presented by

UEFI and the Security Development Lifecycle

Spring 2018 UEFI Seminar and Plugfest
March 26-30, 2018
Presented by Tim Lewis (Insyde Software)

Agenda

- The Threat Is Real
- The Security Development Lifecycle (SDL)
  - How The Lifecycle Works
  - Is Your Firmware Worth Attacking?
  - How Your Firmware Gets Attacked
- What You Can Do About It
- Resources

The Threat Is Real

- Firmware holds a unique, valuable security position.
  - Computer systems are only as secure as their firmware.
- Firmware is under increasing numbers of attacks.
  - Not just from researchers and hackers, but from professionals.
- Firmware threats often appear years after 1st shipment.

We Will Never Get the Code 100% Correct

"We must be 100% correct, 100% of the time, on a schedule, with limited resources, only knowing what we know today.

Oh, and the product has to be reliable, supportable, compatible, manageable, affordable, accessible, usable, global, doable and deployable.

They can spend as long as they like to find one bug, with the benefit of future research."

- *Basics of Secure Design, Development, and Test*, Microsoft, 2010.

The Security Development Lifecycle

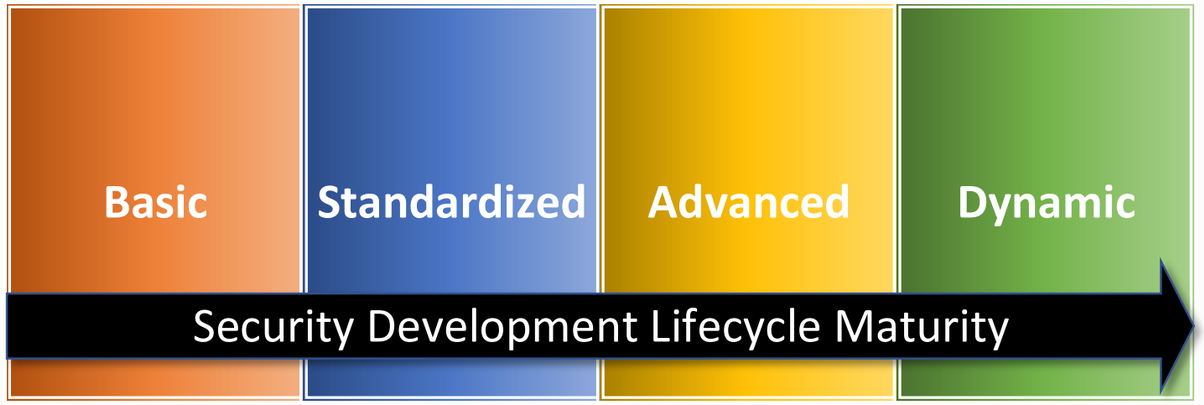
- The Security Development Lifecycle (SDL) is a *process for developing demonstrably more secure software*, pioneered by Microsoft.

- The SDL process improves the capability to support, design, develop, test and release secure software.
  - Improved support through training and in-house security expertise.
  - Improved design using risk assessment and threat modeling.
  - Improved development with best practices that minimize chances of attacks.
  - Improved testing using tools to detect and test for vulnerabilities.
  - Improved response by root causing, deploying fixes, informing customers and updating tests.

*Microsoft calls these: Training/Policy/Organization, Requirements and Design, Implementation, Verification and Release/Response.

Security Development Lifecycle Maturity

- Process maturity level graded into four levels:
  - **Basic** – Process is undefined or inconsistent.
  - **Standardized** – Activities are expert led.
  - **Advanced** – Activities are led by a central security team.
  - **Dynamic** – Activities are co-led by product teams and central security team.



- Each of the capability areas (support, design, develop, test, release) has key requirements which must be met in order to go to the next maturity level.
- Where is your company?

STRIDE – How Your Firmware Gets Hacked?

