MANEWS Issue Number 30

the Mainframe Audit News

This newsletter tells you stuff you need to know to audit IBM mainframe computers running with z/OS and the MVS operating system. This issue we describe how to audit the security software whether it's RACF, ACF2, or TopSecret.

Table of Contents

1) z/OS 2.1 off support as of Sept 2018

Release 2.1 of z/OS is now off support. A standard audit check is to ensure that all software in use is on a current, vendor-supported release. You can use the output of the operator command DISPLAY IPLINFO to learn what release you are auditing. If it is not a current release, you will likely have an audit finding. The risk would include: the extra cost of support from the vendor, the missing of vendor-supplied security updates, and the possibility of system unavailability.

You can learn more about z/OS releases and end of support dates at https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=ZSL03438USEN&

2) How to Audit RACF, ACF2, or TopSecret (Part 1)

We advocate a structured approach to audit planning and execution, one which allows us to address the entire question of access contol, while breaking the question into sub-questions, so we can address them one at a time.

Our structured approach to mainframe security auditing consists of two high level components: the MVS operating system security and the authentication/authorization tool, which for mainframes with z/OS is always one of three choices: RACF, ACF2, or TopSecret.

Each of these three has a basic report (or two) showing the essential option settings, a pre-requisite to the audit. For RACF, the reports are called **SETR LIST** and **DSMON**. For ACF2, the report is called **SHOW ALL**. For TopSecret, the report is called **TSS MODIFY(STATUS)**.

Whichever one of the three you have, it answers two basic questions:

Q1 **Who is This User?** (Authentication)

Q2 **Can He/She Do X?** (Authorization)

When we want to discuss the security software when we don't know whether it's RACF, ACF2, or TopSecret, we'll use the generic term **SAF** (System Authorization Facility).

Taking one piece at a time, we break the authentication/authorization (that is, SAF) part of the audit into evaluation of controls over each of five sub-components: access to the system, access to data, access to resources, delegation of authority, and separation of duties.

For this issue we start to address the first sub-component: “Can people access the system without being authorized?”. Our audit plan consists of a structured approach made up of: data gathering followed by analysis of these basic questions: Do the SAF tool and the way it is used provide effective:

