research plaza

news from leading universities and
research institutes in the Netherlands

```
15:22:35.684070 IP morgan.58007 > 157.55.130.140.40016: Flags [.], ack 2335, win
 327, options [nop,nop,TS val 4827411 ecr 2521054868], length 0
15:22:35.893256 IP 157.55.130.140.40016 > morgan.58007: Flags [P.], seq 2335:234
5, ack 4290, win 83, options [nop,nop,TS val 2521054928 ecr 4827411], length 10
15:22:35.893303 IP morgan.58007 > 157.55.130.140.40016: Flags [.], ack 2345, win
 327, options [nop,nop,TS val 4827463 ecr 2521054928], length 0
15:22:35.893433 IP morgan.58007 > 157.55.130.140.40016: Flags [P.], seq 4290:429
4, ack 2345, win 327, options [nop,nop,TS val 4827463 ecr 2521054928], length 4
15:22:36.090333 IP 157.55.130.140.40016 > morgan.58007: Flags [.], ack 4294, win
 83, options [nop,nop,TS val 2521054979 ecr 4827463], length 0
15:22:36.350279 IP morgan.47661 > tb-in-f113.1e100.net.http: Flags [F.], seq 850
732267, ack 3853135942, win 123, options [nop,nop,TS val 4827577 ecr 2907336134]
, length 0
15:22:36.350361 IP morgan.49997 > tf-in-f94.1e100.net.http: Flags [F.], seq 4576
, ack 38846, win 331, options [nop,nop,TS val 4827577 ecr 4051426219], length 0
15:22:36.354281 IP tb-in-f113.1e100.net.http > morgan.47661: Flags [F.], seq 1,
ack 1, win 264, options [nop,nop,TS val 2907451490 ecr 4827577], length 0
15:22:36.354316 IP morgan.47661 > tb-in-f113.1e100.net.http: Flags [.], ack 2, w
in 123, options [nop,nop,TS val 4827578 ecr 2907451490], length 0
15:22:36.354356 IP tf-in-f94.1e100.net.http > morgan.49997: Flags [F.], seq 3884
6, ack 4577, win 390, options [nop,nop,TS val 4051541455 ecr 4827577], length 0
15:22:36.354370 IP morgan.49997 > tf-in-f94.1e100.net.http: Flags [.], ack 38847
, win 331, options [nop,nop,TS val 4827578 ecr 4051541455], length 0
```

```
  /* 2^255 - 2^5 */ square(t1,t0);
  /* 2^255 - 21 */ mult(out,t1,z11);
}

int crypto_scalarmult(unsigned char *q,
  const unsigned char *n,
  const unsigned char *p)
{
  unsigned int work[96];
  unsigned char e[32];
  unsigned int i;
  for (i = 0;i < 32;++i) e[i] = n[i];
  e[0] &= 248;
  e[31] &= 127;
  e[31] |= 64;
  for (i = 0;i < 32;++i) work[i] = p[i];
  mainloop(work,e);
  recip(work + 32,work + 32);
  mult(work + 64,work,work + 32);
  freeze(work + 64);
  for (i = 0;i < 32;++i) q[i] = work[64 + i];
  return 0;
}
                                                265,1
```

RESEARCHERS

- Daniel J. Bernstein, Eindhoven University of Technology, the Netherlands, and University of Illinois at Chicago, USA
- Tanja Lange, Eindhoven University of Technology, the Netherlands
- Peter Schwabe, Academia Sinica, Taiwan.

High-security and high-speed protection for computer networks

SECURING COMMUNICATION

Internet and mobile communication has become a vital part of our lives in the past decade, but almost all of it is exposed to criminals. Researchers at the Eindhoven University of Technology have developed a new cryptographic library that is fast enough to allow universal deployment of high-security encryption.

We often assume that communication over the internet is just as secure as traditional forms of personal communication. We assume that we know who we are communicating with; we assume our conversations are private, that only the person we talk to can hear what we are saying; and we assume that what we are saying will reach the recipient without being modified.

The importance of these three aspects of security can be illustrated using a simple example of internet communication: buying and downloading a game from an online store. Users begin by accessing the online store, and want to be sure that they are in fact accessing the right website and not a look-alike that will take their money but not let them download the software. Users then submit their credit-card details or other banking information, and want to be sure that this information is protected from eavesdroppers who could misuse it. Users then download the purchased game, and want to be sure that they have the genuine product and not some kind of malware.

