Gaps in Mainframe Defenses

(often unreported in SMF)

Bottom Line - When your mainframe falls victim to a Ransomware attack or a module gains authorized status invisibly, the least of your worries is that there is no trace in SMF. That leaves your security console in the dark as well (if you bother uploading SMF data to your SIEM). There are literally dozens of attack vectors that can be used to gain access from phishing campaigns to more exotic Ransomware or Key 0 attacks. Integrity monitoring provides early warning of impending attacks by detecting and alerting on these and other red flags in your day-to-day operations. By implementing Early Warning with Integrity Monitoring (IM) you have new weapons to prevent even the most sophisticated attacks. At the same time your mainframe systems will have resiliency never before available in z/OS. Can you afford to wait?

Exposures and attack vectors – Following is a list of exposures, typically found in penetration testing and security audits of modern IBM mainframes. A list of the more common exposures, ranked by risk and consequence follows, along with potential solutions and mitigations. Thanks to industry experts such as Mark Wilson, Chad Rikansrud, Phil Young, Brian Marshall and others for their continued work to highlight multiple attack vectors exploited by bad actors.

1. Time Bombs, Multiple Back Doors and persistent malicious software: Most sites worry about restoring data but ignore the trusted systems, applications, and parameters that provide the hiding place for malware. Before the real attack, hackers install multiple malicious software to enable subsequent attacks. If you only recover your data, hackers can simply use another back door to re-encrypt the files and you are back to Square 1. You must restore both data AND all the compromised components of your infrastructure to its trusted state. Data must be recovered to a very recent state but not so for infrastructure. With attacks now taking on a degree of sophistication not previously seen, recovery of components from the last known trusted state can mean retrieving from weeks previous.

    - **Problem**: Ransomware is one type of malicious attack and a good example of what occurs in a breach. Bad actors commonly implement several back doors, often obscuring the original entry point to avoid detection. Another common practice is to install other malicious software and time bombs that are preset to activate if a key is not provided. Once imbedded in your infrastructure layer (system software, parameters and applications) the malware is still there. Threat actors can now simply use a different back door and re-encrypt their original target – your critical databases. Attacks often take days or weeks to implement, so your recovery process will need to restore data from the most recent uncorrupted version along with system and application components from earlier times. What do you do?

    - **Solution**: Integrity Monitoring (IM) was invented for exactly this purpose. Regular IM scans will detect unauthorized changes, which components have been compromised and when they were last correct. This will reveal all of the back doors, timebombs and unauthorized changes that were installed. Used in a preventative manner it can prevent attacks before they really occur but at a minimum gives the point in time to which each infrastructure component needs to be restored. To ensure resiliency, recovery of both the data and infrastructure is needed. To avoid seemingly unrelated events from slipping through, an overview report can should provide all changes made in a client specified period of time. Modern IM products can also help with remediation of these issues by generating the correct restore jobs for resolution.

    - Advanced IM tools can auto-discover and monitor sensitive system files and also be extended to monitor critical applications running on the mainframe.

    - To eliminate false positives, different versions of software can be automatically tracked as rolled across multiple LPARs at different points in time. To simplify administration, planned changes implemented by authorized staff will not generate an alert thus eliminating a major source of false positives.

2. Need for Early Warning: There are many warning signs that indicate an attack is imminent. Typically, both external and disgruntled internal attackers display uncharacteristic behavior to determine an appropriate attack point during the early phases of an attack (snooping). This activity typically involves accessing places not normal for that user, or in much larger volumes than normal. Specific monitoring activities can detect these usage patterns and raise an alert, ideally preventing the attack rather than recovering from it. Although different from integrity monitoring, early warning software can use the same extensive alerting and response mechanisms to ensure corrective action takes place long before the final attack is launched.

