Ray Overby
Key Resources, Inc.

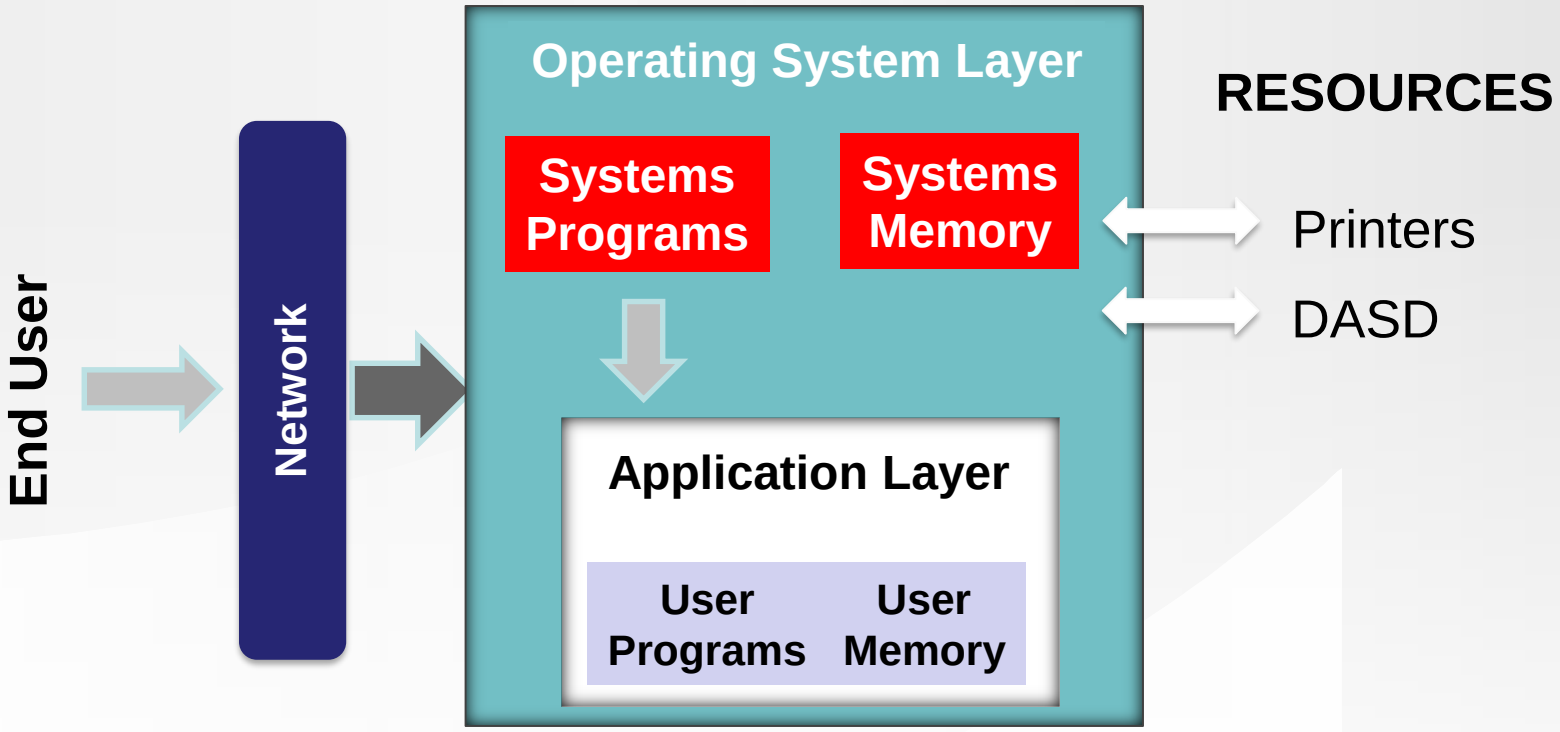
Closing the Integrity Gap in Your Mainframe

Ray.Overby@KRIsecurity.com
www.KRIsecurity.com

AGENDA

- Introduction: Is the Mainframe Secure?
  - Enterprise Quadrant Architecture
  - IBM's z/OS System Integrity Statement
- Mainframe Security Methods
  - Configuration-based Security
  - Code-based Security
- The Mainframe Vulnerability Lifecycle
- Mainframe Configuration Vulnerabilities
- Mainframe Software Vulnerabilities
- Summary
- Questions

Enterprise Quadrant Architecture



Is the Mainframe Secure?

