UNIVERSITY OF STUTTGART

Hauptseminar: The Logjam Attack

Li Yang Wu

June 7, 2017

CONTENTS

ABSTRACT

[Adrian et al., 2015] developed a new attack called Logjam to break many cryptographic systems using the Diffie-Hellman key exchange mechanism. This review elucidates its technical details and summarize overall results of their paper.

1 INTRODUCTION

If it comes to network security, the Diffie-Hellman key exchange is the standard wildly used in practice and declared to be safe. However, [Adrian et al., 2015] could break the security on many popular and browser certified websites. They demonstrated this so called Logjam attack on the Transfer Level Security protocol that supports Diffe-Hellman as key exchange. They also implemented an attack exploiting weak and misconfigured groups using the Pohling-Hellman algorithm.

The second result the authors present is an estimate on cost of computing logs from key sizes of 1024 bits. I will briefly summarize their assumptions for the estimate and the results.

2 DIFFIE-HELLMAN KEY EXCHANGE

[Diffie and Hellman, 1976] was the first paper presenting a practical solution to the privacy and the authentication problem that does not rely on secure communication channels. These kinds of channels are often not given or so inefficient to use that the benefit of telecommunication is nullified. Hence, it was a historical and great advance in computer networking and cryptography. They presented an asymmetric approach with trap door functions that have an exponential cipher-cryptanalyst ratio. Because of this, a security system can choose feasible large private keys that are at the same time infeasible to infer from public keys and encrypted messages w.r.t. computation time.

2.1 PROBLEM FORMULATION

Given:

- A, B, the communication partners.
- Only insecure communication channels are available.
- $\{M\}$, a finite message space which contain all possible messages M to be exchanged.

Determine:

- $C = (\{E_K\}_{K \in \{K\}}, \{D_K\}_{K \in \{K\}})$, a *public key cryptosystem* with mutual inverse functions

  - $E_K : \{M\} \to \{M\}$
  - $D_K : \{M\} \to \{M\}$

  denoted as *encryption* and *decryption transformation* respectively.

- $\{K\}$, a suitable set of keys.
- Such that
  1. $\forall K \in \{K\} : M = D_K(E_K(M))$
  2. $\forall K \in \{K\}, M \in \{M\} : E_K(M), D_K(M)$ are easy to compute.
  3. $\forall^\infty K \in \{K\}, \alpha : \{E_K\}_{K\in\{K\}} \to \{D_K\}_{K\in\{K\}} : D_K = \alpha(E_K)$ is infeasible to compute.
  4. $\forall K \in \{K\} \exists \beta : \{K\} \to (\{E_K\}_{K\in\{K\}}, \{D_K\}_{K\in\{K\}}) : \beta(K) = (E_K, D_K)$ is feasible to compute.

2.2 Solutions

KEY EXCHANGE The solution the authors propose is to draw private and public keys (D, E) from finite fields of sizes equal to prime numbers $\mathrm{GF}(q) \cong \mathbb{Z}/q\mathbb{Z}$, i.e.

$$D, E \in \mathrm{GF}(q) = \{x \mod q \mid x \in \mathbb{Z}\}. \tag{2.1}$$

By choosing the private key as the logarithm of the public key and an arbitrary basis $\alpha \in \mathrm{GF}(q)$,

$$D = \alpha^E \mod q, \tag{2.2}$$
$$E = \log_\alpha Y \mod q, \tag{2.3}$$
$$1 \leq D, E \leq q - 1, \tag{2.4}$$

the requirements 3. and 4. are fulfilled. Calculating D from E has logarithmic time complexity in q. Inversely, calculating E from D has a complexity of $\sqrt{q}$. Therefore the cipher-cryptanalyst ratio is exponential. Note that this is not a proven bound. We use the best known algorithms for these problem to get this ratio. Better algorithms have not been found for decades, therefore this method is considered safe. Also, one need to find good primes q in order to assure worst or near worst case log-computation time.

Now if A and B want to exchange a session key, they first choose public and private keys D_A, D_B, E_A, E_B with a q of sufficient size and exchange their public keys over the insecure communication channel. Then, they can calculate the common session key

$$\begin{aligned} K_{A,B} &= \alpha^{E_A E_B} \mod q && (2.5)\\ &= (\alpha^{E_A})^{E_B} \mod q && (2.6)\\ &= {D_A}^{E_B} \mod q && (2.7)\\ &= (\alpha^{E_B})^{E_A} \mod q && (2.8)\\ &= {D_B}^{E_A} \mod q. && (2.9)\end{aligned}$$

While A calculates the right term in 2.7, B calculates the right term in 2.9 and both of them have now the same session key $K_{A,B}$.

The trap-door property of this mechanism allows us to use it for one-way authentication. A signature s of some user A is $D_A(s)$. No one can forge it, since D_A is private. A cannot deny his or her signature, since everyone can confirm $E_A(D_A(s)) = s$.

