The Elliptic Curve Diffie-Hellman (ECDH)

Rakel Haakegaard and Joanna Lang

December 2015

1 Introduction

Abstract: The Elliptic Curve Diffie-Hellman (ECDH), a variant of the Diffie Hellman, allows two parties that have no prior knowledge of each other to establish a shared secret key over an insecure channel.[3] The Diffie-Hellman works over any group as long as the DLP in the given group is a difficult problem.[2] It is one of the first public key protocols, and it is used to secure a variety of Internet services. However, newly research from October 2015 suggests that the security of Diffie-Hellman key exchange is less secure than widely believed, and maybe not strong enough to prevent very well-funded attacks. We will first discuss the usage and the security of the ECDH specificly, and then look into the newly published article from October 2015 [1] to see if the discoveries that have been made also apply to the ECDH

2 Description of ECDH

The Elliptic Curve Diffie Hellman (ECDH) distincts from the general Diffie Hellman (DH) in the way that it is based on the elliptic curve discrete logarithm problem (ECDLP) instead of the discrete logarithm problem (DLP). ECDH is an anonymous key agreement protocol which allows two parties, A and B, to establish a shared secret key over an insecure channel, where each of the parties have an elliptic curve public-private key pair[7].

The ECDH works as follows. A and B agree on the elliptic curve group E of order n and a primitive element P in E, which then also has the order n. E, n and P are assumed to be known to the adversary. The ECDLP, which the ECDH is based on, is defined as the computation of the integer k given P and Q such that $Q = [k]P$. The ECDH let A and B compute a shared secret key S, using the property of the ECDLP as described below.

A selects an integer a in the range $[2, n-1]$, computes $Q = [a]P$ and sends Q to B. B on the other hand selects an integer b in the range $[2, n-1]$, computes $R = [b]P$ and sends R to A. A and B receives R and Q respectively, and computes the shared secret key S; $S = [a]R = [b]Q = [a][b]P = [a * bmod(n)]P$. Both A and B get the same value for S, and the shared key is established.[2]

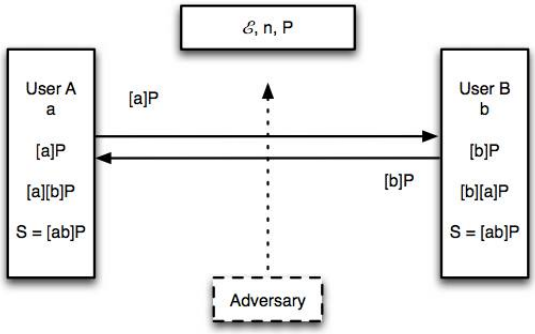

Figure 1: Elliptic Curve Diffie-Hellman key exchange method[2]

3 Security for ECDH

The computational elliptic curve Diffie-Hellman problem (ECDHP) is the problem of trying to find S=[ab mod(n)]P, given E, n, P and the two points Q=[a]P and R=[b]P. This is the problem the adversary will try to solve to get the secret key S, and the ability to defeat this type of attacks is an important part of the security of ECDH.

If the ECDLP in $< P >$ can be efficiently solved, then the ECDHP in $< P >$ can also be efficiently solved by finding a from (P, Q) and then

computing $S = [a]R$. In other words, the ECDHP is no harder than the ECDLP. It is unknown whether the hardness of ECDHP is equal to the hardness of ECDLP. Anyhow, for the ECDHP to have a high degree of security, it is essential that the corresponding ECDLP has a high degree of security. This topic will be discussed in the following section.[8]

3.1 Security and hardness of the ECDLP

The elliptic curve parameters for cryptographic schemes should be carefully chosen to be able to resist all known attacks on the ECDLP. The most naive approach for solving the ECDLP is exhaustive search, which can be defeated by choosing a sufficiently large n ($n >= 2^{80}$). The best known attack to the ECDLP is a combination of the Pohlig-Hellman algorithm and Pollard's rho algorithm, with a running time of $O(\sqrt{(p)})$, with p being the largest prime divisor of n. To defeat this type of attack, one should choose the elliptic curve parameters such that n is divisible by a significantly large prime number p. The size of p should be so large that $\sqrt{(p)}$ steps is an infeasible amount of computation ($p >= 2^{160}$).[8] The ECDLP is believed to be infeasible by the state of today's computer technology, given that the elliptic curve parameters are carefully chosen to defeat the known attacks to the ECDLP. As of today there has been no discovery of a general-purpose subexponential-time algorithm for solving the ECDLP.[2]