- The Mainframe is ***Only*** as Secure as the ***TIME*** **and** ***EFFORT*** You Spend on Maintaining the Integrity of the Environment
- IBM's z/OS Authorized Assembler Services Guide states that you are responsible for making sure that anything you install on each z/OS system you maintain meets the criteria of the Integrity Statement.
- The Integrity of the Environment is Based on the ***INABILITY*** of any program, not authorized by a mechanism under your control to:
  - Disable or circumvent the store or fetch protection encoded in z/OS,
  - Access an operating system resource that is password protected or protected by a mainframe security product, also called an External Security Manager (ESM),
  - Obtain control in an authorized state such as APF-authorized, supervisor state, or with a protection key less than eight exploitable to illicitly and invisibly browse, corrupt, or steal data associated with an application

Is the Mainframe Secure?

- All installation-supplied authorized code (for example, an installation SVC) must perform the same or an equivalent type of validity checking and control that the system uses to maintain its integrity.
- Modifications and additions (3rd Party Software) to the system do not introduce any integrity exposures.
- The IBM z/OS Statement of Integrity **only** applies to IBM software. It does not apply to any ISV code or installation written code. You, the z/OS system owner, are responsible for continually verifying the integrity of any configuration based modifications and code you add to your mainframe.

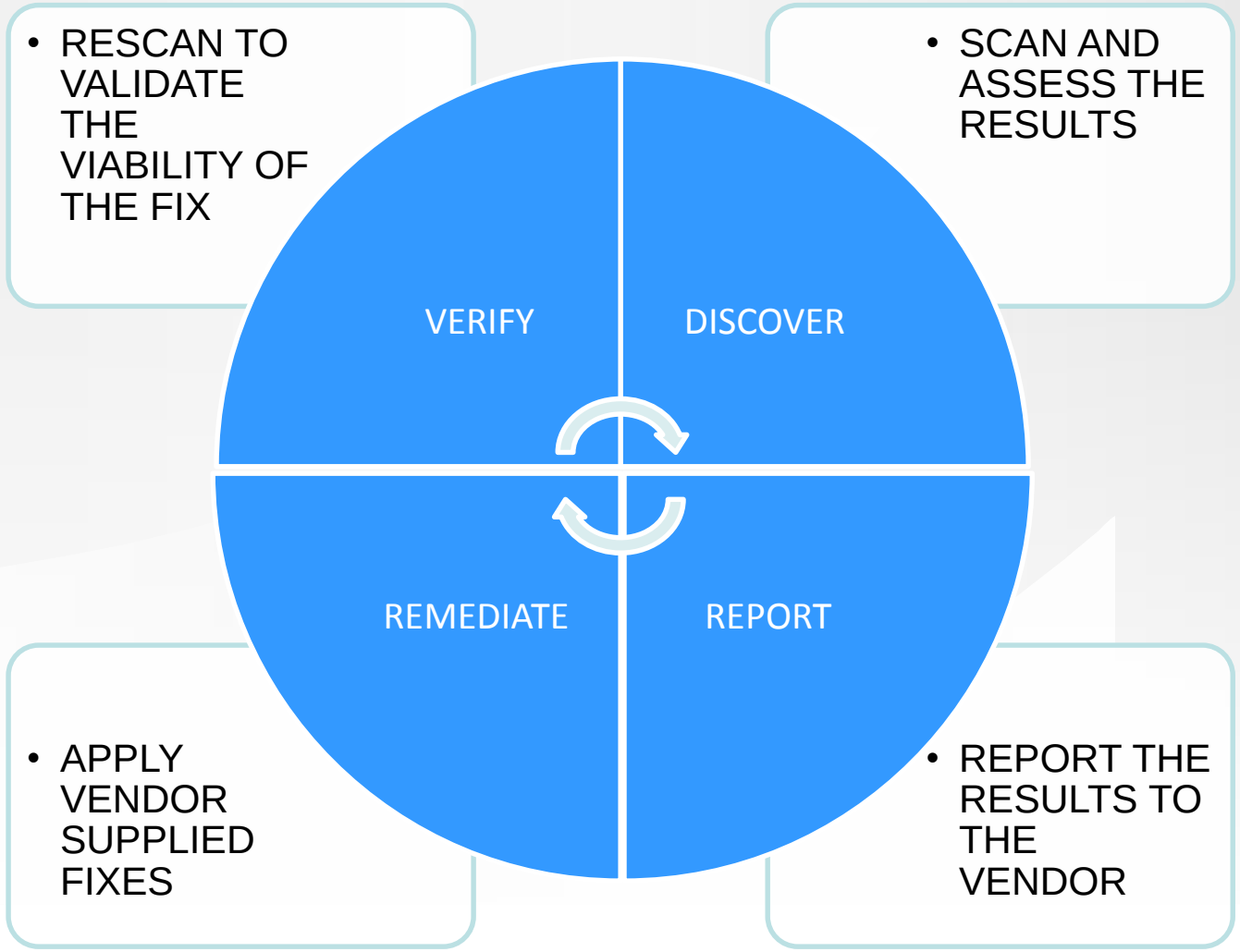
What are the Security Methods Under My Control?

