



# ***Survivable Trust for Critical Infrastructure***

**David M. Nicol, Sean W. Smith, Chris Hawblitzel,  
Ed Feustel, John Marchesini, Bennet Yee\***

Cybersecurity Research Group  
Institute for Security Technology Studies, Dartmouth College  
Dartmouth College

\* UCSD



# Secure Coprocessor

A computing device attached to a host

- that is tamper-resistant
- that contains a cryptographic accelerator, and keys (loaded at the factory) private even from its host

Such a device can provide authentication

- Authentication of identity
- Authentication of computational result
- Protection against insider attack

Such a device is a Trustable Third Party (TTP)

IBM sells one, the 4758. Smith on design team.



# Survivable TTP

We are exploring a *survivable* TTP to support trust applications in critical infrastructure

- communications, power, water, transportation

Create a network of TTPs

Survivability derives from

- distribution
- ubiquity
- obscurity



# Marianas\*

- A peer-to-peer network of Islands (host + secure coprocessor).
- Protects against insider attack, even though the hosts may be compromised.
- Network gives a means of authenticating results of remote computation, even though the host may not be trusted.

\*a chain of islands in the Pacific Trust Territory



# Application Areas

- Survivable, interoperable, self-organizing PKI
- Distributed control systems
- Distributed maintenance
- Highly available forward-secure logging
- Secure network Weather Service
- Highly available certificate revocation
- Shell game with critical information
- Privacy in delayed binding networks



# Research Issues

- Myriad issues in trust management, beyond the scope of this talk
- Issues in peer-to-peer design
  - Dynamic and multi-dimensional trust relationships alter the notion of “peer”
    - Trust might be broken by software upgrade
    - A known neighbor may be trustworthy for only certain types of interactions
    - A neighbor’s host may be suspected of having been compromised
  - Alternatives for query response include
    - Playing dead
    - Forward, but don’t participate
    - Warn others of suspected rogue

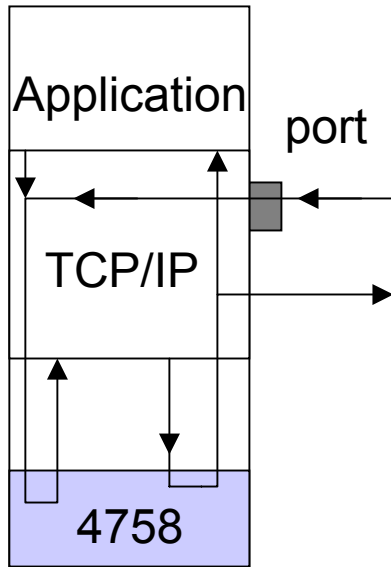


# Prototype

- We are developing a protocol which allows islands to communicate in a secure fashion.
- The protocol is entirely decentralized, modified from Gnutella.
- Since there are no network facilities in the card, we put the packet processing logic inside secure hardware, and used TCP/IP services provided by the host to ship the packets.



# Communication: High level view



Host side : send/receive ciphertext

Card side : packet processing, crypto, routing, p2p logic

Vulnerable "only" to denial of service

Island



# Software - Gnutella

- Gnutella is a *protocol* that is used to locate information over a decentralized network.
  - Five packet formats (descriptors).
  - A few general routing rules.
  - That's it!
- Gnutella is **not** a piece of software, a network, or a file sharing service.



# Why We Chose It

- We were only interested in the protocol and a servent (SERVer + cliENT) for a starting point.
- We were **not** interested in locating mp3's, our initial app. locates fragments of cryptographic keys.
- We took the Gnut (v0.4.25) servent and split it in two. One piece to implement the host side, and the other was ported to run inside the card.
- Gnut is open source, professionally coded, well documented, and runs on Linux.



# Armor via Authentication

- We began to add some of the protections by using the services made available by the card.
- 1) Authentication – for our definition of secure, we need mutual authentication
  - We implemented FIPS 196, which is the core of most authentication protocols in use (e.g. SSL).
  - Card-side software entities can authenticate and even if an intruder were watching the transaction, they would still not be able to get any important information.



# Armor via Encryption

- 2) Plans for encryption - Once two cards have authenticated, we plan to encrypt all subsequent traffic.
  - The parties need to agree upon symmetric keys to be used, and encrypt all packets with the keys.
  - We have the design, and are in the process of implementing this.
- Our command line Gnut servent - Once we have finished the encryption, we plan to make available for download our servent.



# Application : Ephemeral Private Key

Imagine a CA whose private key is partitioned with distributed replicates---gnutella used to retrieve and assemble

Protection :

- No island has entire key
- Suspicion of rogue CA may trigger withholding of key fragement

We will explore mediated RSA / threshold crypto



# Summary

We're exploring the join of secure coprocessing  
with peer-to-peer

Lots of interesting thesis problems here

Lots of significant applications

Stop on by :

[www.ists.dartmouth.edu/cybersecurity/index.htm](http://www.ists.dartmouth.edu/cybersecurity/index.htm)