and used for the first public test: revision d70986b2c4da.
- The "modified" version, that took into account our first attack: revision 1d4f348681e9.
- The "final" version that was used for the election: revision 51aa4300aceb.

Attack on the Original Implementation

Bộ nguồn giải thuật được xây dựng trên phương pháp mã hoá ELGamal cải tiến, kết hợp với smart-contracts

Giải thuật sẽ sử dụng các biến số đầu vào: m, g, pk và số nguyên tố p để xây dựng mô hình.

Giải thuật ElGamal sẽ sinh ra 3 cặp khoá là:
(sk i , pk i), (sk 2 , pk 2), (sk 3 , pk 3)

Được dùng để mã hoá các lá phiếu.

Công thức mã hoá được biểu diễn như hình bên:

The multilevel variant is obtained by successively applying the ElGamal encryption, with three different parameter sets, first on the message m, and then on the a-part of the successive ElGamal ciphertexts. In the Moscow system, there are 3 levels. Each level uses a group G_i which is the multiplicative group of a finite field $\mathbb{F}_{p_i}$, where p_i is a safe prime. An important remark, here, is that the p_i's being different, there is no algebraic map from one group to the other. It is necessary to lift an element of $\mathbb{F}^*_{p_1}$ to an integer in $[1, p_1 - 1]$ before mapping it to $\mathbb{F}^*_{p_2}$. This mapping will be without loss of information only if p_2 is larger than p_1; and similarly we need p_3 bigger than p_2. These conditions are indeed enforced in the source code.

2.1 Attack on the Original Implementation

In the original version of the source code (rev d70986b2c4da), the encryption scheme can be found in the files `elGamal.js` and `multiLevelEncryptor.js` of the `smart-contracts/packages/crypto-lib/src/` subdirectory. The first file contains a textbook version of the ElGamal encryption algorithm, while the second one builds on top of it a "multilevel" variant that we are going to describe here since this is a non-standard construction.

Let us first fix the notations for the textbook ElGamal encryption. Let G be a cyclic group generated by g of order q. An ElGamal keypair is obtained by choosing a (secret) decryption key sk as a random integer in $\mathbb{Z}_q$, and the corresponding (public) encryption key pk is given by $\mathrm{pk} = g^{\mathrm{sk}}$. Let us denote by $\mathsf{Enc}_{g,\mathrm{pk}}(m) = (a, b)$ the ElGamal encryption of the message $m \in G$ with a

public key pk and a generator g. This is a randomized encryption: an integer r is picked uniformly at random in $\mathbb{Z}_q$, and then the encryption is obtained as

$$\mathsf{Enc}_{g,\mathrm{pk}}(m) = (a, b) = (g^r, \mathrm{pk}^r \cdot m)\,.$$

The corresponding decryption function $\mathsf{Dec}_{g,\mathrm{sk}}(a, b)$, that uses the secret key sk corresponding to pk is then given by

$$\mathsf{Dec}_{g,\mathrm{sk}}(a, b) = b \cdot a^{-\mathrm{sk}} = m.$$

Let us denote by g_1, g_2, g_3 the generators of the 3 groups G_1, G_2, G_3. There are 3 ElGamal key pairs $(\mathrm{sk}_1, \mathrm{pk}_1)$, $(\mathrm{sk}_2, \mathrm{pk}_2)$, $(\mathrm{sk}_3, \mathrm{pk}_3)$ used for the encryption and decryption of the ballots. In order to encrypt a message $m \in G_1$, we compute the following successive ElGamal encryptions:

$$\begin{aligned}
(a_1, b_1) &:= \mathsf{Enc}_{g_1,\mathrm{pk}_1}(m); \quad \text{map } a_1 \text{ to } G_2;\\
(a_2, b_2) &:= \mathsf{Enc}_{g_2,\mathrm{pk}_2}(a_1); \quad \text{map } a_2 \text{ to } G_3;\\
(a_3, b_3) &:= \mathsf{Enc}_{g_3,\mathrm{pk}_3}(a_2),
\end{aligned}$$

Attack on the Original Implementation

Theo thông tin từ mã nguồn, số nguyên tố pi thì nhỏ hơn 256bits.

