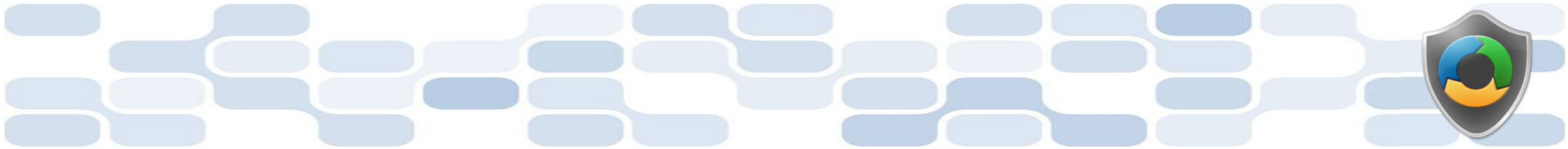
Introduction to Microsoft® Security Development Lifecycle (SDL) Threat Modeling

Secure software made easier

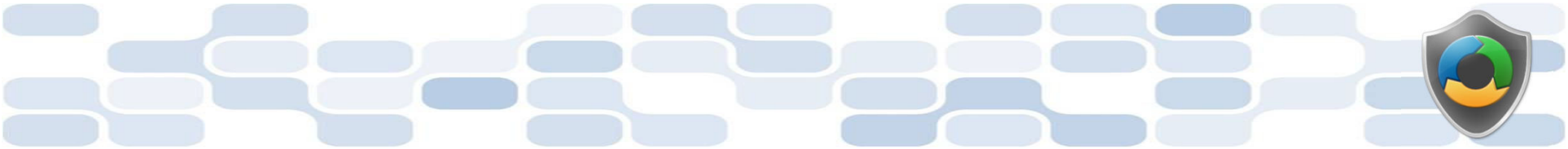
Presenter Name

Date



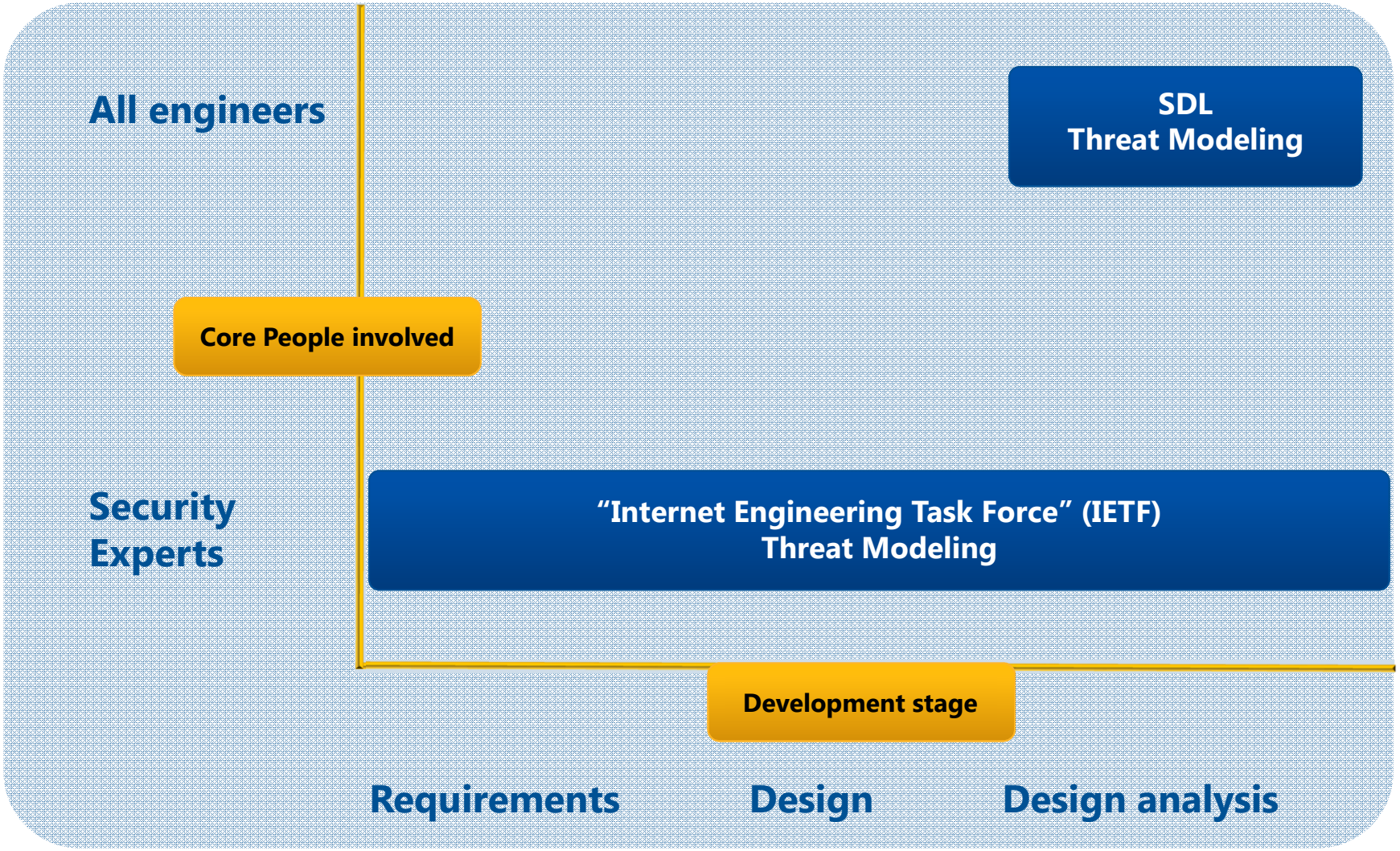
Course Overview

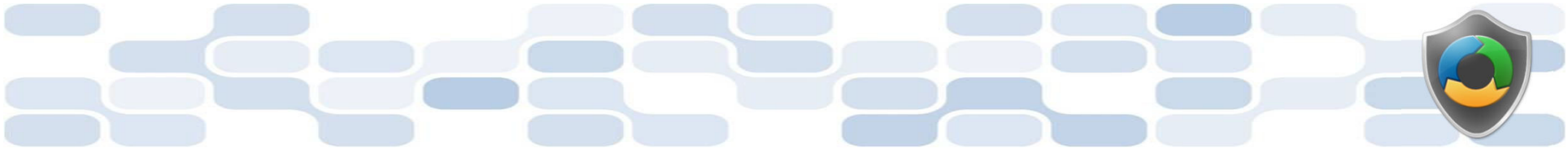
- Introduction and Goals
- How to Threat Model
- The STRIDE per Element Approach to Threat Modeling
- Diagram Validation Rules of Thumb
- Exercise
- Demo



Introduction and Goals

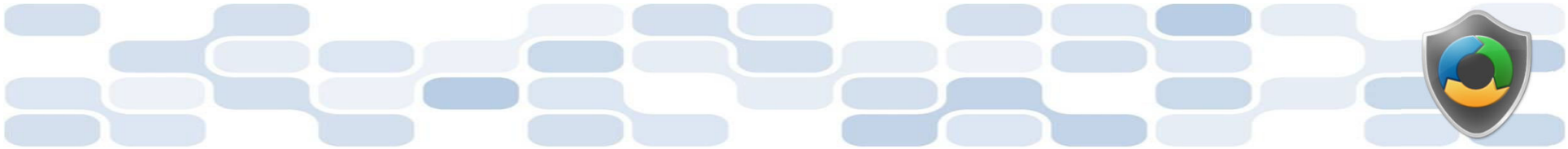
Terminology and Context





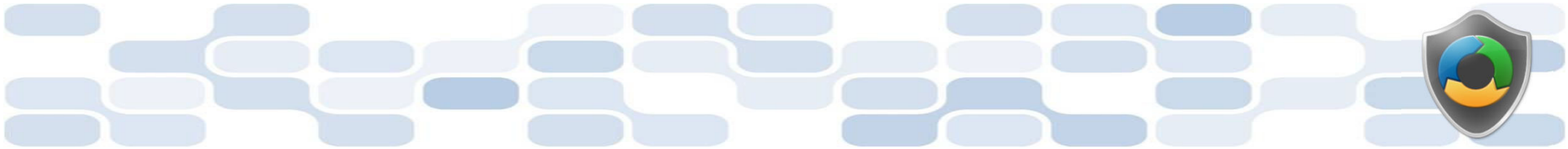
Threat Modeling Basics

- **Who?**
  - The bad guys will do a good job of it
  - Maybe you will…your choice
- **What?**
  - A repeatable process to find and address all threats to your product
- **When?**
  - The earlier you start, the more time to plan and fix
  - Worst case is for when you're trying to ship: Find problems, make ugly scope and schedule choices, revisit those features soon
- **Why?**
  - Find problems when there's time to fix them
  - Security Development Lifecycle (SDL) requirement
  - Deliver more secure products
- **How?**



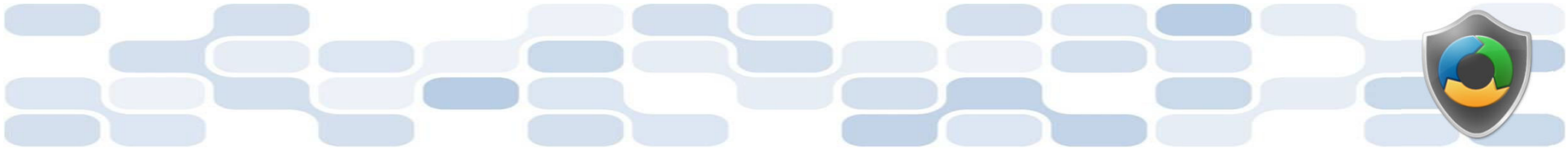
Who

- Building a threat model (at Microsoft)
  - Program Manager (PM) owns overall process
  - Testers
    - Identify threats in analyze phase
    - Use threat models to drive test plans
  - Developers create diagrams
- Customers for threat models
  - Your team
  - Other features, product teams
  - Customers, via user education
  - “External” quality assurance resources, such as pen testers
- You’ll need to decide what fits to your organization



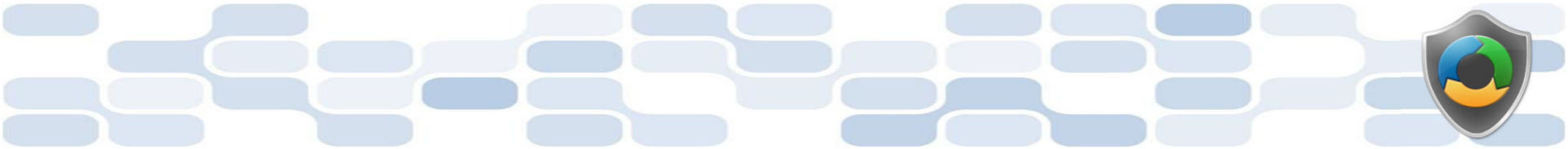
What

- Consider, document, and discuss security in a structured way
- Threat model and document
  - The product as a whole
  - The security-relevant features
  - The attack surfaces
- Assurance that threat modeling has been done well