On the other hand, it should be noted that there is no mathematical proof of that an efficient algorithm for solving the ECDLP does not exist. If someone were to prove that such an efficient polynomial-time algorithm does not exist, this would imply that $P \neq NP$. This question is as of today known for being one of the most fundamental and outstanding open questions in computer science, so such a proof would be revolutionary (and not very likely to appear). There is either no proof for that the ECDLP is intractable, as ECDLP is not known to be NP-hard. This is not likely to be proved either.[8]

The ECDLP was introduced to computer science only 30 years ago (1985), and because of this it is not as researched as the commonly used DLP, which has a subexponential solution. This, together with the lack of proof for its hardness, are reasons for that some scepticism exist around the security of ECDLP.

3.2 Other attacks to ECDH: Man in the middle attack

The ECDH is also concerned with other types of attacks than finding the shared secret key S. One of these is the man-in-the-middle attack, which we will look further into in this section.

A man-in-the-middle attack is an attack where the attacker secretly relays and possibly alters the communication between two parties, while they believe they are directly communication with each other. A third party, who is attacking, retrieves A's public key and sends it's own public key to B. Then, when B transmits his public key, the third party interrupt and substitutes the value with her own public key and sends it to A. Therefore, A has now come to an agreement on a common secret key with the third party instead of B. The exchange can be done in reverse. Therefore it is now possible for the third party to decrypt any messages sent out by A or B. It can read and possibly modify them before the re-encryption with the appropriate key, and then transmit them to the other party.

To address this attacking problem, generally a process of authentication will be needed. The public keys created in the key exchange are either static or ephemeral. A static key is intended for use for a relatively long period of time. Typically, it is intended for use in many instances of a cryptographic key establishment scheme. An ephemeral key is a key which is generated for each execution of a key establishment process. The ephemeral keys are not necessarily authenticated, and this is necessary to avoid man-in-the-middle attack. So, authenticity assurances must be obtained by other means.

If one of A or B's public key is static, man-in-the-middle attacks are thwarted. A secure communication protocol is said to have forward secrecy if compromise of long-term keys does not compromise past session keys. This protects past sessions against future compromises of secret keys or passwords. Static public keys provide neither forward secrecy nor key-compromise impersonation resilience. Therefor, to avoid leaking information about the static private key, holders

should validate the other public key and should apply a secure key derivation function to the raw Diffie-Hellman shared secret.[5]

4 Usage of ECDH in secure internet protocols

Among other security protocols, the Diffie-Hellman protocol has often been applied to SSL (Security Sockets Layer) and SSH (Secure Shell).

SSL (Security Sockets Layer) is the predecessor to TLS (Transport Layer Security) and they are both referred to as 'SSL'. SSL is the standard security technology developed to establish an encrypted link between a web server and a browser. The link should ensure privacy and integrity of all data passed between the web server and the browser. Before a client and server can begin to exchange information protected by SSL they must exchange or agree upon an encryption key and a cipher to use when encrypting data. The key and chiper must both have high security. Elliptic Curve Diffie-Hellman is one of the secure methods used for the key exchange.

SSL is composed of two layers, the lower layer which manages the symmetric cryptography so the communication is private and reliable, and the upper layer called the handshake protocol. Diffie-Hellman is used in this upper layer. It is possible to use several different Diffie-Hellman methods, in many cases Elliptic Curve Diffie-Hellman is preferable. The handshake allows the server to authenticate itself to the client using public-key techniques. This is also called asymmetric encryption. The key exchange process uses Diffie-Hellman to ensure each party that the other is who they say they are. After this exchange, the keys are computed and the parties begin encrypting all traffic between them, using the computed keys. SSL is among other uses useful for business traffic and to ensure confidentiality, authenticity and integrity.[4]

SHH is a cryptographic network protocol to allow remote login and other network services to operate securely over an unsecured network. It is often used and very common for secure login on the internet. This protocol can automatically encrypt, authenticate and compress transmitted data. SSH proceeds in three stages, the "hello" phase where the first identification is done. In the second stage the parties agree upon a shared secret key. This is where the ECDH method is implemented and used, and the secure key exchanges is done. Ordinary Diffie-Hellman can also be used. At the third and final stage, the shared secret key are used to generate the application keys.