Theo công bố vào năm 1995-1996, thì từ 215 đến 281 bits tại thời điểm đó sẽ không thể nào tính toán được private key trong vòng 12h.

Tuy nhiên máy tính tiến bộ về mặt xử lý và tốc độ tính toán tăng dần trong suốt 20 năm.

Tác giả đã dùng thử 3 phần mềm phổ thông để đánh giá:

- SageMath (Free)
- Magma (Tính phí bản quyền)
- CADO-NFS (Free)

In the published source code, the primes p_i's have less than 256 bits. Discrete logarithms in finite fields defined by such small primes have been computed for the first time in the middle of the 90's: Weber, Denny and Zayer did a series of computation in 1995–1996, starting from 215 to 281 bits [33]. At that time, the computing resources required for the computations were rather high, and solving the 3 discrete logarithm problems to get the private keys would not have been easily feasible in less than 12 h as required by the challenge.

More than 2 decades later, computers are much faster and have much more memory. Furthermore, the Number Field Sieve algorithm [21], which is the fastest known method asymptotically was still a very new algorithm in the mid-90's, and many theoretical and practical optimizations have been developed since then [7,18,25,30]. The current record is a computation modulo a 768-bit prime [19].

We have tried the following software products that contain a full implementation of discrete logarithm computations in prime fields:

Software	SageMath [29]	Magma [4]	CADO-NFS [27]
Version	8.8	2.24-2	rev. 6b3746a2e

Attack on the Original Implementation

Kết quả chạy thử:
Ban đầu chạy trên máy Intel i5-4590 4 nhân tốc độ 3,3 GHz và RAM 16 GB, với mong muốn giải trong vòng 12h.

- SageMath: mất hơn 4 ngày vẫn chạy chưa xong
- Magma: bị tràn bộ nhớ nên không thể ra kết quả.

Tác giả thử nâng cấp trên nút máy chủ 64 lõi với RAM 192 GB.

Thì Magna đã có thể giải được trong vòng gần 24h với lượng sử dụng 130 GB.

Dường như không thể nào giải trong vòng 12h.

Software	SageMath [29]	Magma [4]	CADO-NFS [27]
Version	8.8	2.24-2	rev. 6b3746a2e

Note that Magma is proprietary software, while the others are free software.

The experiments were first made on a typical personal computer equipped with a 4-core Intel i5-4590 processor at 3.3 GHz and 16 GB of RAM. It is running a standard Debian distribution. SageMath uses GP/Pari [28] internally for computing discrete logarithms. On this machine, the computation took more than 12 h, and actually we had to stop it after 4 days while it was still running. According to GP/Pari documentation, the algorithm used is a linear sieve index calculus method. As for Magma, the handbook tells us that depending on arithmetic properties of the prime, the algorithm used can be the Gaussian integer sieve or a fallback linear sieve. The prime we tested was compatible with the Gaussian integer sieve. But during the linear algebra step, the memory requirement was much larger than the available 16 GB. We started the computation again, on a 64-core server node with 192 GB of RAM. On this machine, Magma computed the discrete logarithm in a bit less than 24 h with 130 GB of peak memory usage. It should be noted that both Magma and SageMath use only one of the available computing cores, so that there does not seem to be an easy way to go below the 12 h limit with them, even with an access to a powerful machine.

Attack on the Original Implementation

CADO-NFS được chạy thử nghiệm trên máy tính cá nhân như trên.
Và thời gian tính toán đảo ngược private key có kết quả như sau:

Key 1 = 7 min 5 s
Key 2 = 8 min 27 s
Key 3 = 5 min 14 s

Nguyên nhân là nhờ vào giải thuật Number Field Sieven, có thể giảm thời gian tính toán đảo ngược, được nêu chi tiết trong loại tấn côn LogJam.