These essential requirements of communication over computer networks are ensured through *cryptographic protection*. *Encryption* is what provides communication with confidentiality, the assurance that transmitted information is only read by the recipient and not by an eavesdropper. Authentication of users and data is provided by *message-authentication codes* and *digital signatures*. The security of these functions relies on the fact that a legitimate user knows some secret information, a *key* unknown to attackers. If attackers somehow figure out this key, they can fully breach the system's security.

The scientific literature contains well-studied cryptographic functions for encryption and authentication that are believed to be secure. Security in this context is not absolute; all cryptographic protection used for internet communication can be broken by a large enough effort. However, even all of the world's supercomputers working together would take thousands of years or more to actually carry out the computations required to break a good cryptographic function.

For each of these functions there are various implementations in software, typically bundled into cryptographic libraries. Libraries are collections of software that can be used to integrate features into computer programs. The use of these established libraries in the development of programs that need cryptographic protection is now common best practice.

One might think that the security of network communication is now fully protected by well-established implementations of well-studied cryptographic functions. Unfortunately, quite the opposite is true, as demonstrated by frequent international news stories about new information-security disasters caused by failures of cryptography.

A NEW CRYPTO LIBRARY: NaCL

Researchers at the Eindhoven University of Technology are tackling these problems. Daniel J. Bernstein (also of the University of Illinois at Chicago, USA), Tanja Lange, and their former PhD student Peter Schwabe (now at the Academia Sinica, Taiwan) have identified the fundamental sources of security failures in existing cryptographic libraries. They have designed and implemented a new *Networking and Cryptography library* (NaCl, pronounced salt) that systematically avoids these failures.

USABILITY AND SELECTION OF FUNCTIONS

A typical cryptographic library is a collection of many different functions and supports a plethora of parameter sets. It is left to the software developer to choose from these functions and parameters, and combine them in a way that offers the desired security. These choices come with various pitfalls, not only because most libraries still contain highly insecure functions for 'historical' or 'compatibility' reasons, but also because it is easy to combine secure functions in an insecure way.

The Eindhoven researchers have found that this level of complication is unnecessary for most applications. NaCl offers an easy-to-use high-level interface for exactly what applications need: secure authenticated encryption. The underlying functions and parameters are chosen by experts in cryptography, namely the NaCl designers.

HIGH SPEED FOR HIGH SECURITY

Almost all internet communication is unencrypted and unauthenticated, leaving it completely unprotected against attacks. One might wonder why any programmer would fail to protect communication if free cryptographic libraries are readily available. The reason is often simply that cryptography is too slow; keeping up with high network loads requires many expensive computers with high electricity and maintenance costs. Analogous problems apply to smartphones and tablets, which have smaller network loads but also much smaller central processing units (CPUs) and limited battery life.

Sometimes, rather than not deploying cryptographic protection at all, programmers react to performance problems by deploying low-security cryptography. Many cryptographic libraries allow trade-offs between security and performance. The Eindhoven researchers are world leaders in evaluating the security of cryptography; they have found that many cryptographic systems can be breached using the level of computer power that is readily available today to rogue governments, large companies and botnets, and that will soon be available to attackers with far fewer resources at their disposal.

As stated above, NaCl does not provide any low-security options; its choice of functions is very conservative. It nevertheless offers exceptionally high speed, keeping up with even very large network loads. The Eindhoven researchers selected the functions in NaCl with close attention to software performance, and developed highly optimized implementations of those functions for a broad spectrum of commonly used CPUs, ranging from powerful Intel server CPUs down to energy-efficient ARM smartphone CPUs. Their implementations hold various speed records published at international conferences (see box 'What's under the hood?').

FUNCTIONS AND IMPLEMENTATIONS

Imagine a computer program that reads two numbers x and y from the user, multiplies x by itself to obtain x^2, multiplies y by itself to obtain y^2, and subtracts the results to obtain $x^2 - y^2$. Now imagine a second computer program that reads two numbers x and y from the user, adds x to y to obtain $x + y$, subtracts y from x to obtain $x - y$, and multiplies the results to obtain $x^2 - y^2$.

