Using the TPM:
Data Protection and Storage

Ariel Segall
ariels@alum.mit.edu

Day 2

Approved for Public Release: 12-2749.
Distribution unlimited
License

All materials are licensed under a Creative Commons “Share Alike” license.

- http://creativecommons.org/licenses/by-sa/3.0
What We’ll Be Covering

- The TPM’s Tamper Resistance
- Using Storage Keys
- Using Binding Keys
- NVRAM
Revisiting the TPM’s Tamper Resistance

My most frequently asked question about TPMs: “I can use it to protect my data if the machine is stolen, right?”

- Reminder: TPM is *tamper-resistant*, not tamper-proof
  - Not up to government standards
  - Not designed for nation-state adversaries!
- Far better than software protection, but keys *can* be removed
  - Expensive to break: $100,000+ for the publicized attack
  - High failure rate: destroyed a dozen to remove keys from one
  - Still not sufficient for sponsor high-value data with high theft risk
TPM’s Storage Protection Scenarios

- **Evil Maids**
  - Can’t copy hard drive and pull keys out at leisure
  - Combined with PCRs, can’t reboot into evil OS and steal secrets

- **Software data theft**
  - Can’t freely vacuum data and send off machine
  - Combine TPM keys and user passwords for best security
  - Note: At-rest protection, not during use!

- **Casual physical theft**
  - Not good enough for nation-states, plenty good against even competent thieves
What This Means For Use

- The TPM is strongest when protecting data at rest...
- ...therefore, protecting data in bulk less effective than small, focused chunks
- Storage most effective when used in multi-part security:
  - TPM as thing you have; authorization value as thing you know
  - State verification one of the most powerful tools for data protection...
- ...and can also cause self-inflicted DoS. Use with care.
What We’ll Be Covering

- The TPM’s Tamper Resistance
- Using Storage Keys
- Using Binding Keys
- NVRAM
Using Storage Keys

- **TPM_Sea1**: Encrypt data for later decryption with **TPM_Unseal**
  - Local platform only!
- **Storage keys are also used to protect TPM keys**
  - Every **TPM_CreateWrapKey** operation must provide a storage key parent
  - When migrating keys, encrypted to storage key: new parent
- **Note**: The SRK is a storage key!
Key Storage Hierarchy Review

- Storage Root Key
- Identity Key
- Wrap Key (inc. Storage)
- Storage Key
- Wrap Key (inc Storage)
Sealing

- When sealing data, several options:
  - Which storage key to use
  - Whether to require authorization data (password)
  - Whether to provide PCR constraints for decryption
  - Whether to provide locality constraints for decryption

- Sealed data always contains unique TPM internal value
  - Locked to this TPM even if the key migrates

- Sealing also records the current PCR state
  - Ensure that decrypted data can be trusted

- Returns “sealed blob”
Unsealing

- Use same storage key to decrypt
- Verifies authorization, current PCRs, current locality against blob
  - Note: two authorizations may be required! One for key, one for blob
- Verifies creation data to ensure real creation value matches public value
- Returns decrypted data
- Note: blob can be unsealed multiple times

Note: Once unsealed, data is in the clear!

- Use PCR values and authorization to minimize risk of loss during use
Storage Key Summary

- Utility key for protecting secret data, including keys
- Directly protect user data with Seal
  - Optionally additionally protect with password, PCR constraints
  - Local system only
- Decrypt with Unseal
What We’ll Be Covering

- The TPM’s Tamper Resistance
- Using Storage Keys
- Using Binding Keys
- NVRAM
Binding Keys

- Another utility key for data protection
- Different paradigm than storage
- Anyone on any platform can Bind data
- Only TPM can decrypt, using TPM_Unbind
- No fancy options—just decryption
  - PCR constraints and authorization still possible, but on key not blob
## Storage vs. Binding Key Summary

<table>
<thead>
<tr>
<th>Storage</th>
<th>Binding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local use only</td>
<td>Local or remote use</td>
</tr>
<tr>
<td><em>Seals</em> user data, optional extra constraints</td>
<td><em>Binds</em> user data, constraints only on key</td>
</tr>
<tr>
<td>Can be used as key parent</td>
<td>Encrypts user data only</td>
</tr>
<tr>
<td>Only authenticates local data</td>
<td>Usable for machine authentication</td>
</tr>
</tbody>
</table>
What We’ll Be Covering

- The TPM’s Tamper Resistance
- Using Storage Keys
- Using Binding Keys
- NVRAM
NVRAM Summary

- Storage area inside TPM
- Very limited in size: only 1280 bytes required, though can be bigger
  - No hard data on actual implementations
- Controlled by owner; permissions can be delegated
- Some sections reserved for specific purposes (e.g., root credentials)
- Customizable constraints per region for read or write access
  - PCR contents
  - Locality
  - Authorization data
- Limited number of writes; can be burned out
  - Order of 10,000; only a minor DoS issue for most applications
Why NVRAM is Useful

- Stores data that can serve as reference
  - Much harder to modify than data on disk!
  - Hashes for integrity checking
  - Owner or trusted authority public key
  - Very powerful for system sanity checking!
- Stores high-value data that should not be accidentally deleted
  - Keys
  - Certificates
NVRAM Use Case Examples

- Storing user-chosen pictures for 'trusted boot'
  - If correct picture retrieved, PCR values in known state
- Preventing attacks which replace trusted authority
  - IT-approved CA key or DNS server in read-only NVRAM
- Integrity reference for software
  - Put hash of file in write-limited NVRAM
  - Current AV definitions? Most recent save file? Policy approved by owner? Approved OS list for boot loader?
  - If file is public key, can use to verify owner signature.
- Resources for early boot, DRTM
  - Limited space, but easy to constrain access
Using NVRAM – Quick Summary

- Establish a region of NVRAM with desired size and permissions
  - TPM_NV_DefineSpace
  - Owner only, unless permissions delegated
- Separate commands for owner, non-owner access
  - TPM_NV_WriteValue, TPM_NV_ReadValue for non-owner
  - TPM_NV_WriteValueAuth, TPM_NV_ReadValueAuth for owner
- Note: Access control enforcement on NVRAM is not automatic!
  - Manufacturers need ability to write certs into NVRAM without being owner
  - Supposed to set flag enabling access control afterwards
  - Don’t always!
  - How to check and set flag in next section.
Problems with NVRAM

- Space is very limited
  - For one application, not a problem; if commonly used, potentially serious
  - Many hashes; very few certificates or keys
- Limited number of writes in lifetime
  - How many? Good question!
  - Not suitable for applications with frequent updates
TPM Data Protection Review

- TPM designed for protection of data at rest
- Storage keys protect data on this platform
  - Many protection options
- Binding keys protect data from anywhere
- NVRAM protects limited, high-value data
  - Good for integrity verification and system checks
Questions?