Nên CADO-NFS có thể nhanh chóng giải mã được các đoạn mã khoảng 256 bits

CADO-NFS is an implementation of the Number Field Sieve for integer factorization and discrete logarithms in prime fields (and some experimental support for small degree extensions of prime fields). The last stable release 2.3.0 is two years old, so we used the development version, available on the public git repository. With CADO-NFS, on the standard personal machine, the running times to retrieve the private keys of August 18 were as follows:

Key number	Time
1	7 min 5 s
2	8 min 27 s
3	5 min 14 s

Note that the variation in the running time from one key to the other is not unusual for computations with moderately small primes. Also, we should mention that when doing this work, we realized that the development version was not robust for numbers of this size: it sometimes failed in the final step called "individual logarithm" or "descent". The revision number we gave above corresponds to a version where we have fixed these problems, so that CADO-NFS can reliably compute discrete logarithms in finite fields of about 256 bits.

Due to mathematical obstructions, the Number Field Sieve is an algorithm that can compute discrete logarithms only in a sub-group of prime (or prime-power) order. In the present situation where the order of the generator is twice a prime, a small Pohlig-Hellman step must be added. This part is not included in CADO-NFS and must be done by hand. Similarly, peculiarities of the Number Field Sieve imply that the base for the discrete logarithm computed by CADO-NFS is arbitrary. Therefore, in order to compute one of the sk_i, the program must be run twice, once for the generator and once for the public key. Fortunately, in the Number Field Sieve algorithm, many parts of the computation can be shared between the two executions modulo the same prime (this is the basis of the LogJam attack [2]), and CADO-NFS indeed shares them automatically. The running times given above include those 2 runs for each key. For completeness and reproducibility, we provide in Appendix A a script to obtain the keys; this includes the few additional modular operations to be done apart from the calls to CADO-NFS.

Attack on the Modeified Implementation

Sau khi cuộc tấn công đầu tiên được gửi tới các nhà phát triển hệ thống và công khai vài ngày sau đó, mã nguồn công khai đã được sửa đổi.
Kích thước khóa đã được tăng lên 1024 bit và ElGamal đa cấp đã bị loại bỏ và thay thế bằng mã hóa ElGamal duy nhất.

Tuy nhiên, chúng tôi phát hiện ra rằng các phần khác của quá trình triển khai không được thay đổi tương ứng, do đó một cuộc tấn công vẫn có thể xảy ra.

2.2 Attack on the Modified Version

After the first attack was sent to the developers of the system and made public a few days later, the public source code has been modified. The key size has been increased to 1024 bits, and the multilevel ElGamal has been removed and replaced by a single ElGamal encryption.

In the original version, the generators in all the involved groups were generators for the full multiplicative group of the finite fields, thus their orders were twice a prime numbers. This exposed the danger of leakage of one bit of information on the message, with a subgroup attack. This is an old technique [23], but there are still frequent attacks, in particular when an implementation forgets the key validation step [31]. Although we did not push in this direction in the first attack, it was explicitly mentioned as a weakness. Therefore in the modified version, the generator was chosen to be a quadratic residue, thus having prime order.

We discovered however that the other parts of the implementation were not changed accordingly, so that an attack was still possible.

Let $p = 2q + 1$ be the 1024-bit safe prime used to define the group, where q is also a prime. Let Q_p be the group of quadratic residues modulo p; it has order $|Q_p| = (p-1)/2 = q$. The chosen generator g belongs to Q_p, and therefore, so is the public key pk, since it is computed as before as $\mathrm{pk} = g^{\mathrm{sk}}$, where sk is randomly chosen in $\mathbb{Z}_q$.

The problem with the modified implementation is that the message m is allowed to be any integer from $[1, q-1]$ which is naturally mapped to an element of $\mathbb{F}_p^*$. For semantic security (under the Decisional Diffie-Hellman assumption), the message m should instead be encoded as one of the q elements of the group Q_p generated by g. In the case where m is not necessarily picked from the group of quadratic residues, the Decisional Diffie-Hellman assumption does not hold and indeed it is possible to build an efficient distinguisher, thus showing that the encryption scheme in the modified version is not semantically secure.