These two pieces of software are two different implementations of the same mathematical function. The function produces $x^2 - y^2$, given x and y. The implementations compute this function in different ways, with different speeds: the first implementation uses two multiplications and a subtraction, while the second implementation uses one multiplication, one addition, and one subtraction.

Cryptography uses more complicated functions. Each function has a wide range of implementations, and those implementations vary dramatically in speed.

WHAT'S UNDER THE HOOD?

The core of NaCl is *public-key authenticated encryption*, consisting of three components:

- the Curve25519 Diffie–Hellman key-exchange function, based on fast arithmetic on a strong elliptic curve, computes a secret shared between the sender and receiver, using the sender's secret key and the receiver's public key (or vice versa);
- the Salsa20 stream cipher, which has been recommended by ECRYPT after four years of extensive study in the eSTREAM project, encrypts a message using the shared secret; and
- the Poly1305 *message-authentication code,* a fast function that is information-theoretically secure if used together with a secure cipher, authenticates the encrypted message using the shared secret.

End-to-end two-party communication is not the only communication scenario that requires high-security cryptographic protection. NaCl also has a fourth component, the Ed25519 public-key signature system, for unforgeable and undeniable broadcast communication.

SIDE-CHANNEL SECURITY

Even when information-security systems use high-security cryptographic functions and use them in the right way, they may not steer clear of cryptographic failures. The reason is that a particular implementation of a secure function can be insecure.

Timing attacks are a powerful attack strategy targeting implementations. An attacker measures the time that the legitimate user takes to perform some procedures involving the secret key. If this time depends on the key, the attacker may be able to deduce information about the key.

This type of attack has been known since 1996 when Paul Kocher (Cryptography Research, USA) introduced it as part of a larger class of attacks called *side-channel attacks*. Since then, the power of these attacks has been demonstrated many times in practice. Maybe the most impressive result was presented in 2006 by cryptographers Adi Shamir and Eran Tromer (Weizmann Institute of Science, Israel) and Dag Arne Osvik when they used a timing attack to discover, in 65 milliseconds, the secret key used in widely deployed software for hard-disk encryption.

In principle it is possible to implement *constant-time* software for every cryptographic function. Constant-time software means software whose running time does not depend on secret data. However, for many cryptographic functions this comes with huge performance penalties. This is why most cryptographic libraries are still vulnerable to timing attacks.

The NaCl designers carefully selected the functions used in NaCl to allow constant-time implementations without huge performance penalties. All implementations in NaCl are constant-time implementations; they are thus inherently protected against timing attacks.

USERS

The researchers' long-term aim is to have the entire internet secured by NaCl. Although this target might be

PUBLIC KEY CRYPTOGRAPHY

Since the beginning of cryptography more than 2000 years ago, users who wanted to communicate securely needed to first agree on a secret key. This secret key needed to be transmitted from one user to another in person, or through a pre-existing secure channel.

In 1976, Whitfield Diffie and Martin Hellman from Stanford University, USA proposed *public-key encryption* as a way for two users to communicate securely through a *public* channel, with no secret keys shared in advance. The sender encrypts data using a *one-way function*, i.e. a function that is easy to compute but hard to invert. The receiver specifies the one-way function with a secret *trapdoor*, allowing the receiver to invert the function and decrypt the data. The sender does not know the trapdoor.

Diffie and Hellman suggested the following specific method for users to agree on a secret key through a public channel. User *A* picks a secret integer *a*, computes the power g^a, where g is a standard group generator, and sends g^a to user *B*. User *B* picks a secret integer *b*, computes the power g^b, and sends g^b to user *A*. Now *A* computes the common key as $(g^b)^a = g^{ab}$; *B* computes the same key as $(g^a)^b = g^{ab}$. An eavesdropper sees only g^a and g^b being transmitted; a successful attacker would have to compute g^{ab} from these two values, for example by computing a as the logarithm of g^a base g. If g is a point on a strong *elliptic curve* then this logarithm computation is extremely difficult.

years and many committee decisions away, NaCl already has an expanding user base as an easy-to-use tool for standalone projects that provide both sides of the secured communication. Here are two examples of projects using NaCl.