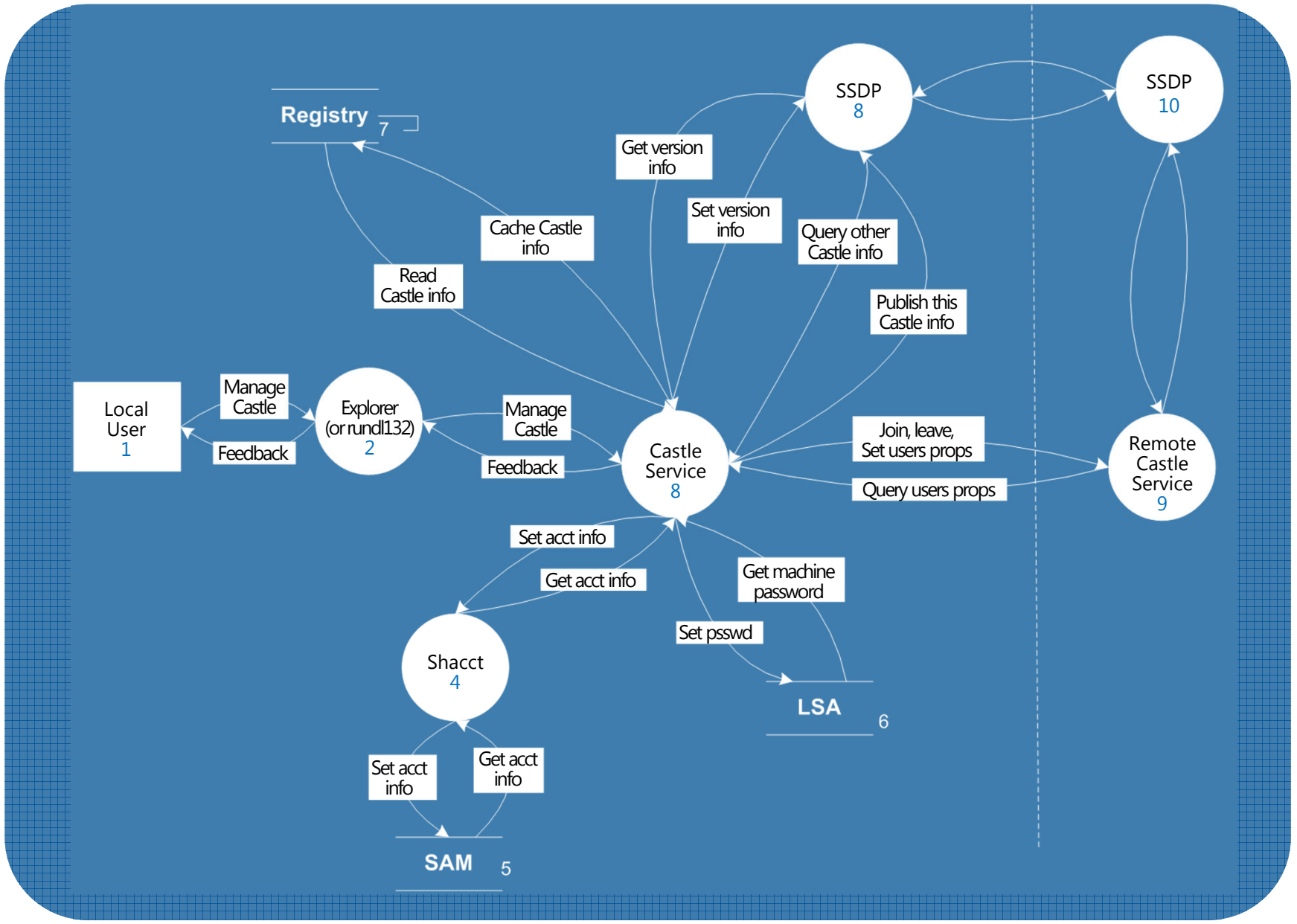


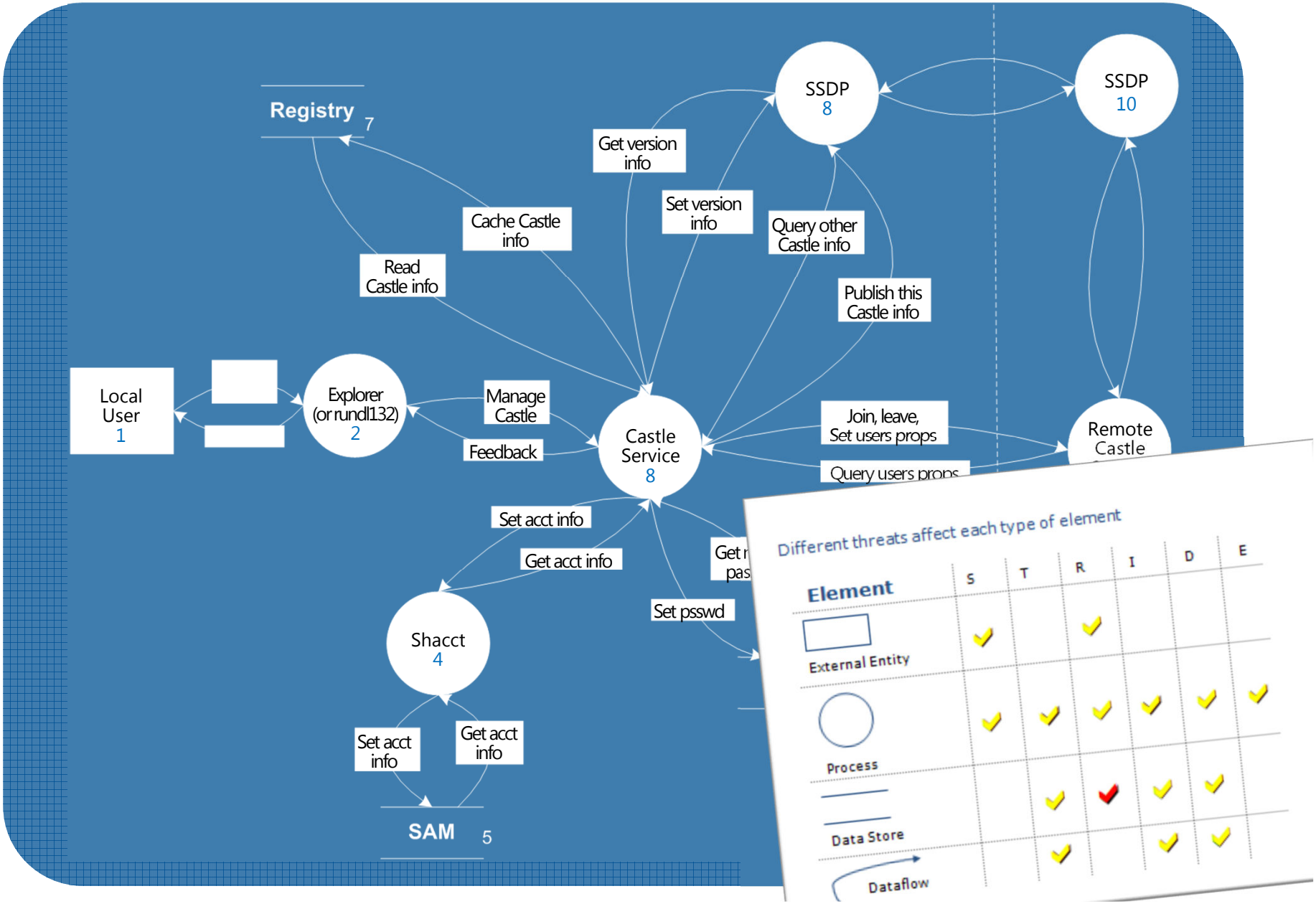
Why

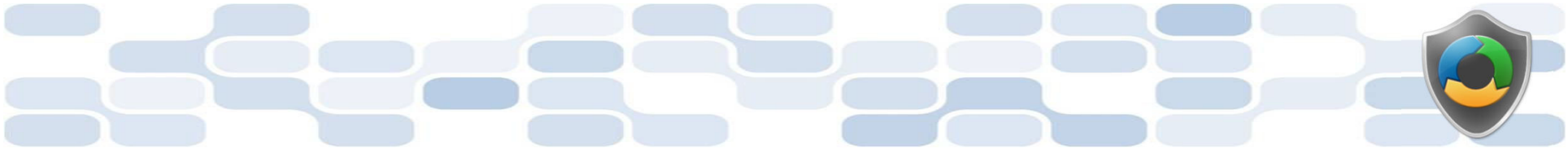
- Produce software that's secure by design
  - Improve designs the same way we've improved code
- Because attackers think differently
  - Creator blindness/new perspective
- Allow you to predictably and effectively find security problems early in the process



How to Threat Model

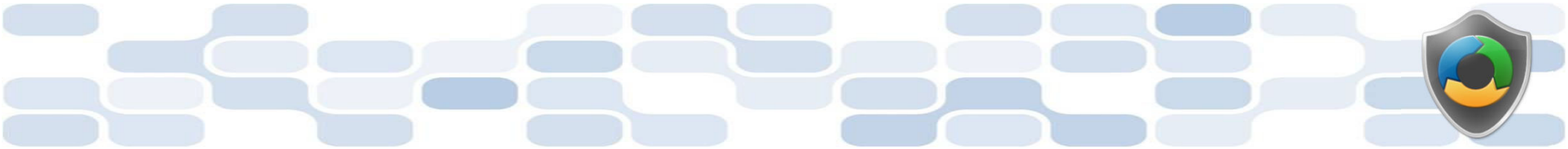

Registry
7
SSDP
8
SSDP
10
Get version info
Set version info
Query other Castle info
Cache Castle info
Read Castle info
Publish this Castle info
Local User
1
Manage Castle
Feedback
Explorer (or rundll32)
2
Manage Castle
Feedback
Castle Service
8
Join, leave, Set users props
Query users props
Remote Castle Service
9
Set acct info
Get acct info
Get machine password
Set psswd
Shacct
4
LSA
6
Set acct info
Get acct info
SAM
5


Registry 7
SSDP 8
SSDP 10
Get version info
Set version info
Query other Castle info
Cache Castle info
Read Castle info
Publish this Castle info
Local User 1
Explorer (or rundl132) 2
Manage Castle
Feedback
Castle Service 8
Join, leave, Set users props
Query users props
Remote Castle
Set acct info
Get acct info
Set pssw
Shacct 4
Set acct info
Get acct info
SAM 5
Different threats affect each type of element
Element
S
T
R
I
D
E
External Entity
Process
Data Store
Dataflow



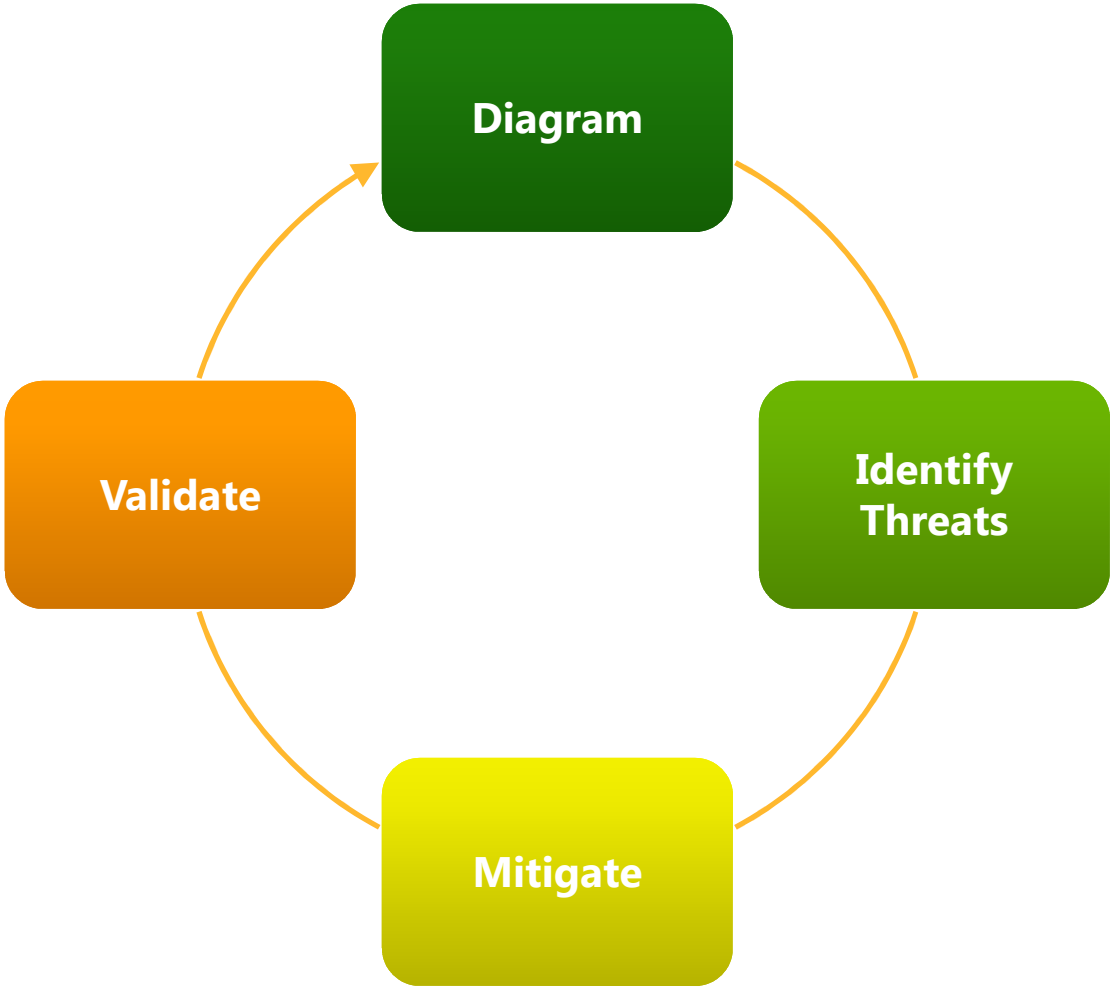
Any Questions?

- Everyone understands that?
- Spotted the several serious bugs?
- Let’s step back and build up to that

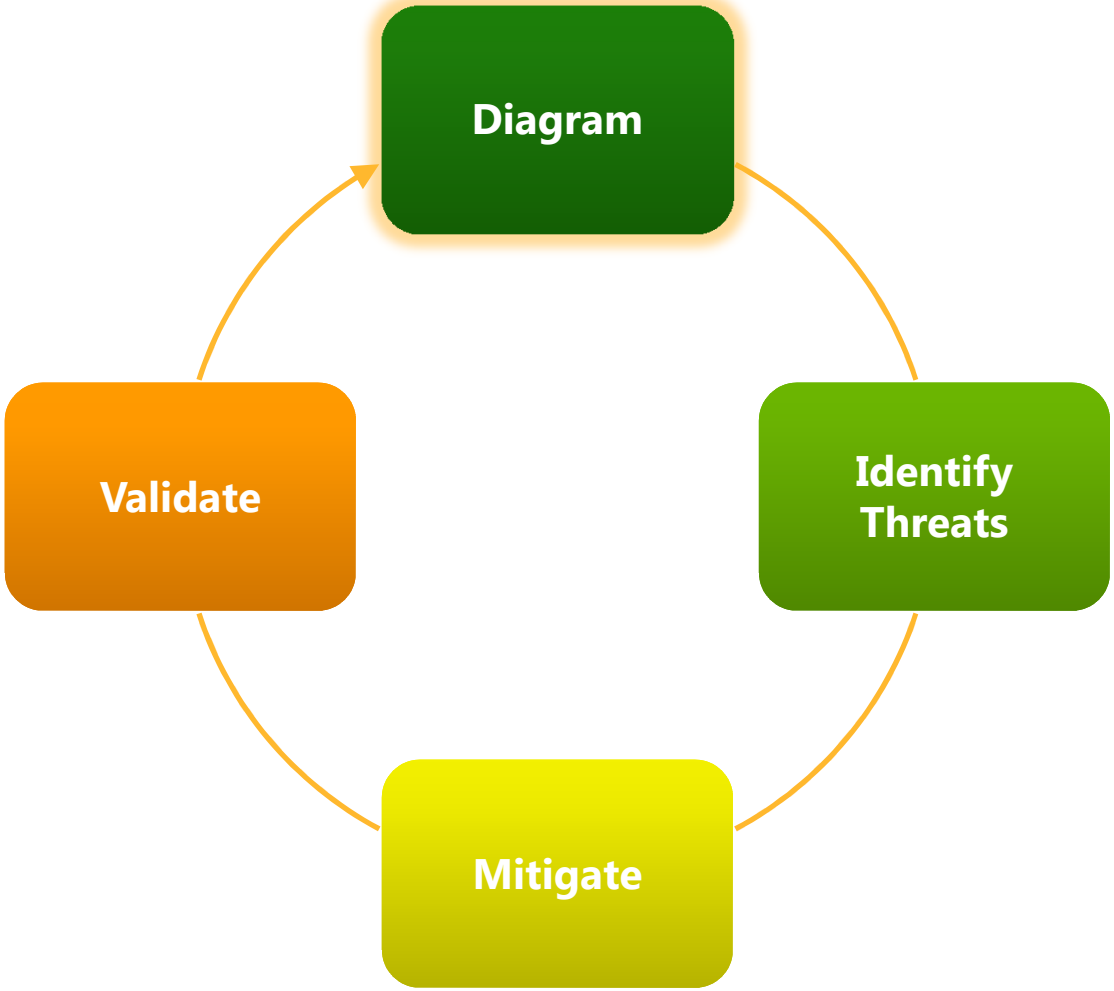


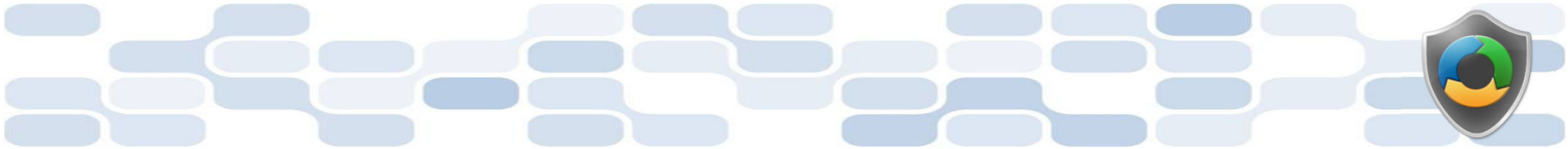
The STRIDE per Element Approach to Threat Modeling

The Process in a Nutshell



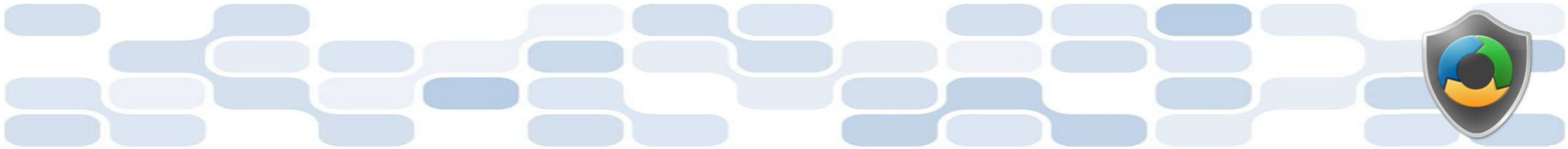
The Process: Diagramming





How to Create Diagrams

- Go to the whiteboard
- Start with an overview which has:
  - A few external interactors
  - One or two processes
  - One or two data stores (maybe)
  - Data flows to connect them
- Check your work
  - Can you tell a story without edits?
  - Does it match reality?