Attack on the Modeified Implementation

Trong phiên bản gốc của sơ đồ mã hóa, ngay khi cuộc bầu cử bắt đầu, 3 khóa chung của ElGamal đa cấp phải được công khai và từ chúng, chỉ trong vài phút, các khóa giải mã có thể được suy ra.

Sau đó, điều này giống như thể sự lựa chọn của cử tri đã được thể hiện rõ ràng trong suốt quá trình. Ngay cả khi có giả định về độ tin cậy mạnh mẽ trên máy chủ nhận những phiếu bầu này và ngay cả khi nó trung thực và quên mất mối liên hệ giữa cử tri và lá phiếu, thì vẫn có vấn đề đưa chúng vào blockchain để xác minh.

Vì các lá phiếu (về cơ bản) ở dạng văn bản rõ ràng, nên kết quả từng phần sẽ được công khai trong suốt ngày bầu cử, điều này có thể ảnh hưởng mạnh mẽ đến kết quả. Trên thực tế, ở Nga việc công bố bất kỳ kết quả sơ bộ nào trong khi cuộc bầu cử vẫn đang diễn ra là bất hợp pháp

2.3 On the Role of Encryption in the Protocol – What Did We Break?

As in many e-voting protocols, the encryption scheme is used to encrypt the choice of the voter to form an encrypted ballot. From the Javascript source code (under a sub-directory called `voting-form`) that is supposed to be run on the voting device of the voter, we deduce that the encrypted data consists solely of

In the original version of the encryption scheme, as soon as the election starts, the 3 public keys of the multilevel ElGamal must become public, and from them, in a matter of minutes the decryption keys can be deduced. Then, this is as if the choices of the voters were in cleartext all along the process. Even if there is a strong trust assumption on the server that receives these votes, and even if it is honest and forgets the link between the voters and the ballots, there is still the issue of putting them in the blockchain for verifiability. Since the ballots are (essentially) in cleartext, the partial results become public all along the day of the election, which can have a strong influence on the result. Actually, it is illegal in Russia to announce any preliminary result while the election is still running.

Attack on the Modeified Implementation

Cuộc tấn công thứ hai của chúng tôi sẽ không cung cấp thông tin đầy đủ về lá phiếu. Chỉ một chút thông tin bị rò rỉ từ một lá phiếu được mã hóa, đó là liệu ứng cử viên được chọn có id phó là dư lượng bậc hai hay không.

Sau đó, từ một lá phiếu được mã hóa, bằng cách tính toán biểu tượng Legendre, người ta có thể suy ra lựa chọn của cử tri trừ khi cô ấy bỏ phiếu cho một ứng cử viên ít được ưa chuộng hơn.

Do đó, giống với cuộc tấn công đầu tiên, cuộc tấn công thứ hai này có nghĩa là tính bí mật của phiếu bầu dựa trên sự tin cậy rất cao được giả định trong máy chủ bỏ phiếu và kết quả một phần sẽ bị rò rỉ trong suốt quá trình.

Mặc dù mã nguồn công khai chưa được chính thức sửa đổi, nhưng 1 bản cập nhật được cung cấp cho các tình nguyện viên trong quá trình thử nghiệm đã bao gồm một bản vá chống lại cuộc tấn công thứ hai.

Our second attack will not give full information about a ballot. Just one bit of information is leaked from an encrypted ballot, namely whether or not the chosen candidate has a deputy id which is a quadratic residue. As the deputy ids seem to be chosen at random with no specific arithmetic property, there is a one-half probability that they belong to Q_p, as for any element of $\mathbb{F}_p^*$. There could be some bias if the deputy ids had only a few bits, but with 32-bit integers, according to standard number theoretic heuristics this will not be the case. A plausible scenario for the attack is then a district where two candidates concentrate most of the votes, one of them having a deputy id in Q_p and the other not. Then, from an encrypted ballot, by computing a Legendre symbol, one can deduce the voter's choice unless she voted for a less popular candidate.

