Sizzle: SSL on Motes

Vipul Gupta, Sun Labs

(Joint work with S. Chang Shantz, H. Eberle, S. Fung*, N. Gura, M. Millard*, A. Patel*, A. Wander*, M. Wurm*, Y. Zhu*)

*Student intern

CENTS Retreat,
Granlibakken Conference Center,
Tahoe City, Jan 12-14, 2005

Sun microsystems

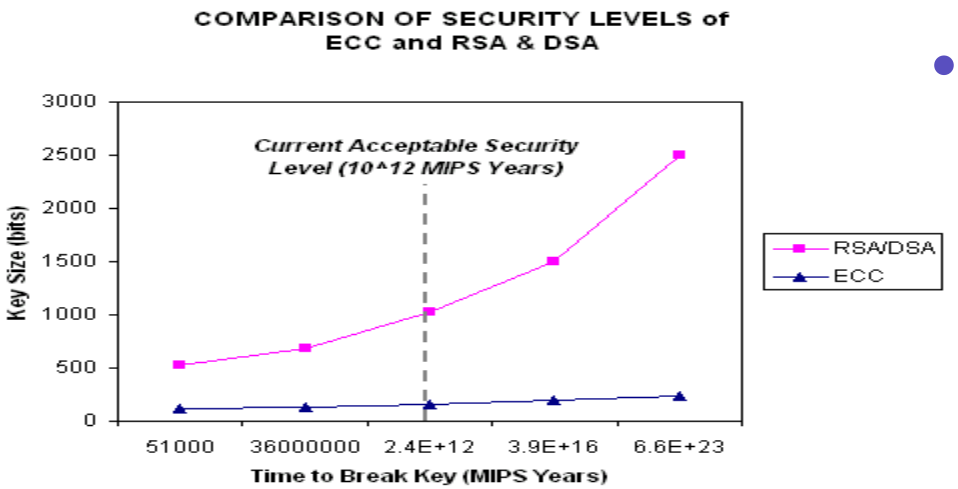
Outline

- Sensor network security background
- Elliptic Curve Cryptography (ECC) overview
- Sizzle (Slim SSL) – HTTPS server on motes
- Demo
- Conclusion

Sensor Network Security

- General perception: public-key cryptography is impractical
- Previous symmetric-key based approaches:
  - Key distribution problem
  - Link level security (not end-to-end)
  - Compromising a few nodes jeopardizes security of entire network
- Sizzle: Standards-based end-to-end security architecture (ECC + SSL)

Elliptic Curve Cryptography



Sym.	RSA	ECC	Ratio	MIPS yrs
80	1,024	160	**6:1**	10^{12}
112	2,048	224	**9:1**	10^{24}
128	3,072	256	**12:1**	10^{28}
192	7,680	384	**20:1**	10^{47}
256	15,360	521	**30:1**	10^{66}

- Computationally highly efficient public-key cryptosystem, highest security strength per bit
  - Savings in memory, bandwidth, power
  - Advantage improves as security needs increase
- Endorsed/standardized by NIST, ANSI, IEEE, IETF
- Good match for AES

