

# F3ildCrypt: End-to-End Protection of Sensitive Information in Web Services

Matthew Burnside and Angelos D. Keromytis

Department of Computer Science  
Columbia University  
{mb, angelos}@cs.columbia.edu

ISC 2009

# Motivation

- Identity-related information is valuable
- You must provide such information when using an online merchant
- This information is vulnerable to disclosure at the endpoints and in transit
- Can we protect this information end-to-end without revealing details of the logical corporate architecture?

# Outline

- 1 Introduction
- 2 Related work
- 3 Architecture
- 4 Evaluation
- 5 Conclusion

# Introduction

Users have to trust online merchants:

- Merchant is not malicious
- Merchant will always protect sensitive information
- Merchant site is maintained by diligent sysadmins

In Service Oriented Architectures, users have to trust:

- Merchant *and peer SOAs* are not malicious
- Merchant *and peer SOAs* will always protect sensitive information
- Merchant *and peer SOAs* are maintained by diligent sysadmins

# Data in transit

- In this work, we focus on data in transit
- Approach does not protect against nodes with legitimate access to the data
- We protect the data from the web browser to the back-end database

# Design alternative

## Pair-wise key distribution

- Generate a certificate for each potential target host in the SOA pipeline
- Deliver certificate set to each web browser
- Browser encrypts each field direct to its destination host

# Pair-wise key distribution

- Certificates for all hosts in any partner SOAs must also be delivered
- Certificate set must be updated each time the architecture of the SOA or SOA partners varies
- Reveals the logical architecture of the SOA and its SOA partners



# Related work: Proxy re-encryption

For all plaintext  $P$ , Alice  $\langle pk_A, sk_A \rangle$ , Bob  $\langle pk_B, sk_B \rangle$ :

$$pk_B(p) = rk_{A \rightarrow B}(pk_A(P))$$

- [Blaze et al., 1998]
- [Ateniese et al., 2005]

- Introduced end-to-end encryption in web pipelines
- Firefox plugin for application-level crypto
- Requires disclosure of corporate network details

# Architecture

- Design goals
- F3ieldCrypt architecture
- Example session

# Network model

- SOA-style network
- Each SOA may have multiple partner SOAs
- SOAs wish to prevent disclosure of logical architecture and peering

# Threat model

- XXX

# Design goals

- End-to-end protection of XML fields – even across SOA boundaries
- Confidentiality of logical architecture of each SOA must be respected

*Do not provide protection against compromise or failure of entities with legitimate access to sensitive information.*

# F3ieldCrypt architecture

- Each SOA  $s$  publishes a public key  $pk_{E_s}$
- Browser  $b$  generates plaintext  $P$
- $b$  sends  $C = pk_{E_s}(P)$
- At  $s$ , proxy re-encrypt  $C$  to internal hosts and partner SOAs  $0...n$

# Key generation

- Key pair  $\langle pk_{E_s}, sk_{E_s} \rangle$  generated at the external-key holder
- Public keys of applications  $pk_{I_0} \dots pk_{I_n}$  are collected
- Used in conjunction with  $sk_{E_s}$  to generate  $rk_{E \rightarrow I_0} \dots rk_{E \rightarrow I_n}$



## F3ieldCrypt architecture (cont.)

- By proxy re-encryption:

$$pk_{I_j}(P) = rk_{E \rightarrow I_j}(pk_E(P))$$

- Keys  $rk_{E \rightarrow I_0} \dots rk_{E \rightarrow I_n}$  stored at **proxy re-encryption engine**

# Proxy re-encryption engine

- Fields arrive at PRE encrypted under  $pk_{E_s}$
- Each field  $f$  is re-encrypted under  $pk_{E \rightarrow I_j}$
- The mapping  $f \rightarrow j$  is determined from a XACML policy

# Client policy and crypto engines

Web clients receive a re-cryptography engine and a policy engine.

- **Policy engine** uses a XACML policy to determine which fields to encrypt
- **Re-crypto engine** encrypts XML fields as directed by the policy engine.

# Architecture summary

- XXX: insert image

# Implementation

- Java-based Re-crypto engine based on JHU-MIT Proxy Re-cryptography Library for each web client
- Python-based XML proxy for each internal application to store keys and unwrap XML
- XML gateway at the SOA stores the re-encryption engine

# Testbed servers

## Dell PowerEdge 2650 Servers

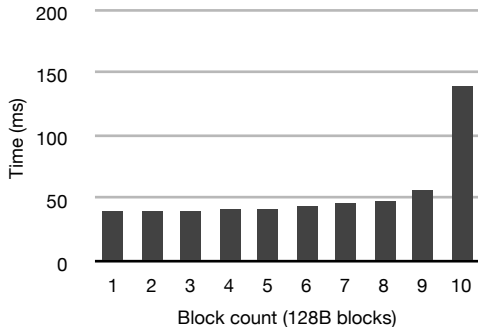
- 2.0GHz Intel Zeon processor, 1GB RAM, Gigabit Ethernet
- OpenBSD 4.2
- OpenBSD PF firewall, Apache 1.3.29, PHP 4.4.1, MySQL 5.0.45

# Testbed client

## Macbook Pro

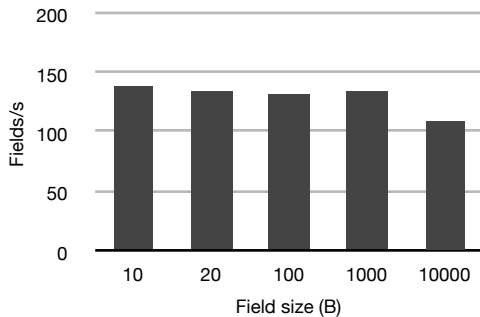
- 2.4 GHz Intel Core 2 Duo, 2GB RAM, Gigabit Ethernet
- OS X 10.5.2, Darwin kernel 9.2.2, Mozilla Firefox 2.0.0.13

# Block encryption on the client

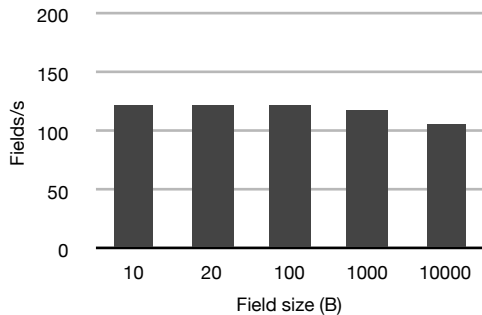




# Re-encryption rate at an XML gateway



# Decryption rate at an XML proxy



# Conclusion

- End-to-end protection to users and SOAs
- XXX



G. Ateniese, K. Fu, M. Green, and S. Hohenberger.

Improved proxy re-encryption schemes with applications to secure distributed storage.

*In Proceedings of the 12th Annual Network and Distributed Systems Security Symposium (NDSS 2005), 2005.*



Matt Blaze, G. Bleumer, and M. Strauss.

Divertible protocols and atomic proxy cryptography.

*In Proceedings of Eurocrypt '98, pages 127–144, 1998.*



Angelos Stavrou, Michael Locasto, and Angelos Keromytis.

W3bcrypt: Encryption as a stylesheet.

*In Proceedings of the 4th Applied Cryptography and Network Security Conference (ACNS 2006), pages 349–364, 2006.*