   - **Problem**: Detecting warning signs of suspicious behavior and making use of different information sources both on and off the mainframe have never been a part of the z/OS operating system. Cross referencing of modifications to change control tools (like ServiceNow or BMC Helix), that may be running in the Cloud, have not been possible. Loading of integrity monitoring information to an SIEM (like Splunk or QRadar) has not been available. Utilization of these different data sources is at best cumbersome and often impossible. In short cohesive early warning systems don't really exist on z/OS.

   - **Solutions**: Monitoring of who is accessing critical files (DB2, IMS, USS, RACF, APF libs and backups), and raising an alert when someone does so, is imperative. By creating a whitelist of processes trusted to access such files, means alerts are only raised when actions are taken by unknown agents. Further, integration to open a problem ticket, quarantine a suspect job or suspend users is needed.

   - Honey pots or decoy datasets can be created and monitored by an Early Warning System (EWS). These datasets only exist to attract the interest of users snooping around the system. When someone references the decoy in some manner, EWS raises an alert, and you have caught them red handed.

   - Alerts can also be generated when a user begins to exhibit uncharacteristic behavior. A baseline of expected access and update behavior could be maintained by EWS. If suddenly a user starts accessing new data or is much more active, a client defined weighting criteria can be applied. If it exceeds an allowable threshold the EWS can typically generate an alert.

   - **Opportunity:** If your organization is implementing Zero Trust these features are foundational. By ensuring that all credentials are monitored, legitimate or not the requirement to trust no one can be enforced rigorously.

3. Missed malicious activity: Using legitimate credentials both internal attacks from disgruntled employees and external agents with stolen IDs and Passwords are difficult to detect. Many tools that depend on processing the millions of SMF records created daily fail to identify unusual user behavior or recognize unauthorized changes. Often there is no automated process to alert when suspicious actions take place. Even then alerts can be missed if key staff are away or the suspicious behavior is lost in a blizzard of false positives. If an alert is missed it is never regenerated and the problem goes unrecognized.

   - **Problem**: Due to pure volumes of records generated, difficulty in identifying malicious acts when using legitimate credentials and human error tools, that process SMF records to report suspicious behavior often are error prone. Before restoration you must be sure that the backups have not been compromised.

   - **Solution**: IM products detect malicious changes by comparing the contents of programs, parameters and logs to trusted versions. False positives are eliminated because the contents is either correct or not. By automatically learning about new maintenance applied and new versions created, IM tools know what has is the correct state and what is unauthorized. A single step added to your DevOps or update process tells your IM product that the changes are legitimate and anything else is unexpected. This prevents even real insiders from slipping in unapproved code preventing a SolarWinds build system attack from slipping through.

Further, since newer IM tools can determine every component compromised, it can create the required recovery steps from verified trusted components, eliminating all the Timebombs, back doors and other malicious software introduced. This enables a fast, precise recovery without causing more damage by regressing weeks of legitimate system updates.

4. APF Authorization. An authorized program can do virtually anything in z/OS. The consequence of having an untrusted module gain APF Authorization is potentially catastrophic. Authorization is granted at the dataset level (APF libraries, LPALIB, LNKLIST and other system libraries). Any program linked with the AC1 parameter executed from one of those libraries can then operate in an authorized mode. The program can put itself in supervisor state or any system key, modify system control blocks, execute privileged instructions, turn off SMF recording, disable security tracking or even quiesce the whole system.

   - **Problem**: It is often believed that every instance of authorization is specified in the APF member in Parmlib. However new APF libraries can be added dynamically (typical) via operator command rendering the list incomplete. There are commonly 200 to 300 APF datasets and thousands of programs authorized on modern z/OS systems.

   - **Solution**: Ensuring that the APF libraries are locked down using RACF, TSS or ACF/2 is vital. However, in the case of these powerful programs, any change to one of these modules, outside of a normal change process, even by a trusted set of credentials, should be investigated. The only real solution is to have an ongoing integrity monitoring process that verifies that all authorized programs are in their trusted state and alerts are generated when any discrepancies from the trusted state are detected.