First, iPhones and other iOS devices encrypt (with the user's unlock password) files stored on SD cards, so that a criminal who steals a phone cannot read the files stored on it (as long as the unlock password is long and unguessable). This poses an interesting challenge when an iPhone writes a file, such as a mail attachment downloading in the background, while the phone is locked. In effect, the iPhone is talking to itself, using the SD card as a communication channel, and needs to be able to encrypt data without being able to decrypt it. Apple uses the Curve25519 component of NaCl to solve this problem (see box, 'Public key cryptography').

Second, web browsers locate web servers by sending queries to the internet's Domain Name System (DNS): the query is the server's name, and the response is the server's contact information. OpenDNS, a company with its headquarters in San Francisco, USA, handles billions of DNS queries a day from millions of computers around the internet. OpenDNS automatically uses *DNSCurve* to encrypt and authenticate communication to servers that announce DNSCurve support, and *DNSCrypt* to encrypt and authenticate communication to users who install the DNSCrypt software freely available from OpenDNS. DNSCurve uses the Curve25519, Salsa20, and Poly1305 components of NaCl; DNSCrypt also uses the Ed25519 component of NaCl. Earlier this year, two months after the introduction of DNSCrypt, OpenDNS announced that DNSCrypt was already in use by tens of thousands of users who have downloaded the software to their personal computers. ■

RESEARCH ORGANIZATION

Coding Theory and Cryptography, Eindhoven Institute for the Protection of Systems and Information (Ei/ψ), Eindhoven University of Technology, the Netherlands.

PROJECT WEBSITE

The Networking and Cryptography (NaCl) library, together with extensive documentation, is available at http://nacl.cr.yp.to
The library has been placed in the public domain, and avoids all known patents.

SOURCE PUBLICATION

- Bernstein, D.J., Lange, T. and Schwabe, P. (2012) The security impact of a new cryptographic library. In: A. Hevia and G. Neven (Eds.): *LATINCRYPT 2012, Lecture Notes in Computer Science* 7533. Berlin, Springer, pp. 159–176.
- Bernstein, D.J. and Schwabe, P. (2012) NEON crypto. In: E. Prouff and P. Schaumont (Eds.): *CHES 2012, Lecture Notes in Computer Science* 7428. Berlin, Springer, pp. 320–339.

OTHER REFERENCES

- Bernstein, D.J., Duif, N., Lange, T., Schwabe, P. and Yang, B-Y (2012) High-speed high-security signatures. *Journal of Cryptographic Engineering* 2(2): 77–89. (Short version: (2011) Cryptographic hardware and embedded systems. *CHES 2011*. *Lecture Notes in Computer Science* 6917. Berlin, Springer, pp. 124–142.
- Bernstein, D.J. (2008) The Salsa20 family of stream ciphers. In: M. Robshaw and O. Billet: New stream cipher designs: the eSTREAM finalists. *Lecture Notes in Computer Science* 4986. Berlin, Springer, pp. 84–97.
- Bernstein, D.J. (2006) Curve25519: New Diffie–Hellman speed records. In: M. Yung, Y. Dodis, A. Kiayias, and T. Malkin (Eds.), *PKC 2006, Lecture Notes in Computer Science* 3958. Berlin, Springer, pp. 207–228.
- Bernstein, D.J. (2006) The Poly1305-AES message-authentication code. In: H. Gilbert and H. Handschuh (Eds.), *FSE 2005, Lecture Notes in Computer Science* 3557. Berlin, Springer, pp. 32–49.

FUNDING AND CONTRIBUTIONS

- Development of NaCl was funded by EU FP7 project 216499, Computer-Aided Cryptography Engineering (CACE), and is now supported by EU FP7 project 216676, European Network of Excellence in Cryptology (ECRYPT II). The ECRYPT website is www.ecrypt.eu.org
- NaCl includes software contributed by Matthew Dempsky (Mochi Media, now at Google); Niels Duif (Eindhoven University of Technology); Emilia Käsper (KU Leuven, Belgium, now at Google); Adam Langley (Google); and Bo-Yin Yang (Academia Sinica).

COLOPHON

Publisher: Rutger Engelhard
Managing editor: Valerie Jones
English-language editor: Mark Speer
Layout: Anita Weisz-Toebosch
Editorial assistant: Britt Myren
Website: Stephan Csorba, David Ennis

Research Plaza is a project of Contactivity bv,
Stationsweg 28, 2312 AV Leiden
www.contactivity.com