LAYER one

Security Engineering in Windows Vista™

Michael Weiss
Lead Security Program Manager
Microsoft® Windows® Security Assurance
Michael.Weiss@Microsoft.com

LAYER one

Security Engineering in Windows Vista™

Michael Weiss
Lead Security Program Manager
Microsoft® Windows® Security Assurance
Michael.Weiss@Microsoft.com

Agenda

- Introduction
  - Who Am I?
  - Goals of This Talk
- Windows Vista Security Approach
- Retrospective Case Study
- Q & A

LAYER one

A Quick Heads-Up

- Did you see this presentation at BlackHat, or HitB?
- No significant new information here
  - New organization of information, though

Who Am I?

- Lead Security Program Manager in Windows Security Assurance
  - Windows Security Assurance does the following for Windows feature teams:
    - Evangelize security
    - Consult on design and implementation
    - Train on attacker and defense techniques
    - Develop and enforce security policies
- Joined Microsoft 14 years ago
  - Deployment and management support
  - Financial services
  - Online security
  - Windows security

Goals of This Talk

- Explain what we did in Windows Vista
  - Overview of security engineering activities
  - Some detail on our major security initiatives
  - Overview of our work on mitigations
- Listen to
  - Any engineering-focused feedback you have
  - How you think we're doing
- Ungoal:
  - Security features (e.g., BitLocker)

Agenda

- Introduction
  - Who Am I?
  - Goals of This Talk
- Windows Vista Security Approach
- Retrospective Case Study
- Q & A

LAYER one

Windows Vista Security Approach

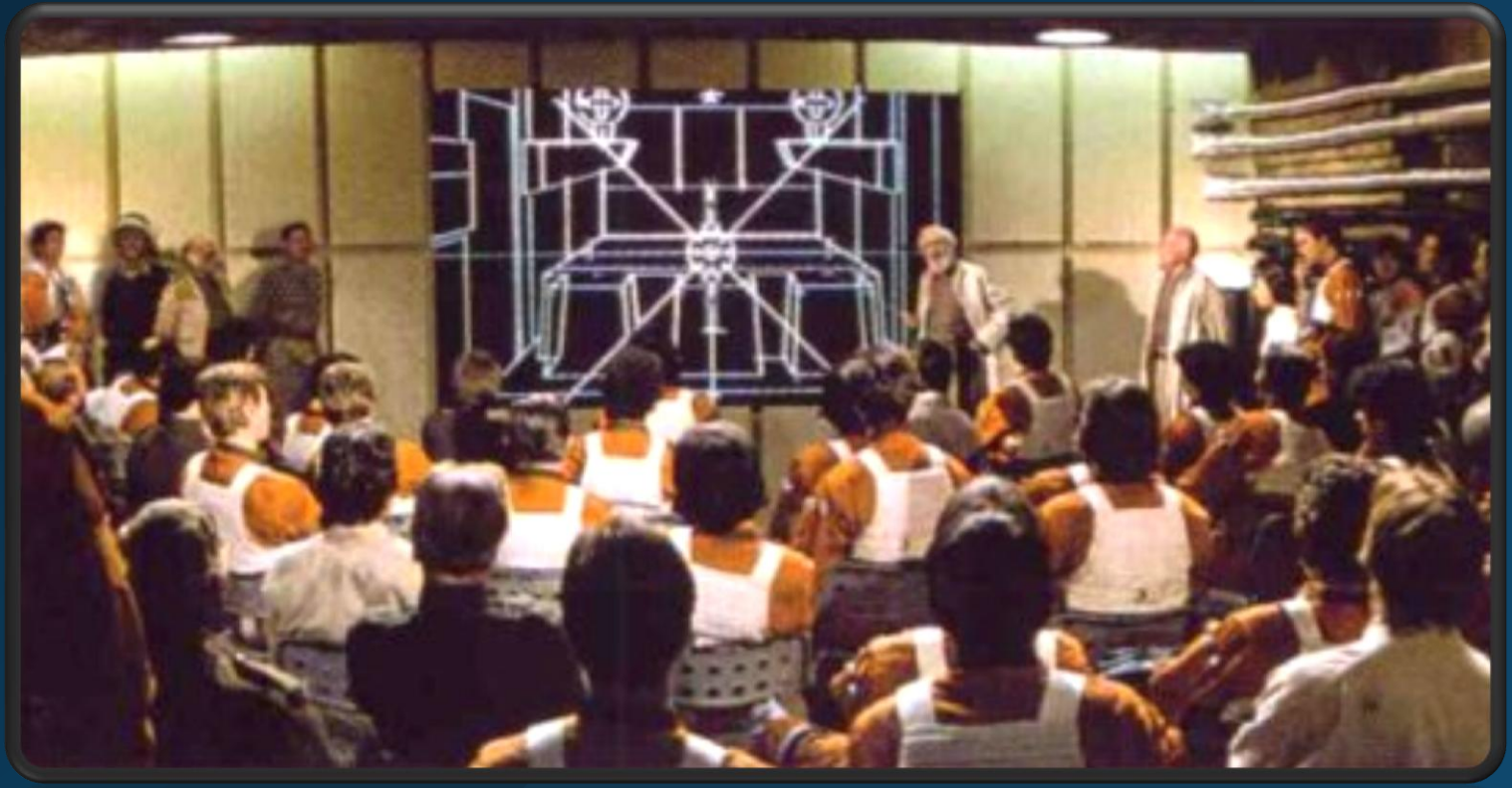
Everything in Windows XP® SP2 SDL *plus*

