ZERO TO ECDH IN 30 MINUTES

Illustrating the elliptic-curve Diffie-Hellman key-agreement scheme

Table of contents

Introduction

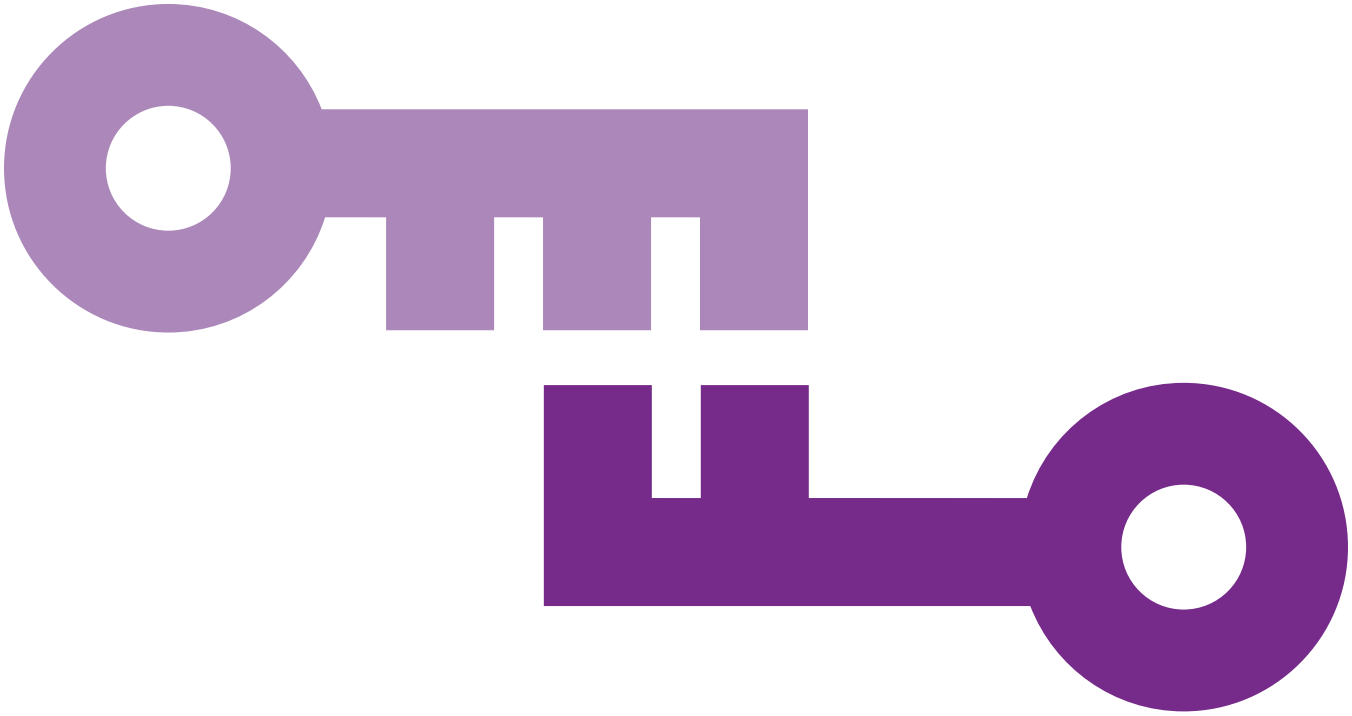
This is a quick primer on the elliptic-curve Diffie-Hellman (ECDH) key-agreement scheme. It provides a simple illustration of how the properties of elliptic-curve cryptography (ECC) can be used to build a useful security scheme.

Key agreement

"... key-agreement schemes are anonymous."

A key agreement scheme is a procedure by which two or more parties agree upon a value from which they can subsequently derive one or more keys for use in a symmetric encryption and/or data authentication scheme. Neither party completely determines the key value on their own. Instead, they both contribute to the final key value. And, most important, anyone who observes the exchanges between the two parties cannot tell what the final result will be.

It is important to remember that, in their basic form, key-agreement schemes are anonymous. In other words, they don't tell either party the identity of the other party (the one with whom they have agreed a key), nor whether that party is the one they believe it to be.



One-way function

"A keyed one-way function is a function that takes two inputs, one of which is private … and produces one output."