- **Configuration-Based Security**
  - Configuration-based Security is built around a Security Policy, the Procedures enacted to maintain the Security Policy, and a correctly configured ESM and Operating System.
- **An Installation has Responsibility for Securing the following:**
  - IPL (boot) Parameters,
  - Subsystem Startup Parameters,
  - Adoption of security procedures (for example, the password protection of appropriate system data sets) that are a necessary complement to the integrity support within the operating system itself,
  - Setup and management of External Security Managers like IBM RACF®, CA ACF2™ or CA Top Secret®.

What are the Security Methods Under My Control?

- **Code-Based Security**
  - IBM's Statement of Integrity is the basis for Operating System Integrity Testing™ (OSIT)
  - Code-based Security is built around a strong Vulnerability Management program as outlined in your Security Policy.
  - Vulnerability Management across all platforms is now a component of PCI, SOX, etc.
  - Vulnerability Management on the Mainframe is no different than on any other Platform or Technology
    - A Mainframe Based Vulnerability Program should include the operating system layer and the application layer
    - A Mainframe Based Vulnerability Program should encompass all aspects of the Vulnerability Lifecycle

The Mainframe Vulnerability Lifecycle



Mainframe Configuration Vulnerabilities

- Configuration-based vulnerabilities are caused by improper configuration settings.
- Configuration-based vulnerabilities are remediated by changing configuration parameters.
- How to Protect Yourself
  - Configure all security mechanisms based on your security policy,
  - Turn off all unused services,
  - Continually monitor roles, permissions, and accounts,
  - Disabling all default accounts and/or change their passwords,
  - Set up and monitor logs and alerts.

Mainframe Software Vulnerabilities

- KRI classifies z/OS integrity vulnerabilities as Severe Security Code Vulnerabilities™ (SSCV).
- Severe Security Code Vulnerabilities™ (SSCV's) are caused by poor design or coding errors in products that reside in the operating system layer on your mainframe.
- SSCV's have the following characteristics IN ALL CASES:
  - *Exploitable*
  - *Violates the IBM Statement of Integrity*
  - *If exploited, can compromise all data on the system and the system itself*
  - *A single SSCV would allow someone with access to z/OS to bypass its controls without forensic evidence of a breach. They can:*
    - *Dynamically elevate External Security Manager (ESM) authorities*
    - *Dynamically elevate z/OS authorities*
  - *Alter operating systems processing:*
    - *To suppress forensic evidence*
    - *To collect information such as userids and passwords*