Threat	Property	Definition	Example
SPOOFING	Authentication	Impersonating someone or something else.	Pretending to be supervisor, OEM or Microsoft
TAMPERING	Integrity	Modifying data or code.	Modifying an authenticated variable, SPI flash part or S3 resume script.
REPUDIATION	Non-repudiation	Claiming to have not performed an action.	“I didn’t log into that system” or “I didn’t send that e-mail.”
INFORMATION DISCLOSURE	Confidentiality	Exposing information to someone not authorized to see it.	Allowing someone to read a password or private keys or personal information.
DENIAL OF SERVICE	Availability	Deny or degrade service to users.	Can’t boot the system or can’t update the system.
ELEVATION OF PRIVILEGE	Authorization	Gain capabilities without proper authorization.	Allow an application to gain MM privileges.

Based on Microsoft’s “Basics of Secure Design Develop and Test"

Attack Threats (Using STRIDE)

Based on Microsoft's "Basics of Secure Design Develop and Test"

Threat	Property	Definition	Example
SPOOFING	Authentication	Impersonating someone or something else.	Pretending to be supervisor, OEM or Microsoft
TAMPERING	Integrity	Modifying data or code.	Modifying an authenticated variable, SPI flash part or S3 resume script.
REPUDIATION	Non-repudiation	Claiming to have not performed an action.	"I didn't log into that system" or "I didn't send that e-mail."
INFORMATION DISCLOSURE	Confidentiality	Exposing information to someone not authorized to see it.	Allowing someone to read a password or private keys or personal information.
DENIAL OF SERVICE	Availability	Deny or degrade service to users.	Can't boot the system or can't update the system.
ELEVATION OF PRIVILEGE	Authorization	Gain capabilities without proper authorization.	Allow an application to gain MM privileges.

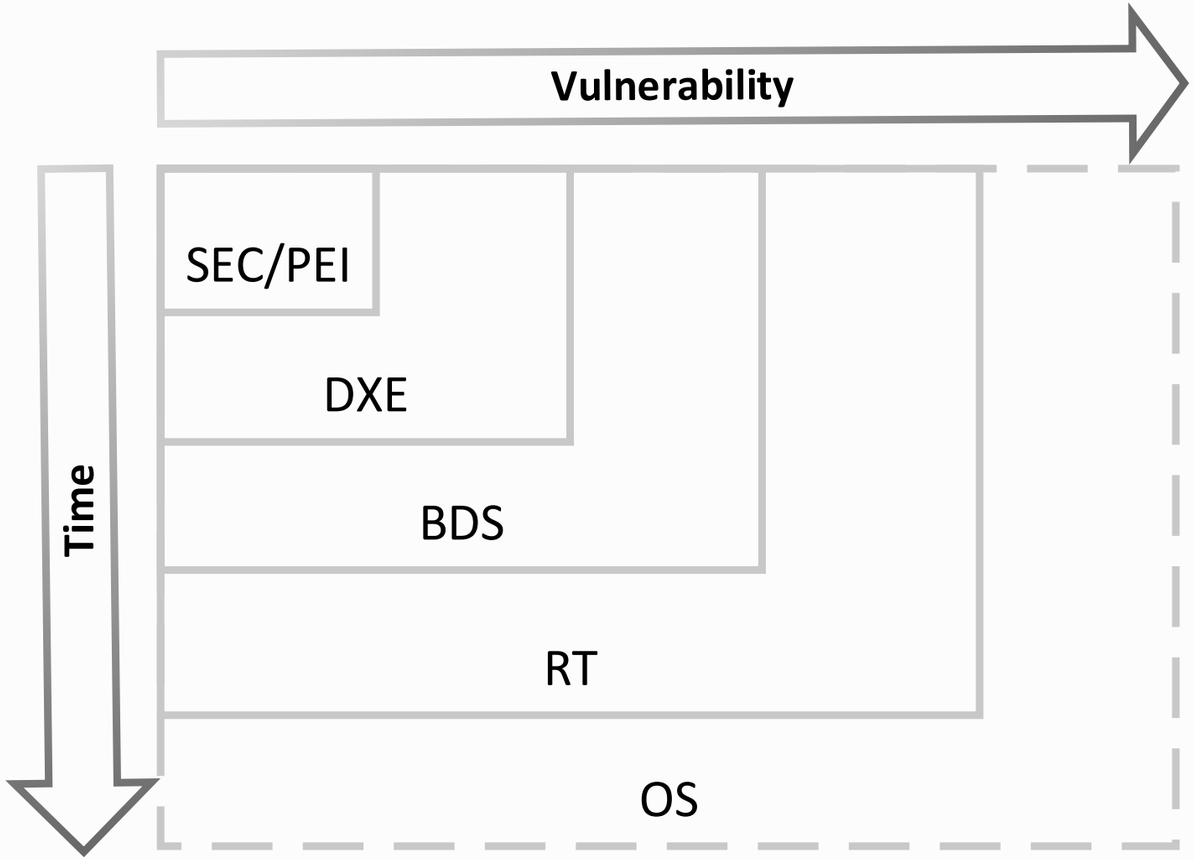
Attack Threats (Using STRIDE)