The original Diffie-Hellman key agreement scheme is based on multiplication of integers modulo a large prime number, specifically numbers greater than one and less than p, where p is a large prime. ECDH is an analogous scheme based on addition of points on an elliptic curve.

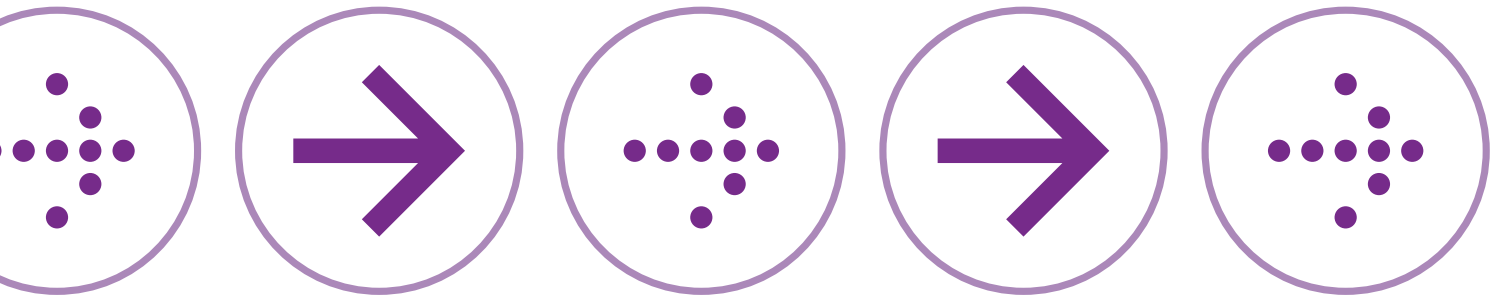
In both schemes, the basic operations are combined to create a primitive function known as a keyed one-way function. A keyed one-way function is a function that takes two inputs, one of which is private (e.g., the key), and produces one output. Given the two inputs, it must be straightforward to calculate the output.

But, it must be computationally infeasible to calculate the key, using only the other input and the output. In this way, each party can use their private key without revealing it to anyone else, either the other party or an eavesdropper.

In the original Diffie-Hellman scheme, the keyed one-way function is formed by multiplying the input to the function by itself repeatedly a number of times determined by the value of the key (i.e., raising the input to the power of the key).

Note that the input and output are integers modulo a large prime number, and the key is an integer. Raising a number to a power (i.e., exponentiation) in the group of integers modulo a large prime number is a relatively straightforward calculation, whereas the inverse operation (i.e., finding the power to which a known input must be raised in order to produce a known output) is computationally infeasible, if the prime modulus is sufficiently large.

Because calculating the inverse of exponentiation is called extracting the logarithm, this problem is known as the discrete logarithm problem.



Elliptic-curve one-way function

"The keyed one-way function is formed by adding the input to itself ..."

In the elliptic-curve Diffie-Hellman scheme, the input and output are points on the curve, while the key is an integer. See Figure 1. The keyed one-way function is formed by adding the input to itself, repeatedly, a number of times determined by the value of the key (i.e., multiplying the input by the key).

Multiplying a point by an integer is a relatively straightforward calculation, even for curves with a very large underlying field, whereas the inverse operation (i.e., finding out what multiple of a known input point produces a known output point is computationally infeasible, if the underlying field is sufficiently large).

The inverse of multiplication is, of course, division, so one might expect this problem to be called the elliptic-curve division problem. But, in fact, it is referred to as the elliptic-curve discrete logarithm problem.

