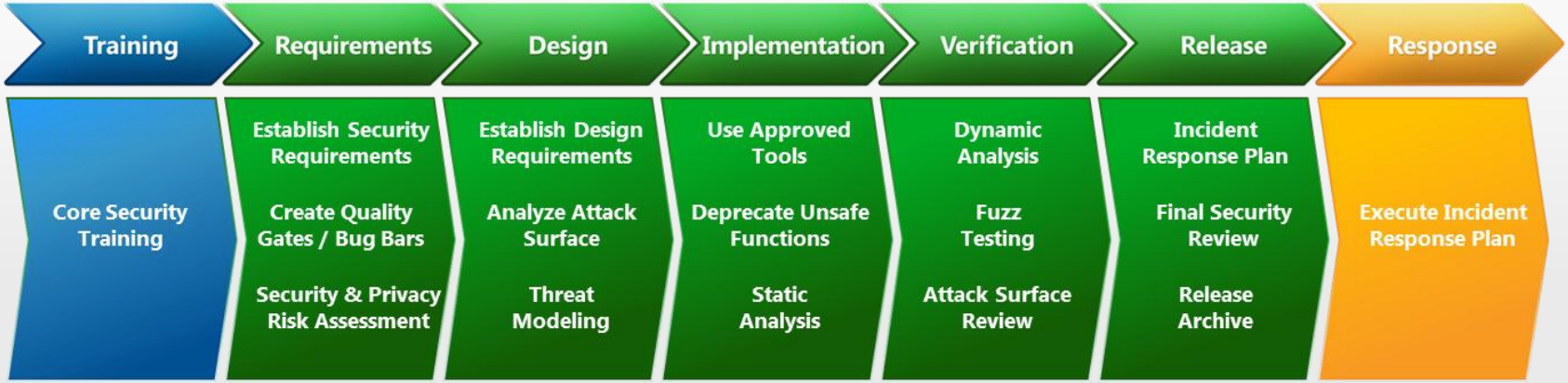
Fending Off Attacks by Reducing an Application's Attack Surface

Jason Taylor
CTO, Security Innovation
an SDL Pro Network member company

The Microsoft Security Development Lifecycle (SDL)

The industry-leading software security assurance process

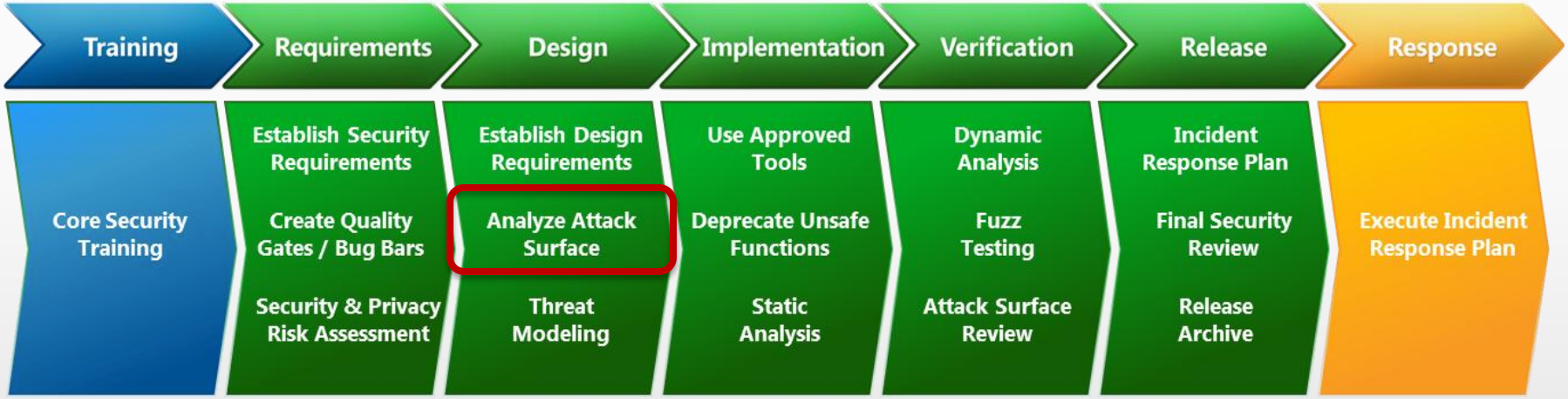
Combining a holistic and practical approach, the SDL introduces security and privacy throughout all phases of the development process.