- Apply least privilege throughout architecture
- Automate proven techniques
- Methodically apply security expertise on whole OS
- Additional Defense-in-Depth mitigations

The Goal:

Stop playing Whac-a-Mole
Find and fix vulnerabilities before shipping

LAYER one



LAYER one

Least Privilege
The target area is only two meters wide

- Problem:
  - If a program runs as SYSTEM or Administrator, any compromise is catastrophic
- Defenses:
  - Run Applications with Least Privilege
    - Enhance the standard user account
    - Administrators use full privilege only for administrative tasks or applications
    - Run some applications (e.g., IE) with heightened restrictions
  - Harden Services
    - Minimize privilege user pervasively in services
    - Define restrictions to ensure behavior conforms to expected activity

UAC

Why Focus on Hardening Services?

- Services are attractive targets for malware
  - Sasser, Blaster, Slammer, Zotob, CodeRed, …
- No need for user interaction
- Often run in elevated identities
- Many worms do nasty things to services
  - Alter the OS
  - Open network connections to propagate

Get Services Out of SYSTEM

- LOCAL SERVICE or NETWORK SERVICE instead
  - LOCAL SERVICE and NETWORK SERVICE are not members of the Administrators group
  - LOCAL SERVICE and NETWORK SERVICE are denied the most powerful privileges
    - SeDebug
    - SeTcb
    - etc.

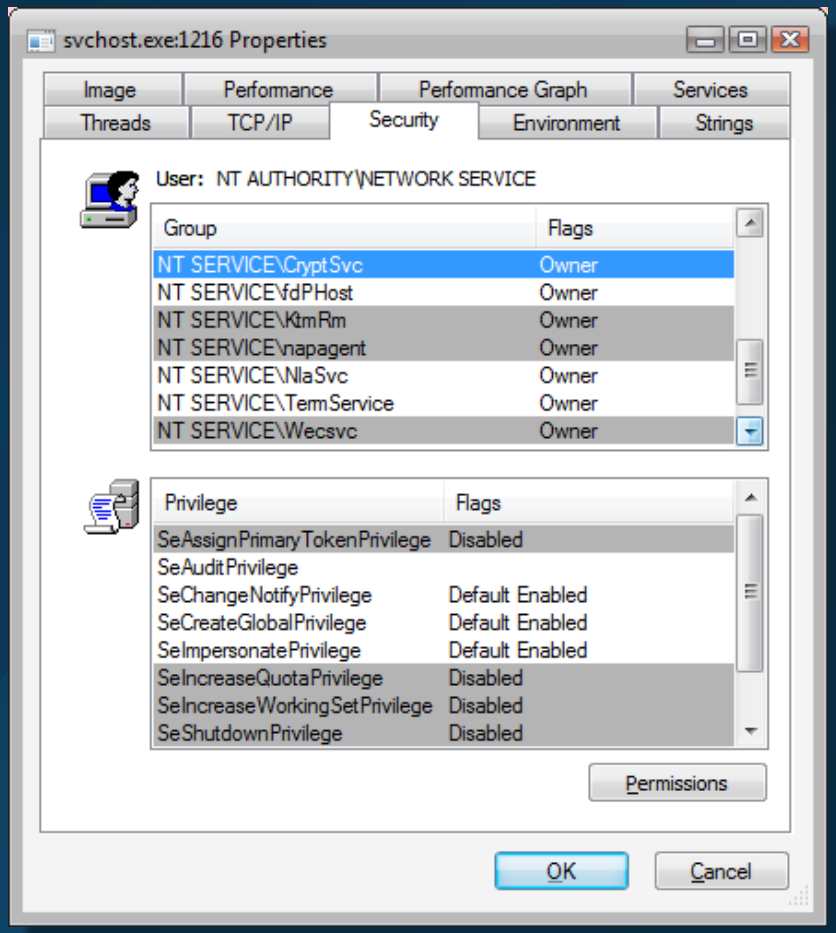
What Did We Do to Windows XP SP2 SYSTEM Services?