Diagramming

- Use DFDs (Data Flow Diagrams)
  - Include processes, data stores, data flows
  - Include *trust boundaries*
  - Diagrams per scenario may be helpful
- Update diagrams as product changes
- Enumerate assumptions, dependencies
- Number everything (if manual)

Diagram Elements: Examples

External Entity

- People
- Other systems
- Microsoft.com

Process

- DLLs
- EXEs
- COM object
- Components
- Services
- Web Services
- Assemblies

Data Flow

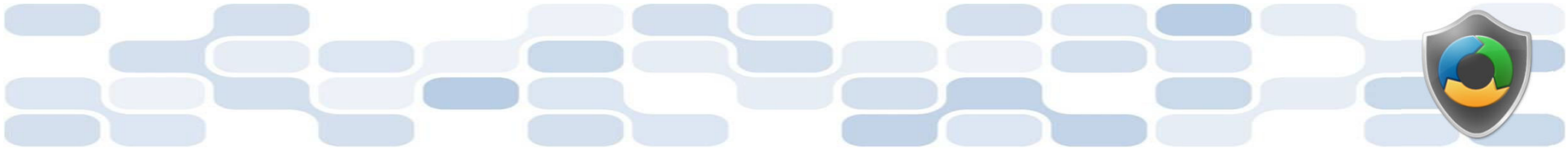
- Function call
- Network traffic
- Remote Procedure Call (RPC)

Data Store

- Database
- File
- Registry
- Shared Memory
- Queue / Stack

Trust Boundary

- Process boundary
- File system



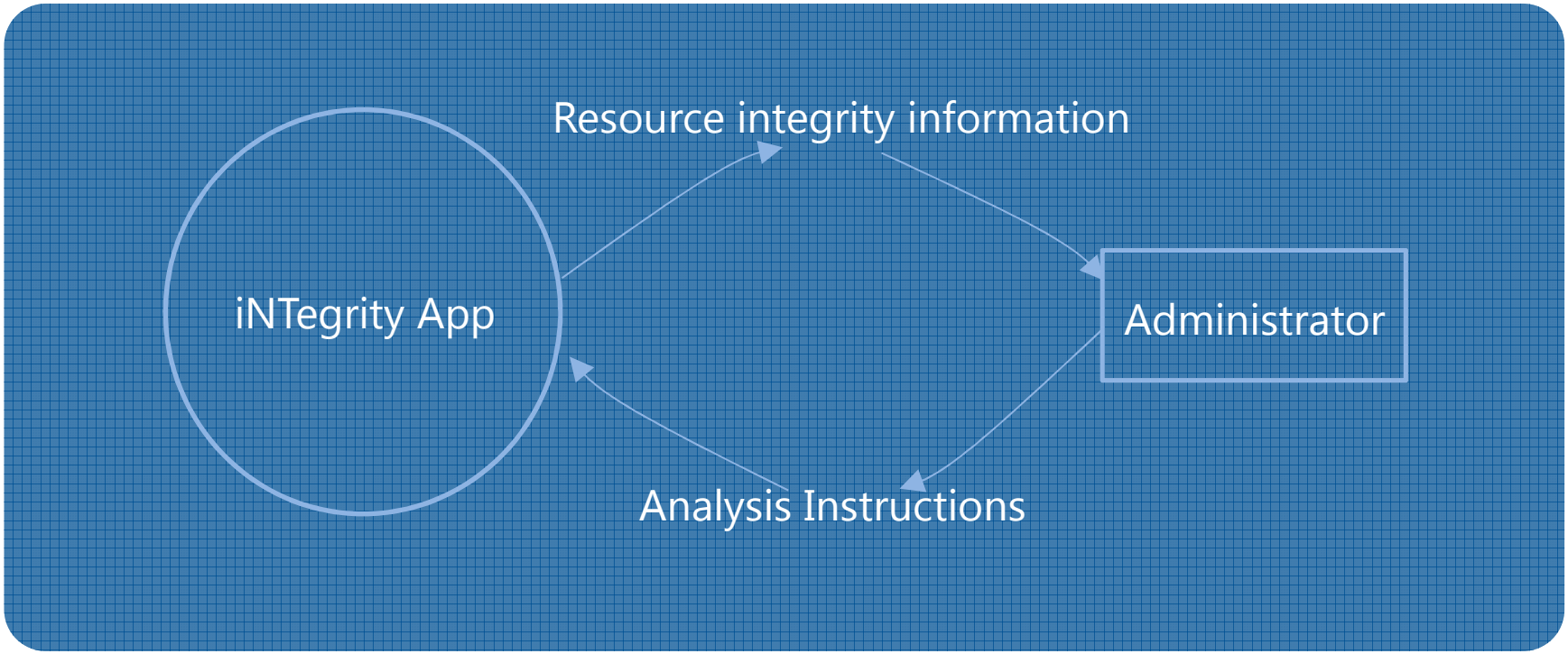
Diagrams: Trust Boundaries

- Add trust boundaries that intersect data flows
- Points/surfaces where an attacker can interject
  - Machine boundaries, privilege boundaries, integrity boundaries are examples of trust boundaries
  - Threads in a native process are often inside a trust boundary, because they share the same privs, rights, identifiers and access
- Processes talking across a network always have a trust boundary
  - They make may create a secure channel, but they're still distinct entities
  - Encrypting network traffic is an 'instinctive' mitigation
    - But doesn't address tampering or spoofing

Diagram Iteration

- Iterate over processes, data stores, and see where they need to be broken down
- How to know it “needs to be broken down?”
  - More detail is needed to explain security impact of the design
  - Object crosses a trust boundary
  - Words like “sometimes” and “also” indicate you have a combination of things that can be broken out
    - “Sometimes this datastore is used for X”…probably add a second datastore to the diagram

Context Diagram



Level 1 Diagram

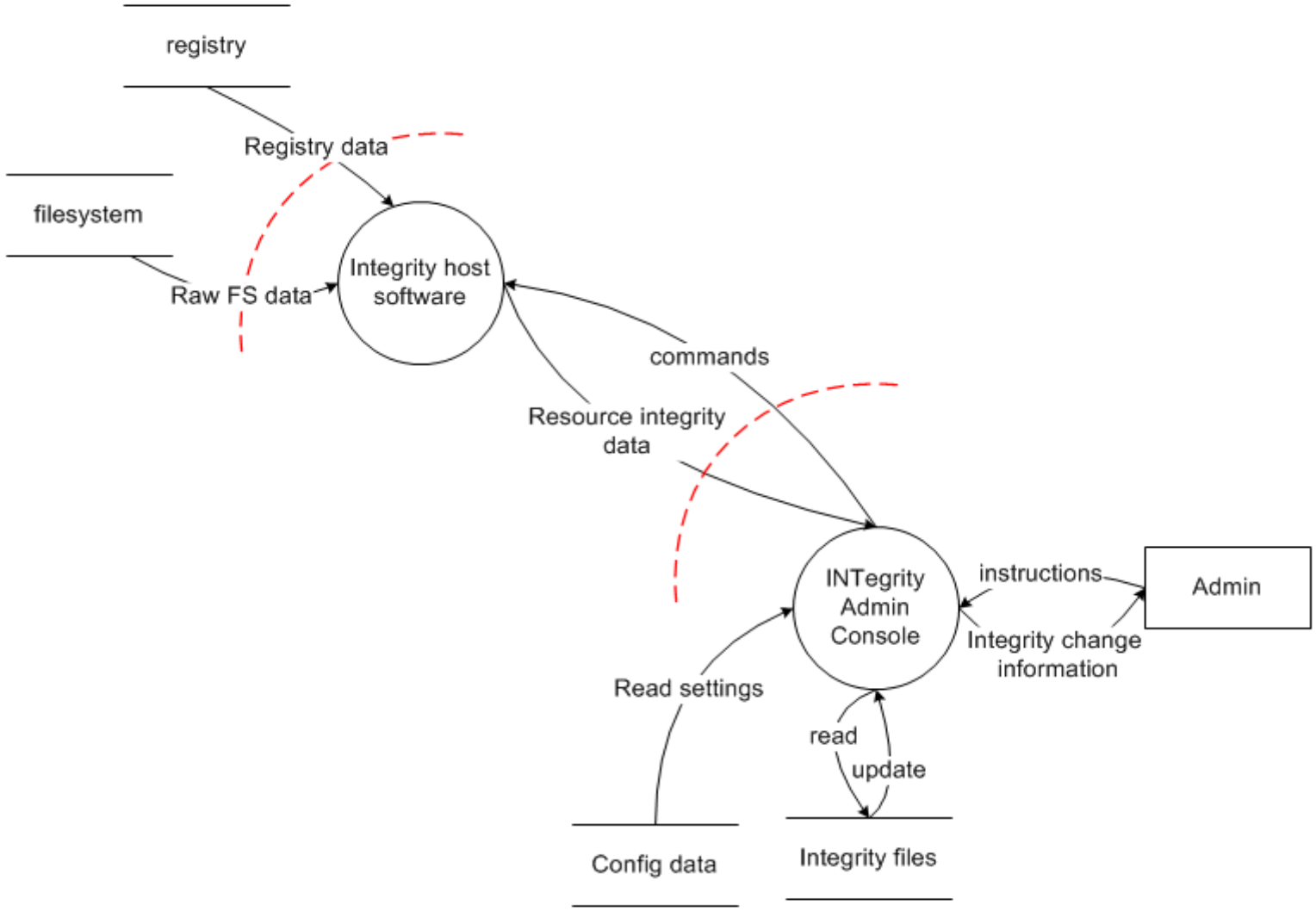
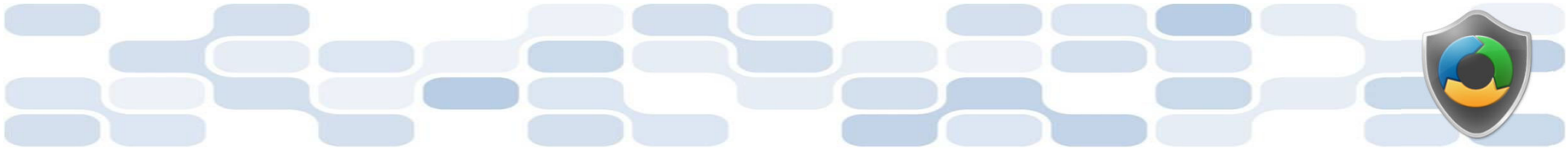


Diagram layers

- Context Diagram
  - Very high-level; entire component / product / system
- Level 1 Diagram
  - High level; single feature / scenario
- Level 2 Diagram
  - Low level; detailed sub-components of features
- Level 3 Diagram
  - More detailed
  - Rare to need more layers, except in huge projects or when you're drawing more trust boundaries



Creating Diagrams: analysis or synthesis?

- Top down
  - Gives you the “context” in context diagram
  - Focuses on the system as a whole
  - More work at the start
- Bottom up
  - Feature crews know their features
  - Approach not designed for synthesis
  - More work overall

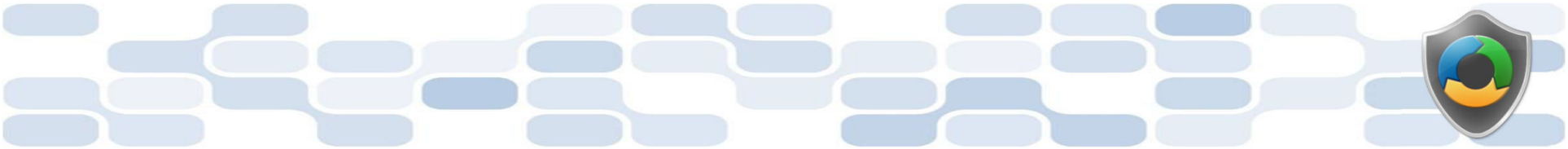
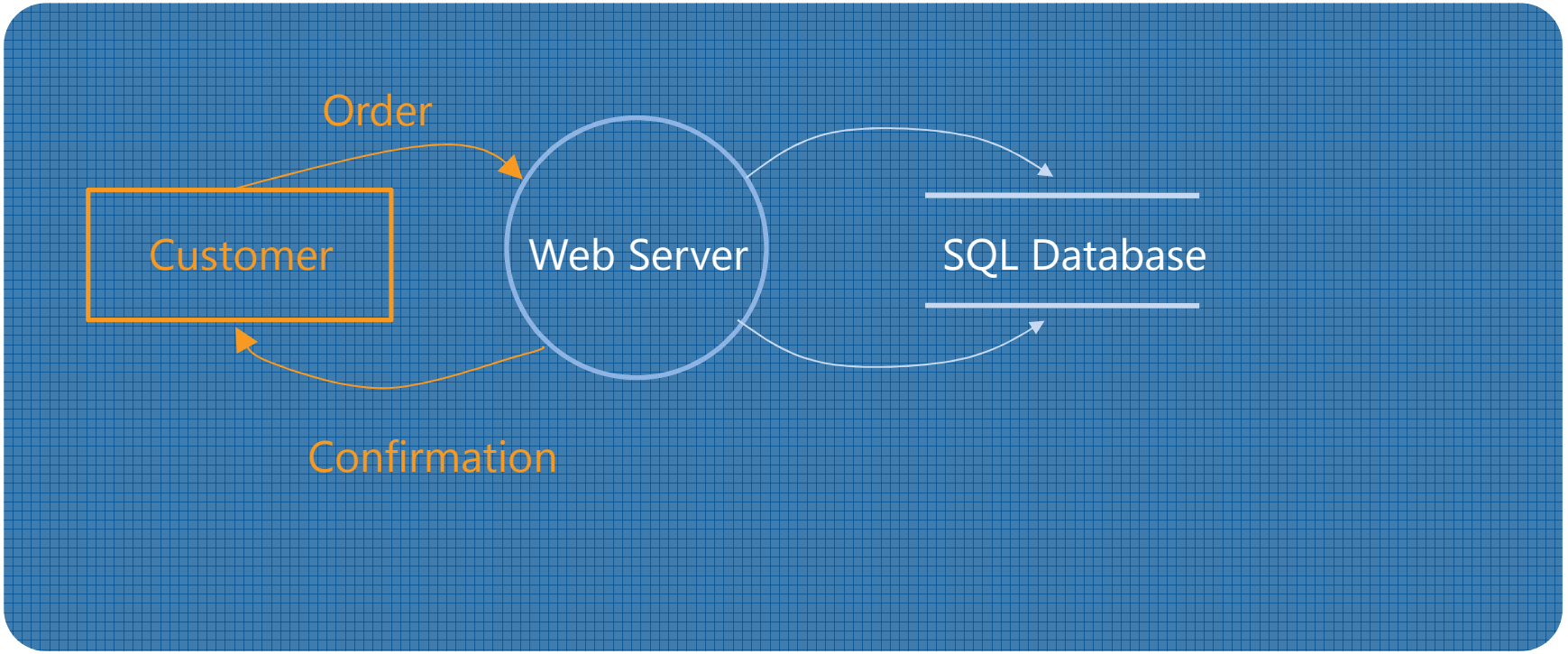