Therefore, as for the first attack, this second attack means that vote secrecy relies on a very strong trust assumption in the voting server, and that the partial results are leaked all along the process.

Lessons Learned and Conclusion

Bài học đầu tiên rút ra từ câu chuyện này là, các nhà thiết kế nên hết sức cẩn thận khi sử dụng mật mã.
Các tác giả của hệ thống Moscow đã mắc nhiều sai sót với sơ đồ mã hóa mà họ quyết định sử dụng. Và trên thực tế, ngay cả bây giờ, về mặt kỹ thuật, việc mã hóa vẫn còn yếu vì hai lý do. Đầu tiên, khóa 1024-bit quá nhỏ để bảo mật trong thời gian trung hạn và nếu giao thức thay đổi để quyền riêng tư của phiếu bầu dựa vào nó thì điều này sẽ không đủ. Hơn nữa, theo những gì chúng tôi có thể thấy, cách chọn số nguyên tố không được công khai, do đó nó có thể bao gồm một cửa bẫy khiến cho các nhà thiết kế dễ dàng tính toán logarit rời rạc.Thứ hai, giải thuật ElGamal hiện đang được triển khai, không phải là chuẩn IND-CCA2. Tùy thuộc vào giao thức, điều này có thể dẫn đến các cuộc tấn công nhỏ hoặc có sức tàn phá lớn. Như một ví dụ về trường hợp thứ hai, trong một giao thức bao gồm một lời giải mã cho phép giải mã bất kỳ văn bản mật mã nào không có trong thùng phiếu (ví dụ: vì mục đích kiểm toán), sẽ dễ dàng sử dụng các thuộc tính đồng hình của ElGamal để giải mã tất cả các lá phiếu.

The first lesson learned from this story is, not surprisingly, that designers should be very careful when using cryptography. The authors of the Moscow system made many mistakes with the encryption scheme they decided to use. And in fact, even now, technically the encryption is still weak for two reasons. First, the 1024-bit key is too small for medium term security, and if the protocol changes so that vote privacy relies on it, this will not be enough. Furthermore, as far as we could see, the way the prime was chosen is not public, so that it could include a trapdoor making discrete logarithms easy to compute for the designers [8]. Second, textbook ElGamal, which is what is implemented now, is not IND-CCA2. Depending on the protocol, this might lead to minor or devastating attacks. As an example of the latter, in a protocol that would include a decryption oracle that allows to decrypt any ciphertext that is not in the ballot box (for instance, for audit purpose), it would be easy to use the homomorphic properties of ElGamal to get all the ballots decrypted.

The second lesson is that using a blockchain is not enough to guarantee full transparency. There are various notions of verifiability in the e-voting literature [5], and the designers must clearly say which property they have, under precise trust assumptions. These trust assumptions must be made even more carefully when using a permissioned blockchain, where the nodes running the blockchain are probably specifically chosen for the election, and where the access to the blockchain can be cut at any time.

Lessons Learned and Conclusion

Bài học thứ hai là việc sử dụng blockchain là không đủ để đảm bảo tính minh bạch hoàn toàn. Có nhiều khái niệm khác nhau về khả năng kiểm chứng trong - tài liệu về bỏ phiếu điện tử [5], và các nhà thiết kế phải nói rõ ràng họ có thuộc tính nào, theo các giả định tin cậy chính xác. Các giả định về độ tin cậy này phải được thực hiện cẩn thận hơn nữa khi sử dụng chuỗi khối được cấp phép, trong đó các nút chạy chuỗi khối có thể được chọn cụ thể cho cuộc bầu cử và nơi có thể cắt quyền truy cập vào chuỗi khối bất cứ lúc nào.

Cụ thể hơn nữa đối với bỏ phiếu điện tử, hệ thống Moscow là một ví dụ điển hình về khó khăn đối với hệ thống bỏ phiếu Internet trong việc khiến tính bí mật của phiếu bầu chỉ dựa vào việc cắt liên kết giữa cử tri và lá phiếu được mã hóa của họ khi họ đến một máy chủ cũng sẽ xác thực cử tri. Trong một cuộc bầu cử có tính rủi ro cao như vậy, nhiều đặc tính bảo mật dường như không tương thích phải được đáp ứng (bí mật và minh bạch), và hầu như không thể tránh khỏi các công cụ mật mã tiên tiến.

