Eleven Steps to Make Mainframe Security Audits More Effective and Efficient

These are some things I've learned about auditing IBM mainframe computers by trying a lot of approaches, some of which worked for me, and some of which didn't. I've been the system programmer being audited and the auditor trying to conduct a good audit. As you'll see below, much of what works is based on establishing a factual basis for the findings and identifying a clear standard against which to evaluate them.

Some of these steps you may already be following. None should conflict with whatever formal audit methodology you are using. Feel free to take what works for you.

1. Focus your scope by recognizing that mainframe security with z/OS has two basic parts: **MVS** (the operating system) security and the security software. The security software is always one of: **RACF**, **ACF2**, or **TopSecret**. So make your focus be either MVS security or the security software. (You could make the focus be the mainframe network or an application's security or how encryption is managed, but that would be different sets of Steps, which we intend to cover in other documents.)

2. Before the audit starts (or at least by the end of the first hour), get these basic documents in hand:

    - For MVS, names of the parmlib datasets and copies of them (get the output of the operator command **D PARMLIB** to get the names);
    - For MVS, the output of the operator command **D IPLINFO**, to learn what release of z/OS (MVS) is in effect
    - For MVS, dataset rules from the security software covering all parmlibs, proclibs APF authorized datasets, the SMF datasets, the page files; JES spool file, JES checkpoint file, security software datasets, and all datasets whose names begin **SYS1.** .
    - For RACF, output of **SETR LIST** and **DSMON**
    - For ACF2, output of **SHOW ALL**
    - For TopSecret, output of **TSS MODIFY(STATUS)**
    - Policy statements, standards, and relevant baseline documents describing how options are supposed to be set, who is responsible for what, how change control functions, applicable standards and regulations, and administrative procedures

3. Verify the financial audit control objectives (reliability of numbers, compliance, protection of assets, going concern, etc.) and decide which if any will be the foundation for the IT audit. If none of these, then decide what is the foundation. The foundation should not be someone's subjective opinion, nor a standard which has not been adopted as company policy.

4. Decide what will be the standard you compare against for the audit. You want to avoid arguments over your findings that amount to your opinion against some system programmer's.

5. Understand that there is no finding unless you identify either significant risk or violation of an applicable law or standard.

6. Keep the findings fact-based. For example, "Password limits are automatically set to a minimum length of five characters." is easy to prove. However, "We believe that this is too short." is an opinion that will not survive a closing meeting.

7. Avoid basing findings on such vague, subjective concepts as "privileges consistent with their job descriptions" or "necessary to do their jobs". These turn out to be meaningless.

8. If there is no standard to compare a finding against, and no one responsible for making a decision, that may be the audit finding. Consider "*Twelve system programmers have privileges that give them access to every dataset*." How do you judge whether this is acceptable or not? If you ask the system programmers, they will tell you that they need the privileges to do their jobs.

   Now consider, "*Twelve system programmers have privileges that give them access to every dataset. We were unable to identify any formal approval for these privileges. We were unable to identify any manager responsible for approving them. We were therefore unable to determine whether these system programmers should have these privileges, since the organization has not set a standard we could compare to. The heads of several business units were not aware that this many people could access their applications' data without approval from the business unit. We recommend that the organization implement formal approval procedures with a specific individual responsible for*

the approval of such privileges and annual re-certification, based on actual need."

Such a finding directly addresses the issue of IT governance.

9. Enlarge your audience whenever necessary. If you are discussing whether an application programmer should have access to his application's production data, include the business unit manager responsible for the application in the discussion. Consider including a financial auditor, or at least reference to the financial control objectives.

10. For an MVS security audit, your starting point is **IBM's integrity statement for MVS** that gives us their assurance that the architecture of the MVS operating system provides reliable security. You will want to address programs which have been added to MVS with privileges that permit the programs to bypass all the security on the system. Such programs are often called "back doors" and are commonly found on every computer platform, including Windows and UNIX. The concept of a back door is not a security problem, so long as the back door can be demonstrated to be secure. (You may have a back door to your house. Your insurance company probably expects you to keep it locked at night.)

    Your control objectives might be to evaluate whether system programming management has the tools available for them to know and for them to be able to demonstrate that all the "back-doors" to the system:

    - Are "safe"
    - Have been approved
    - Can't be modified without formal approval, testing, and detection

    For each of these, you should be able to find a standard against which to compare your findings.

10 For a security software audit, whether RACF, ACF2, or TopSecret, you will want to learn basic settings from the reports in Step 2 above. These will include:

- Password length and content and other settings
- How passwords are encrypted
- How residual data on disk is handled
- How tape datasets are protected
- What resource classes (to protect resources such as sensitive programs, online transactions, and printouts) are active
- How batch job userids are handled
- How started task userids are handled

For each of these, you should be able to find a standard against which to compare your findings.

11. Recognize that IBM mainframes with MVS and z/OS now have full-fledged UNIX systems running under the control of MVS and protected by the security software. This UNIX is called **USS**, for UNIX System Services, and is arguably the most secure, reliable, standard, flexible, and cost-effective UNIX you will find anywhere. It supports full-fledged **TCP/IP**, including FTP, telnet, an httpd daemon (like Apache or IIS), and connection to the Internet. IBM provides excellent tools to secure this all, but these tools are often not properly implemented. This is usually a result of various organizational issues. Frequently this is a major opportunity for auditors to help make things better, but should be addressed in a separate audit.

For More Information:

- The ***Mainframe Audit News*** provides lots of background information, audit tips,and explanations of terms. See back issues or subscribe at: http://www.stuhenderson.com/Newsletters-Archive.html

- The ***RACF User News*** provides technical and administrative advice for IBM's mainframe security software. See back issues or subscribe at: http://www.stuhenderson.com/Newsletters-Archive.html

- The Henderson Group website offers several free, no-registration-required, white papers and webinars on mainframe audit topics. See what's available to download at: http://www.stuhenderson.com/XARTSTXT.HTM

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

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

About the Author:

Stu Henderson has worked both as a system programmer and as an auditor. As a consultant and trainer, he likes to share with others what he has learned on both sides of the fence.