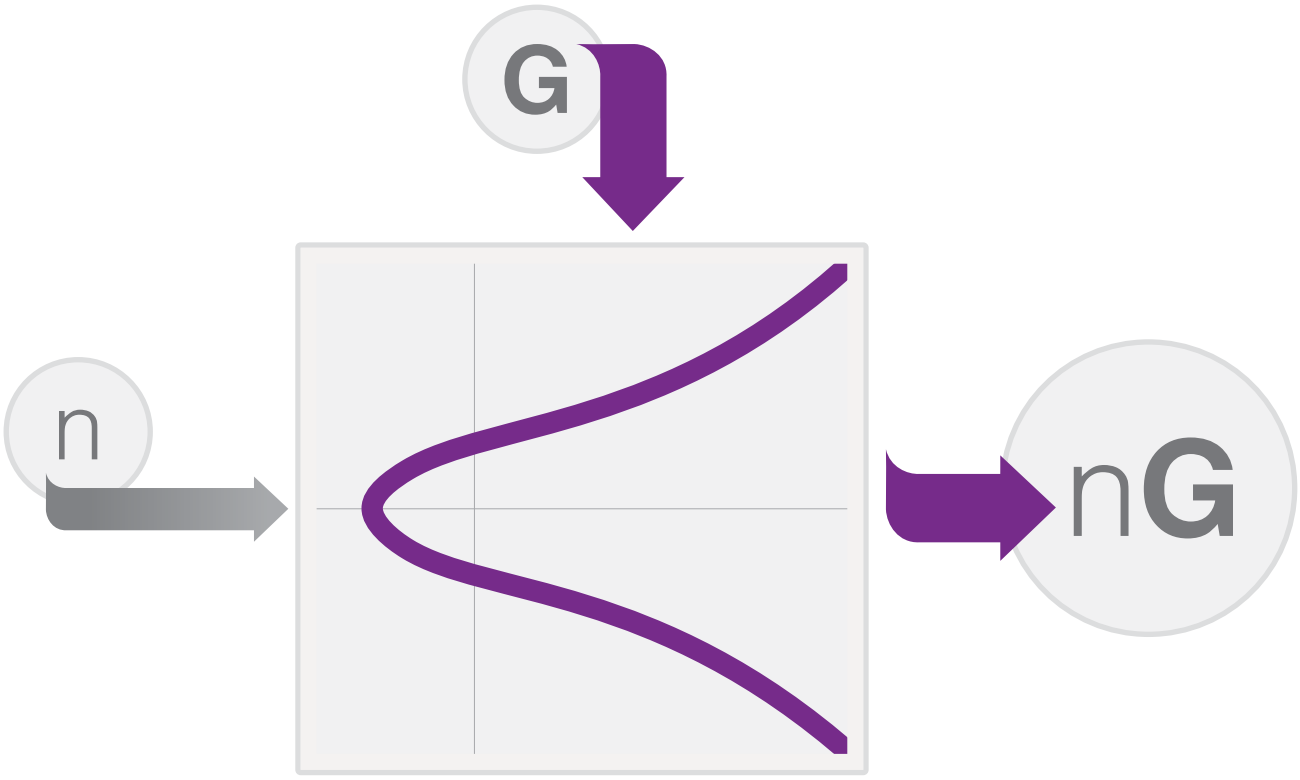


Figure 1: Elliptic-curve primitive

Elliptic-Curve Diffie-Hellman

"An eavesdropper may be able to observe the agreed parameters and may see the exchange of public keys. But, she is not able to determine what either private key is, nor the key that the two parties have agreed upon."

If two people go out to buy beer, and one buys four six-packs, while the other buys six four-packs, they will return with the same number of beers: 24. The order of the multiplication operations does not affect the result:

6 x 4 beers = 4 x 6 beers = 24 beers

It is this property of integer-point multiplication that allows the parties to the ECDH exchange to agree upon a key, while it is the one-way property of the elliptic-curve primitive that allows them to do this while keeping their keys and the result of the exchange secret from all but themselves.

