

# The Security of Trusted Computing

Rick Wash

April 13, 2004

## What is Trusted Computing?

- Historical context
  - Trusting the software (kernel, root)
  - How do you know what's running?
- Hardware-assisted security
  - Do things that can only be done in hardware
  - Minimal hardware
- Requires software support also

## Implementations

- *Microsoft's* Palladium / NGSCB
  - Stuff
- *IBM / TCG's* TCGA

## TCGA Functions (\*)

- Public Key Functions
  - Key Generation
  - Signature
  - Encryption / Decryption
- Trusted Boot
  - Hash Functions
  - Platform Configuration
- Initialization and Management

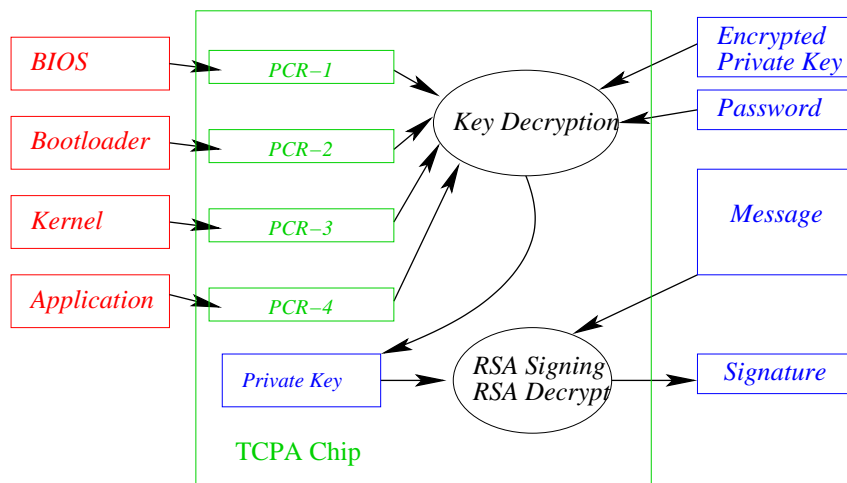
## TCPA Chip (\*)

Functional Units	Non-volatile Memory	Volatile Memory
RNG	Endorsement Key (2048b)	RSA Key Slot 0 ●●●
Hash	Storage Root Key (2048b)	RSA Key Slot 9
HMAC	Owner Auth Secret (160b)	PCR-0 ●●●
RSA Key Generation		PCR-14
RSA Encrypt / Decrypt		Key Handles
		Auth Session Handles

## TCPA Threat Model

- Key Management / Security
  - Keys **never** leave chip unencrypted
- System State Security Model
- **NOT** secure against physical attack

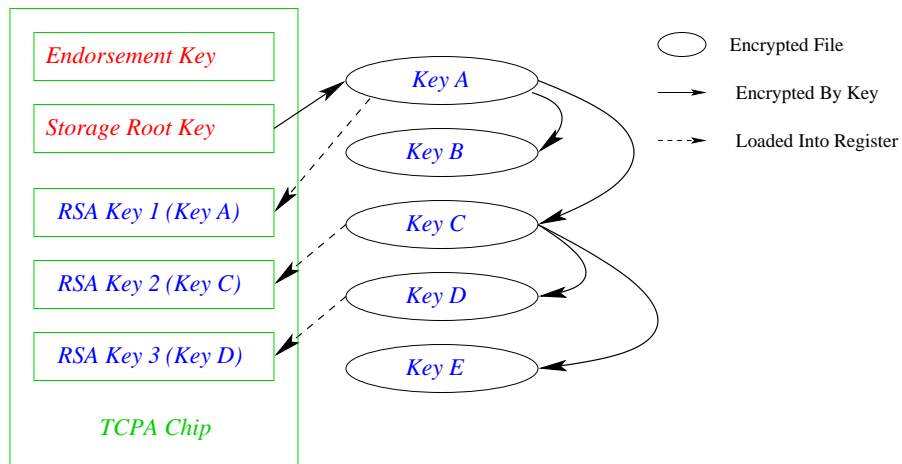
## Secure Boot / Signatures (\*)



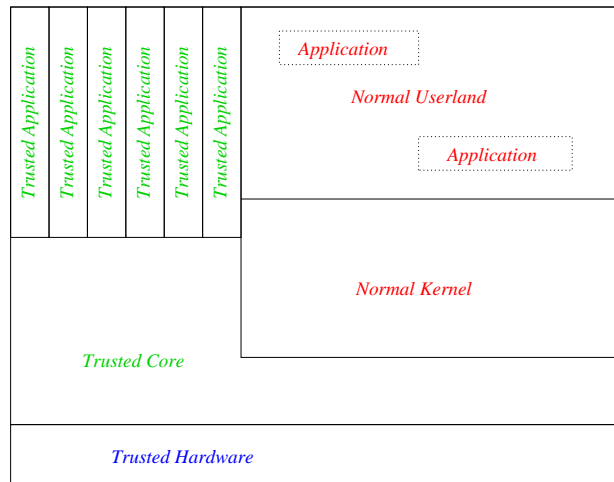
- Walk through the boot process
- Walk through loading a key

- Walk through signing a message
- Explain why this is more secure

### Tree of Keys (\*)



### Trusted OS Architecture (\*)



- VM-type layer under the OS
- Trusted apps run in separate space

### Higher-level Primitives

- Trusted Boot

- Attestation
- Key Management

## Trusted Boot

- Used to verify system state
- Enable recovery from compromise
- ALA tripwire
- Can also prevent access if incorrect state
  - encrypt binary
  - bind key to PCR's
- **Local** computer

## Attestation

- **Remote** computer statement about this machine
- Used for authentication
  - Of Computer
  - Of State
  - Of Person
- Enables to be built:
  - DRM
  - Verify State of Patches
  - Remote Authentication

## Key Management

- Keys not stored in memory
- Limited to encryption services on chip
  - Currently only RSA

## Challenges

- Security Model
  - Can you really trust it if its not physically secure?
  - What security is provided by secure boot?
  - What security is provided by attestation?
  - What benefits does trusted key management have?
- Key Mobility?
- Multi-user environment?

## My Work

- Drivers for NetBSD, OpenBSD
  - IBM has Linux drivers
- Integration with SSH, Kerberos, OpenSSL, KCA
- Key Mangement System ala *Factotum*
- Credential Translation

## What it Does NOT Do (\*)

- Prevent Buffer Overflows, etc.
- Make Software More Secure
- Eliminate Open Source
- Prevent Copying