   - **Opportunity**; If you trust your current system components you simply run auto-discovery against that image and you are done. If you wish to compare it against the components as they arrived from IBM and 3rd party vendors, an alternate baseline can be created from the original distribution libraries to identify changes in your production systems.

   - **Problem:** By default, LNKLST, SVCLIB and all concatenated datasets are authorized. These, and other systems datasets are not identified in the APF list. As such they may be overlooked when validating the status of APF program access.

   - **Solution:** Continuous automatic discovery of both static and dynamic modifications to APF libraries and other sensitive system dataset (LINKLST, SVCLIB, LPALIB, PARMLIB's, PROCLIB's, TCPPARMs, VTAMLST, etc) is needed. An automated process is ongoing eliminates the administrative overhead of manual review process. Modern integrity monitoring tools ensure that the contents of all programs are in their correct and trusted state. If a change is detected from the expected, trusted state, the IM tool simply raises an alert, potentially avoiding an attack planned for later initiation.

   - **Problem:** Display mechanisms such as Operator and SDSF display commands can show all of the available libraries that are authorized. That makes it easy for hackers to find the full list of eligible datasets and try to find one to which their stolen user IDs has update authority. Even easier is to find a previously authorized ID that no longer exists. Then the hacker can create a dataset with malware inside, change the name to the obsolete entry and presto have an authorized process.

   - **Solution:** Use of the display commands can be monitored in real time and provide valuable early warning that suspicious snooping (or reconnaissance) is underway and should be investigated with advanced forensic tools to prevent problems before they occur. Again whitelists of valid processes that are allowed to use such display commands can avoid unnecessary alerts. Modern monitoring tools can report multiple types of suspicious snooping behavior including SDSF and OPER commands.

5. USS attacks: Unix Systems Services (USS) is now installed on every z/OS system. This is an operational Unix system where users have normal Unix processes available to them. USS can also interchange data and share programs with the z/OS host. Attacking a z/OS system from USS requires less z/OS specific knowledge than would otherwise be required. This type of attack is more likely and thus more dangerous since it is in the knowledge sphere of the most common type of attacker. It is also quite well known as the entry point for one of the more notorious mainframe attacks.

   - **Problem**: Working from within the USS domain hackers can create programs that are authorized in a z/OS context by simply turning on the +A option in the extended attributes. This also requires that the modules are linked AC1 requires limited knowledge of the victim mainframe environment as most of the attack originates from the Unix environment.

   - **Solution:** Programs with the +A attribute turned on still need to be loaded to an APF library to cause harm. Regular integrity scanning to identify unapproved program modifications and additions can detect exposures before the attack is launched. Attacking via USS was a key aspect of the successful attack against the mainframe at Nordea Bank.

6. Compromised Backups: Backups are the primary resource in recovering both infrastructure (system software and applications) as well as data after a successful attack. Knowing this, hackers now typically compromise backups before they attack the real target, critical production data.

   - **Problem**: Before restoration you must be sure that the backups have not been compromised.

   - **Solution**: Once a backup has been created it should never change. Modern integrity monitoring tools create a validation hash key when the backup is taken. Subsequently, the backup can be scanned again and compared to the original. If the keys match, the backup is verified. However, backups can be very large. Newer integrity monitors are able to verify even large backups in seconds ensuring the integrity of critical restore processes.

   - **Problem:** The gold standard is certainly to have an immutable backup created on a frequent basis to ensure that you are able to restore your data at a moment's notice. Determining which immutable copy has not been compromised is not supplied with basic snapset and safeguarded copy implementations.

   - **Solution**: By integrating SMF information with the results of ongoing scans, modern Integrity Monitoring products provide real insight into which snapset or safeguarded copy is safe for restore.