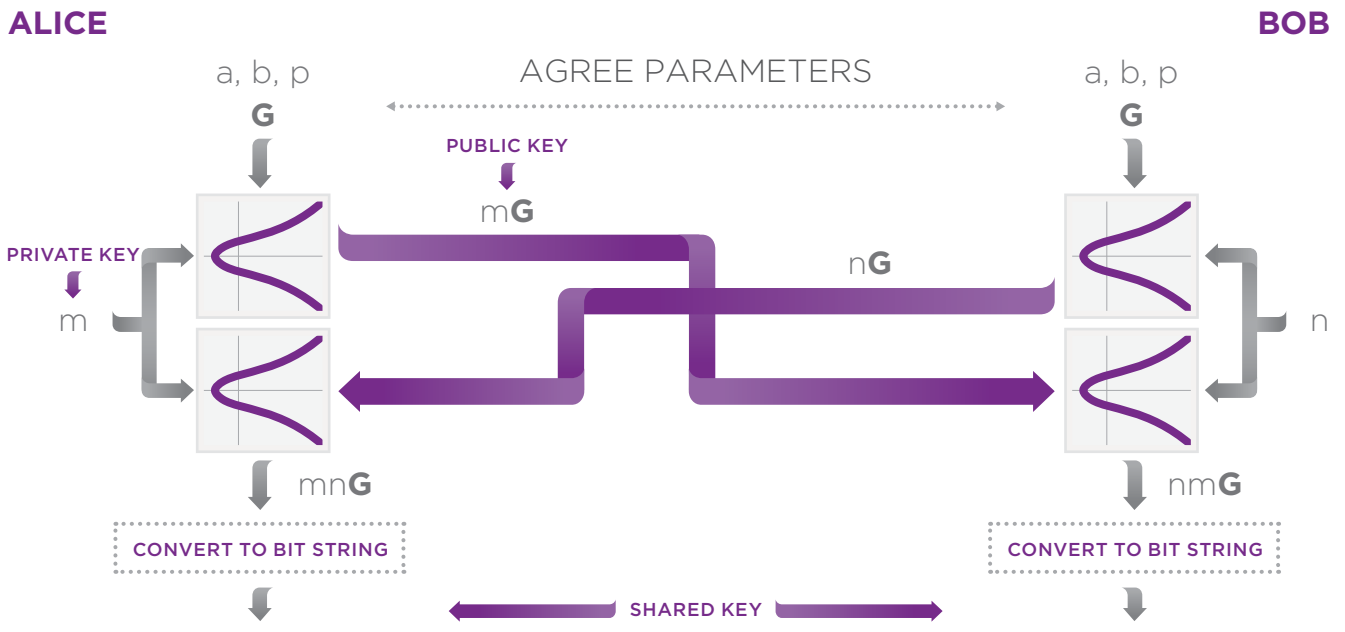
Before the protocol runs, the two parties must agree to system parameters, a, b, p and G. See Figure 2. This may be achieved by simply agreeing upon a standard set of parameters, such as those defined by the NIST P-256 standard.

Each party generates a random integer to use as its private key. For Alice, this is m, and for Bob it is n. Each then multiplies the base-point, G, by their private key to form a new point that represents their public key. Remember that each point comprises an x coordinate and a y coordinate.

They exchange their public keys and multiply the other's public key by their own private key. This produces a new point which is the same for each party. It remains only to convert this point to a bit string suitable for use as a key.

An eavesdropper may be able to observe the agreed parameters and may see the exchange of public keys. But, she is not able to determine what either private key is, nor the key that the two parties have agreed upon.

Figure 2: ECDH



Variants

"... ECDHE can provide Perfect Forward Secrecy ..."

As noted earlier, key-agreement schemes do not inherently provide entity-authentication. However, this limitation can be overcome by conveying the parties' public keys in the form of a certificate from a public-key certification authority.

The resulting scheme is called static ECDH, because the parties' key-pairs change only infrequently. And, because there is no random input to the scheme other than the parties' keys, the resulting agreed key is the same every time the same two parties communicate.

This is undesirable from a security point of view. And, for this reason, it is a common choice not to use certificates, but to generate new key pairs for each run of the protocol. This variant is called anonymous ECDH.

Because anonymous ECDH does not provide entity authentication, it is often used in combination with a separate authentication scheme, such as one based on digital signatures and certificates. This combination is called ephemeral ECDH, or ECDHE for short.

Used in this way, ECDHE can provide Perfect Forward Secrecy; meaning that disclosure of the key used to protect one message cannot lead to the disclosure of keys protecting other messages, and that there is no single secret value whose compromise can lead to the compromise of multiple messages.

Conclusion

Elliptic-curve cryptography is often described as fiendishly complicated. However, as this explanation of the most simple and elegant elliptic-curve cryptographic scheme illustrates, it can, in fact, be very easy to understand.

About Entrust Datacard

Consumers, citizens and employees increasingly expect anywhere-anytime experiences — whether they are making purchases, crossing borders, accessing e-gov services or logging onto corporate networks. Entrust Datacard offers the trusted identity and secure transaction technologies that make those experiences reliable and secure. Solutions range from the physical world of financial cards, passports and ID cards to the digital realm of authentication, certificates and secure communications. With more than 2,000 Entrust Datacard colleagues around the world, and a network of strong global partners, the company serves customers in 150 countries worldwide.

For more information about Entrust products and services, call **888-690-2424**, email **entrust@entrust.com** or visit **www.entrust.com**.

Headquarters
Entrust Datacard
1187 Park Place
Shakopee, MN 55379
USA

Twitter Facebook Youtube LinkedIn Google+ RSS