Diagram Validation Rules of Thumb

Diagram Validation Rules of Thumb

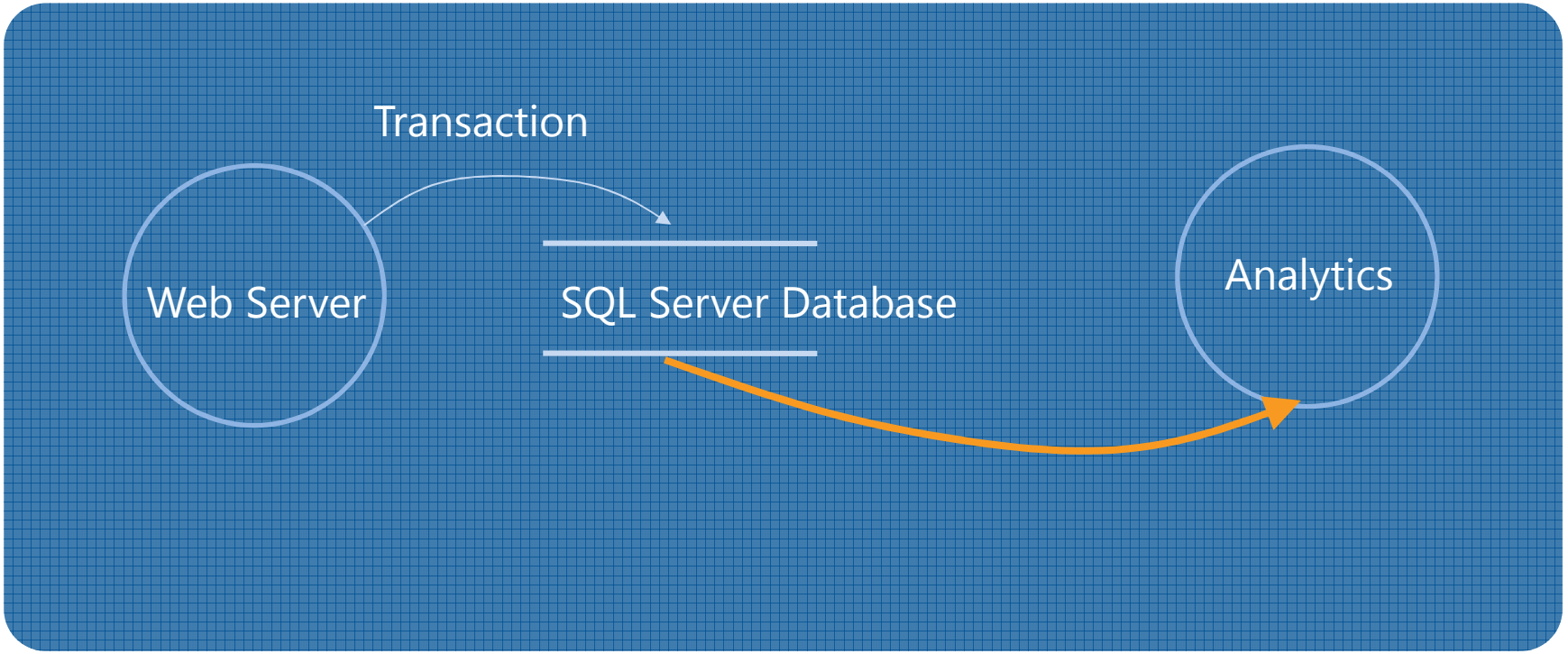
Does data magically appear?



Data comes from external entities or data stores

Diagram Validation Rules of Thumb

Are there data sinks?



You write to a store for a reason: Someone uses it.

Diagram Validation Rules of Thumb

Data doesn't flow magically

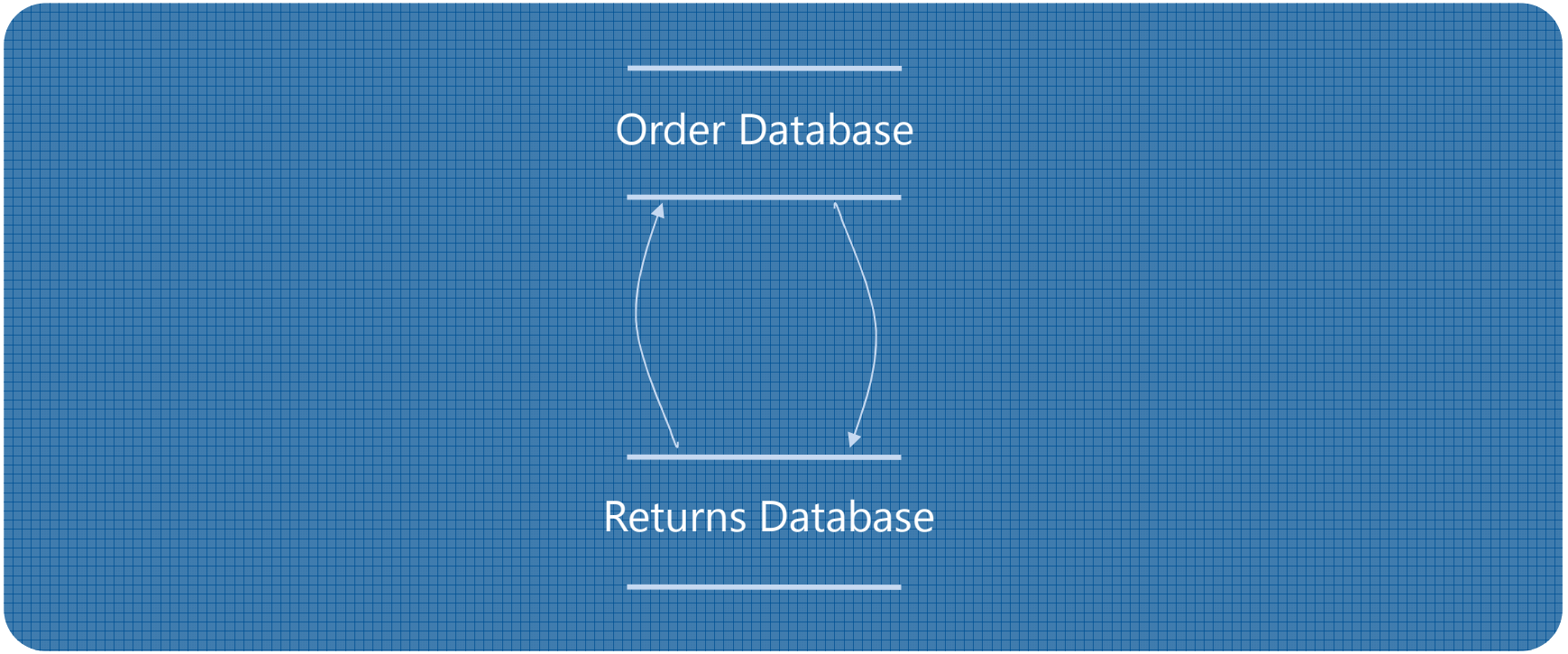
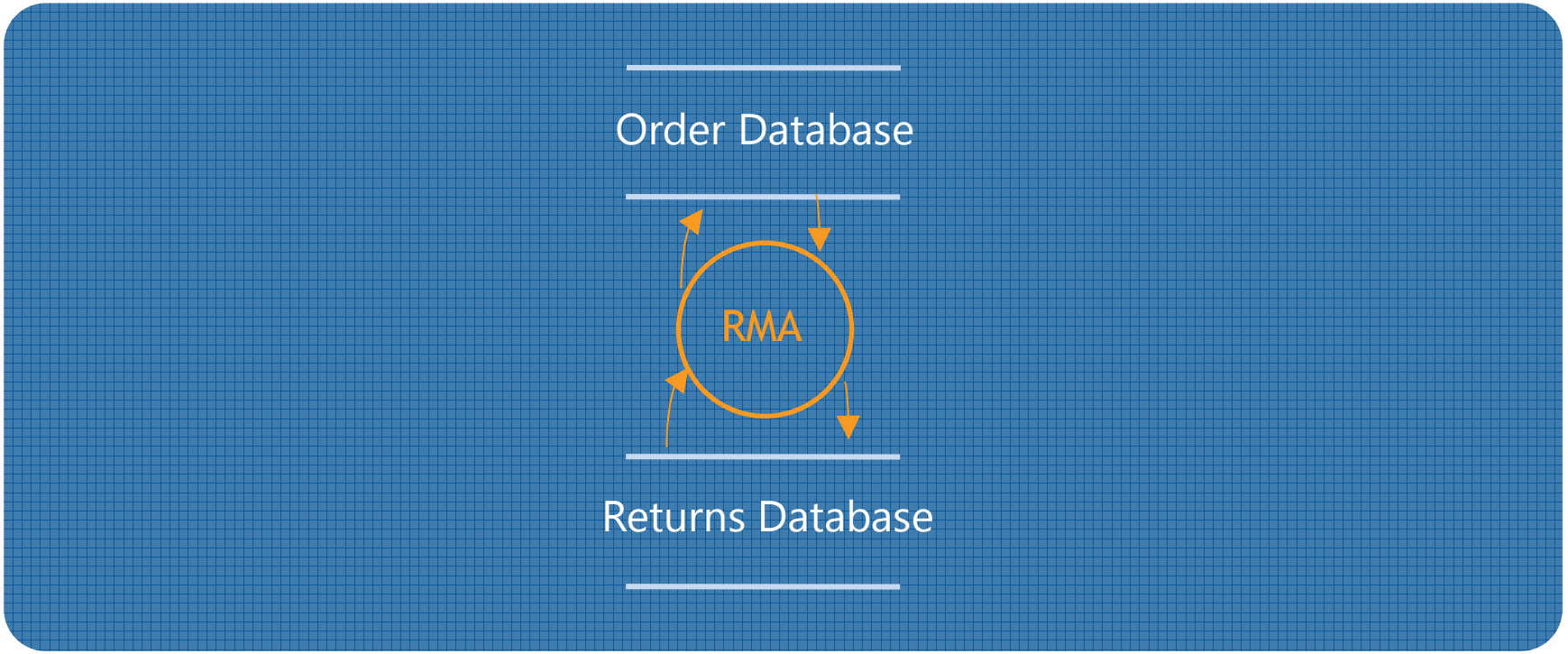


