Integrating Elliptic Curve Cryptography into the Web's Security Infrastructure

Vipul Gupta
Sun Microsystems, Inc.
vipul.gupta@sun.com

Douglas Stebila
University of Oxford
douglas@stebila.ca

Sheueling Chang Shantz
Sun Microsystems, Inc.
sheueling.chang@sun.com

ABSTRACT

RSA is the most popular public-key cryptosystem on the Web today but long-term trends such as the proliferation of smaller, simpler devices and increasing security needs will make continued reliance on RSA more challenging over time. We offer Elliptic Curve Cryptography (ECC) as a suitable alternative and describe our integration of this technology into several key components of the Web's security infrastructure. We also present experimental results quantifying the benefits of using ECC for secure web transactions.

Categories and Subject Descriptors: E.3 [Data Encryption]: Public key cryptosystems; H.4.3 [Information Systems Applications]: Communications Applications – *Information browsers, Electronic mail*; C.2.2 [Computer Communication Networks]: Network Protocols – *Applications*

General Terms: Security, Standardization, Experimentation.

Keywords: Elliptic Curve Cryptography, Mozilla, Apache, OpenSSL

1. INTRODUCTION

The Internet today is a truly global marketplace, with a wide variety of goods and services available online. Secure communication is an intrinsic requirement for many popular online transactions such as e-commerce, stock trading and banking. These transactions employ a combination of *public-key* and *symmetric-key* cryptography to authenticate participants and guarantee the integrity and confidentiality of information in transit. Our focus, for this paper, is on public-key cryptography as it is the main bottleneck in Internet security protocols including the most popular — SSL [2].

RSA is the most widely used public-key technology today but the following trends will make continued reliance on RSA more challenging over time:

- *More and simpler connected devices:* In the last several years, the Internet has grown rapidly beyond servers, desktops and laptops to include handheld devices like PDAs and smart phones. We expect this trend to continue as increasing numbers of even simpler, more constrained devices (sensors, home appliances, personal medical devices) get connected to the Web. Many of these devices will not have the computational resources to provide adequate levels of security using RSA.
- *More transactions requiring security:* More and more individuals and businesses continue to establish and expand their Internet presence, resulting in ever larger volumes of online transactions. In the future, not only will there be more of the kinds of transactions that already demand security, heightened privacy concerns will drive the need to secure additional kinds of transactions, *e.g.* book lovers might demand confidentiality of their browsing habits, not just their credit card numbers, at Amazon.com. This will significantly increase the cost of supporting RSA computations.
- *Demand for higher levels of security:* As processor speeds increase and the number of Internet-connected devices grows, potential attackers will have more resources at their disposal to attack the cryptography used in secure connections. This will necessitate further increases in RSA key sizes and worsen the performance bottleneck. Already, 512-bit RSA is considered insecure and most transactions use 1024-bit keys. Before the end of this decade, RSA keys will need to grow to 2048-bits [4]

These trends highlight a clear need for an efficient public-key cryptosystem that can: (i) lower the capability threshold for small devices to perform strong cryptography, and (ii) increase a server's capacity to handle secure connections. Elliptic Curve Cryptography (ECC) [5] promises to fulfill this need.

Compared to traditional counterparts (RSA, Diffie-Hellman and DSA), ECC offers the same level of security using much smaller keys [6]. Smaller keys result in faster computations and savings in memory, power and bandwidth that are especially important in constrained environments. More importantly, ECC's performance advantage increases as security needs increase over time.

Recently, the National Institute of Standards and Technology approved ECC for use by the U.S. government. Several standards organizations (*e.g.* IEEE, ANSI, OMA IETF) have ongoing efforts to include ECC as a required or recommended security mechanism.

2. SEEDING ECC ADOPTION

For any new security technology to be widely adopted, it must be integrated into end-user applications like email and web browsing. Interoperable standards, both at the algorithm (ECDH, ECDSA) and protocol (S/MIME, SSL) levels are essential prerequisites. Most importantly, the new technology must demonstrate a compelling value proposition to offset the cost and inconvenience of migration. Our research group is addressing all of these issues with the goal of creating a vibrant, complete ecosystem around ECC which promises to play a significant role in the Web's future security needs.

2.1 Standardization

Besides HTTP, SSL is used to secure many other protocols including IMAP, SMTP and LDAP and we view the standardization of ECC in SSL as critical. The first author of this paper is also

WWW2004, May 17–22, 2004, New York, New York, USA.
ACM 1-58113-912-8/04/0005.

the lead author of the internet-draft specifying new ECC cipher suites for SSL [3]. This specification describes the use of ECDH key agreement in the SSL handshake and ECDSA signing as an authentication mechanism. Special provisions are included to accommodate constrained devices that may only implement a small subset of available curves or point formats.

2.2 Open Source Software

We have integrated ECC technology into OpenSSL, Apache, Netscape Security Services (NSS) and Mozilla [7]. The OpenSSL and NSS libraries include implementations of cryptographic algorithms, and the SSL and S/MIME protocols. OpenSSL is used by Apache, the most widely used web server, and other applications including Lynx, Apple's Safari and Ximian's Evolution. NSS powers the security features in the Mozilla and Netscape browsers as well as Sun Microsystems' web, directory and messaging servers.