Download the *Simplified Implementation of the Microsoft SDL* to learn more about the Security Development Lifecycle process and practices.

Analyze Attack Surface

Required design phase activity for the Microsoft SDL



Agenda

- **Objectives of Attack Surface Analysis & Reduction**
- Measuring and Reducing your Attack Surface
- Conclusion

A Software's Attack Surface

Software exposes key business and customer assets via entry points:

- User interfaces
- Web services
- Direct database access
- Network channels, pipes, files, APIs …

The entire collection of entry points in a product is called its Attack Surface

- Set of ways in which adversaries can attack a software system
- Not limited to local resources; channels used to communicate with remote resources are vectors for attack

Big Attack Surface = Big Security Work= Big Security Problems

- With big attack surface, you must spend more time making sure the code is excellent quality, conservative, and defensive

Understanding Attack Surface

Attack Surface can Gauge Risk of Attack

- Attack surface measures potential avenues of attack
- Attack surface measures potential impact of an attack

Useful as a Metric in Design through Deployment

- Measurement is both point-in-time and relative

Attacks over the years show that certain vectors are more likely to be opportunities of attack than others

- Known (and un-patched) vulnerabilities
- Services running as privileged user root (in UNIX) are more likely to be targets of attack than services running as non-root users
- Files with full control (e.g., rwxrwxrwx) are more likely to be attacked than files with less generous permissions
- Symbolic links are highly likely to be used as enablers in attacks
- Windows apps with VBScript, JScript, or ActiveX controls enabled are more likely to be enablers of attack than if such scripting engines and controls were disabled

Main Goals of Attack Surface Reduction

Reduce the amount of code executing by default

Reduce the volume of code that is accessible to untrusted users by default

Close doors (access points) that can be easily opened/exploited

Limit the damage if the access point is exploited

Bottom Line: Fend off Future Attacks

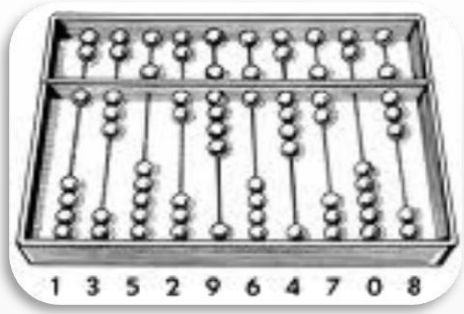
Agenda

Objectives of Attack Surface Analysis

Measuring and Reducing your Attack Surface

Conclusion

Common Metrics for Software Security



At the Code Level

- We count the number of bugs found
- Not an ideal metric because:
  - It misses bugs that are not found (or fixed), perhaps the one that gets exploited
  - Equal importance is given to all bugs, even though some bugs are easier to exploit than others
  - The exploit of one may result in more damage than the exploit of another

At the Products/System Level

- We count the number of times a system version is mentioned in CERT, Microsoft Security Bulletins, MITRE CVEs, etc
- Not an ideal metric because it ignores the specific system configuration that gave rise to the vulnerability
  - Whether a security patch has been installed
  - Whether defaults are turned off
  - Whether it always runs in system administrator mode

New Metric to Add: Attack Surface

Rather than count bugs at code level or vulnerability reports at system level, you can measure the avenues of attack

- The counts of these "more likely to be attacked" features determine a system's attack surface

Attack Surface attempts to measure security in relative terms

- Useful as a delta measurement (i.e. v1.2 to v1.n of a product)
- Not useful for comparing app to app or as stand-alone metric

Measuring attack surface

Start with baseline, then measure each week

- Define your minimal attack surface early on during design

If the attack surface count goes up, determine why and see if you can drive it back down

- When engineers know you are measuring the attack surface, they may improve efforts

Baseline	Baseline + 1 month	Comment
3 x TCP ports	2 x TCP ports	Good; one fewer port to worry about.
1 x UDP port	2 x UDP port	Which functionality opened the new UDP port? Why is it open by default? Is it authenticated? Is it restricted to a subnet?
2 x Services (both SYSTEM)	3 x Services (2 x SYSTEM, 1 x LocalService)	Why is another service running by default? Why are any running as SYSTEM?
3 x ActiveX controls	4 x ActiveX controls	Why is the new control installed? Is it safe for scripting?
No additional user accounts	1 x application account	Turns out this is a member of the administrators group too! Why? What's the password?

