Hardening the UEFI Attack Surface
How to Harden an Attack Surface

• Threat Modeling
• Secure Coding
• Security Code Audits
• Fuzz Testing
• Software Security Defenses
Media Player Threat Model
UEFI Threat Model
Secure Coding (one aspect)

Validation of untrusted input!

Poor validation of untrusted input may result in:
  – Buffer overflows
  – Integer and pointer corruption
  – Memory overwrites
  – ...

Leading to:
  – Compromised runtime integrity of authenticated components
  – ...


Fuzz Testing

Applying malformed data against the attack surface
Software Security Defenses

- Writing Secure Code
- Stack Buffer Overrun Detection (GS)
- Data Execution Prevention (DEP/NX)
- Address Space Layout Randomization (ASLR)
- Heap Corruption Detection
- Migration to Safer Functions
Tianocore Revisions of Interest

Revisions of interest EDK2 trunk between 2012/01/01 and 2012/04/15. It is important to review these changes and, if the revision is applicable to implementations, integrate them.

12927    13144
13094    13156
13095    13157
13104    13158
13109    13162
13110    13185
13120
How to Harden and Attack Surface

• **Secure Coding:** helps to avoid problems

• **Threat Modeling:** helps to define trust boundaries and potentially malicious data input points

• **Security Code Audits:** helps identify vulnerabilities through manual code inspection

• **Fuzz Testing:** helps find input parsing and other vulnerabilities

• **Software Security Defenses:** helps provide blanket protection against some threats