- Services moved to LOCAL SERVICE
  - Windows Audio
  - DHCP Client
  - Windows Event Log
  - COM+ Event System
  - Workstation Service
  - Windows Time
  - Security Center
  - Windows Image Acquisition
- Services moved to NETWORK SERVICE
  - Cryptographic Services
  - Policy Agent
  - Telephony
  - Terminal Services
- *And* 48% of new services in Windows Vista run under a low privilege account

LAYER one

Compartmentalize with Service SIDs

- Per Service SIDs
  - Derived from service name in SCM
  - S-1-5-80-xxxx
- ACL objects such that only your service can manipulate them
- Integrated into:
  - LookupAccountSid
  - LookupAccountName



Example:
S-1-5-80-242729624-280608522-2219052887-3187409060-2225943459
Resolves to NT SERVICE\CryptSvc

LAYER one

Eliminate Unnecessary Privileges

- Enumerate required privileges
  - All others are removed
- Processes that host multiple services get union of required privileges

```
// Set up the required privileges
SERVICE_REQUIRED_PRIVILEGES_INFOW servicePrivileges;
servicePrivileges.pmszRequiredPrivileges =
       (L"SeChangeNotifyPrivilege\0"
        L"SeCreateGlobalPrivilege\0"
        L"SeImpersonatePrivilege\0");

fRet = ChangeServiceConfig2(
        schService,
        SERVICE_CONFIG_REQUIRED_PRIVILEGES_INFO,
        &servicePrivileges);
```

Restrict Network Behavior

- Define service's network requirements
  - OS enforces network access policy
  - e.g.: foo.exe can only open port TCP/123 inbound
    - `|Action=Allow|Dir=In|LPORT=123|Protocol=17|App=%SystemRoot%\foo.exe`
  - If foo.exe has a vulnerability, rogue code cannot
    - Make outbound connections
    - Be accessed through any port other than 123 over TCP
- Enforced by firewall

LAYER one

New svchosts

Group Services to Take Advantage of Restrictions

Svchost Name	*Service Account*	*Network Access*	*Write-Restricted Token*
LocalServiceNoNetwork	Local	No	Yes
LocalServiceRestricted	Local	Yes*	Yes
LocalServiceNetworkRestricted	Local	Yes*	No
NetworkServiceRestricted	Network	Yes*	Yes
NetworkServiceNetworkRestricted	Network	Yes*	No
LocalSystemNetworkRestricted	Local System	Yes*	No

**To a fixed set of network ports*

LAYER one

Example: Comparison of DHCP Client Service

	Windows xp SP2	Windows Vista
Account	SYSTEM	LOCAL SERVICE
Privileges	24	4
Network Identity?	Yes *(Machine Account)*	No
Uses Fixed Set of Ports?	No	Yes
Data accessible only by service? (Service SID)	No	Yes

Windows Vista Security Approach

Everything in Windows XP SP2 SDL *plus*

- Apply least privilege throughout architecture
- Automate proven techniques
- Methodically apply security expertise on whole OS
- Additional Defense-in-Depth mitigations

The Goal:

Stop playing Whac-a-Mole
Find and fix vulnerabilities before shipping

LAYER one

Code Analysis:
Never send a human to do a machine's job

- Given the code: **`buff[x] = 5;`**
- Q: How big is **`buff`**, and what is the value of **`x`**?
- C/C++ doesn't associate buffers to their sizes

Meet SAL

- Standard Annotation Language (SAL) provides interface contracts to tools
  - The concept is not new: think IDL in RPC
  - Primary focus is finding buffer overruns

How Does SAL Work?

```
void FillString(
        TCHAR* buf,
        size_t cchBuf,
        char ch)
{
  for (size_t i = 0; i < cchBuf; i++)    {
    buf[i] = ch;
  }
}
```

Pointer to a TCHAR buffer

Function writes this many characters to **buf**

- If **cchBuf** > size of **buf**, the loop will walk off the end of **buf**

How Does SAL Work?

```
void FillString(
        TCHAR* __element_count(cchBuf) buf,
        size_t cchBuf,
        char ch)
{
  for (size_t i = 0; i < cchBuf; i++)   {
    buf[i] = ch;
  }
}
```

Out parameter (will be written to) and is non-null

Buffer size is an Element count

Buffer is cchBuf elements in size

How Does SAL Work?

```
void FillString(
        __out_ecount(cchBuf) TCHAR* buf,
        size_t cchBuf,
        char ch)
{
  for (size_t i = 0; i < cchBuf; i++)   {
    buf[i] = ch;
  }
}

void main() {
        TCHAR *buff = malloc(200 * sizeof(TCHAR));
        FillString(buff,210,'x');
}
```

1. Warning C6386: Buffer overrun: accessing 'argument 1', the writable size is '200*2' bytes, but '420' bytes might be written
2. Warning C6387: 'argument 1' might be '0': this does not adhere to 'FillString' `__out`

Remember this Buffer Overrun?

- Buffer overrun found in IE7 Beta 2 on Jan 31, 2006.
  - http://www.security-protocols.com/advisory/sp-x23-advisory.txt

```
<BGSOUND SRC=file://------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
-------------- >
```

- Workaround: Mozilla Firefox ☹

```
WCHAR pwzTempPath[MAX_PATH];

PathCreateFromUrlW(
        pwzPath,
                (LPWSTR) pwzTempPath,
        &cchPath,
        0);
```

(Obviously)

PREfast & SAL in Action

```
LWSTDAPI
PathCreateFromUrlW(
    LPCWSTR pszIn,
    __out_ecount(*pcchOut) LPWSTR pszOut,
    __inout LPDWORD pcchOut,
    DWORD dwFlags )
```

```
WCHAR pwzTempPath[MAX_PATH];

PathCreateFromUrlW(
    pwzPath,
    (LPWSTR) pwzTempPath,
    &cchPath ,
    0)
```

11/24/2005 5:50 AM Bug # ****932 Opened by PREfast

Description:

1. Potential overflow using expression '**`& pwzTempPath`**'
2. Buffer access is apparently unbounded by the buffer size.
3. In particular: cchPath`3485a is not constrained by any constant
4. Buffer is of length 260 elements (2 bytes/element) [size of variable or field]
5. Annotation on function PathCreateFromUrlW@16 requires that {parameter 2} is of length >= *{parameter 3} elements (2 bytes/element)
   where {parameter 2} is **`& pwzTempPath`**; {parameter 3} is **`& cchPath`**

LAYER one

Did SDL Succeed?

- Root cause analysis leads to tools improvement
  - After PnP RPC bug (Zotob worm), PREfast was improved (Warning #2015)
  - Auto-file bugs on Warning #2015
  - IE bug identified immediately, filed by PREfast Nov., 2005
    - Caught by SAL annotation on PathCreateFromUrlW API
    - Found through internal fuzzing 8 days after public vulnerability report
    - Reported through Windows Error Reporting 1 day later
    - Bug was found and would have been fixed by RTM
- Focus on root cause analysis, continuous tools and process improvements in SDL pays off