Potential Uses for Attack Surface Measurement

Developers and architects can better prioritize testing efforts

- If a software's attack surface measurement is high, they may want to invest more in testing

Developers can use it as a guide while implementing patches of security vulnerabilities

- A good patch should not only remove a vulnerability from a system, but also should not increase the system's attack surface

Consumers can use it to guide their choice of configuration

- Since a system's attack surface measurement is dependent on the system's configuration, software consumers would choose a configuration that results in a smaller attack surface exposure

Risk Management/Corporate Security can understand their potential business exposure

Attack Surface & Entry Points

To attack an application, attackers will

- Identify entry points
- Exploit the entry point or the backend exposed behind it
- Try to deny legitimate users access to these entry points

In our software, we have to:

- Understand the attack surface
- Reduce it as much as possible
- Test against it for exposed vulnerabilities

Mapping our Attack Surface

- Architecture and design docs, code..
- But how many entry points are undocumented
  - Temporary files? Pipes? Network communication? Registry keys? Embeddable code?

The Attack Surface Reduction Process

Look at all of your entry points

- Network I/O
- File I/O

Rank them

- Authenticated versus anonymous
- Administrator only versus user
- Network versus local

Attack Surface Reduction is as important as trying to get the code right

It's Not *Just* About Turning Stuff Off!

Higher Attack Surface	Lower Attack Surface
Executing by default	Off by default
Open socket	Closed socket
Anonymous access	Authenticated access
Constantly on	Intermittently on
Admin access	User access
Internet access	Local subnet access
SYSTEM	Not SYSTEM!
Uniform defaults	User-chosen settings
Large code	Small code
Weak ACLs	Strong ACLs

Reducing your Attack Surface – Best Practices

Reduce the amount of code running by default

Reduce access to entry points by untrusted users

- Restrict access to network endpoints used by your application to the local subnet or IP range
- Limit access to network endpoints using authentication

Reduce privilege to limit damage potential

- Sandbox running code
- Give higher permissions for as short a period as possible
- Pay attention to trust boundaries

Define attack surface during design

ASR Best Practices

Reduce the Amount of Running Code

Apply the 80/20 rule

- Ask yourself, "Do 80 percent of users/customers use this feature?"
- If the answer is no, then turn it off and make it easy to turn the feature back on

Services can't be attacked if they are not running

- All forms of dynamic Web content should be opt-in
- If there is security defect in a service, this ensures that the only affected users are those actively running the service (versus all software users)

ASR is not just an "on or off" proposition

- You can reduce attack surface by limiting who can access the code once it is running
- This has an interesting side effect: many features are still usable, but are inaccessible by attackers (must have knowledge of access controls?)

ASR Best Practices

Reduce access to entry points by untrusted users

Example: Web Site

- By default, most Web sites are accessible to all users
- If the Web server limited access only to trusted users, or users on a specific subnet, then fewer users could take advantage of the defect
- This is good because the code works for those who require the functionality
- Restrict access to network endpoints used by your application to the local subnet or IP range

To reduce the number of users who can communicate with potentially vulnerable code, authenticate before they can access the code

- Limit access to network endpoints using authentication. Simply validating the user reduces your component's attack surface
- Most bad guys are unauthenticated, and by raising the authentication bar a little you can eliminate a large population of potential attackers

ASR Best Practices

Reduce access to entry points by untrusted users

Pay attention to trust boundaries

- Review the entire data flow from entry point to data store
- Look for all the potential data flows, data stores, and processes that the code touches (code that an attacker can touch)
- Determine what privilege is required to access that entry point
- Can you raise the authorization bar?