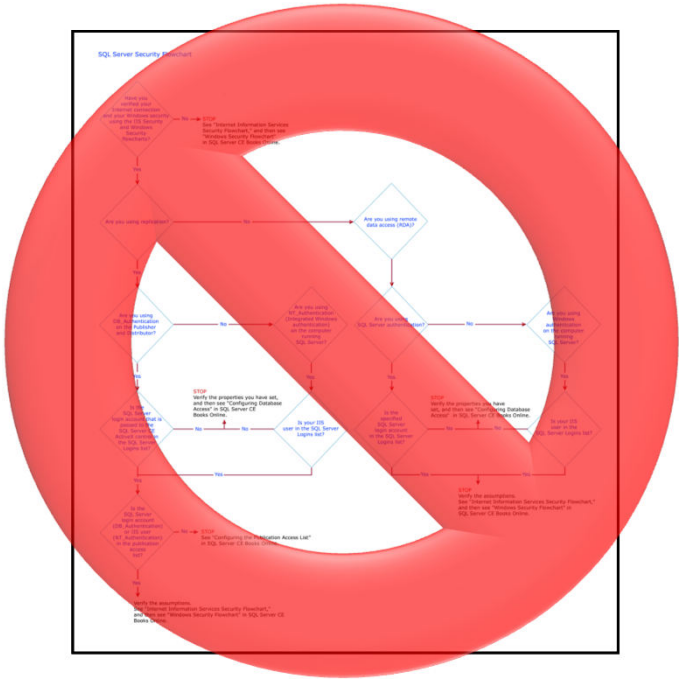
Diagram Validation Rules of Thumb

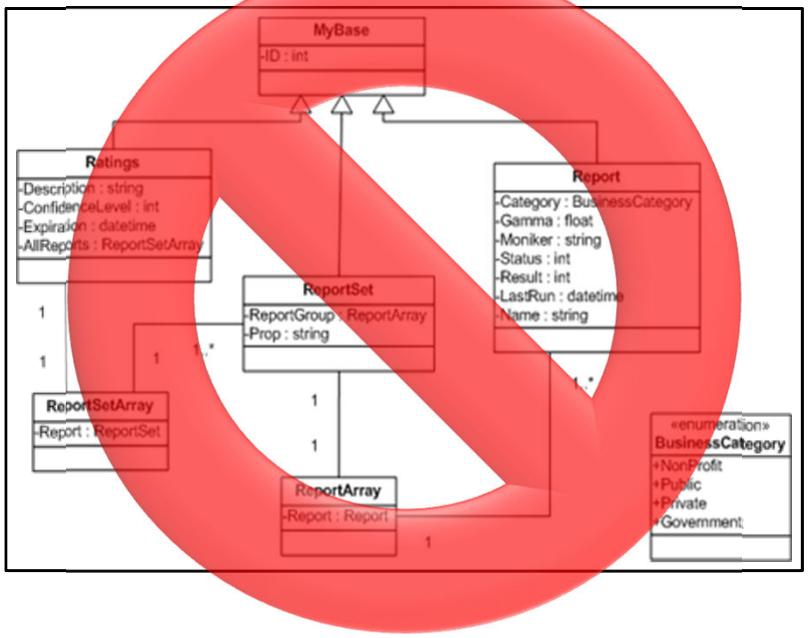
It goes through a process



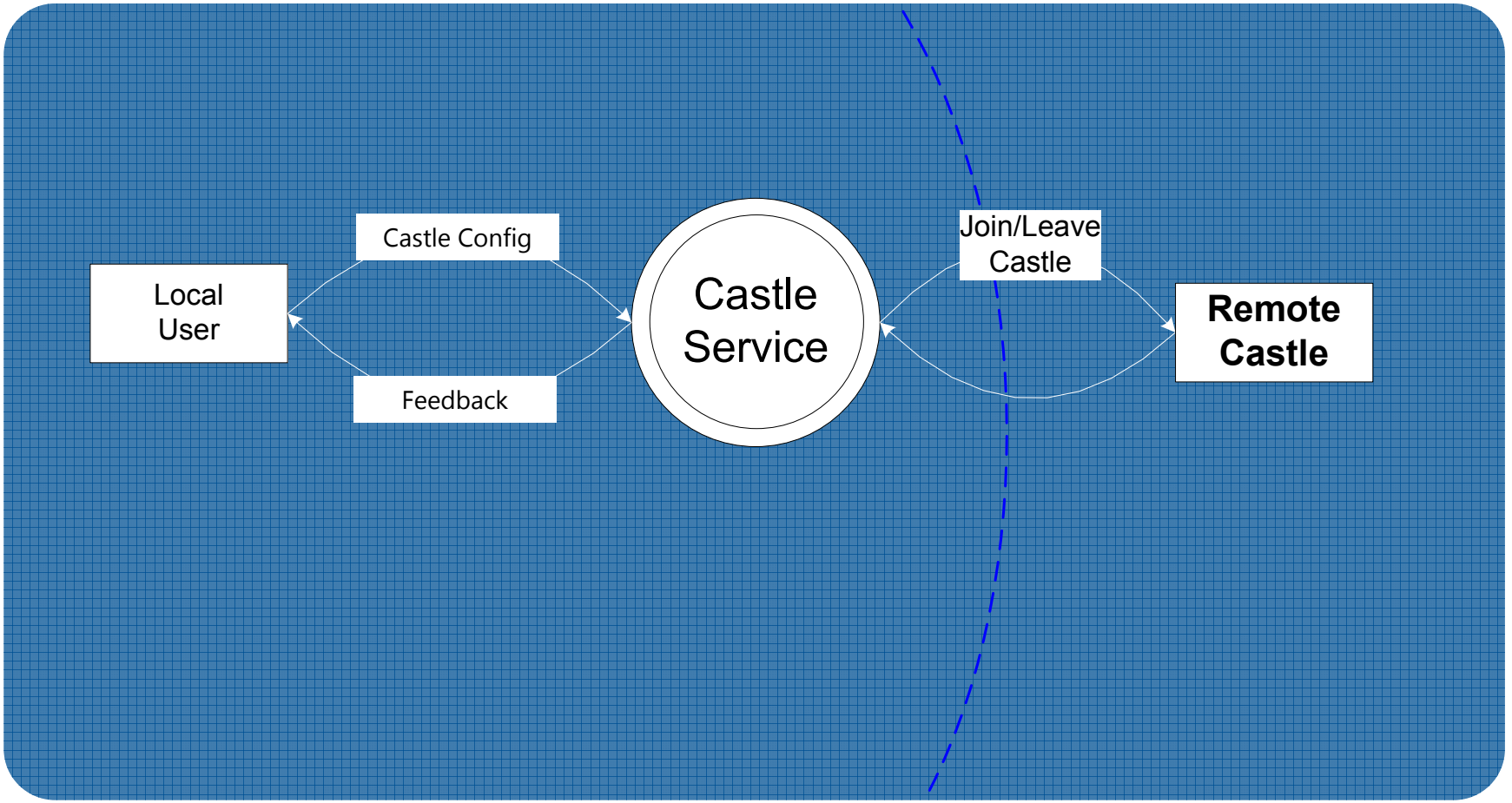
Diagrams Should Not Resemble

- Flow charts
- Class diagrams
- Call graphs

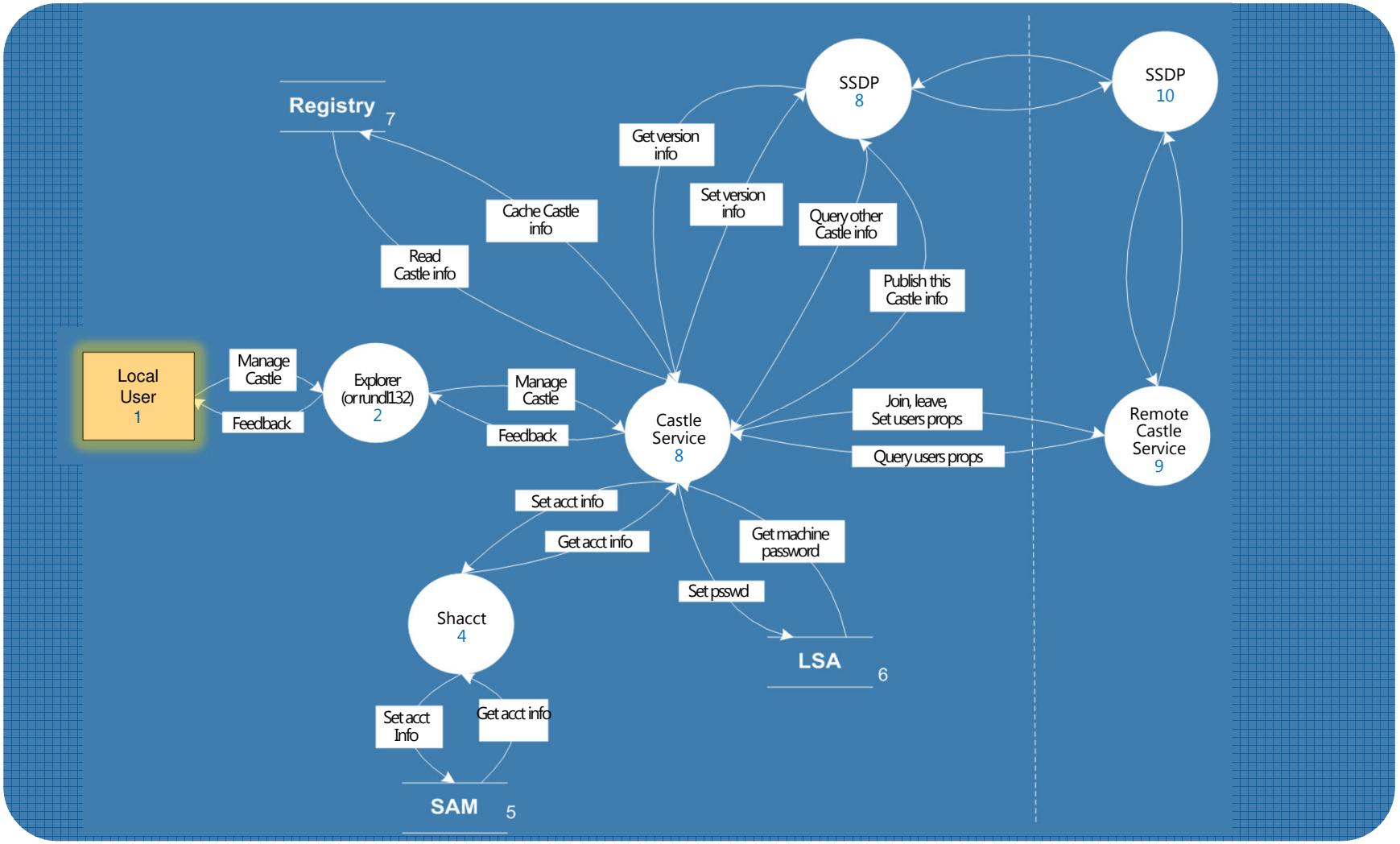




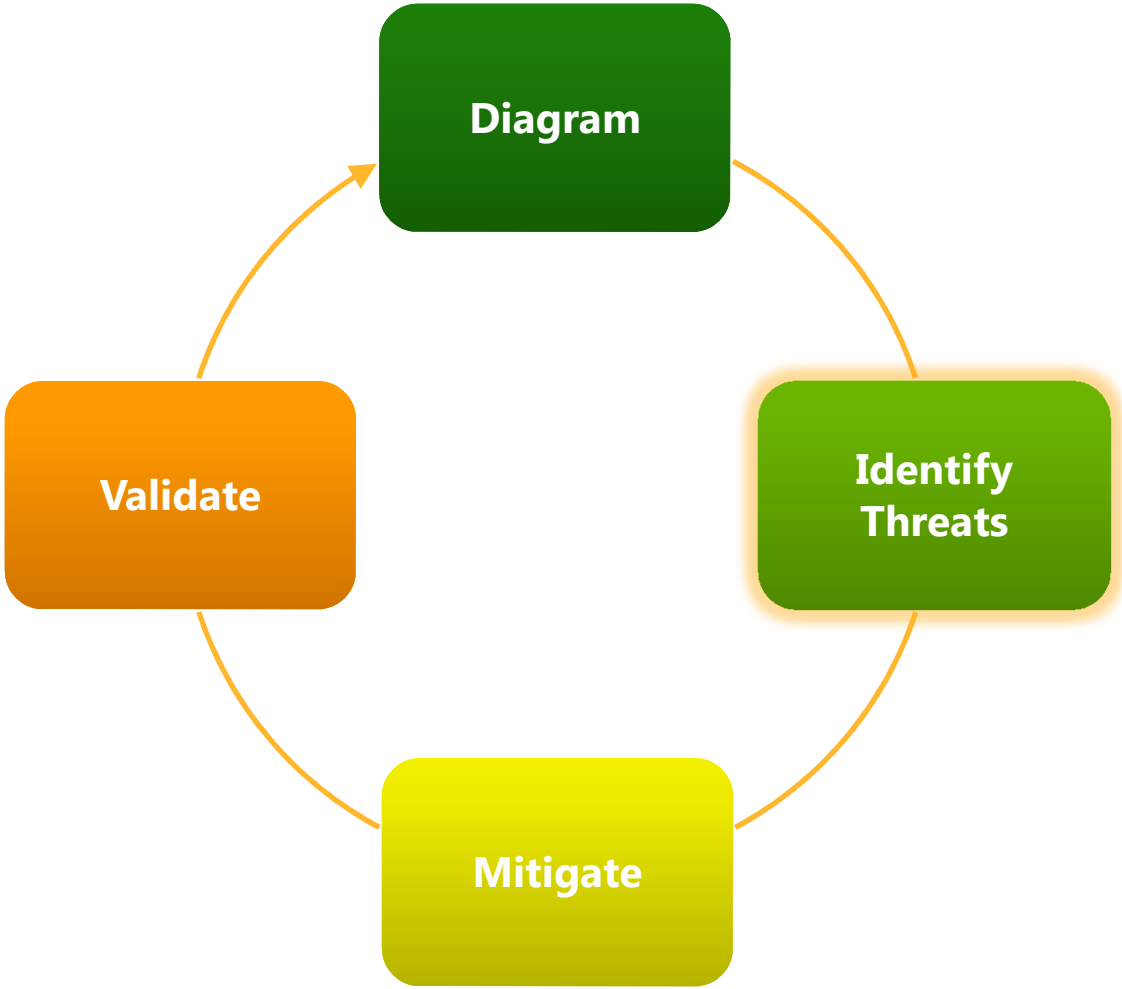
Real Context Diagram ("Castle")



Castle Level 1 Diagram



The Process: Identifying Threats



Identify Threats

- Experts can brainstorm
- How to do this without being an expert?
  - Use STRIDE to step through the diagram elements
  - Get specific about threat manifestation

Threat	Property we want
Spoofing	Authentication
Tampering	Integrity
Repudiation	Nonrepudiation
Information Disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorization

Threat: Spoofing

Threat	**S**poofing
Property	Authentication
Definition	Impersonating something or someone else
Example	Pretending to be any of billg, microsoft.com, or ntdll.dll

Threat: Tampering

Threat	Tampering
Property	Integrity
Definition	Modifying data or code
Example	Modifying a DLL on disk or DVD, or a packet as it traverses the LAN

Threat: Repudiation

Threat	Repudiation
Property	Non-Repudiation
Definition	Claiming to have not performed an action
Example	"I didn't send that email," "I didn't modify that file," "I certainly didn't visit that Web site, dear!"