Our contributions to NSS and Mozilla include — (i) a highly optimized elliptic curve math library supporting both GF(p) and GF(2^m) curves, and ECDH/ECDSA algorithms, (ii) support for generating, storing, importing and exporting elliptic curve keys and certificates in the NSS key and certificate stores, (iii) client- and server-side support for ECC cipher suites in SSL and ECDSA support in S/MIME, (iv) support for EC key type in the KEYGEN tag handling code to make deployment of elliptic curve certificates just as easy as RSA certificates.

Our contributions to OpenSSL and Apache include — (i) an elliptic curve math library over GF(2^m) fields, (ii) the ECDH key exchange mechanism, (iii) client- and server-side support for ECC cipher suites in SSL and (iv) a patch to mod_ssl for exposing the new cipher suites to Apache.

ECC enabled versions of OpenSSL and NSS are available from our web site [8] and can be used to establish a secure connection between Mozilla and Apache with ECC. Additionally, the Mozilla email client can now sign/verify messages using ECDSA.

3. PERFORMANCE IMPACT OF ECC

This section briefly summarizes our experimental results on replacing RSA with ECC in secure web transactions. We used Apache 2.0.45 compiled with OpenSSL-SNAP-20030309 on a 900 MHz UltraSPARC III. For RSA, we used 1024- and 2048-bit keys, representing current and future security needs; the equivalent ECC key sizes are 160- and 224-bits, respectively [6].

RSA decryption and ECDH operations represent the most computationally intensive portion in a typical SSL handshake. Performance measurements for these operations (Table 1) illustrate the advantage of ECC over RSA and show that advantage growing as security needs increase.

In addition to public-key computations, secure webservers incur the cost of message parsing, file system accesses, and symmetric-key encryption and hashing. Several of these depend on the amount of data transferred. SSL also allows for session reuse which amortizes the cost of a public-key operation across multiple connections. These considerations make a server's overall performance highly dependent on its workload characteristics. For our experiments, we used http_load to synthesize a workload based on Badia's sampling of six secure web sites [1]. The survey found aggregate page sizes in the range 10KB to 70KB with a 30KB median and identified two primary usage models impacting session reuse. For the *"shopping cart"* model, on average, a new session is created for every three page fetches (66% reuse). For the *"financial institution"* model, a new session is created for every eight pages (87.5% reuse).

We discovered that replacing RSA with ECC reduces the server's processing time for new SSL connections across the entire range of page sizes from 10KB to 70KB. The measured reduction ranges from 29% for a 70KB page comparing ECC-160 with RSA-1024 up to 85% for a 10KB page comparing ECC-224 with RSA-2048. In our experiments, it was not until we increased the file size to 1MB that we noticed ECC and RSA performing comparably. At these sizes, factors other than public-key costs (*e.g.* network saturation) dominate.

Table 1: Measured performance of public-key algorithms.

	ECC-160	RSA-1024	ECC-224	RSA-2048
Ops/sec	271.3	114.3	195.5	17.8
Speed up	2.4 : 1		11 : 1	

Figure 1 shows server throughput for 30KB page accesses under different session reuse values. As expected, increasing session reuse decreases the impact of choosing any particular public-key cryptosystem. However, even with session reuse as high as 87.5%, an ECC based server handles 13% more requests compared to RSA at current key sizes and 120% more at future key sizes. For 66% reuse, the performance advantage due to ECC increases to 31% for ECC-160/RSA-1024 and 279% for ECC-224/RSA-2048.

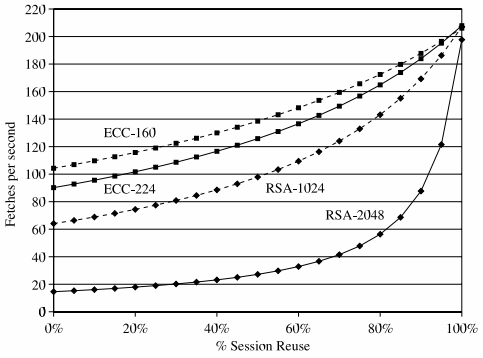

Figure 1: Throughput v/s session reuse plot.

4. CONCLUSION

Our experiments show significant performance benefits from using ECC in secure web transactions. Due to its computational efficiency, ECC can be used in constrained environments where traditional public-key mechanisms are simply impractical. As larger numbers of smaller, simpler devices connect to the Web and users grow more sensitive to security and privacy concerns, there will be a greater demand for efficient cryptographic techniques like ECC. We have contributed this technology to OpenSSL, Apache, NSS and Mozilla with the aim of jumpstarting its widespread adoption.

5. REFERENCES

[1] L. Badia, "Real World SSL Benchmarking", Rainbow Technologies Whitepaper, Sep. 2001.

[2] A. Frier, P. Karlton, P. Kocher, "The SSL3.0 Protocol Version 3.0", 1996.

[3] V. Gupta, S. Blake-Wilson, B. Moeller, C. Hawk, "ECC Cipher Suites for TLS", IETF internet draft <draft-ietf- tls-ecc-05.txt>, Jan. 2004.

[4] B. Kaliski, "TWIRL and RSA Key Size", RSA Labs Tech Note, May 2003.

[5] N. Koblitz, "Elliptic curve cryptosystems", *Mathematics of Computation*, 48:203-209, 1987.

[6] A. Lenstra and E. Verheul, "Selecting Cryptographic Key Sizes", *Journal of Cryptology* 14 (2001) pp. 255-293.

[7] See openssl.org, apache.org and mozilla.org.

[8] See http://research.sun.com/projects/crypto