File Parsers Under Attack

- MS05-002: 3 ANI
- MS05-009: PNG
- MS05-012: OLE/COM
- MS05-014: CDF
- MS05-018: Fonts
- MS05-020: MSRatings (.RAT)
- MS05-022: GIF
- MS05-025: PNG
- MS05-026: ITS
- MS05-036: 9 ICM (JPG, PNG, BMP)
- MS05-050: AVI
- MS05-053: EMF
- MS06-002: EOT
- MS06-003: TNEF
- MS06-004: WMF
- MS06-005: BMP

Multi-Prong Approach on Parsers

- Automate what you can:
  - *All parsers:* Internally developed general purpose fuzzer
    - Over 100M manipulations by Beta 2
  - *Highest risk parsers:* get Data-Definition-Language extensions
  - *Hard targets:* Smart fuzzers (Examples: EMF, HTML)
  - Code coverage helps in "template reduction" to improve efficiency
    - Library of >19,000 JPGs optimized to 47 with same block coverage
- Apply security expertise where you need it:
  - Security code review + detailed program analysis on "problem parsers"
  - Extended SAL annotations for struct members
  - Emit runtime stack protections more aggressively in "attack path"

Windows Vista Security Approach

Everything in Windows XP SP2 SDL *plus*

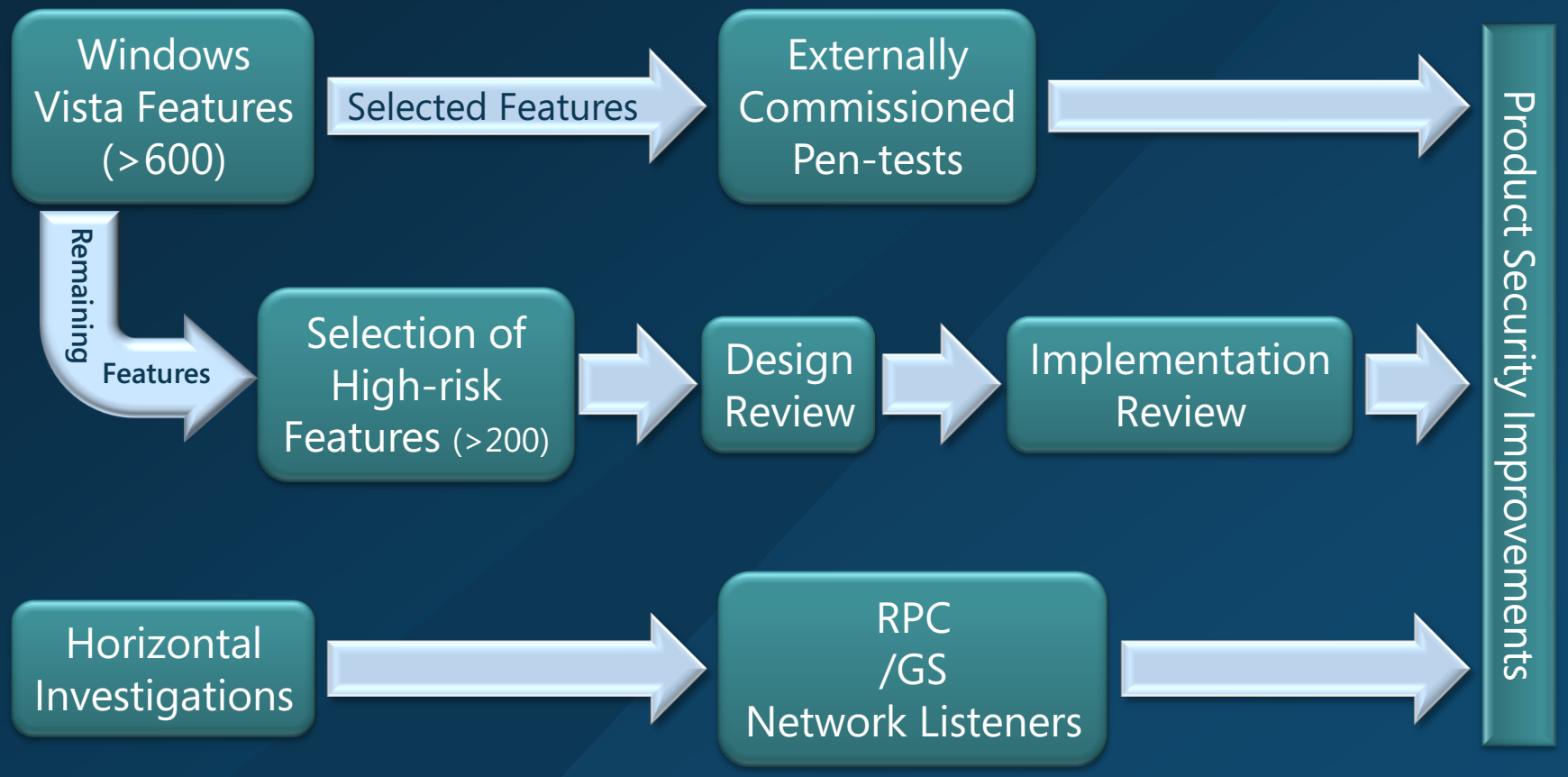
- Apply least privilege throughout architecture
- Automate proven techniques
- Methodically apply security expertise on whole OS
- Additional Defense-in-Depth mitigations