-- **Control over every path into the system?**

-- **Password restrictions?**

-- **Administration of userids and passwords?**

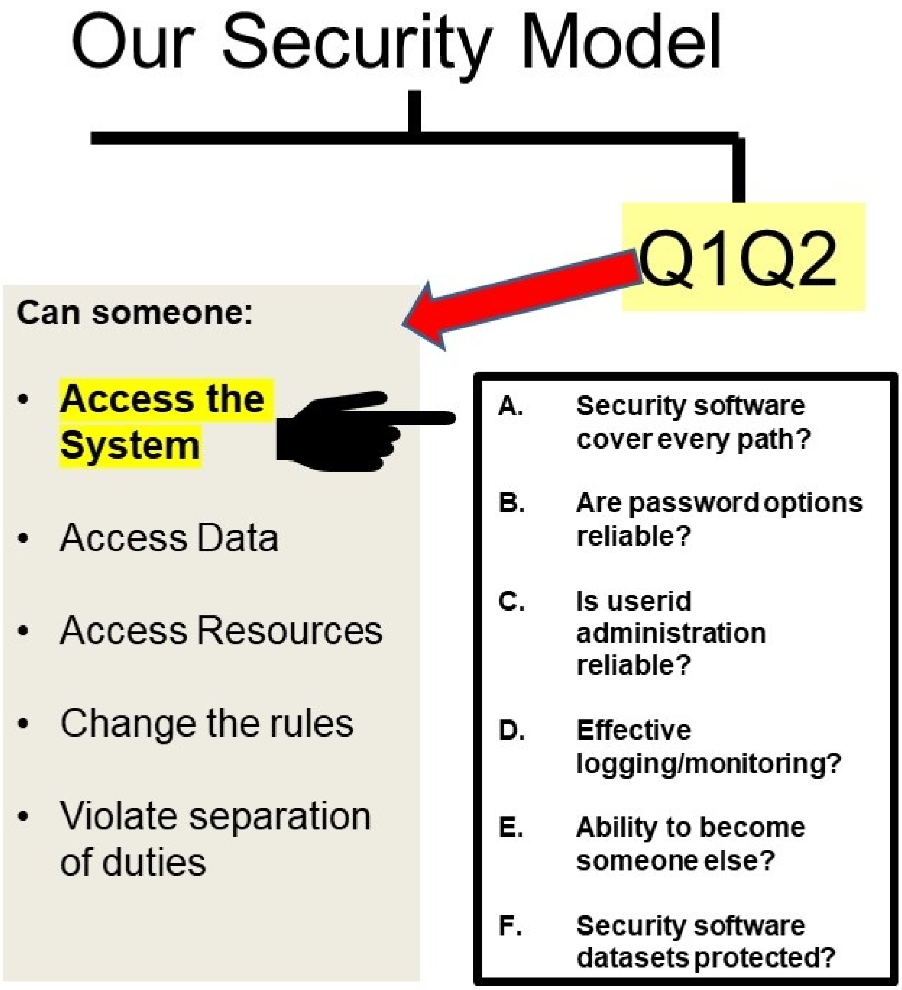
-- **Logging and monitoring?**

-- **Control over the ability to assume someone else’s identity?**

-- **Control over access to the SAF database?**

By “***effective***”, we mean sufficient to ensure that users are reliably identified and that they are only permitted to access the system through paths to which they have been approved. We take this as our control objective, and assume that we address it completely by addressing the sub-components listed above.

SECURITY MODEL



Our Security Model

Q1Q2

Can someone:

- Access the System
- Access Data
- Access Resources
- Change the rules
- Violate separation of duties

A. Security software cover every path?

B. Are password options reliable?

C. Is userid administration reliable?

D. Effective logging/monitoring?

E. Ability to become someone else?

F. Security software datasets protected?

The logic behind this is as follows: if every path into the system is controlled by SAF, and if the controls over passwords and userid administration make it difficult for someone to guess someone else's password, and if the controls over assuming someone else's identity are effective, then we can reasonably conclude that each user is reliably identified and that users can only access the system through paths to which they have been approved.

In the detail below, we provide recommended and "best practice" settings for you to review. Note that if you find actual settings not in agreement with our suggestions, you do not yet have a finding. You have a **condition**, for which you need to identify the **effect**, as per David Hayes' article below.

We recognize that what we describe here does not tell the full, detailed story, but that it does give you a good running start, better than most checklists. As with any audit advice, use only with your own judgement, verifying risks and controls as you go, and inviting descriptions of any compensating controls not covered here.

Control over every path into the system

Paths into the system include: batch jobs, started tasks, signing onto an applid from a terminal, and TCP/IP ports. As an auditor, you would look for Test of Design documents specifying that every path into the mainframe system should be controlled by the security software.

Batch jobs and **started tasks** are programs which execute in the background, as dictated by JCL (Job Control Language), the scripting language which specifies what programs to execute and with which datasets.

An **applid** is a program with a sign-on screen such as TSO or CICS.

TCP/IP is the same communications protocol used with Windows and UNIX and the Internet. Because z/OS includes a full implementation of TCP/IP, often linked indirectly to the Internet, it must be considered a path into the system. **UDP** is a related communications protocol, also included as part of z/OS and should be treated in a fashion similar to that of TCP/IP. Within TCP/IP and within UDP, each port is a number which is associated with a given application such as email, file transfer, or remote logon

Identify all paths into the system, and make sure that each is controlled by SAF.

For RACF, look at the SETR LIST to verify that BATCHALLRACF and XBMALLRACF are active, which fails any batch job executing without a valid RACF userid. Look at the DSMON report to see the STARTED PROCEDURES REPORT, which assigns RACF userids to started tasks. Make sure that there is an asterisk ("*") or catchall entry that matches any started task name not covered by a more specific entry.

An applid is a program with a signon screen such as TSO or CICS which lets you sign on from a terminal.   Review use of resource rules in the APPL class to see how they control who is permitted to log onto which applids. (Obtain the output of the TSO **command RLIST APPL * ALL** to learn who is permitted to which applids.) Determine whether programmers can access production applids, which would violate the separation of production from test.

For TCP/IP and for UDP, you want to verify that all of the ports have been blocked, that it, it is not possible for someone to write a program which "opens a port", creating an uncontrolled path into the system from the Internet. Look in the configuration file for TCP/IP for statements such as **PORT UNRSV TCP * DENY** and **PORT UNRSV UDP * DENY**, which prevent use of any port not specifically defined in that configuration file. Note that there are other techniques for blocking the ports. In any case control over use of TCP/IP and UDP ports should be addressed in Test of Design documents such as policy, standards, procedures, and baseline documents.