Example: Code with only two network-facing entry points

- One is anonymous and the other can only be accessed by administrators
- Hacker is going to attack the code along the anonymous data path simply because they can
  - Not going to exercise the administrative code path because, if they can, then they are already an administrator (and your problems are only just starting) ☺

As you review dataflow, look for anonymous threat paths. It may be prudent to raise the authorization bar on these paths

ASR Best Practices

Reduce privilege to limit damage potential

Even if your code has vulnerabilities, and you don't limit access to the code, you can still reduce the damage potential by operating the code with reduced privileges

- Remove any privileges not explicitly required by the application

Example: Dangerous Backup Privilege

- Any process running under an account with this privilege can read any file in the file system, regardless of the access control list (ACL) on the file
- Wrong way
  Run as SYSTEM, which has backup privilege, but also has restore, debug, and "act as part of operating system" privilege; and is an admin account too
- Right Way:
  Run service under a specific authenticated account that has this privilege

Never run services as **SYSTEM, or daemons as root, or user accounts with administrative rights** unless you have exhausted all other possibilities

Define Attack Surface During Design

When designing your application, outline what the attack surface will look like and identify the following:

- The networking protocols you enabled by default
- The endpoints that should support authentication and authorization. Focus on other anonymous endpoints
- The off-by-default features
  - Look for code that auto-starts or will execute when accessed, such as services, daemons, ISAPI filters and applications, SOAP services, and Web roots
- Reusable components (ActiveX® controls, COM objects, and .NET Framework assemblies, etc)
- Process identities for all the code you run
- User accounts installed

When developers know early what the attack surface looks like, they can design and code defensively from the start

Other Activities to help Minimize your Attack Surface

Design Phase: Threat Modeling

- Identify threats and reduce risk
- Identify trust boundaries
- Discover entry points and design flaw
- SDL Threat Modeling Tool: http://go.microsoft.com/?linkid=9706808

Verification Phase

Attack Surface Review

- Leverage Attack Surface Analyzer developed by Microsoft's Security Engineering group
- Takes a snapshot of system state before and after the installation of product(s)
- Displays the changes to a number of key elements of the Windows attack surface
- BETA version :http://go.microsoft.com/?linkid=9758398

Design Review

- Check design for attack surface reduction opportunities
- Check design against threats uncovered in the threat model

Code Review

- Increase awareness of secure coding principles
- Discover dangerous code changes that add vulnerabilities or impact attack surface

Agenda

Objectives of Attack Surface Analysis

Measuring and Reducing your Attack Surface

Conclusion

Conclusion

Use Attack Surface Reduction to reduce your overall security profile

- Protect against the threats and vulnerabilities you don't know about

Match attack surface risk to the level of effort devoted to secure design, development and testing

- Include Attack Surface analysis as part of your overall strategy
- Include design, development and test best practices as effective ways to minimize your attack surface and overall risk

Resources

The Microsoft Security Development Lifecycle (SDL)
www.microsoft.com/sdl

The Simplified Implementation of the Microsoft SDL whitepaper
http://go.microsoft.com/?linkid=9708425

Attack Surface Analyzer BETA
http://go.microsoft.com/?linkid=9758398

Special offering for webcast live attendees. Benefit from one week's free access to the following eLearning classes

- Attack Surface Analysis
  http://www.securityinnovation.com/products/elearning/attack-surface-analysis.shtml
- Introduction to the Microsoft SDL
  http://www.securityinnovation.com/products/elearning/intro-sdl.shtml

About Security Innovation

MS SDL Pro Network Member Company

Provides Application Security Assessment Services, Consulting and Training

- Code Review, Penetration Testing
- SDLC Optimization
- ELearning and instructor-led training

Contact

- Getsecure@securityinnovation.com
- 1.877.694.1008 x1
- www.securityinnovation.com
- Technical questions: jtaylor@securityinnovation.com

Questions & Answers

- Submit text questions using the "Ask" button.
- Send us your feedback and content ideas in the survey.
- Replay of this webcast will be available in 24 hours.
- Get the latest developer content (webcasts, podcasts, videos, virtual labs) at: www.Microsoft.com/Events/Series/
- For more security webcasts: www.microsoft.com/events/series/securitytalk
- Check out **Windows Azure** Subscriptions: bit.ly/TryWindowsAzure

Microsoft®

Your potential. Our passion.™