Based on Microsoft's "Basics of Secure Design Develop and Test"

Threat	Property	Definition	Example
SPOOFING	Authentication	Impersonating someone or something else.	Pretending to be supervisor, OEM or Microsoft
TAMPERING	Integrity	Modifying data or code.	Modifying an authenticated variable, SPI flash part or S3 resume script.
REPUDIATION	Non-repudiation	Claiming to have not performed an action.	"I didn't log into that system" or "I didn't send that e-mail."
INFORMATION DISCLOSURE	Confidentiality	Exposing information to someone not authorized to see it.	Allowing someone to read a password or private keys or personal information.
DENIAL OF SERVICE	Availability	Deny or degrade service to users.	Can't boot the system or can't update the system.
ELEVATION OF PRIVILEGE	Authorization	Gain capabilities without proper authorization.	Allow an application to gain MM privileges.

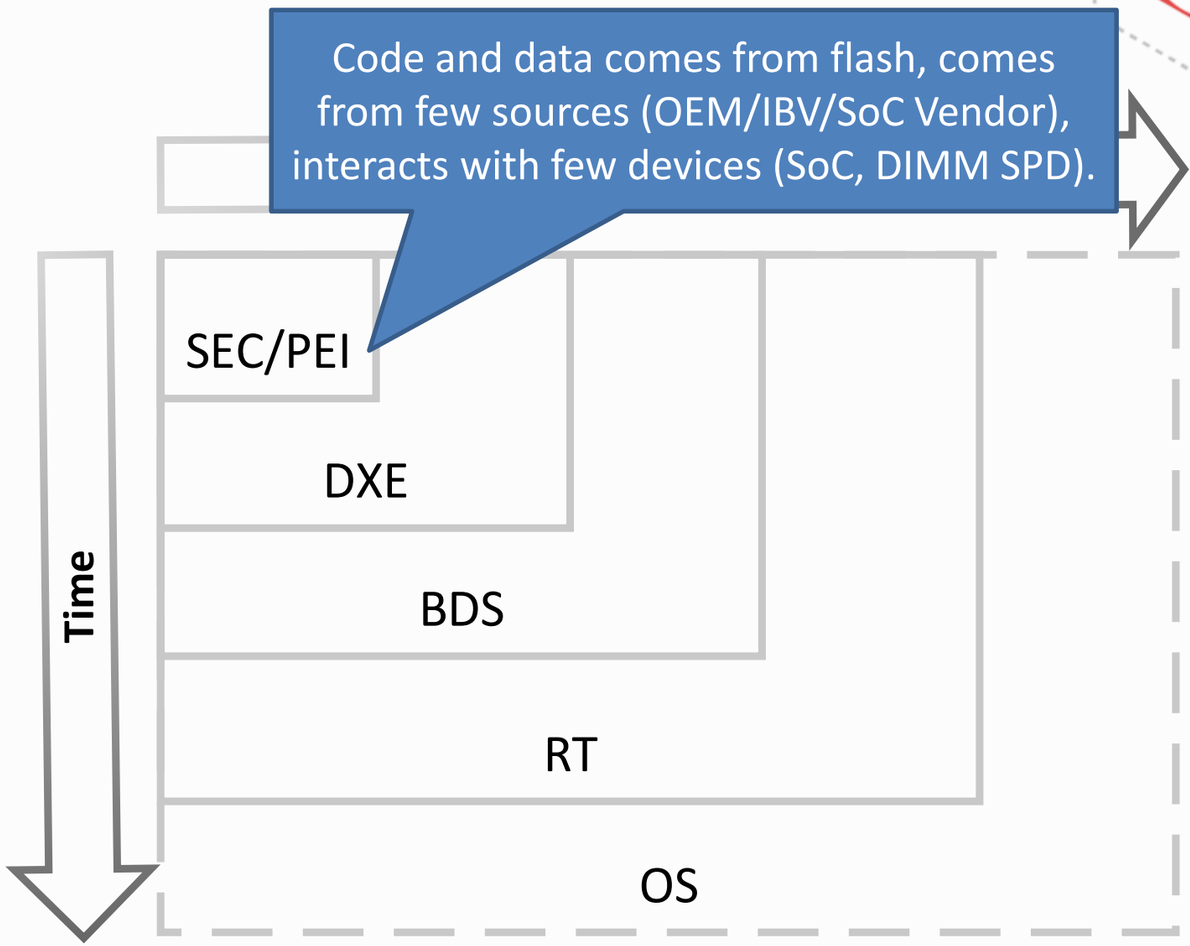
Trust Boundaries in UEFI

- Vulnerability increases as code
  - Executes in more environments
  - Comes from more sources
  - Interacts with more devices



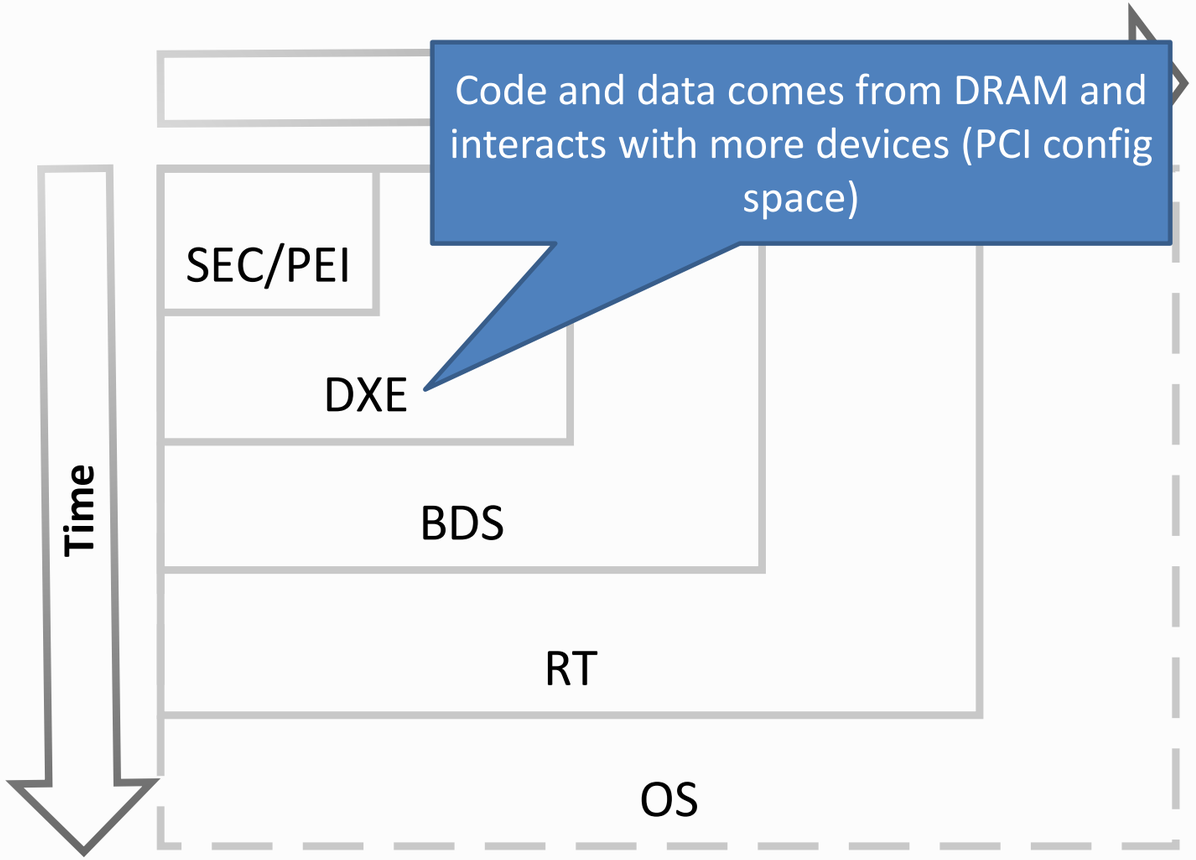
Attack Vectors In UEFI

- Vulnerability increases as code
  - Executes in more environments
  - Comes from more sources
  - Interacts with more devices