For ACF2, look in the SHOW ALL report for the settings: **MODE**, **JOBCHECK**, and **STC OPTION**. (Note that in ACF2, userids as called "LOGONIDs" or "LIDs".)The MODE setting in the SHOW All should be set to **ABORT**, which means that ACF2 is implemented fully. If JOBCHECK is set to YES, then only userids with the JOB privilege specified in their LID (user) record can submit batch jobs. Review the controls over granting of the JOB privlege to a userid.

If STC OPTION is set to ON, then only userids with the STC privilege can be used for started tasks. Review the controls over granting of the STC privilege to a userid.

For TCP/IP and for UDP, you want to verify that all of the ports have been blocked, that it, it is not possible for someone to write a program which "opens a port", creating an uncontrolled path into the system from the Internet. Look in the configuration file for TCP/IP for statements such as **PORT UNRSV TCP * DENY** and **PORT UNRSV UDP * DENY**, which

prevents use of any port not specifically defined in that configuration file. Note that there are other techniques for blocking the ports. In any case control over use of TCP/IP and UDP ports should be addressed in Test of Design documents such as policy, standards, procedures, and baseline documents.

For TopSecret, look at the **MODE** setting in the TSS MODIFY(STATUS) report. It should be set to FAIL, which means that TopSecret is implemented fully. This also means that any batch job not associated with a valid userid will fail. (Note that in TopSecret, userids are called “ACIDs”.)

Each path into the system in TopSecret is defined as a FACILITY. To learn all the FACILITYs that have been defined, get the output of this command: **TSS MODIFY(FAC(ALL))**. Then to learn who is permitted to use, for example, the FACILITY named BATCH, get the output of: **TSS WHOHAS FAC(BATCH)**. To learn which userids are permitted to be used with started tasks, review the output of the command **TSS WHOHAS FAC(STC)**.

Note that when TSS is in FAIL mode, every batch job must be associated with a userid or it will not be allowed to execute.

Since a FACILITY may be defined with a MODE other than FAIL, be sure to check if this is the case. Any mode other than FAIL would constitute a security weakness.

You can learn much more about all the FACILITYs by reviewing the contents of the TopSecret control file. You can learn the name of this dataset from the JCL for TopSecret, which is often a member named TSS in one of the proclibs. This file has large numbers of details about each FACILITY, beyond the MODE. (Many of these are beyond the scope of this article.)

For TCP/IP and for UDP, you want to verify that all of the ports have been blocked, that it, it is not possible for someone to write a program which “opens a port”, creating an uncontrolled path into the system from the Internet. Look in the configuration file for TCP/IP for statements such as **PORT UNRSV TCP * DENY** and **PORT UNRSV UDP * DENY**, which prevents use of any port not specifically defined in that configuration file. Note that there are other techniques for blocking the ports. In any case control over use of TCP/IP and UDP ports should be addressed in Test of Design documents such as policy, standards, procedures, and baseline documents.

Password restrictions

Restrictions can be set over: the minimum length of passwords, the possible content, how often they have to be changed, how they are encrypted, whether they can include lowercase letters, whether they can be supplemented with password phrases, and other options.

Note that with z/OS, passwords have a maximum length of 8 characters. If you need a longer length, then you can supplement passwords with password phrase, which can have a length of up to 100.

For most installations, use of lower case letters and password phrases can be difficult to implement and will not be necessary. The objective is reasonably to minimize the probability that someone could learn someone else's password. We need to understand the risk: If the minimum password length is 7 or 8, and passwords have to be changed every 30 days, and three invalid passwords will cause a userid to be revoked or suspended, and the content must include both letters and numbers, and there is effective monitoring of patterns of invalid password entry and password resets, what is the probability of guessing someone's password? This probability is even lower if users are effectively trained in how and why to make passwords "easy to remember but difficult to guess".

Understand that most people now consider passwords alone as an unreliable method of proving users' identities. Experience has taught us that there will always be some user who is careless keeping his password secret, or a user who falls for a phishing attack. For this reason, most installations are starting to rely on **MFA** (Multi-Factor Authentication), that is, the use of more than just passwords to prove someone's identity. Your audit should address use of MFA, and consideration of the risk if it is not used.

Forcing the use of password phrases and/or mixed case only reduces risk if a hacker is using a password cracker program to learn passwords. Therefore, these options are considered less important so long as no person has read access to a copy of the SAF database containing userids and passwords.