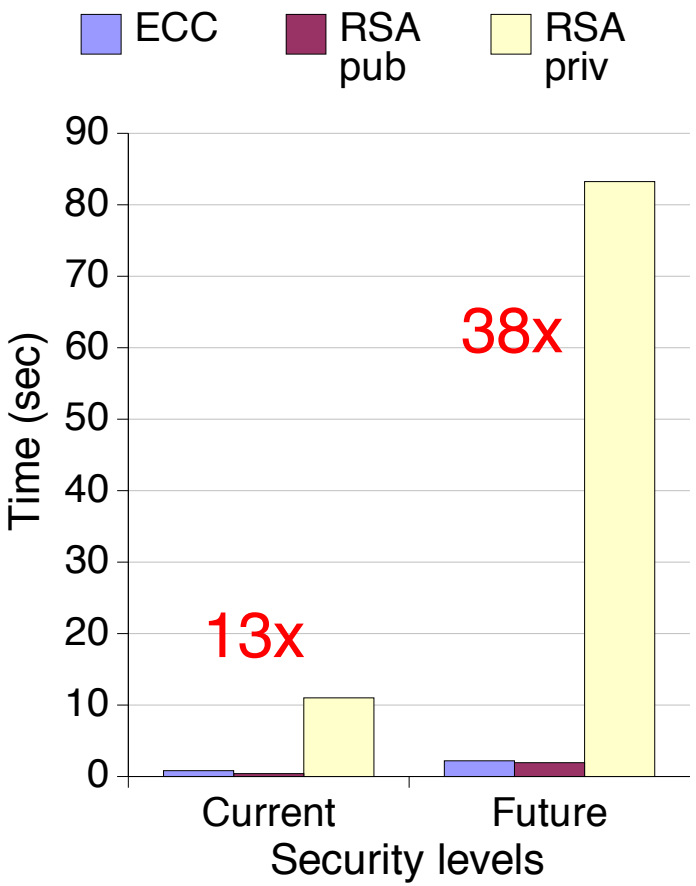
More information: http://research.sun.com/projects/crypto/

ECC on Small Devices

Berkeley/Crossbow
MICA "mote"
(8-bit, Atmel
ATmega processor,
128KB FLASH, 4KB SRAM,
4KB EEPROM)

Algorithm	Time* (s)	Data bytes	Code bytes
ECC secp160r1	0.81	282	3682
ECC secp224r1	2.19	422	4812
RSA 1024 (pub**)	0.43	542	1073
RSA 1024 (priv)	10.99	930	6292
RSA-2048 (pub**)	1.94	1332	2854
RSA-2048 (priv)	83.26	1853	7736

* 8MHz Atmel ATmega ** e=65537



More information: Gura et al., CHES 2004 paper

Sizzle Overview

- World's smallest secure web server
- Uses ECC key exchange in SSL*
- Interoperates with ECC-enabled Mozilla/Firefox/OpenSSL
- Lowers barrier for connecting interesting new devices to the Internet, and controlling/ monitoring them securely