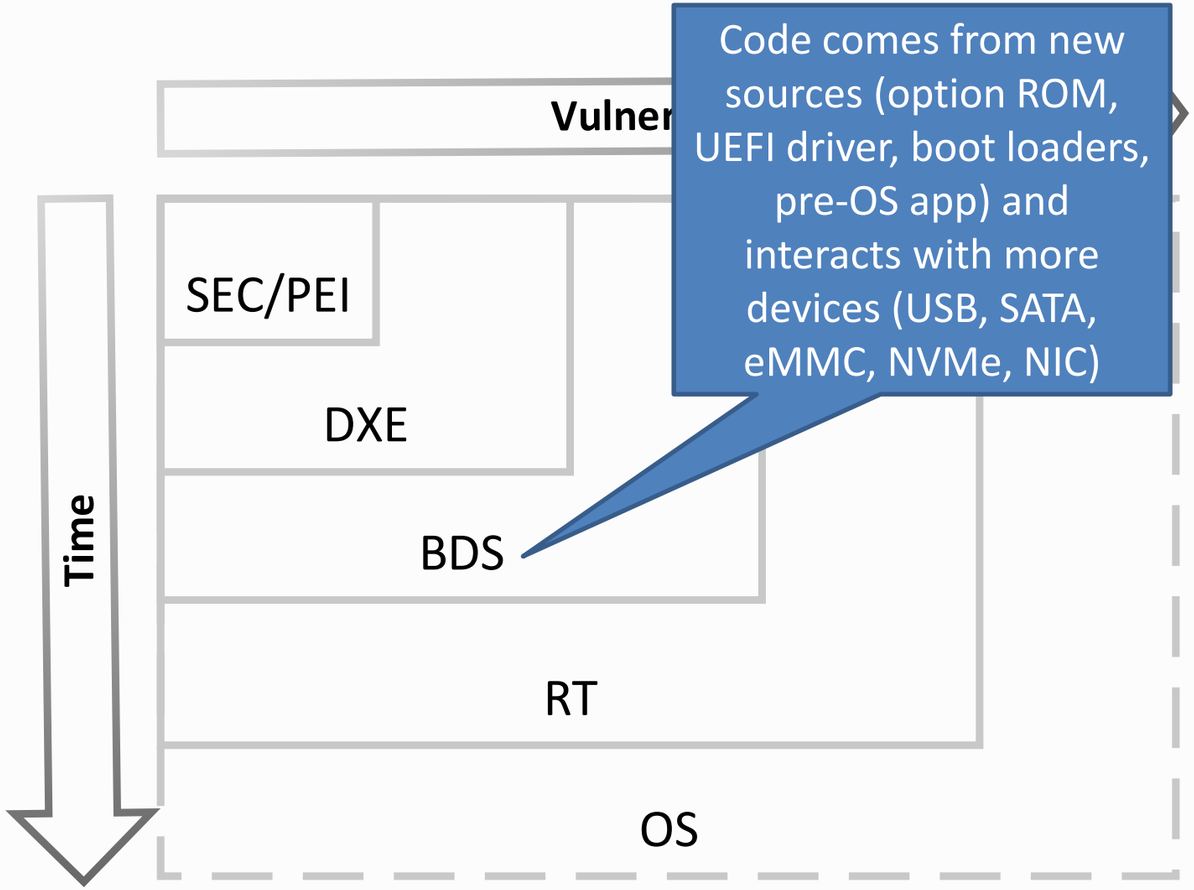
Attack Vectors In UEFI

- Vulnerability increases as code
  - Executes in more environments
  - Comes from more sources
  - Interacts with more devices