Mainframe Software Vulnerabilities

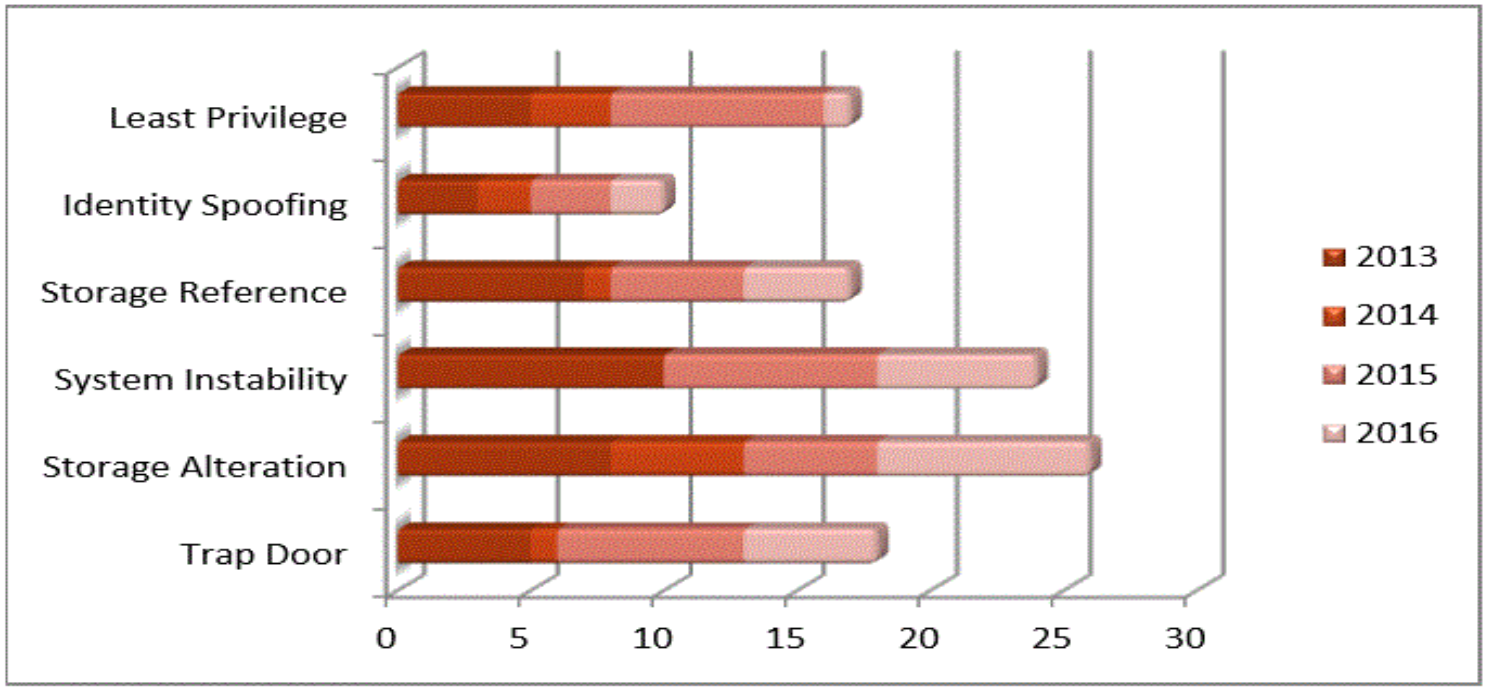
- Common to all of the below is that the vulnerability can be triggered or exploited by a non-authorized user with no special privileges.
- Exploitations of this code vulnerability leaves no forensic evidence or audit trails of what was accessed, copied, and/or modified.
- **Types of Vulnerabilities:**
  - **System Instability** - System instability occurs when the authorized program is invoked other than as designed and the program does not protect itself against improper invocation. This improper invocation can cause z/OS service issues including crashing the system.
  - **Denial of Service**; crashing or slowing the system down.
  - **Identity Spoofing** - Identify Spoofing vulnerabilities allow the non-authorized user to create alternate security credentials.

Mainframe Software Vulnerabilities

- **Types of Vulnerabilities:**
  - **Trap Door** -Trap Door vulnerabilities provide the non-authorized user the ability to call a user provided routine in an authorized state. (Either system key (0-7) or supervisor state.) This enables the user to directly make any changes to the active environment, including:
    - Direct invocation of any authorized z/OS interface
    - Changing user's state (authorized)
    - Making changes to the operating system
    - Making changes to system or application data
    - Impersonating other users
    - Disabling logging auditing (SMF)
    - Disrupt z/OS operation (crash or failure)
  - **Least Privilege** - Least Privilege vulnerabilities occur when a privilege or authority is assigned where a lessor privilege is appropriate.
    - Example: An SVC routine or system exit is improperly linked as AC(1).
    - As SVC routines and system exits are directly invoked by the system already authorized, they do not need to be link edited as AC(1); it has no effect on their normal use. However, the AC(1) attribute allows these routines to be improperly directly invoked via PGM= in JCL, where they were not designed to be invoked.

In Summary

What's Out There



In Summary

- External, independent code validation is essential for both software development “best practices” and controlling enterprise security risk associated with the deployment of third-party applications.
- z/OS Clients believe that their External Security Managers (RACF, CA ACF2, and CA Top Secret) are adequately protecting mission-critical platforms, third-party applications and information.
- SSCVs allow security (RACF, CA ACF2, and CA Top Secret) to be bypassed – data is compromised.
- In general Code-based Vulnerabilities are remediated by:
  - Reporting the problem to the code owner and the code owner provides a PTF that is applied by the user,
  - Removing the software from the system – this is very rare.

QUESTIONS?

TRADEMARKS

IBM, RACF, and z/OS are registered trademarks of IBM in the US

CA and ACF2 are trademarks of Computer Associates Technologies, Inc.

Other names may be trademarks of their respective owners.