Even more specific to e-voting, the Moscow system is a good example of the difficulty for an Internet voting system to make the vote secrecy rely uniquely on cutting the link between the voters and their encrypted ballots when they arrive on a server that should also authenticate the voters. What is really required is to cut the link with the vote in clear, and, for this, classical methods exist like homomorphic decryption or verifiable mixnets. In such a high-stakes election, many seemingly incompatible security properties must be satisfied (secrecy vs transparency), and advanced cryptographic tools are almost impossible to avoid.

Finally, as a conclusion, although our attacks led to the system using a better encryption scheme, it is clear that the system as a whole is still far from being perfect. We consider it likely that if the specification were becoming public in the future, other attacks would be revealed. Therefore, we believe that the main impact of our work was to draw the attention to the system as something that was maybe not as secure as what was claimed. The bad publicity in the press hopefully influenced some potential voters who decided not to take the risk of using this still really problematic system and went for paper ballots instead.

Acknowledgements. Thanks to Iuliia Krivonosova and Robert Krimmer, for sharing some information about the Moscow Internet voting. In particular Iuliia's blog post [20] was quite useful. We also thank Noah Stephens-Davidowitz for his comments on an earlier version of this note. We thank Mikhail Zelenskiy and Denis Dmitriev for sharing some data and information about the voting scheme.

Lessons Learned and Conclusion

Cuối cùng, mặc dù các cuộc tấn công của tác giả đã giúp hệ thống sử dụng sơ đồ mã hóa tốt hơn, nhưng rõ ràng là toàn bộ hệ thống vẫn chưa hoàn hảo.

Chúng tôi cho rằng có khả năng là nếu thông số kỹ thuật được công bố trong tương lai, các cuộc tấn công khác sẽ bị lộ.

Do đó, tác giả tin rằng tác động chính của công việc của tác giả là thu hút sự chú ý đến hệ thống vì nó có thể không an toàn như những gì đã được tuyên bố.

Hy vọng rằng dư luận không tốt trên báo chí đã ảnh hưởng đến một số cử tri tiềm năng, những người đã quyết định không mạo hiểm sử dụng hệ thống vẫn thực sự có vấn đề này và thay vào đó đã đi bỏ phiếu bằng giấy.

Even more specific to e-voting, the Moscow system is a good example of the difficulty for an Internet voting system to make the vote secrecy rely uniquely on cutting the link between the voters and their encrypted ballots when they arrive on a server that should also authenticate the voters. What is really required is to cut the link with the vote in clear, and, for this, classical methods exist like homomorphic decryption or verifiable mixnets. In such a high-stakes election, many seemingly incompatible security properties must be satisfied (secrecy vs transparency), and advanced cryptographic tools are almost impossible to avoid.

Finally, as a conclusion, although our attacks led to the system using a better encryption scheme, it is clear that the system as a whole is still far from being perfect. We consider it likely that if the specification were becoming public in the future, other attacks would be revealed. Therefore, we believe that the main impact of our work was to draw the attention to the system as something that was maybe not as secure as what was claimed. The bad publicity in the press hopefully influenced some potential voters who decided not to take the risk of using this still really problematic system and went for paper ballots instead.

Acknowledgements. Thanks to Iuliia Krivonosova and Robert Krimmer, for sharing some information about the Moscow Internet voting. In particular Iuliia's blog post [20] was quite useful. We also thank Noah Stephens-Davidowitz for his comments on an earlier version of this note. We thank Mikhail Zelenskiy and Denis Dmitriev for sharing some data and information about the voting scheme.

Reference:

1. *Financial Cryptography and Data Security - 24th International Conference, FC 2020 Kota Kinabalu, Malaysia, February 10–14, 2020 Revised Selected Papers*