The Goal:

Stop playing Whac-a-Mole
Find and fix vulnerabilities before shipping

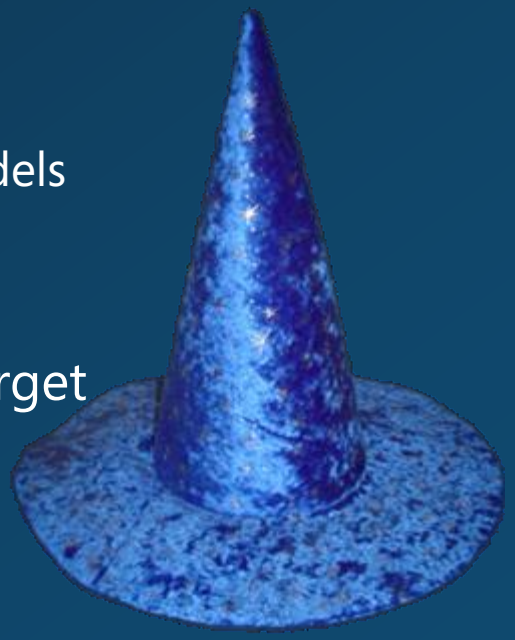
LAYER one

Security Expertise:
It's people



Penetration Testing

- Largest Penetration Test in Microsoft History
  - Internal team of hackers
  - Multiple simultaneous penetration tests
  - "Blue Hat Hackers" ☺
    - 20+ security consultants (aka hackers) in a room
    - Access to Full Source + Symbols, specs, threat models
    - Access to members of product teams, SWI experts
    - All necessary expertise is within 1 building radius
  - Spend anywhere from 1 week to 2 months per target
  - Nothing is out-of-scope



Sampling of Findings

- Process tended to yield “rabbit holes”
- Contradiction in Security Assumptions
  - TM #1: “We have no risk because we don’t parse anything, we just pass things down.”
  - TM #2: “We have no risk because our input is trusted; we just receive validated content.”
- Failures of Imagination
- Unwise filenames
  - wls0wndh.dll
  - Winlogon Session0 Viewer Window Hook DLL

LAYER one

It Came from the Codebase…

```
n -= (e = (e = 0x4000 -
((d &= 0x3fff) > w ? d : w))
> n ? n : e);
```

"It's actually quite beautiful!
Almost like a Haiku or something."

Found by Felix Von Leitner (n.Runs)

Windows Vista Security Approach

Everything in Windows XP SP2 SDL *plus*

- Apply least privilege throughout architecture
- Automate proven techniques
- Methodically apply security expertise on whole OS
- Additional Defense-in-Depth mitigations