*Based on IETF internet-draft draft-ietf-tls-ecc-xx.txt

Sizzle Features

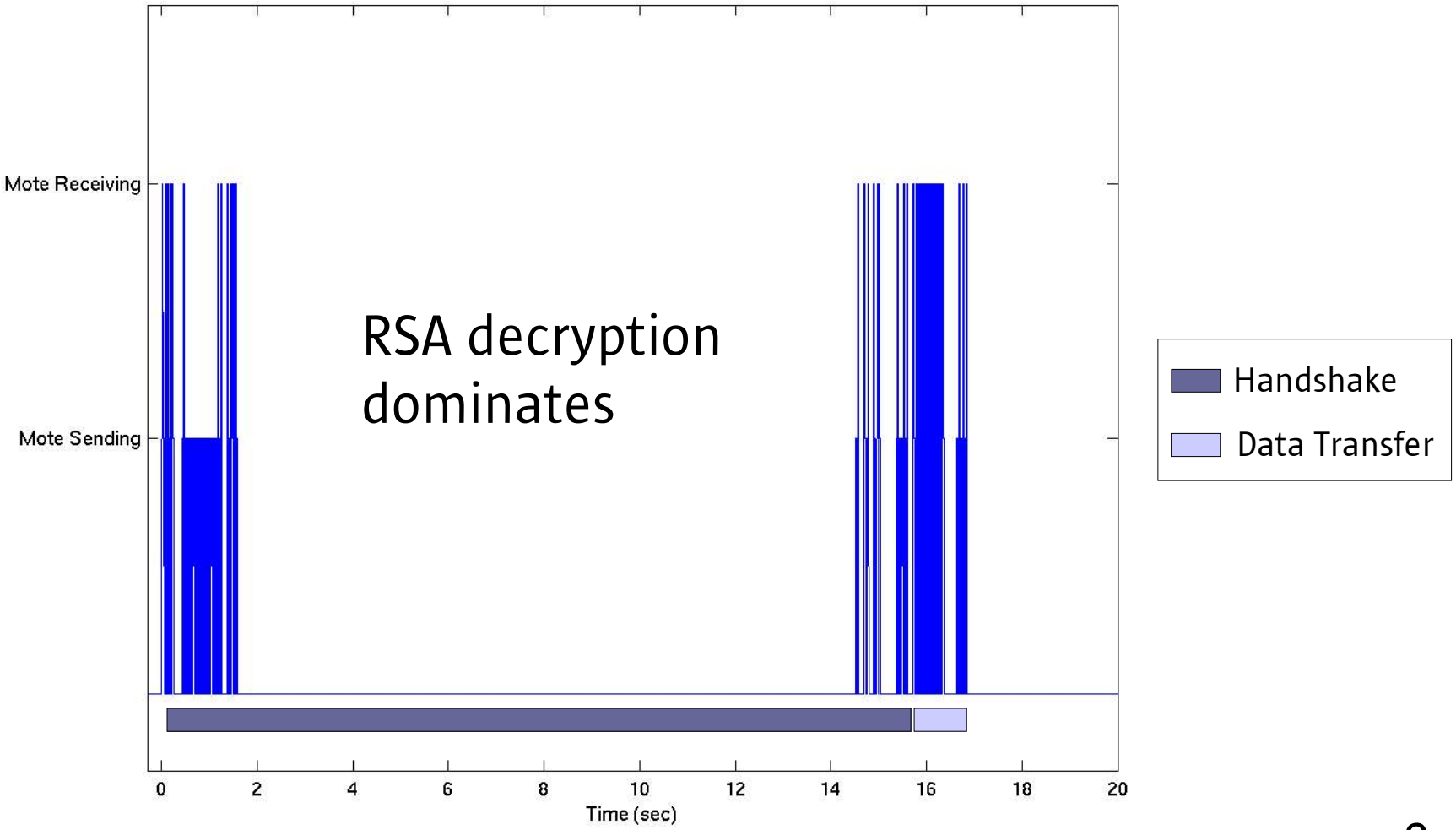
- Uses 160-bit ECC (on curve secp160r1)
- ECDH-ECDSA-RC4-SHA cipher suite
- Minimizes SRAM memory usage and SSL handshake overhead, *e.g.*
  - Static info stored in program memory
  - Small session identifiers, certs
  - Implements session reuse, persistent HTTP(S)