CRYPTANALYSIS The attacker can compute $K_{A,B}$ either by computing E_A from D_B or E_B from D_A, both are log-problems in finite fields. Thus, the attacker is successful, iff he or she can effort exponentially more computation power that A and B.

3 Number Field Sieve Algorithm

As described above, the only known way to break Diffie-Hellman is to compute E from D, i.e. to compute discrete logs for prime fields for a fixed prime q and base α. [Gordon, 1993] first proposed how the *number field sieve* (NFS) algorithm can be used to compute discrete logs. Before that, this technique was known to solve the factorization problem for large primes.

3.1 Approach

The idea is to factorize D and find logs for each of these factors. The logs of these factors can be constructed by the logs of some precomputed logs in the database. This step can be done very efficiently. The difficult problem is to build up such a database for the prime of interest q. But the good thing is that this precomputation only depends on q and not on a specific problem instance D. Once the precomputation is done, the database can be used to compute the log for every $D \in \mathrm{GF}(q)$.

3.2 Precomputation

3.2.1 Smoothness Assumption

Define that an integer number x is B-smooth, if all prime factors of x are smaller that B. NFS assumes the base α to be B-smooth where B is a upper bound for the prime values in the factor base (which is the output of the precomputation). If α is not B-smooth, we can use a B-smooth α' instead. Then, we can compute $\log_{\alpha'} \alpha$ and transform back to the original problem with

$$\log_\alpha D \equiv \frac{\log_{\alpha'} D}{\log_{\alpha'} \alpha} \mod (q-1)\,. \tag{3.1}$$

If α' is not a generator for $\mathrm{GF}(q)^*$, the logs on the right hand side might not exist. α' can be checked for feasibility by factoring $(q-1)$ with the NFS factoring algorithm and check

$$(a')^{\frac{q-1}{p}} \not\equiv 1 \mod q, \quad \forall p \parallel q\,, \tag{3.2}$$

where $p \parallel q$ denotes that p is a prime divisor of q. The reason why α' needs to be B-Smooth is that the factor base needs to have at least one inhomogeneous relation for the logs of the factor base, using the equation

$$log_{\alpha}\alpha = 1 \equiv \sum_{p^t \| a} t \log_{\alpha} p \mod (q-1)\,, \tag{3.3}$$

otherwise we gain no knowledge from the factor base.

3.2.2 Polynomial Selection

The goal in this step to find a polynomial representation of $\mathrm{GF}(q)$. This has the advantage that we can express relations and encode informations in terms of linear equations to reason about them, which are the tasks that actually will be done in the next steps.

First, we represent $\mathrm{GF}(q)$ as $\mathbb{Z}/q\mathbb{Z}$ where elements are identified with their least non-negative residues. Then, choose and integer m and polynomial $f : \mathbb{Z} \to \mathbb{Z}$ of degree k such that f is monic, irreducible over $\mathbb{Q}$ and $f(m) \equiv 0 \mod q$. We can find such a polynomial f by choosing m of suitable size and find coefficients $a_1, ..., a_k$ such that

$$p = \sum_{i=0}^{k} a_i m^i\,. \tag{3.4}$$

It is required that $q \nmid \Delta_f$, where Δ_f is the discriminant of f. If the choice of f or m leads to $q \mid \Delta_f$, then try another m or alter f slightly, for example add m to a_i and subtract 1 from a_{i+1}.

Next, let $\gamma \in \mathbb{C}$ be the root of f, define $K = \mathbb{Q}(\gamma) = \mathbb{Q} \cup \{\gamma\}$ and O_K the ring of integers in K. Since $(q, \Delta_f) = 1$, $\hat{q} := (q, \gamma - m)$ is a first-degree prime factor of $q \in O_K$. It follows that $O_K/\hat{q} \cong \mathrm{GF}(q)$ and we can define a homomorphism $\phi : \mathbb{Z}[\alpha] \to \mathbb{Z}/q\mathbb{Z}$, providing us the representation we aimed for. We denote a prime ideal $s \in O_K$ as *good* if s does not divide the index $[O_K : \mathbb{Z}[\gamma]]$, otherwise *bad*.

The actual factor base $\mathcal{B}$ for which we want to find the logarithms consists of two parts: $\mathcal{B}_{\mathbb{Q}}$ the set of rational primes and $\mathcal{B}_K$ the set of good prime ideals in O_K, where $p, \|p'\|_1 \leq B,\ \forall p \in \mathcal{B}_{\mathbb{Q}}, p' \in \mathcal{B}_K$. Note that B is the same bound for our base α (from our primary problem definition: Find E for $D = \alpha^E \mod q$). Also, let $\mathcal{B}' \subset \mathcal{B}_{\mathbb{Q}}$ denote the set of prime factors of α.

3.2.3 Sieving

3.2.4 Linear Algebra

Given some matrix $A = \mathbb{Z}^{S \times T},\ S > T$, where each row has at most E non-zero entries, find a linear relation over $\mathbb{Q}$ for the rows of A. This is accomplished in four steps, shown for the case $S = T+1$. This procedure was developed by Carl Pomerance (*1944) which was not separately published, instead only for helping out in [Gordon, 1993].