Threat: Information Disclosure

Threat	Information Disclosure
Property	Confidentiality
Definition	Exposing information to someone not authorized to see it
Example	Allowing someone to read the Windows source code; publishing a list of customers to a Web site

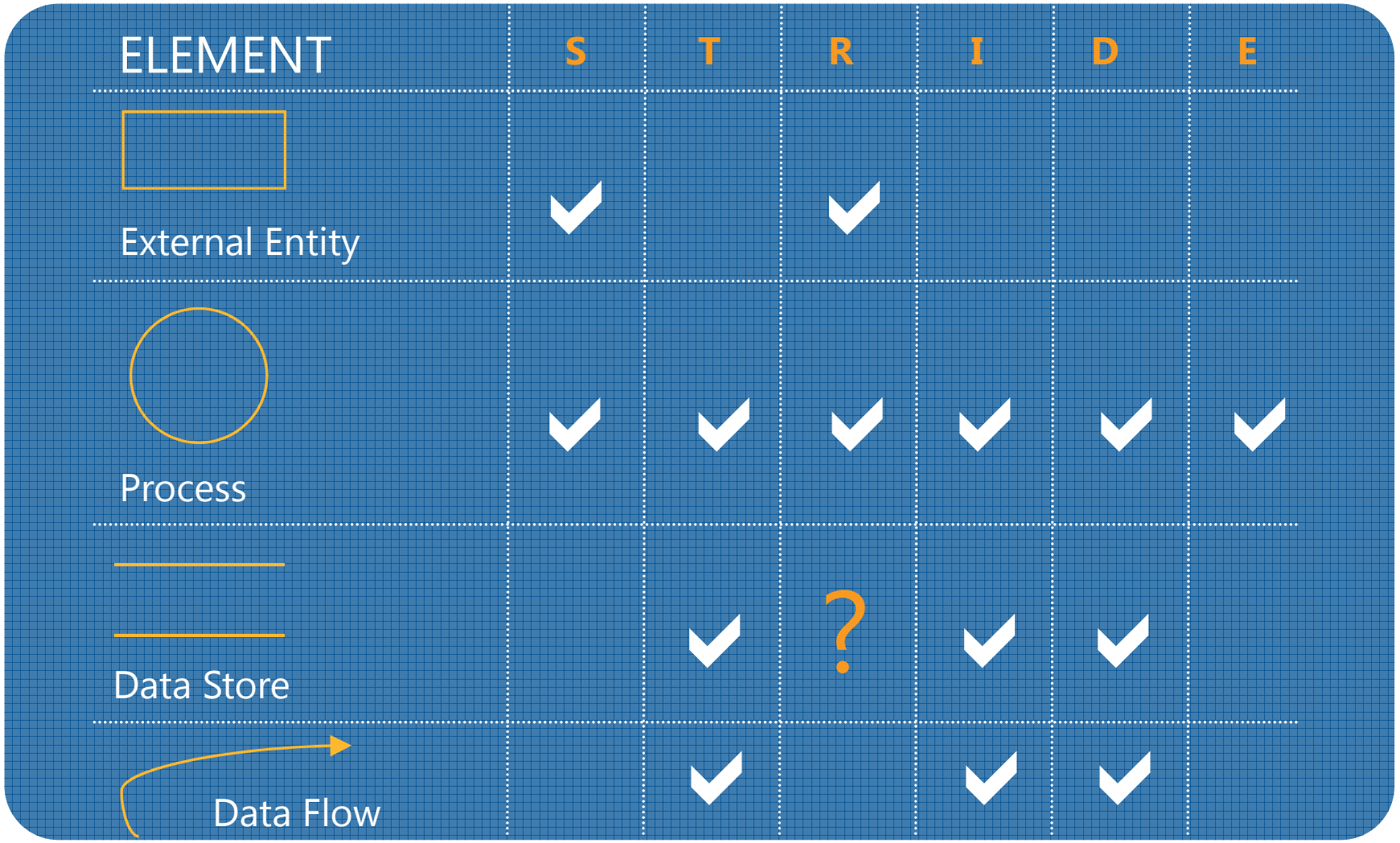
Threat: Denial of Service

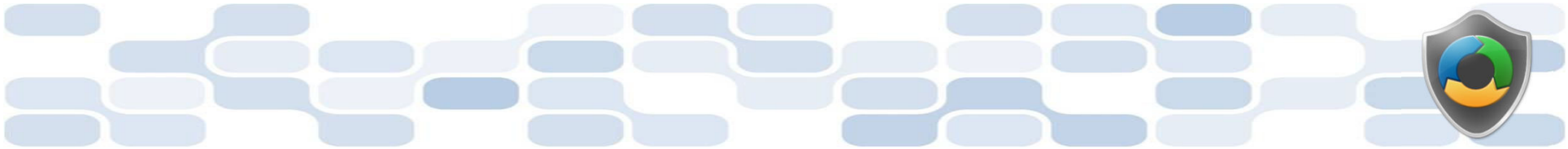
Threat	**D**enial of Service
Property	Availability
Definition	Deny or degrade service to users
Example	Crashing Windows or a Web site, sending a packet and absorbing seconds of CPU time, or routing packets into a black hole

Threat: Elevation of Privilege

Threat	Elevation of Privilege (EoP)
Property	Authorization
Definition	Gain capabilities without proper authorization
Example	Allowing a remote Internet user to run commands is the classic example, but going from a “Limited User” to “Admin” is also EoP

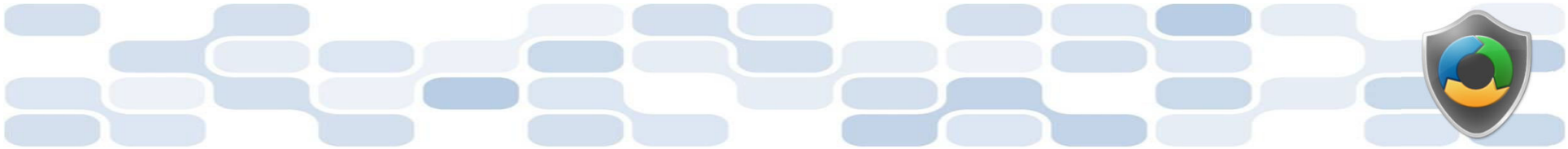
Different Threats Affect Each Element Type





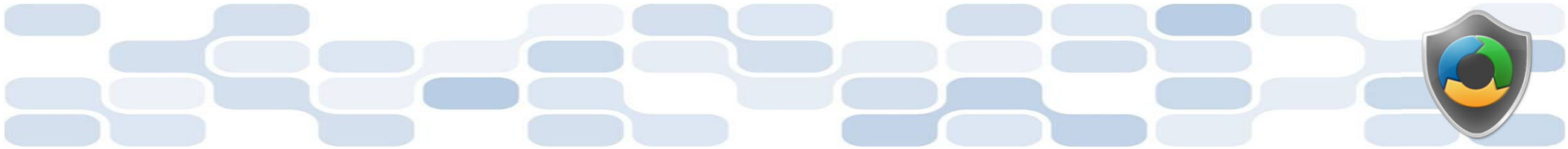
Apply STRIDE Threats to Each Element

- For each item on the diagram:
  - Apply relevant parts of STRIDE
  - Process: STRIDE
  - Data store, data flow: TID
    - Data stores that are logs: TID+R
  - External entity: SR
  - Data flow inside a process:
    - Don’t worry about T, I, or D
- This is why you number things



Use the Trust boundaries

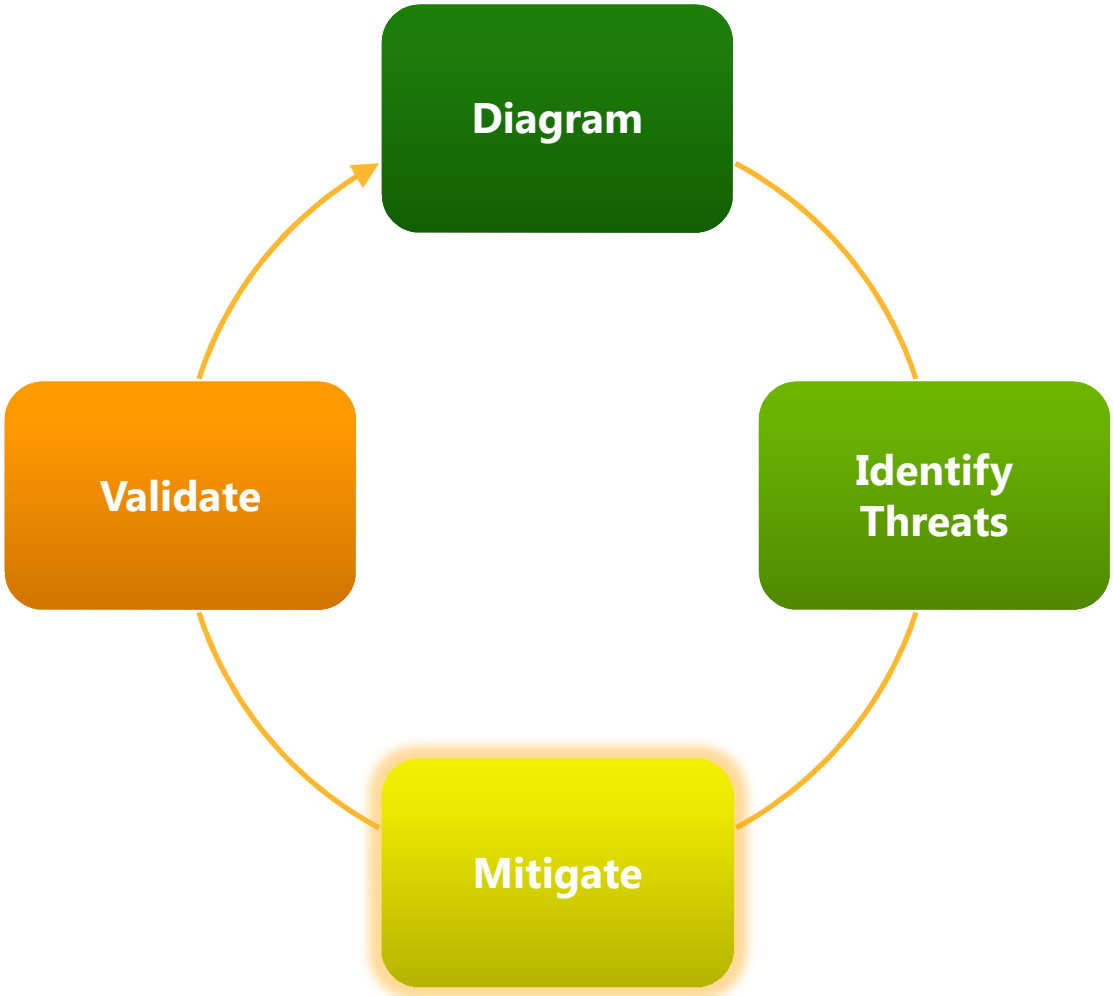
- Trusted/ high code reading from untrusted/low
  - Validate everything for specific and defined uses
- High code writing to low
  - Make sure your errors don’t give away too much

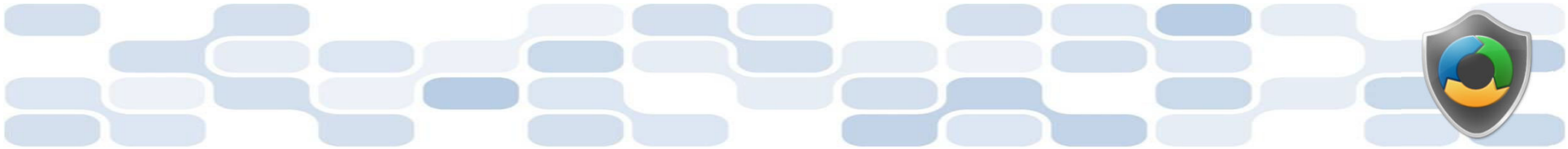


Threats and Distractions

- Don’t worry about these threats
  - The computer is infected with malware
  - Someone removed the hard drive and tampers
  - Admin is attacking user
  - A user is attacking himself
- You can’t address any of these (unless you’re the OS)

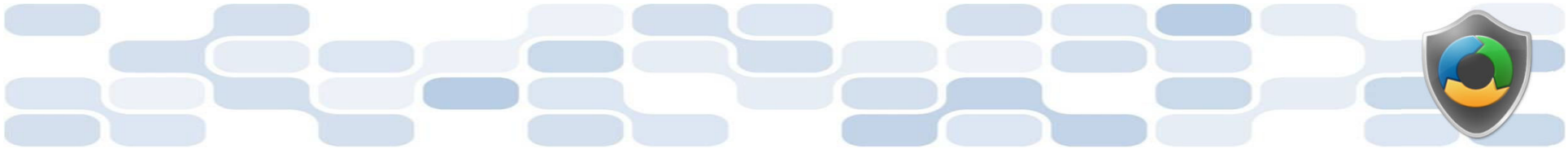
The Process: Mitigation





Mitigation Is the Point of Threat Modeling

- Mitigation
  - To address or alleviate a problem
- Protect customers
- Design secure software
- Why bother if you:
  - Create a great model
  - Identify lots of threats
  - Stop
- So, find problems and fix them

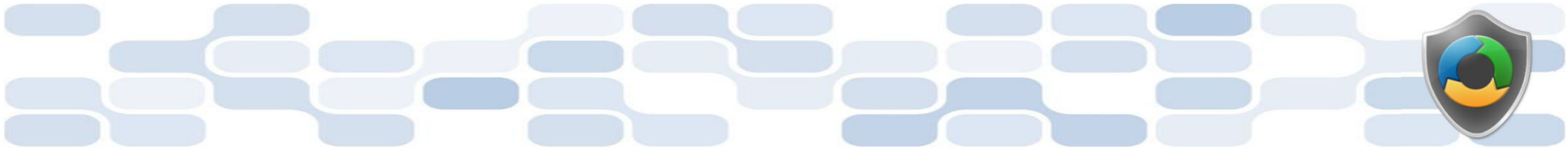


Mitigate

- Address each threat
- Four ways to address threats
  1. Redesign to eliminate
  2. Apply standard mitigations
     - What have similar software packages done and how has that worked out for them?
  3. Invent new mitigations (riskier)
  4. Accept vulnerability in design
     - SDL rules about what you can accept
- Address each threat

Standard Mitigations

Spoofing	Authentication	To authenticate principals: • Cookie authentication • Kerberos authentication • PKI systems such as SSL/TLS and certificates To authenticate code or data: • Digital signatures
Tampering	Integrity	• Windows Vista Mandatory Integrity Controls • ACLs • Digital signatures
Repudiation	Non Repudiation	• Secure logging and auditing • Digital Signatures
Information Disclosure	Confidentiality	• Encryption • ACLS
Denial of Service	Availability	• ACLs • Filtering • Quotas
Elevation of Privilege	Authorization	• ACLs • Group or role membership • Privilege ownership • Input validation



Inventing Mitigations Is Hard: Don't do it

- Mitigations are an area of expertise, such as networking, databases, or cryptography
- Amateurs make mistakes, but so do pros
- Mitigation failures will appear to work
  - Until an expert looks at them
  - We hope that expert will work for us
- When you need to invent mitigations, get expert help