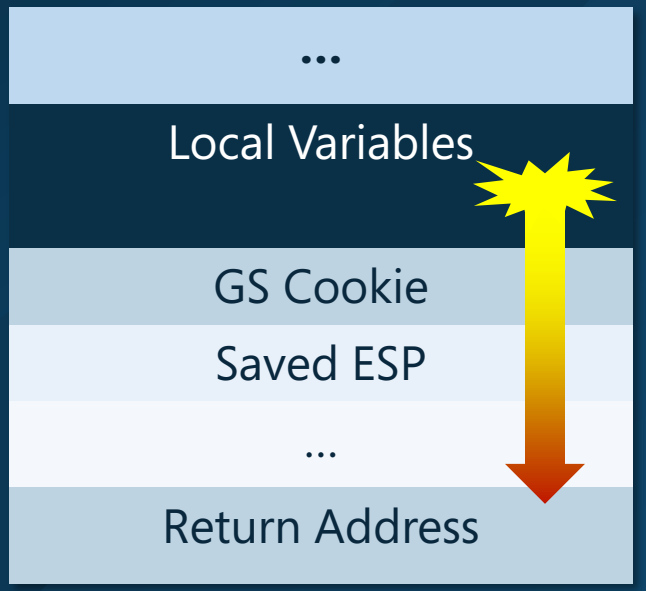
The Goal:

Stop playing Whac-a-Mole
Find and fix vulnerabilities before shipping

LAYER one

MSVC 8.0 (Visual C++ 2005, a.k.a. Whidbey): *How About a Nice Game of Chess?*

- Improved /GS Flag
- Significantly improved stack protection in compiler
- Annotations force more aggressive protection in forward facing areas



Heap Hardening

- Lookasides no longer used
- Early error detection due to block header integrity check
- Dynamic algorithm adjustment based upon usage (target attacks)
- Pseudo-random base address
- Heap TerminateOnCorruption
  - On by default for 64bit
  - On for services and most apps on x86
- See Adrian Marinescu's talk
  (Black Hat US, 2006)

Function Pointer Encoding

- Function pointer encoding
  - Overwriting a function pointer in a predictable location is a common technique to gain control of EIP
  - Encode function pointer with secret; Decode prior to dereference
  - Decreases reliability of exploit by increasing chances of process termination due to AV on EIP, NX exception, invalid instruction, etc.
- APIs
  - EncodePointer - XOR w/per process cookie
  - EncodeSystemPointer - XOR w/shared cookie in SharedUserData

Data Execution Protection *(a.k.a. NX)*

- Most Windows Vista PCs have hardware support for NX
- Default Mode: Opt-in
  - EXEs linked with /NXCOMPAT:YES have NX turned on permanently
  - All Windows services and most EXEs opted in
- You can switch to Opt-out mode
  - Everything has NX turned on
  - Exception list is configurable in registry

What's Been Said About DEP

As can be seen, the technique described in this document outlines a feasible method that can be used **to circumvent the security enhancements provided by hardware-enforced DEP...**

First and foremost, the technique depends on knowing the location of three separate addresses...**The first dependency could be broken by instituting some form of Address Space Layout Randomization** that would thereby make the location of the dependent code blocks unknown to an attacker.

http://www.uninformed.org/?v=2&a=4

mmiller at hick.org, Skywing at valhallalegends.com

Address Space Layout Randomization *(ASLR)*

- Powerful complement to Data Execution Protection
- Images must opt-in via bit in PE header: DYNAMIC_BASE
  - New linker adds support; also emits reloc in EXEs
- Limited number of bits available for randomness
  - Main trade-off
    - How difficult do you want the guess to be? vs.
    - How much contiguous virtual address space do you want to be available to apps?
  - Currently set so that 99.6% of the time, your first guess will fail
- Impact:
  - No major hit on performance. Some wins. Some minor losses.
  - Application compatibility is good with current design
- All of Windows Vista is Opted-in ☺

LAYER one

Agenda

- Introduction
  - Who Am I?
  - Goals of This Talk
- Windows Vista Security Approach
- Retrospective Case Study
- Q & A

SDL Case Study: Zotob Worm

Remote unauthenticated code execution possible (No SDL prior to ship)

Exploit requires authentication (ACL restricted)

No remote security threat (Security RPC Callback added)

No remote security threat (Reviewed and implemented Windows Server 2003 changes)

Even if we had missed it in Windows XP SP2

Blocked by firewall that is on by default

LAYER one

Zotob Worm in Windows Vista?