SSH can be used to secure any network service, but common applications are remote command-line login and remote command execution. Among others, SSH can also be used for setting up automatic login to a remote server, for executing a single command on a remote host and secure file transfer.[4][6]

5 Article: How Diffie Hellman Fails in Practice

A newly published article from October 2015, written by Adrian, Bhargavan Durumeric et al., states that the Diffie-Hellman key exchange frequently offers less security than widely believed.[1] As previously described, Diffie-Hellman is the main key exchange mechanism in SSH and IPsec and a popular option in SSL, so security flaws related to this method are critical. The author of the article states his conclusion based on an examination of how Diffie-Hellman is commonly implemented and deployed with these protocols.

Mainly there are two reasons for that the Diffie-Hellman offers less security than widely believed. The first reason is that a lot of servers use weak Diffie-Hellman parameters. The second, more critical reason is that many use standardized, hard coded or widely shared Diffie-Hellman parameters. This dramatically reduces the cost of large-scale attacks such that some are within range of feasibility.[1]

The article contains complex discussions about the different attacks to be performed on the internet protocols that use Diffie-Hellman. The problems that are found stem from the fact that the Diffie Hellman (based on discrete log) allows an attacker to perform a single precomputation that only depends on the group. This is a well known fact for cryptographers, but has obviously not been fully understood by system builders.[1] The authors suggests some measures to be taken to defeat the problems in its Recommendations section. This conclusion is of great interest of elliptic curve cryptography, and is stated below.

"Transitioning to elliptic curve Diffie-Hellman (ECDH) key exchange with appropriate parameters avoids all known feasible cryptanalytic attacks."..."We recommend transitioning to elliptic curves where possible; this is the most effective long-term solution to the vulnerabilities described in this paper."[1]

The reason for this recommendation is the fact that current elliptic curve discrete log algorithms for strong curves do not gain as much of an advantage from precomputation. ECDH keys are also shorter than the Diffie Hellman based on "mod p", and the computation of the shared secret key is faster.[1]

6 Conclusion

In this report we have discussed the basic properties, the security aspects and the usage of the key agreement protocol Elliptic Curve Diffie Hellman (ECDH).

We divided the security of ECDH into two sections; one concerning the Elliptic Curve Diffie-Hellman problem (ECDHP), and other types of attacks. The ECDHP is essential to the security of the protocol in the way that the adversary shouldn't be able to compute the shared secret key S. The hardness of ECDH is closely related to the corresponding ECDLP, but the equality of the two is not determined. The ECDLP is believed to be infeasible by the state of today's computer technology, but there are some uncertainty associated with it because of the lack of mathematical proof for its hardness.

As a significant example of other types of attacks to the ECDH, we discussed the scenario of a man-in-the-middle attack. An avoiding strategy to this type of attack would in general be a process of authentication.

We also discussed the usage of ECDH in secure Internet protocols. Among other security protocols, the Diffie-Hellman protocols has often been applied to SSL (Security Sockets Layer) and SSH (Secure Shell). The ECDH is believed to be one of the most secure versions of the Diffie-Hellman, and is preferable in many cases.

Finally, we discussed the recently published article "How Diffie-Hellman fails in practice" from 2015. The recommended solution the authors suggests to the problems found, is transitioning to ECDH key exchange whenever possible. With appropriate parameters they state that all known feasible cryptanalytic attacks would be avoided. Based on our discussion for the security of the ECDH, this qualifies as a reasonable solution. On the other hand, because of the uncertainty associated with the ECDH and ECDLP, it may be questioned if this recommendation will be put into practice.

7 References

1. Adrian, Bhargavan, Durumeric et. al. *How Diffie-Hellman Fails in Practice* (2015) Available from: $https://weakdh.org/imperfect-forward-secrecy-ccs15.pdf(01-Nov-2015)$

2. Koç, Çetin Kaya *Elliptic Curve Cryptography Fundamentals.* Available from $http://cs.ucsb.edu/koc/ecc/docx/09ecc.pdf(21-Oct-2015)$

3. *Diffie–Hellman key exchange* (2015) Available from: $https://en.wikipedia.org/wiki/Diffie$

4. Ahmed, Sanjabi, Aldiaz et. al. *Diffie-Hellman and Its Application in Security Protocols* Available from $http://www.ijesit.com/Volume$

5. Ahmed, Sanjabi, Aldiaz et. al. *Man-in-the-middle attack* Available from $https://en.wikipedia.org/wiki/Man-in-the-middle_attack$

6. *Secure Secure Shell* Available from $https://stribika.github.io/2015/01/04/secure-secure-shell.html$

7. *Elliptic Curve Diffie-Hellman* Available from $https://en.wikipedia.org/wiki/Elliptic_curve_Diffie$

8. Hankerson et. al. (2004) *Guide Elliptic Curve Cryptography* University of Waterloo, Springer-Verlag, New York