Sample Mitigation

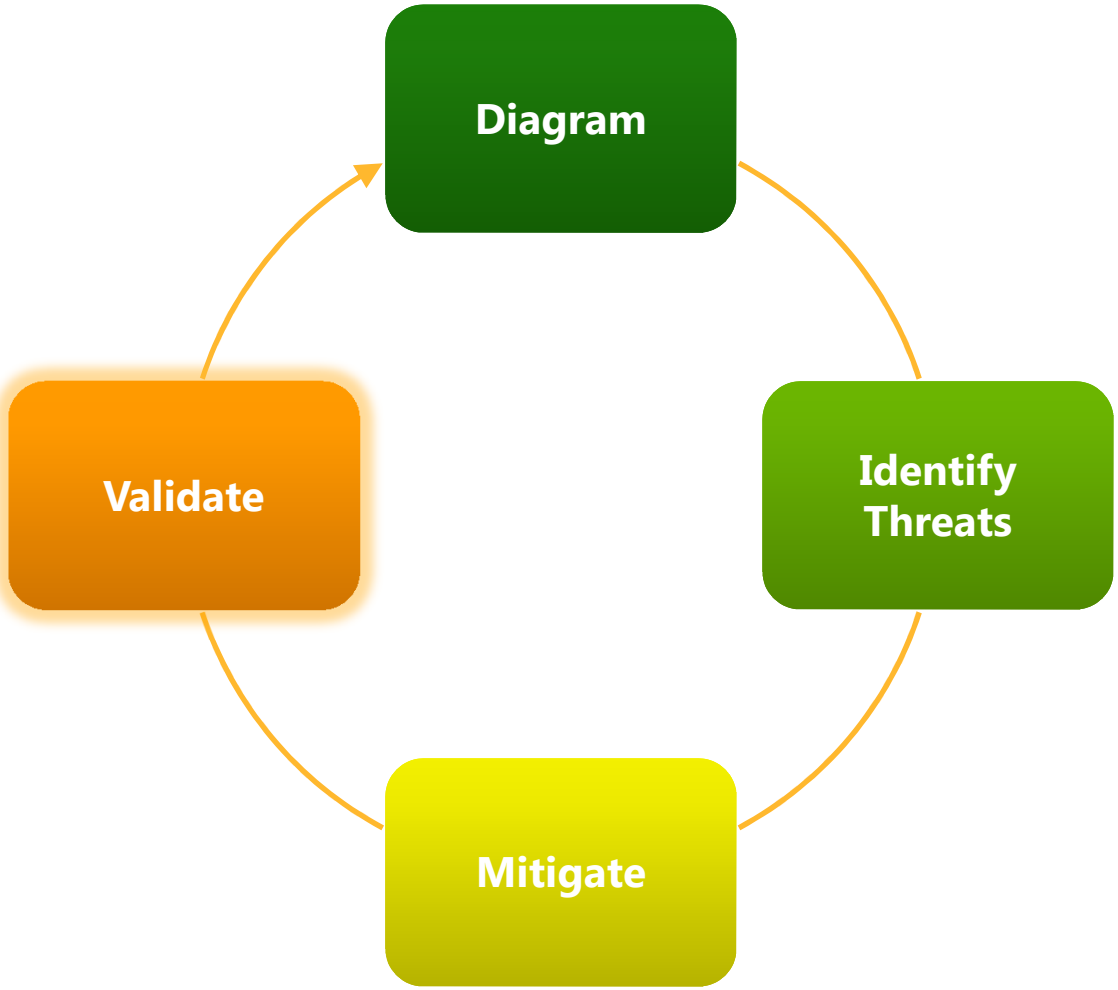
- Mitigation #54, Rasterization Service performs the following mitigation strategies:
  1. OM is validated and checked by (component) before being handed over to Rasterization Service
  2. The resources are decoded and validated by interacting subsystems, such as [foo], [bar], and [boop]
  3. Rasterization ensures that if there are any resource problems while loading and converting OM to raster data, it returns a proper error code
  4. Rasterization Service will be thoroughly fuzz tested

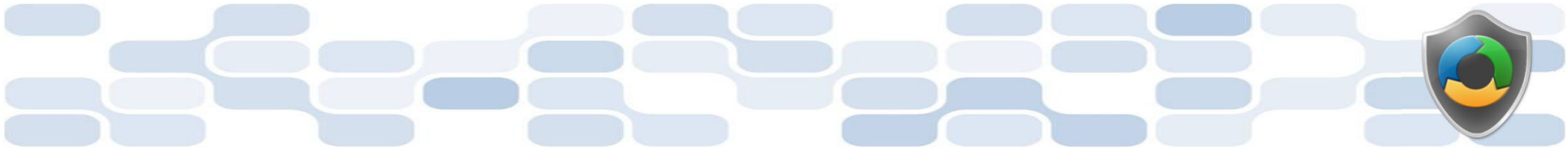
(Comment: Fuzzing isn't a mitigation, but it's a great thing to plan as part 4)

Improving Sample Mitigation: Validated-For

- "OM is validated and checked by [component] before being handed over to Rasterization Service"
- Validated for what? Be specific!
  - "…validates that each element is unique."
  - "…validates that the URL is RFC-1738 compliant, but note URL may be to http://evil.com/ownme.html"
  - (Also a great external security note)

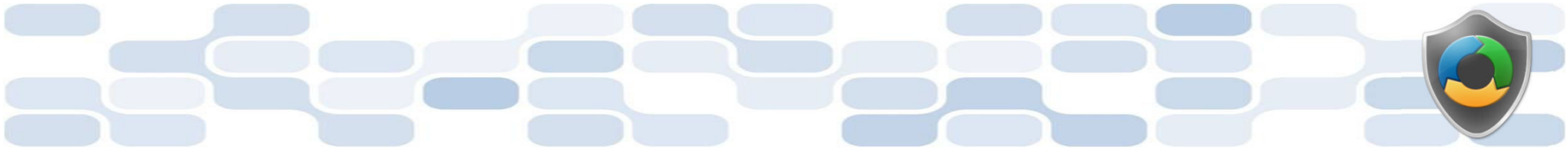
The Process: Validation





Validating Threat Models

- Validate the whole threat model
  - Does diagram match final code?
  - Are threats enumerated?
  - Minimum: STRIDE per element that touches a trust boundary
  - Has Test / QA reviewed the model?
    - Tester approach often finds issues with threat model or details
  - Is each threat mitigated?
  - Are mitigations done right?
- Did you check these before Final Security Review?
  - Shipping will be more predictable

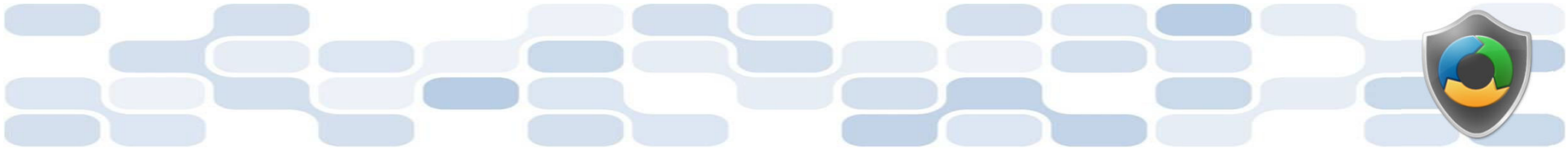


Validate Quality of Threats and Mitigations

- Threats: Do they:
  - Describe the attack
  - Describe the context
  - Describe the impact
- Mitigations
  - Associate with a threat
  - Describe the mitigations
  - File a bug
    - ✖ Fuzzing is a test tactic, not a mitigation

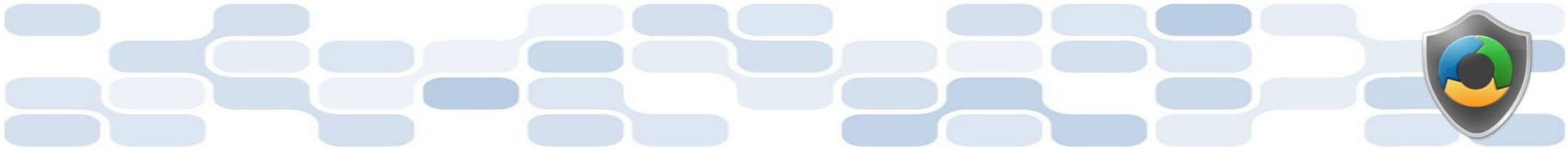
Validate Information Captured

- Dependencies
  - What other code are you using?
  - What security functions are in that other code?
  - Are you sure?
- Assumptions
  - Things you note as you build the threat model
    - ✖ "HTTP.sys will protect us against SQL Injection"
    - ✖ "LPC will protect us from malformed messages"
    - ✔ GenRandom will give us crypto-strong randomness



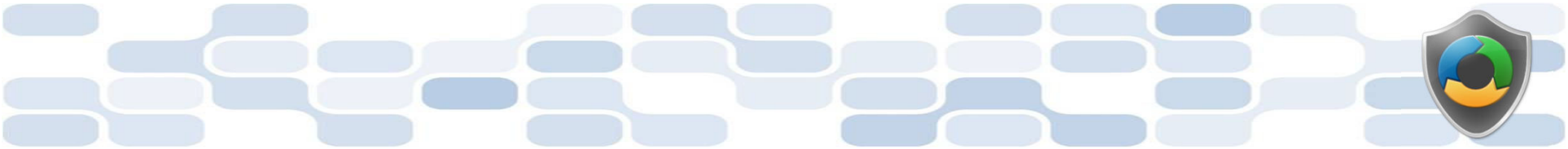
More Sample Mitigations

- Mitigation #3: The Publish License is created by RMS, and we've been advised that it's only OK to include an unencrypted e-mail address if it's required for the service to work. Even if it is required, it seems like a bad idea due to easy e-mail harvesting.
- Primary Mitigation: Bug #123456 has been filed against the RMS team to investigate removing the e-mail address from this element. If that's possible, this would be the best solution to our threat.
- Backup Mitigation: It's acceptable to mitigate this by warning the document author that their e-mail address may be included in the document. If we have to ship it, the user interface will be updated to give clear disclosure to the author when they are protecting a document.

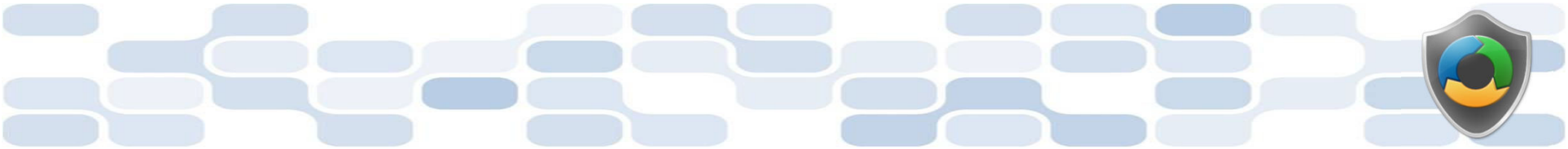


Effective Threat Modeling Meetings

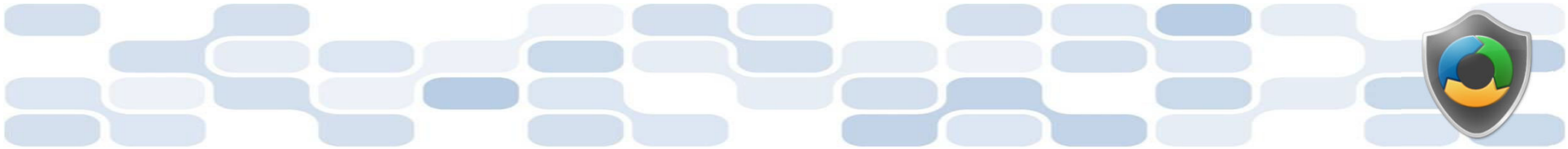
- Develop draft threat model before the meeting
  - Use the meeting to discuss
- Start with a DFD walkthrough
- Identify most interesting elements
  - Assets (if you identify any)
  - Entry points/trust boundaries
- Walk through STRIDE against those elements
- Threats that cross elements/recur
  - Consider library, redesigns



Pause for Questions
Before Exercise



Exercise

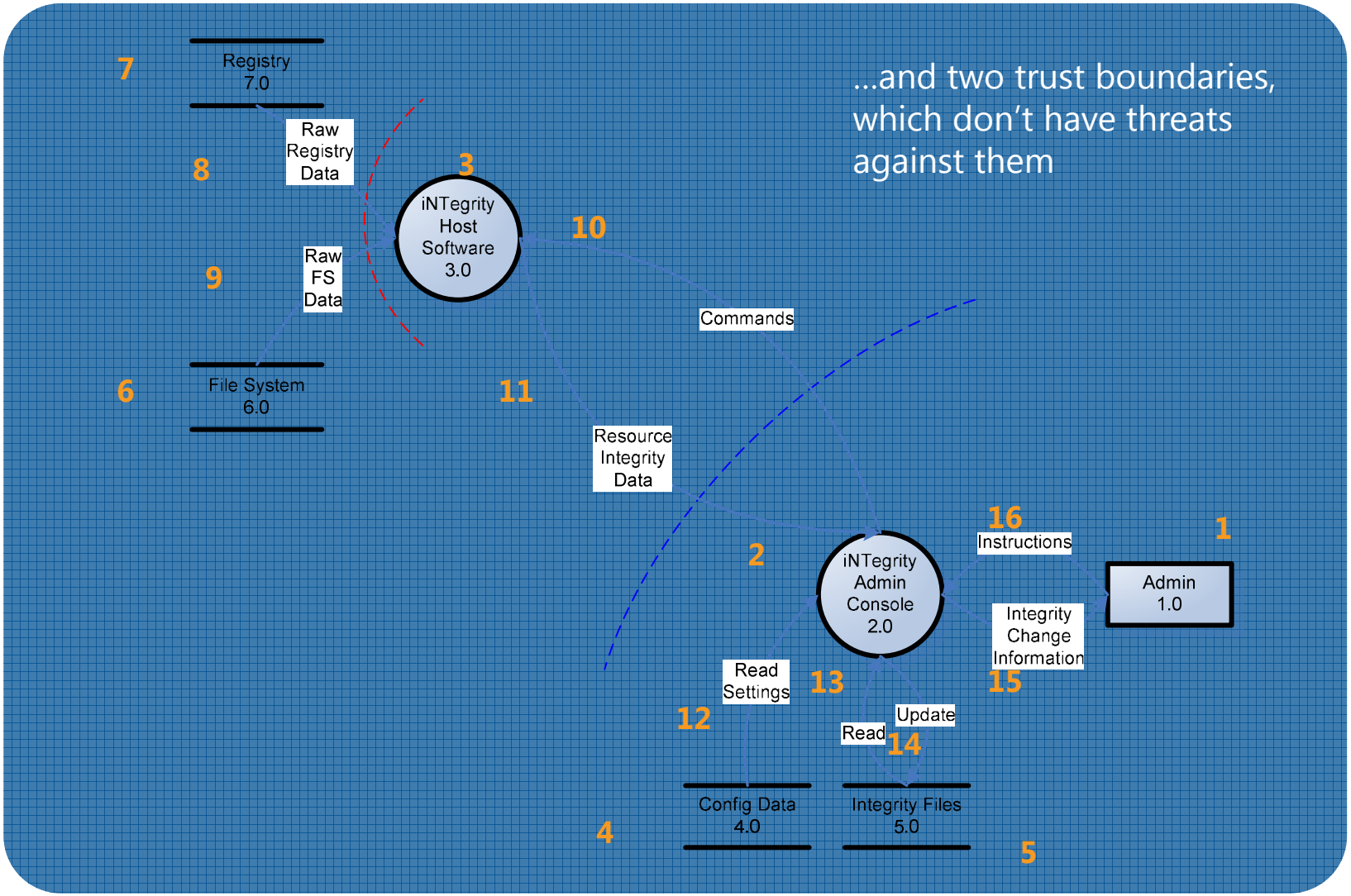


Exercise

- Handout
- Work in teams to:
  - Identify all diagram elements
  - Identify threat types to each element
  - Identify at least three threats
  - Identify first order mitigations