Sizzle Architecture and Statistics

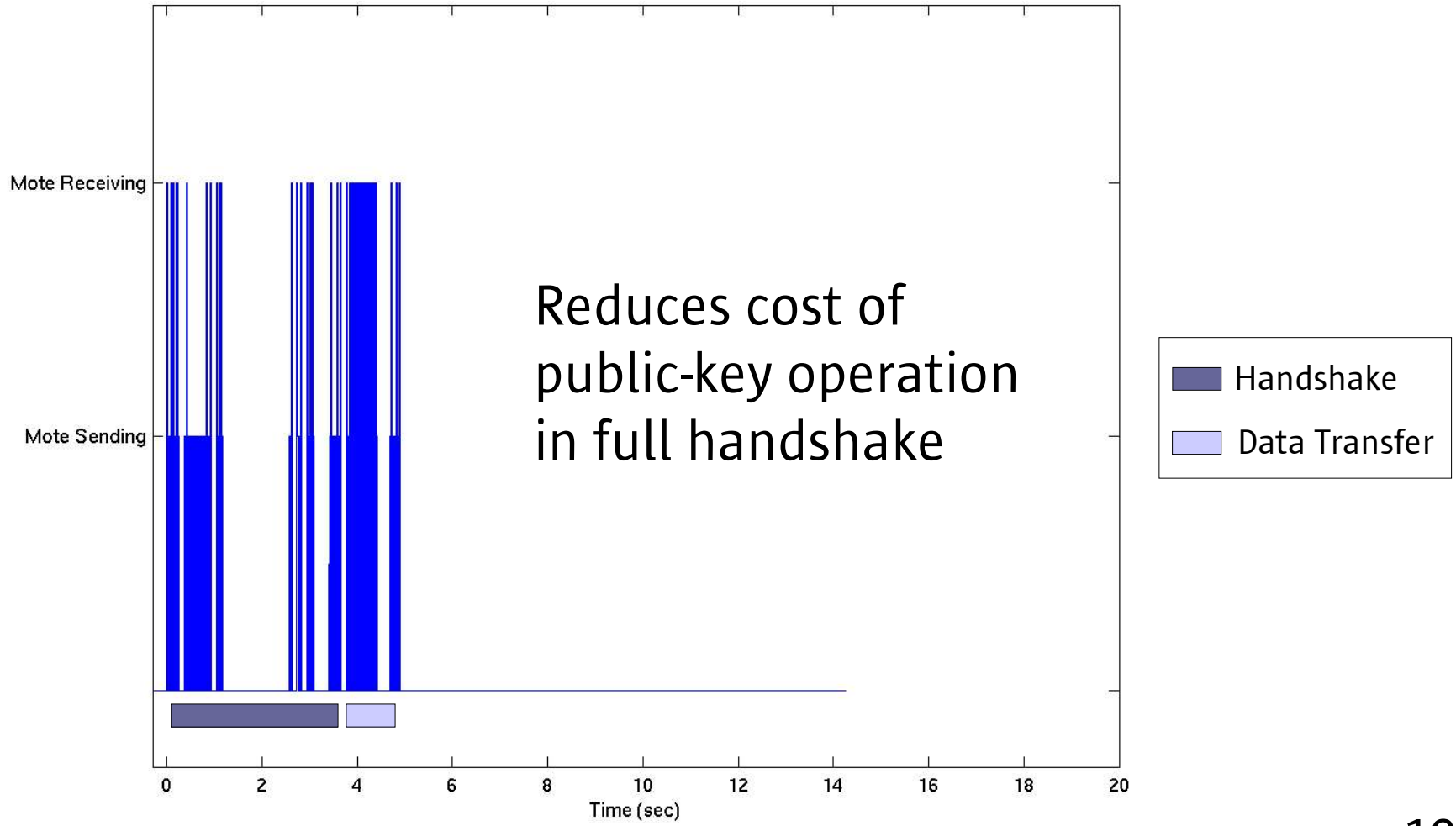
- Memory usage from objdump: ~3KB (RAM), ~60KB (FLASH) on Mica2 mote
- Page load time in sec (450-byte HTTPS transfer on Mica2 w/ Tiny OS 1.1.6):

Full Handshake		Session Reuse	Persistent HTTP(S)	Plain HTTP
RSA	ECC			
16.8	4.9	2.9	1.1	0.9