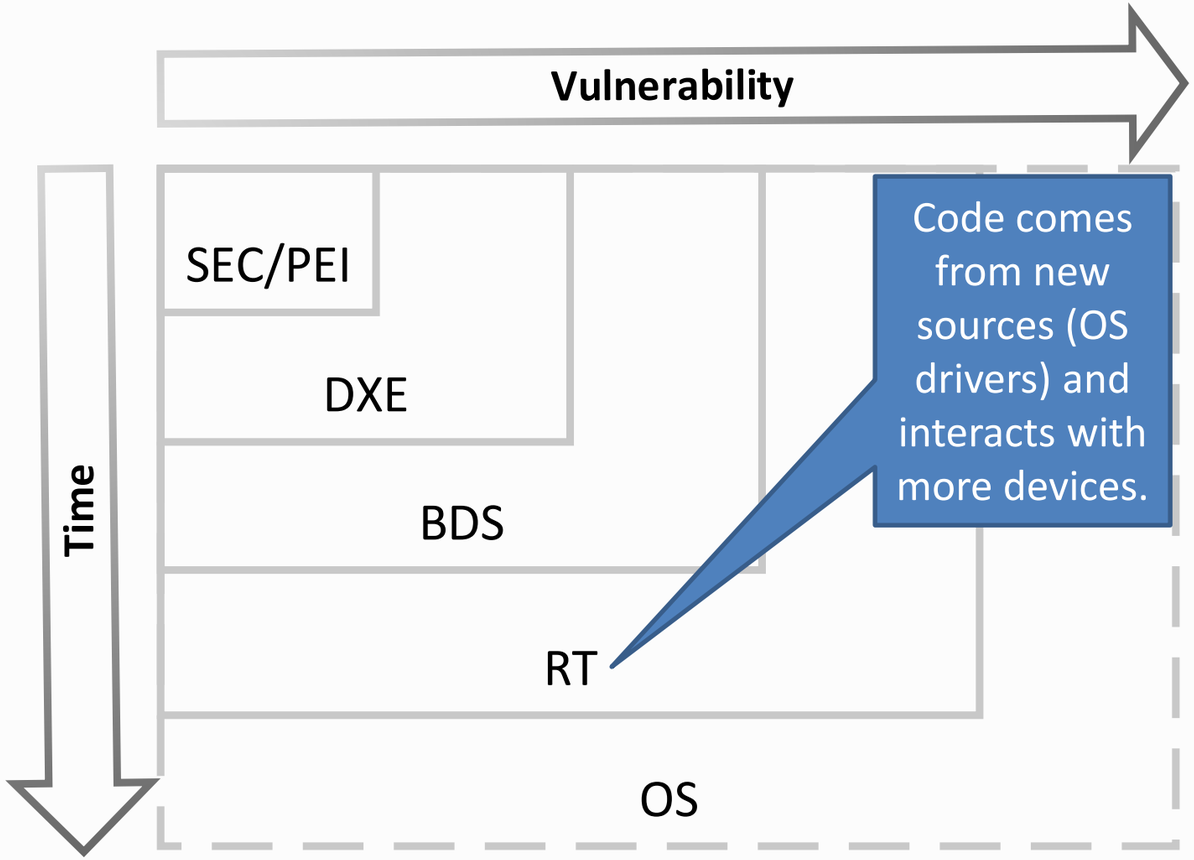
Attack Vectors In UEFI

- Vulnerability increases as code
  - Executes in more environments
  - Comes from more sources
  - Interacts with more devices