Current implementations of RACF should be encrypting passwords with the KDFAES algorithm, currently considered the most rigorous available. ACF2 and TopSecret should be encrypting passwords with the AES algorithm, the most rigorous available with them.

For RACF, look at the SETR LIST report to see the password options: ENCRYPTION ALGORITHM, CHANGE INTERVAL, NUMBER OF UNSUCCESSFUL, and SYNTAX rules. (For more details on these please consult http://www.stuhenderson.com/XSETRTXT.pdf . For more details on interpreting the DSMON report, please consult http://www.stuhenderson.com/XDSMNTXT.HTM .)

For ACF2, look at the SHOW ALL report to see the **PASSWORD OPTIONS IN EFFECT**:

LOGON RETRY COUNT number of bad passwords before an online session is ended

MIN PSWD LENGTH minimum number of characters in a password

MAX PSWD ATTEMPTS number of bad passwords in a single day before a LID is suspended

PSWD ALTER whether a user can change his password at logon time

PSWD FORCE whether a user is forced to change his password after someone else changes it for him

PSWD HISTORY whether users are prevented from using the last four passwords they have used

PSWD-JES whether bad passwords in JOB cards should be counted in suspending a LID

PSWD-LID whether a password will be rejected if it is identical to the LID

PSWD-MAX number of days before a user must change his password

PSWD-MAXL maximum number of characters in a password

PSWD-MINL minimum number of characters in a password

PSWD-MIXD whether mixed case passwords are accepted

PSWD NUMERIC whether all numeric passwords are rejected

PSWD REQUIRED whether a password is required for all LIDs other than STC and RESTRICTED LIDs

For TopSecret, look at the TSS MODIFY(STATUS) report to see the password options, including:

PWEXP is the number of days after which a user must change his password

PWHIST is the number of most recently used passwords stored in each ACID record to prevent users from re-using them

PWENC specifies the password encryption algorithm, such as AES.

PTHRESH is the number of invalid passwords which causes a user ACID to be suspended.

Evaluate **NEWPW** options in TSS MODIFY(STATUS) output, which specify for new passwords: minimum and maximum length, minimum days before a user can change his password a second time, and password content restrictions. NR is maximum number of repeating characters. RS says not to allow new passwords whose initial characters match any entry in the reserved password list.

Summary

What is the actual risk from the current settings? Do actual settings match standards?

We will save the remaining sub-components for a future issue. In the meantime, be sure to look first for Test of Design documents such as policies and standards. The situation can't be considered stable and reliable if management hasn't provided for clear documentation of how the controls are to work. Any time you think you have a finding, make sure you understand the risk and any possible compensating controls. We'll continue this discussion in future issues.

3) How to Tell Which datasets Are Important

An auditor asked us how to tell which datasets are important to an IS audit. The answer of course is that this should be addressed in the risk assessment step of the audit, likely linked to the financial auditors' control objectives. For example, in a General Ledger application audit, if the control objectives include "the numbers in the financial statements are reliable", then any dataset containing those numbers, would be considered important.

For a security audit of the MVS operating system, there are two sets of datasets cconsidered “important to the audit”. The first is a set of files which we call the “key system datasets”. We use this term to mean datasets for which the following is true: any user who can write to these datasets can bypass all the security on the system. These include: the parmlibs, the APF authorized datasets, the LPA datasets, the proclibs, and more.

The second set of datasets is a set of files which we call the “sensitive system dataset”, that is ones for which read access would let a user view sensitive information such as customer data or social security numbers. These include: the SPOOL dataset, the JES checkpoint dataset, the security software database, and the SMF (System Management Facility) log dataset.

(For more detail, please see http://www.stuhenderson.com/Mainframe%20Audit%20News/MANEWS21.pdf)

4) What Is a Finding?

We asked David Hayes to provide advice for IT staff on what auditors mean when they “have a finding”. His answer was so good that we reproduce it here.

“Get All of the Parts of an Audit Finding

When your auditors inform you that they have *found* something, make sure you get the entire story. Every valid audit finding has four components and should include one or more recommendations for corrective action that specifically address the problem the auditors believe are relevant to what they have identified and their audit objectives.

The first component of a finding is obvious – **the condition**: the auditor’s factual description of some aspect of your control environment that does not adequately correspond to the standards you are being audited against.