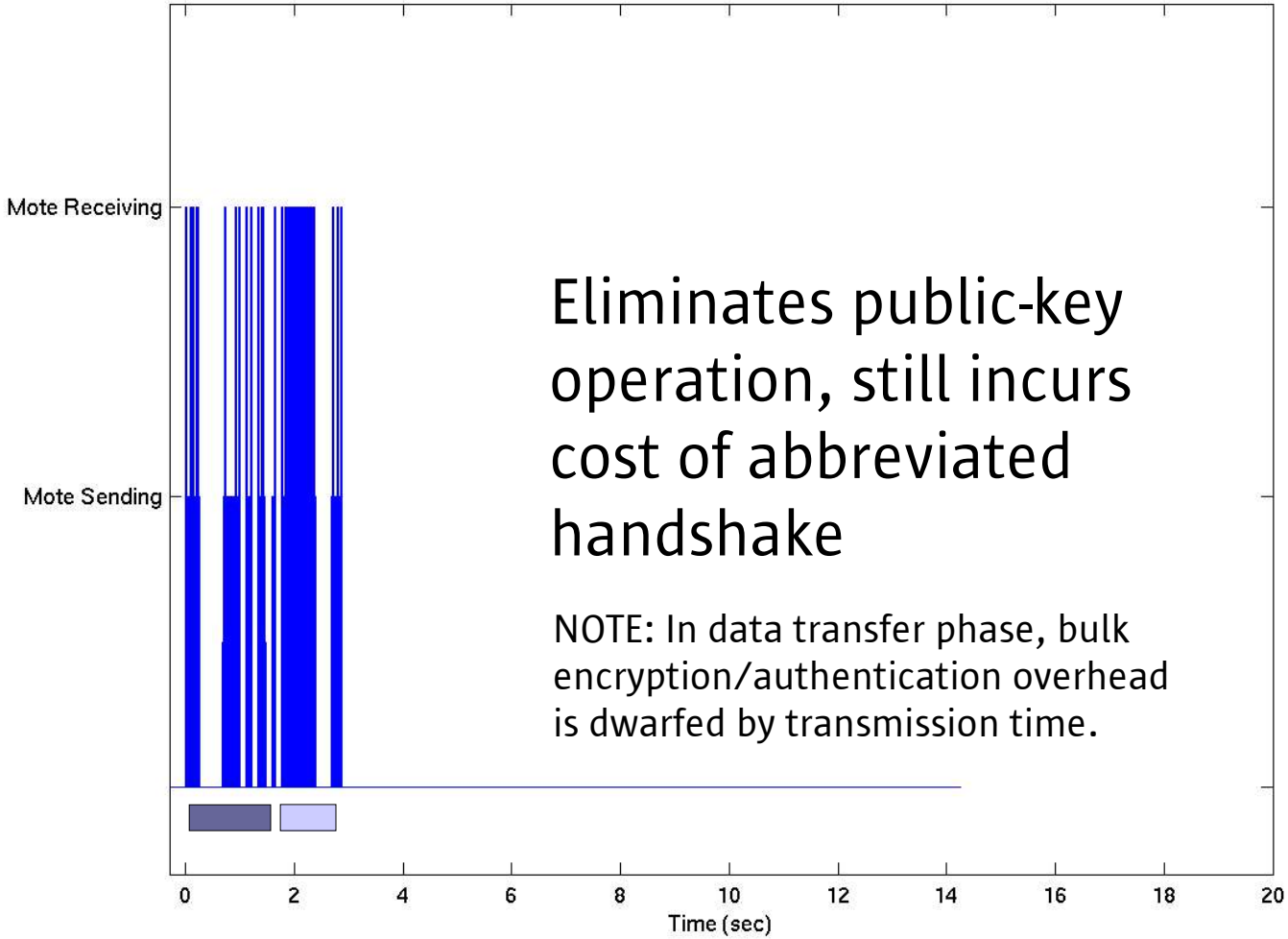
Performance Details (RSA)



Performance Details (ECC)



Performance Details (Session Reuse)