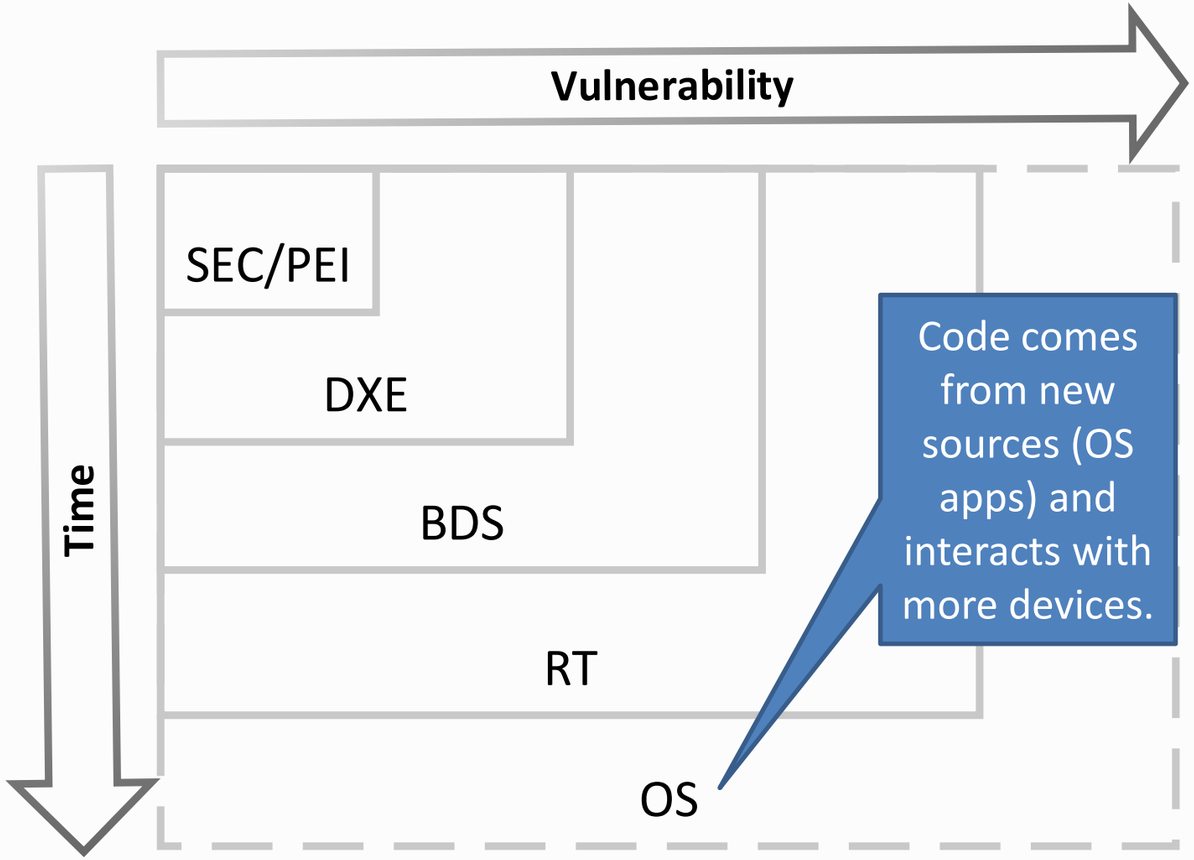
Attack Vectors In UEFI

- Vulnerability increases as code
  - Executes in more environments
  - Comes from more sources
  - Interacts with more devices



Attack Vectors In UEFI

- Vulnerability increases as code
  - Executes in more environments
  - Comes from more sources
  - Interacts with more devices



Where Would I Look for Weaknesses?

- Assuming flash is immutable, DRAM can't be tampered with, and the attacker has source code, the best targets are:
  - (Software) Management Mode Interrupts
  - UEFI Runtime Services
  - S3 Resume Path
  - UEFI variable data
  - Hardware device data, such as DIMM SPD, USB descriptors, ATA identify results, etc.
  - UEFI capsule data
  - Network packets

What To Do About It

- Train your engineers on basic security concepts.
- Check SDL process of your code suppliers.
- Ensure you have all tools to reproduce your build.
- Start with the riskiest part of your code.
- Use tests that focus on security, not just functionality.
- Improve your tools (compiler, static analysis).
  - Asserts/page faults at runtime are a denial-of-service.

Resources

- Where To Get Started With SDL?
  - https://www.microsoft.com/en-us/SDL/process/training.aspx
- Where Are You In SDL?
  - http://studylib.net/doc/7123174/sdl-optimization-model-self-assessment-guide#
- Threat Model Analysis
  - https://docs.microsoft.com/en-us/biztalk/core/threat-model-analysis?redirectedfrom=MSDN
- Basic Security Training
  - https://download.microsoft.com/download/9/3/5/935520EC-D9E2-413E-BEA7-0B865A79B18C/Basics%20of%20Secure%20Design%20Development%20Test.ppsx

Questions?

Thanks for attending the Spring 2018 UEFI Plugfest

For more information on the UEFI Forum and UEFI Specifications, visit http://www.uefi.org

presented by