STEP 1   Attempt to compute the rank $r = \mathrm{rk}(A)$. For that, choose a random prime $q_0 \leq ET \log T$. Later, we will check if that choice was feasible. If not, come back here and sample another one. Then, use Gaussian elimination mod q_0 to find the rank r_0 of $A \mod q_0$. Rearrange the rows so that the first r_0 rows are linearly independent mod q_0. Now the rearranged matrix is

$$A' = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{T+1} \end{pmatrix} = \begin{pmatrix} a'_{11} & a'_{12} & \cdots & a'_{1T} \\ a'_{21} & a'_{22} & \cdots & a'_{2T} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{T+11} & a'_{T+12} & \cdots & a'_{T+1T} \end{pmatrix} . \tag{3.5}$$

The result of the Gaussian elimination provide a submatrix $\hat{A} = \mathbb{Z}^{r_0 \times r_0}$ of the first r_0 rows that is nonsingular mod q_0,

$$\hat{A} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{r_0} \end{pmatrix} = \begin{pmatrix} a'_{11} & a'_{12} & \cdots & a'_{1r_0} \\ a'_{21} & a'_{22} & \cdots & a'_{2r_0} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{r_01} & a'_{r_02} & \cdots & a'_{r_0r_0} \end{pmatrix} . \tag{3.6}$$

STEP 2   Attempt to express v_{r_0+1} as a linear combination of $v_1, v_2, ..., v_{r_0} \mod q$ for each $q \leq ET \log T$. For some q it will be successful, for others not. Denote the product of all successful primes as $\boldsymbol{P}$ and the product of all unsuccessful primes as $\boldsymbol{P'}$. If $\boldsymbol{P'} > ET \log T$, then repeat Step 1 (sample a new random prime number). Otherwise, continue with the next step.

STEP 3   Attempt to find the determinant D of $\hat{A}$. For each prime $q|\boldsymbol{P}$, use the Wieldemann's probabilistic determinant algorithm [Wiedemann, 1986] to compute an integer $D_q \in \{0, ..., q-1\}$, which is the determinant of $A \mod q$ with probability of at least $1 - (ET)^{-2}$, i.e.

$$\mathbb{P}(D_q = \det(A \mod q) \mid A, q) \geq 1 - (ET)^{-2} . \tag{3.7}$$

For each D_q we can write an congruence

$$D_0 \equiv D_q \mod q, \quad \forall q|\boldsymbol{P} \tag{3.8}$$

and use the Chinese remainder theorem to compute D_0 fulfilling these congruences and being closest to zero. Repeat this step until we have found a value that lies within the bound

$$0 < |D_0| \leq (E^{\frac{1}{2}}T)^T . \tag{3.9}$$

STEP 4   In the final step we try to find the actual coefficients $c_1, ..., c_{r_0}$ for the linear relation between v_{r_0+1} and the rest of the rows. Again, use the Chinese remainder theorem to find these coefficients closest to zero such that

$$D_0 v_{r_0+1} \equiv \sum_{i=1}^{r_0} c_i v_i \mod \boldsymbol{P} . \tag{3.10}$$

In case one of these coefficients c_i exceeds $(E^{\frac{1}{2}}T)^T$ in absolute value, repeat Step 3 to find another D_0. Otherwise, we have found the relation

$$D_0 v_{r_0+1} = \sum_{i=1}^{r_0} c_i v_i \; . \tag{3.11}$$

It can also be shown that the expected runtime of this algorithm is

$$\mathcal{O}(E^2 T^3 \log^3 T) \; . \tag{3.12}$$

Note that improved versions of this algorithm exist, such as [Joux and Lercier, 2003].

4 The Logjam Attack

5 Other Attacks

6 Pohling-Hellman Algorithm

7 Discussion

TODO

References

Adrian, D., Bhargavan, K., Durumeric, Z., Gaudry, P., Green, M., Halderman, J. A., Heninger, N., Springall, D., Thomé, E., Valenta, L., VanderSloot, B., Wustrow, E., Zanella-Béguelin, S., and Zimmermann, P. (2015). Imperfect forward secrecy: How diffie-hellman fails in practice. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 5–17, New York, NY, USA. ACM.

Diffie, W. and Hellman, M. (1976). New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654.

Gordon, D. M. (1993). Discrete logarithms in gf(p) using the number field sieve. *SIAM Journal on Discrete Mathematics*, 6(1):124–138.

Joux, A. and Lercier, R. (2003). Improvements to the general number field sieve for discrete logarithms in prime fields. a comparison with the gaussian integer method. *Mathematics of computation*, 72(242):953–967.

Wiedemann, D. (1986). Solving sparse linear equations over finite fields. *IEEE transactions on information theory*, 32(1):54–62.