TCPA Programming in Linux

Dr. David Safford
Mgr., Global Security Analysis Lab
IBM Research

Outline

- Resources
- What is TCPA?
- Getting Started
- Programming in Linux
- Next Steps

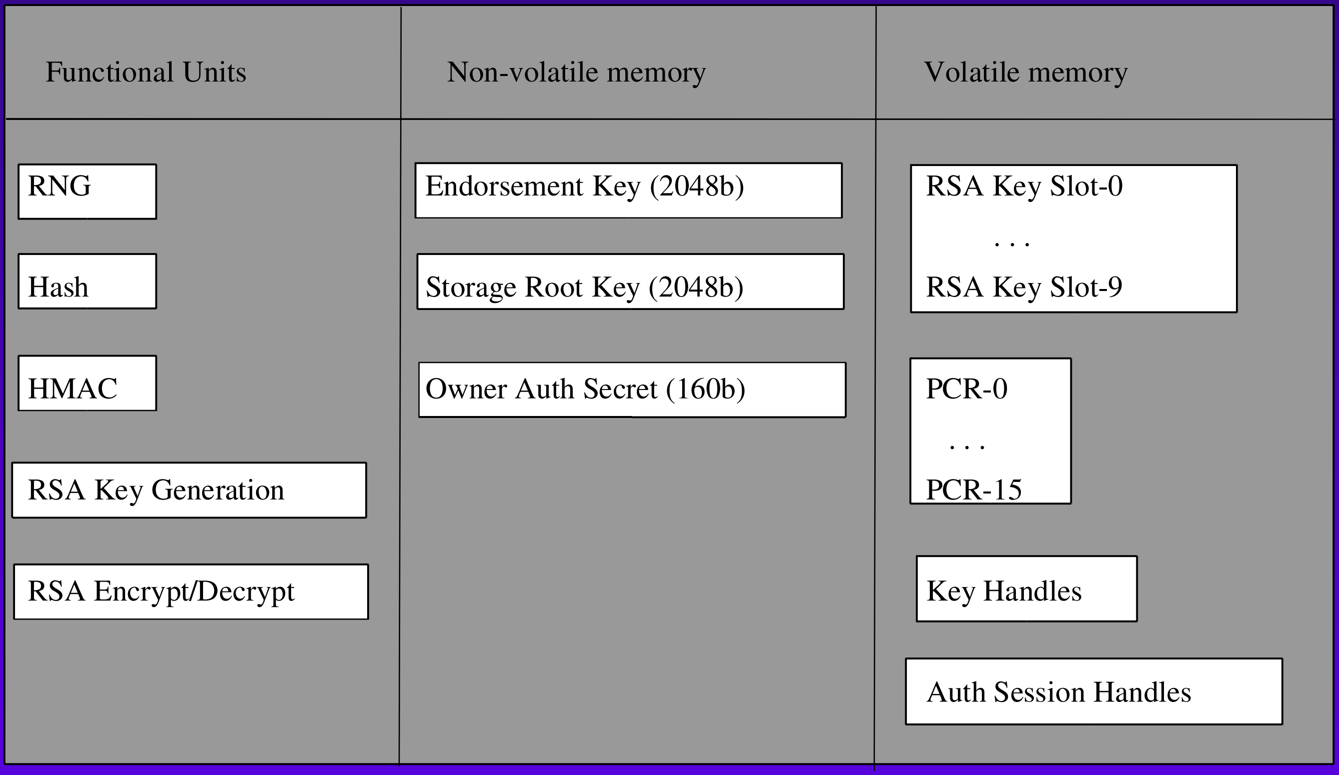
Resources:

- Trusted Computing Platform Alliance (TCPA) home page and main specification v1.1b:
  - http://www.trustedcomputing.org
- Trusted Computing Group (TCG) home page:
  - http://www.trustedcomputinggroup.org
- Research external page (papers, linux driver):
  - http://www.research.ibm.com/gsal/tcpa
- Research internal page (tutorial/libtcpa/driver):
  - http://dr.watson.ibm.com/gsal/tcpa

What is TCPA?:

- Officially IBM Embedded Security Subsystem (ESS) 2.0
- Shipped October 2002 on Thinkpads, desktops
- Hardware RNG, on chip RSA key generation
- RSA signature, encryption/decryption
- Non-volatile key storage
- PCR (platform configuration register)
  - Register extend – state hashed into 160 bit register.
  - Seal and unseal data depending on PCR value
  - Wrap and unwrap keys dependent on PCR
  - can be used for trusted boot, secure key release

Programming view of the TPM



Functional Units

RNG

Hash

HMAC

RSA Key Generation

RSA Encrypt/Decrypt

Non-volatile memory

Endorsement Key (2048b)

Storage Root Key (2048b)

Owner Auth Secret (160b)

Volatile memory

RSA Key Slot-0
. . .
RSA Key Slot-9

PCR-0
. . .
PCR-15

Key Handles

Auth Session Handles

The Non-Volatile Keys

- Pubek: the RSA Public Endorsement Key
  - Created, but not recorded at manufacture
  - Cannot be changed
  - Used to encrypt sensitive data to TPM
- SRK: the RSA storage root key
  - Erased by BIOS or owner “clear”
  - Created on chip with TPM_TakeOwnership command
  - Access protected with authentication shared key
- Ownerauth: 160 bit symmetric shared key
  - Used to authenticate all owner sensitive commands
  - Given by owner to TPM in TPM_TakeOwnership