   - **Opportunity**: In addition, full function IM tools with restore assistance can populate a customized restore process on the fly, recovering the compromised infrastructure components from trusted sources and identifying the optimal snapset for data restore. This integrated restore process can avoid repeated attacks and time bombs imbedded in your compromised infrastructure with precision, while minimizing recovery time and effort.

7. Limitations of SMF recording: There is a perception that mainframe SMF catches all activity, suspicious and benign. Although SMF is an excellent service there are many actions that will not be visible. In fact, most of the exposures in this document may never raise red flags using SMF recording fed into a SIEM. Many sites upload SMF data to their security console (SIEM) unaware that the information being provided may be incomplete, potentially creating significant risk.

   - **Problems**: Many types of Indicators of Compromise are missed by SMF and as a result SIEMs and other reporting tools are deficient in their diagnostic capability. We are unaware of any SMF information which identifies the best snapset for recovery or any information indicating when the components were last known to be correct.

   - Some of the snooping techniques used to determine what to go after in a site are just not "seen" by SMF. In one example, a hacking tool retrieves the list of APF datasets and then issues a `RACROUTE REQUEST=AUTH` command on each to determine what access the user has to each APF dataset. This is a non-authorized program

and no record of its operation may be created in SMF. In addition, SDSF, a standard z/OS tool, and ubiquitous OPER commands can display the list of APF, Linklist, and LPA datasets. This is an existing security exposure which generates no SMF records.

- **Solutions**: System exits can trap what types of searches and system modifications are being initiated with TSO, SDSF and OPER. Interception of commands to check authorization levels, prior to granting the request hold significant promise. Research is on-going in this and associated areas. Continuous monitoring from an Early Warning System, integrated with modern IM products, can be enhanced to alert on such activity before subsequent processing.

- Modern IM solutions can provide information to SIEM tools in real time. IM solutions do not replace SIEMs, or SMF, but add significant new information source from a z/OS perspective. In addition, they add new forensic analysis, new recovery and new verification assist features to get your z/OS system back on the air quickly and reliably.

8. Exits in Key 0 state: Exits in z/OS provide many useful functions. Because they provide services to multiple different address spaces, they need to run in the super-powerful Key 0 state.

   - **Problem:** When some exits are loaded dynamically, they can originate from an un-authorized library. Also, LPA modules can be replaced on the fly from an un-authorized dataset. This effectively gives the exit Key 0. Capabilities. The exit address can then be modified programmatically giving access to the modified LPA module.

   - **Solution**: It is critical that sites need to lock down the SET and SETPROG commands. Monitoring that RACF, TSS and ACF/2 rules have been updated to enforce this should be mandatory. Additional research is being conducted in this space to alleviate this problem. Please check back for more emerging solutions in this area.

9. Zero Trust: Integrity monitoring is foundational when implementing a Zero Trust Architecture. The reason is that integrity monitoring verifies that the contents of infrastructure components match the trusted state. It notifies you of all changes to monitored components no matter who is making the change. As such it can apply Zero Trust scrutiny to systems, subsystems, applications and parameter settings, to ensure nothing has been altered in any manner. By definition, this is Zero Trust.

   - **Problem:** Zero Trust is difficult to achieve. Multiple platforms, tools, software and users must be subjected to rigorous control. Malware such as back doors, time bombs, malicious programs are typically incorporated into the infrastructure layer of operating systems and applications. Creating trusted baselines and comparison to components in use is virtually impossible by manual means.

   - **Solution**: Modern integrity monitoring does not create a Zero Trust environment by itself. However, for z/OS it is impossible to achieve Zero Trust without it. By creating trusted baselines for multiple versions of system and applications software IM can detect and alert on any accidental or malicious changes made by an insider, or outsider, even with stolen legitimate credentials. These tools automate the investigation of alerted actions and guides response teams through the required remediation steps. No SIEM or event monitoring tool provides the ability to compare operational software and systems to trusted baselines and is therefore incomplete in a Zero Trust context. Integrity monitoring is a necessary component to achieving Zero Trust.