The second component is called **criteria**, those standards referred to in the preceding sentence. Effective audits are based on auditors doing extensive preliminary research of the relevant and applicable criteria that apply to your organization and are consistent with the purpose of the audit. Keep in mind that relevant criteria can (and should) include technical standards (from respected sources, especially from vendors), your own organization’s documented policies and procedures, and regulatory edicts that apply.

The third component is a **factual, supported description** of why the problem exists. This is an important and frequently overlooked part of an audit finding. Unless the cause of a problem is identified, how can effective recommendations for corrective action be made?

The last component of an audit finding is **effect**. Quite literally, the auditors must be able to describe in detail how the problem they have identified has a bearing on your organization’s ability to achieve a relevant control objective. The tremendous flexibility of the IBM mainframe computing platforms provides a multitude of techniques available for achieving a level of control consistent with your organization’s control objectives. Some of the auditors engaged in audits involving IBM mainframes may not be cognizant of all of the relevant controls you have in place for a specific operational control objective.

Recently, an audit team cited a datacenter for not having effective controls over monitoring because a certain data field in a log file was not populated with the user's ID. They were using one of the STIGs from DISA as their criteria. What these auditors did not know, or take the time to find out, was that other logging (in the same log file) DID include the user ID. These auditors were using a list and when a setting didn't match their list, they were concluding that something was wrong. When asked about the fourth component of their finding, these auditors could not show that anything adverse would occur due to not recording a user ID multiple times in the same monitoring log.

Two lessons to take from this are: find out if the criteria the auditors plan to use is relevant and appropriate and require that any audit finding contains all four components. When these lessons are followed, auditors and organizations will obtain more value from audits."

Thanks David.

Appendices: Seminar Information and Miscellanea

Appendix A) >>>>Seminar Information

Henderson Group seminars are available for in-house as well as public sessions. For more info, please visit www.stuhenderson.com/XAUDTTXT.HTM

The Henderson Group offers this and other "How to Audit..." courses :

- How to Audit **z/OS with MVS, RACF, ACF2, TopSecret, CICS, DB2, and MQ Series Security** (February 26-March 1, 2019 in Clearwater, FL , and then November 18-21, 2019 in Bethesda, MD)

- **Effective RACF Administration** (November 4-7, 2019 in Bethesda, MD)

Appendix B) >>>>This Issue's Proverb of the Day

“*Become what you are, having learned what it is.*”
— Pindar

Appendix C) >>>>Useful Information

Museful information sources to help you audit more effectively:

1. Articles on Mainframe Security
http://www.stuhenderson.com/Articles-Archive.html

2. New Era offers free webinars by top speakers, and free books to help you audit mainframes better. You can see the seminar schedule and get handouts from previous sessions at
http://www.newera-info.com/The-z-Exchange.html

3. The NIST STIGs (Security Technical Information Guides) for various types of computer, including mainframes
https://web.nvd.nist.gov/view/ncp/repository

4. Useful guidelines for knowing that your InfoSec is comprehensive (Note especially Publication 800-53):
http://csrc.nist.gov/publications/PubsSPs.html#800-53

5. The current release of z/OS is 2.3. Previous releases are z/OS 2.1 and z/OS 2.2. The end of support for z/OS 2.1 is September, 2018. You can monitor end of support dates for IBM software at
http://www.ibm.com/software/support/lifecycle/index_z.html

6. An additional source of free, practical information on mainframe security and auditing, from a variety of sources:
http://www.stuhenderson.com/XINFOTXT.HTM

Appendix D) >>>>About the Mainframe Audit News; Subscribe/Unsubscribe

The MA News is a free, email, newsletter for auditors who need (or suspect that they will need) to be auditing IBM mainframe systems (primarily **MVS**, **z/OS**, and the system software associated with them). This software includes: **CICS**, **DB2**, **JES**, **VTAM**, **MQSeries**, **TSO**, **USS** (UNIX System Services), **TCP/IP**, and others.

It also includes the **httpd daemon** software which connects a mainframe to the Internet. (Note, we expand each of these acronyms and explain how the software works in past and future issues.) The MA News is for auditors who are new to IBM mainframes, and also for experienced MVS auditors who want to keep up to date with the latest developments. We will not make the list of subscribers available to anyone else for any reason.

To Subscribe or Unsubscribe Click on http://www.stuhenderson.com/subscribe.html.

To see Back Issues: www.stuhenderson.com/Newsletters-Archive.html

Feel free to contact us at (301) 229-7187 or stu@stuhenderson.com.