Eliminates public-key operation, still incurs cost of abbreviated handshake

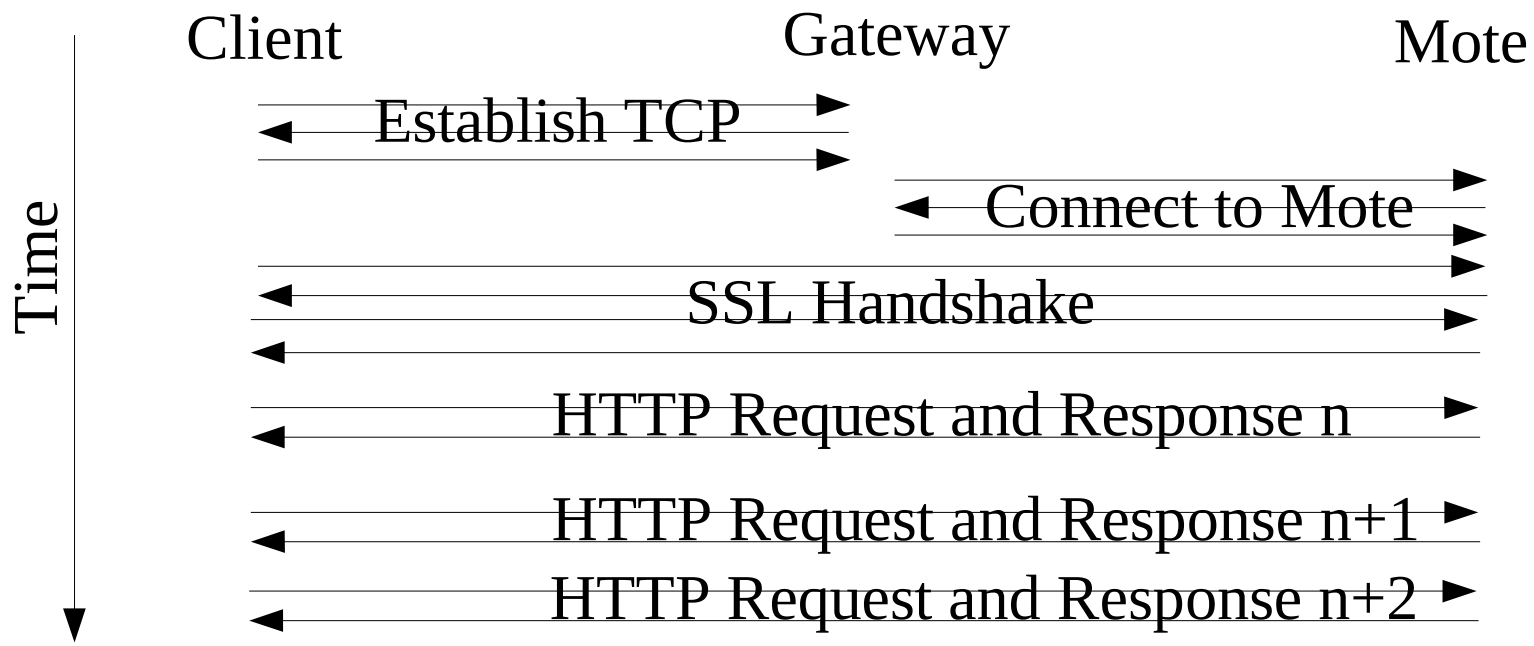
NOTE: In data transfer phase, bulk encryption/authentication overhead is dwarfed by transmission time.

Handshake

Data Transfer

Performance Details (Persistent HTTPS)

- Amortizes the cost of an SSL handshake (full or abbreviated) across multiple data transfers



Sizzle Demonstration

- ECC-enabled Mozilla communicating with Sizzle
- Secure monitoring and control of a “wireless thermostat”
- Comparison of ECC v/s RSA-based handshake
- Impact of session reuse and persistent HTTP(S)

Takeaway

Elliptic Curve Cryptography (ECC) makes public-key cryptography feasible on mote-like devices and creates the opportunity to reuse standard security protocols on the “embedded” Internet.

References

- V. Gupta et *al*., “Sizzle: A Standards-based end-to-end Security Architecture for the Embedded Internet”, PerCom 2005, Mar. 2005*
- N. Gura et *al*., “Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs”, CHES 2004, Aug. 2004
- V. Gupta et *al*., “ECC Cipher Suites for TLS”, IETF internet-draft, Dec. 2004
- V. Gupta et *al*., "Integrating Elliptic Curve Cryptography into the Web's Security Infrastructure", WWW 2004, May 2004

*Mark Weiser Best Paper Award at PerCom 2005

sheueling.chang@sun.com
hans.eberle@sun.com
vipul.gupta@sun.com
nils.gura@sun.com

http://research.sun.com/projects/crypto