Improved PREfast & PREfix Code Scanners

And if that failed…

RPC Fuzzing and Penetration Testing

And if that failed…

Protected by improved version of /GS and SafeSEH

And if that failed…

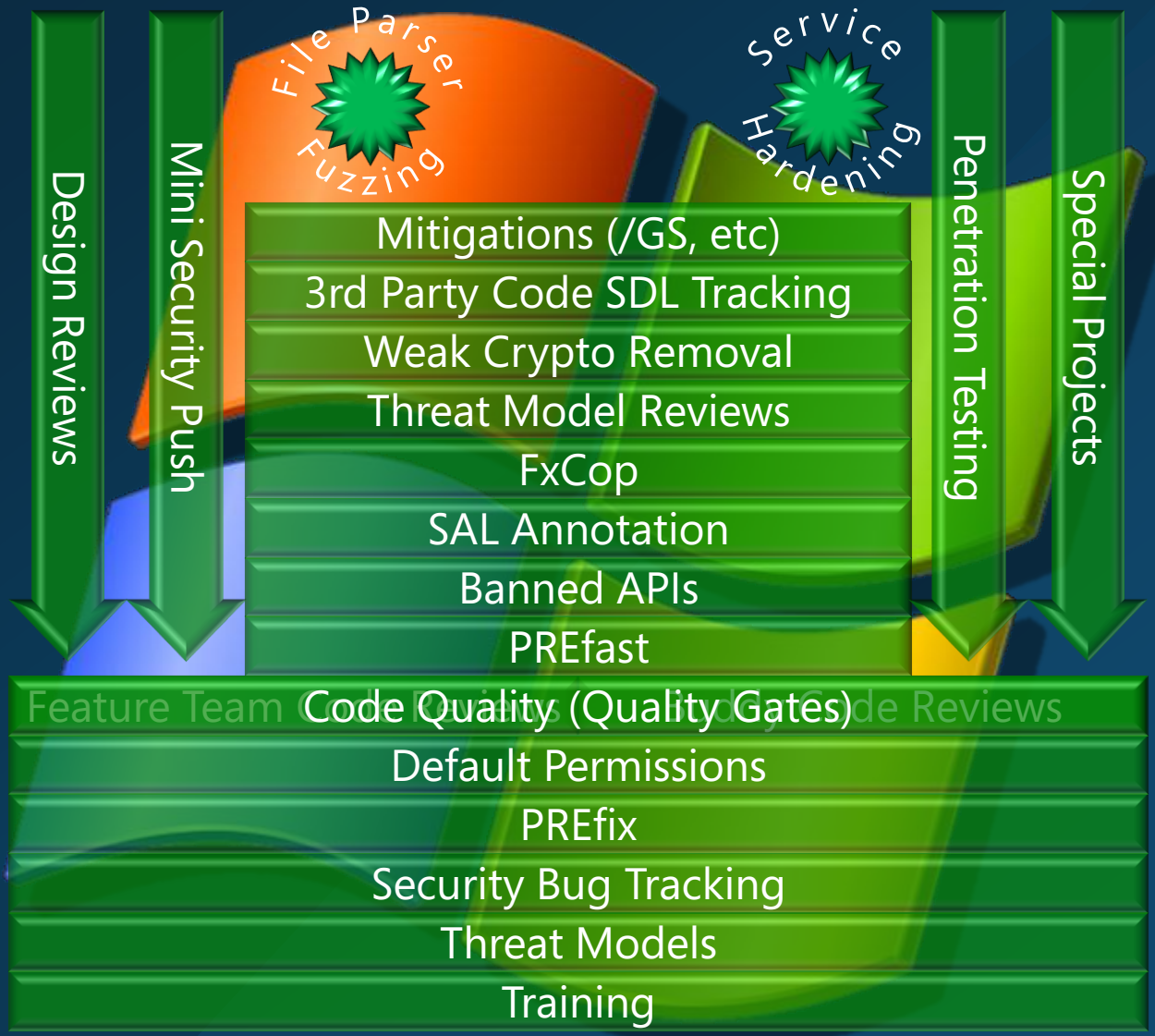
Protected by NX and ASLR

And if *that* failed…

Still blocked by firewall that is on by default

LAYER one

Windows xp SP2



LAYER one

Agenda

- Introduction
  - Who Am I?
  - Goals of This Talk
  - Quick Overview of Security Development Lifecycle
- Windows Vista Security Approach
- Retrospective Case Study
- Q & A

Secure@Microsoft.com

LAYER one

Questions?

LAYER one