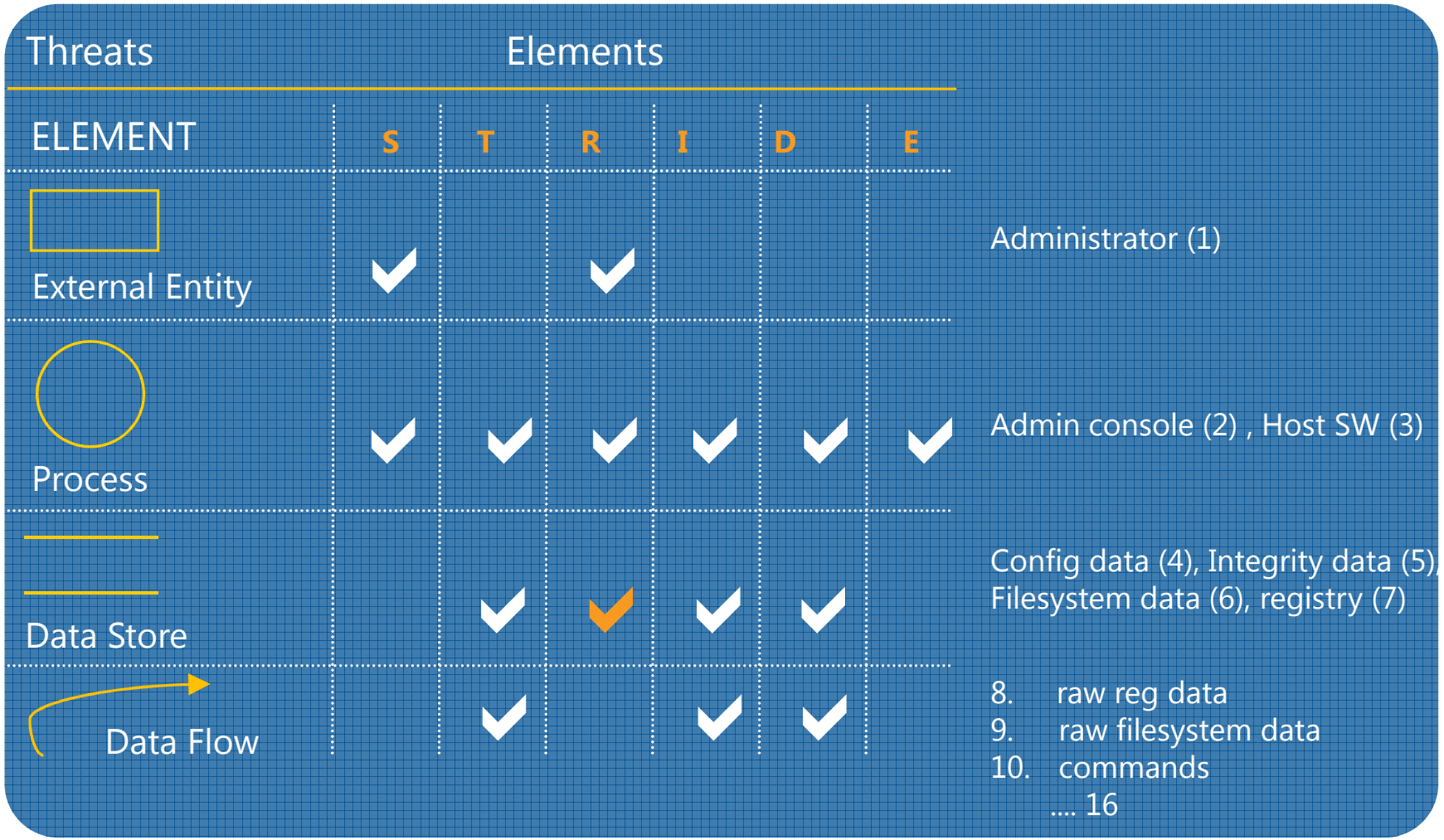
Extra credit: Improve the diagram

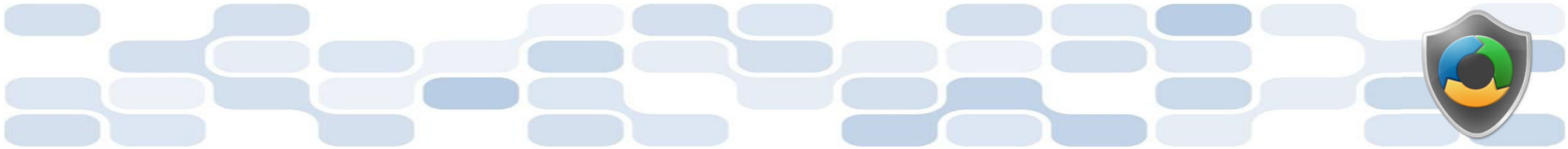
Identify All Elements (16 Elements)



Identify Threat Types to Each Element

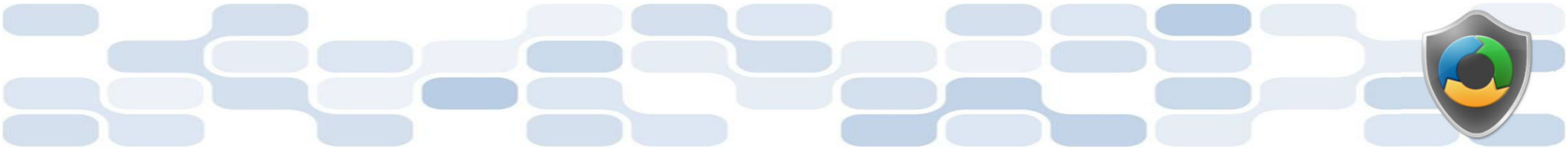
Identify STRIDE threats by element type



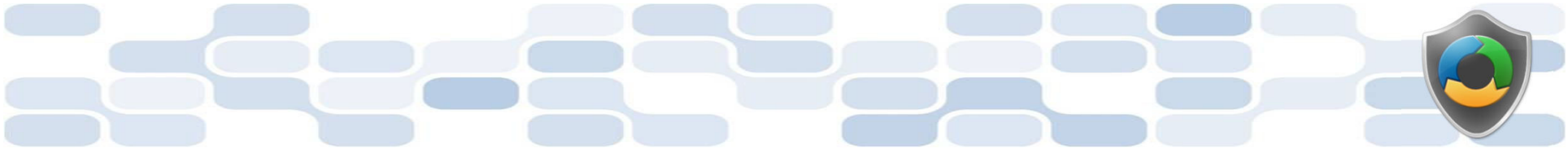


Identify Threats!

- Be specific
- Understand threat and impact
- Identify first order mitigations



Demo



Call to Action

- Threat model your work!
  - Start early
  - Track changes
- Work with a Security Advisor!
- Talk to your “dependencies” about security assumptions
- Learn more

Threat Modeling Learning Resources

MSDN Magazine

Reinvigorate your Threat Modeling Process
http://msdn.microsoft.com/en-us/magazine/cc700352.aspx

Threat Modeling: Uncover Security Design Flaws Using The STRIDE Approach
http://msdn.microsoft.com/msdnmag/issues/06/11/ThreatModeling/default.aspx

Article

Experiences Threat Modeling at Microsoft
http://download.microsoft.com/download/9/D/3/9D389274-F770-4737-9F1A-8EA2720EE92A/Shostack-ModSec08-Experiences-Threat-Modeling-At-Microsoft.pdf

SDL Blog

All threat modeling posts
http://blogs.msdn.com/sdl/archive/tags/threat%20modeling/default.aspx

Books

The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software
(Howard, Lipner, 2006) “Threat Modeling” chapter
http://www.microsoft.com/mspress/books/authors/auth8753.aspx

Resources

SDL Portal

http://www.microsoft.com/sdl

SDL Blog

http://blogs.msdn.com/sdl/

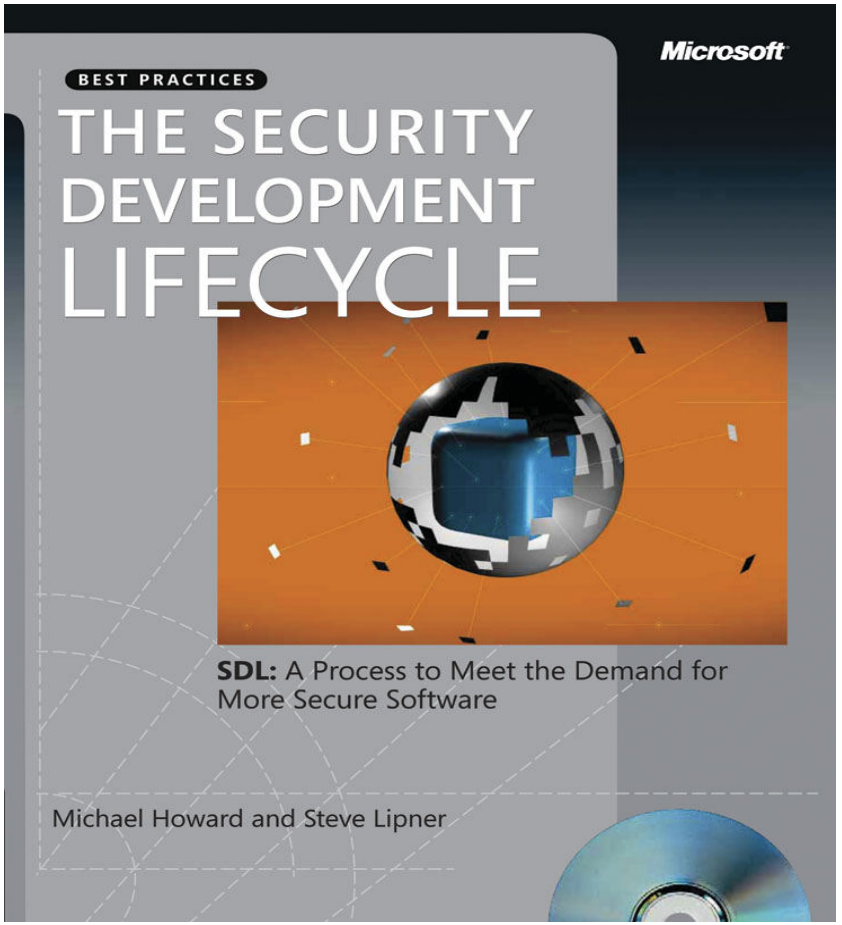
SDL Process on MSDN (Web)

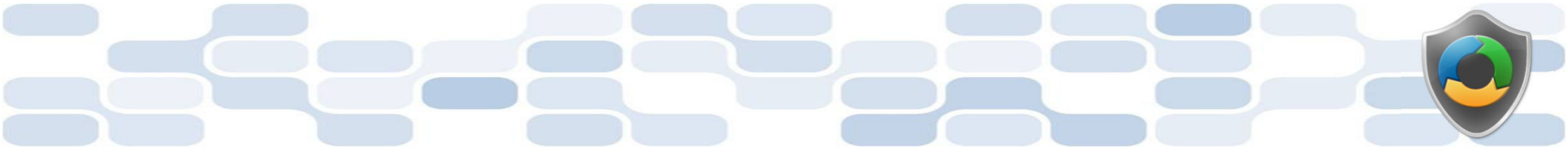
http://msdn.microsoft.com/en-us/library/cc307748.aspx

SDL Process on MSDN (MS Word)

http://go.microsoft.com/?linkid=9694872

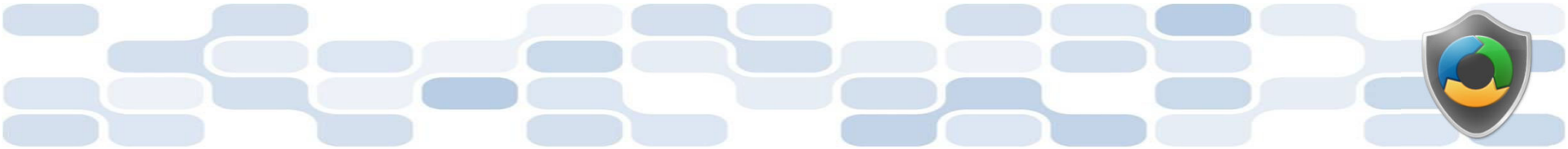
Questions?


Microsoft
BEST PRACTICES
THE SECURITY DEVELOPMENT LIFECYCLE
SDL: A Process to Meet the Demand for More Secure Software
Michael Howard and Steve Lipner



Microsoft®

Your potential. Our passion.™



Backup Slides

Standard Mitigations

S T R I D E

Threat	Property we want
Spoofing	Authentication
Tampering	Integrity
Repudiation	Nonrepudiation
Information Disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorization

Standard Mitigations

STRIDE

Threat	Property	
Spoofing	Authentication	To authenticate principals: • Basic authentication • Digest authentication • Cookie authentication • Windows authentication (NTLM) • Kerberos authentication • PKI systems, such as SSL or TLS and certificates • IPSec • Digitally signed packets To authenticate code or data: • Digital signatures • Message authentication codes • Hashes

Standard Mitigations

S T R I D E

Threat	Property	
Tampering	Integrity	• Windows Vista mandatory integrity controls • ACLs • Digital signatures • Message authentication codes

Standard Mitigations

S T R I D E

Threat	Property	
Repudiation	Nonrepudiation	• Strong authentication • Secure logging and auditing • Digital signatures • Secure time stamps • Trusted third parties

Standard Mitigations

S T R I D E

Threat	Property	
Information Disclosure	Confidentiality	• Encryption • ACLs

Standard Mitigations

S T R I D E

Threat	Property	
Denial of Service	Availability	• ACLs • Filtering • Quotas • Authorization • High-availability designs

Standard Mitigations

S T R I D E

Threat	Property	
Elevation of Privilege	Authorization	• ACLs • Group or role membership • Privilege ownership • Permissions • Input validation