TCPA Software Stack (TSS) Spec:

- Application
- TSP: TCPA Service Provider (concurrency)
- TCS: TCPA Core Services (resource/audit mgt)
- TDDL: TCPA Device Driver Library (TPM commands)
  - (Linux libtcpa)
- TDD: TCPA Device Driver (open/read/write)
  - (Linux tpm.o device driver module)

Libtcpa

- Intended as tutorial/introduction to TCPA on Linux
- Implements interesting subset of commands
  - Ownership, sign, seal/unseal
- Uses openssl for crypto support
- Not a full featured TSS stack:
  - No synchronization for concurrent requests
  - No resource management
  - No audit management
  - No key migration

Getting Started

- Compile and install tpm.o
- Compile libtcpa and example programs
- Enable and clear chip:
  - Power on while holding down “function” (Fn) key (this establishes “physicalpresence”)
  - At bios prompt, release Fn, and press F1 for setup
  - config -> security subsystem
    - “enable” chip
    - “clear” chip

TPM startup

- At power on, chip starts up “activated”
- BIOS is responsible for “startup”, or “deactivate”
- Even deactivated TPM will respond to “safe” cmds:
  - TPM_Reset
  - TPM_GetCapability (particularly version)
  - (these are good test commands, as they should always work.)

The TPM device

- /dev/tpm -> character, major 10, minor 224
- open, write/read as normal character device
- All TPM commands are synchronous send/receive
  - write() command blob to TPM
  - read() result blob from TPM
- Key and authentication handles give context

TPM Command format

- All data is in network (big-endian) byte order!
- Blobs:

  2 byte TAG (to-TPM or from-TPM)
  4 byte total blob length
  4 byte command or result code
  ... command/result specific data

TPM_Transmit()

```
#include <stdio.h>
#include <stdint.h>
#include <netinet.h>
#define TCPA_PARAMSIZE_OFFSET 2
#define TCPA_RETURN_OFFSET    6
#define TCPA_MAX_BUFF_SIZE   4096
int TPM_Transmit(unsigned char *blob)
{
    int tpmfp, len;
    uint32_t size;
    if(tpmfp = open("/dev/tpm", O_RDWR)) < 0) {
        fprintf(stderr,"Can't open TPM Driver\n");
            return(ret);
    }
    size = ntohl(*(uint32_t *)&blob[TCPA_PARAMSIZE_OFFSET]);
    len = write(tpmfp, blob, size);
    /* error handling omitted */
    len = read(tpmfp, blob, TCPA_MAX_BUFF_SIZE);
    /* error handling omitted */
    return(ntohl(*(uint32_t *)&blob[TCPA_RETURN_OFFSET]));
}
```

TPM_Reset

```
uint32_t TPM_Reset()
{
    unsigned char blob[] = {0,193,         /*TPM_TAG_RQU_COMMAND*/
                            0,0,0,10,      /* blob length, bytes */
                            0,0,0,90};     /*TPM_ORD_Reset */
    return(TPM_Transmit(blob));
}

should return a blob {0,196,      /* TPM_TAG_RSP_COMMAND */
                      0,0,0,10,   /* blob length, bytes */
                      0,0,0,0}    /* return code, success */
```

TPM_GetCapability: Version

```
uint32_t TPM_GetCapability_Version()
{
    unsigned char blob[4096] = {0,193,      /* TPM_TAG_RQU_COMMAND */
                                0,0,0,18,   /* blob length, bytes */
                                0,0,0,101,  /* TPM_ORD_GetCapability */
                                0,0,0,6,    /* TCPA_CAP_VERSION */
                                0,0,0,0};   /* no sub capability */
    return(TPM_Transmit(blob));
}

should return a blob {0,196,      /* TPM_TAG_RSP_COMMAND */
                      0,0,0,18,   /* blob length, bytes */
                      0,0,0,0,    /* return code, success */
                      0,0,0,4,    /* length of return data */
                      1,1,0,6}    /* version on my T30 */
```

TPM_TakeOwnership()

- Done (immediately) after clearing TPM.
- Takes two arguments:
  - Unsigned char ownerauth[20] /* hashed owner pass */
  - Unsigned char srkauth[20] /* hashed SRK pass */
- Returns public SRK
- Auth fields are encrypted under Pubek
- Uses Object Independent Authorization Protocol (OIAP)
- User now has authentication on Owner, SRK.

Signing and Wrapping

- TPM_CreateWrapKey
  - On-chip generation and wrapping of RSA key
    - Keys are typed for signature or encryption
  - SRK is the top level encryption key
  - Returns encrypted key blob to user
- TPM_LoadKey
- TPM_EvictKey
- TPM_Sign
- TPM_Seal, TPM_Unseal

Next Steps

- “tutorial” paper and code published in Linux Journal
- Interesting applications
  - Loopback key sealing
  - OpenPKCS11 support
  - OpenSSL support
- Full TSS library/daemon