BMC HACKATHON IDENTIFIES AUTHORIZED STATE VULNERABILITIES

Part of BMC's culture is the fostering of ideas from our field and research and development personnel. To achieve this, BMC instituted a hackathon—a marathon of activities to bake new ideas that improve our offerings, exploit our own solutions, or create ideas for brand new products altogether. Some of our software consultants have over 30 years of detailed mainframe knowledge, so letting them try on the hacker hat for a while uncovered some devastating vulnerabilities among common privileges that are often available on production IBM® z/OS® systems. This is part one of a technical series where we will detail some of those possible vulnerabilities so you can identify potential areas of concern in your own environments.

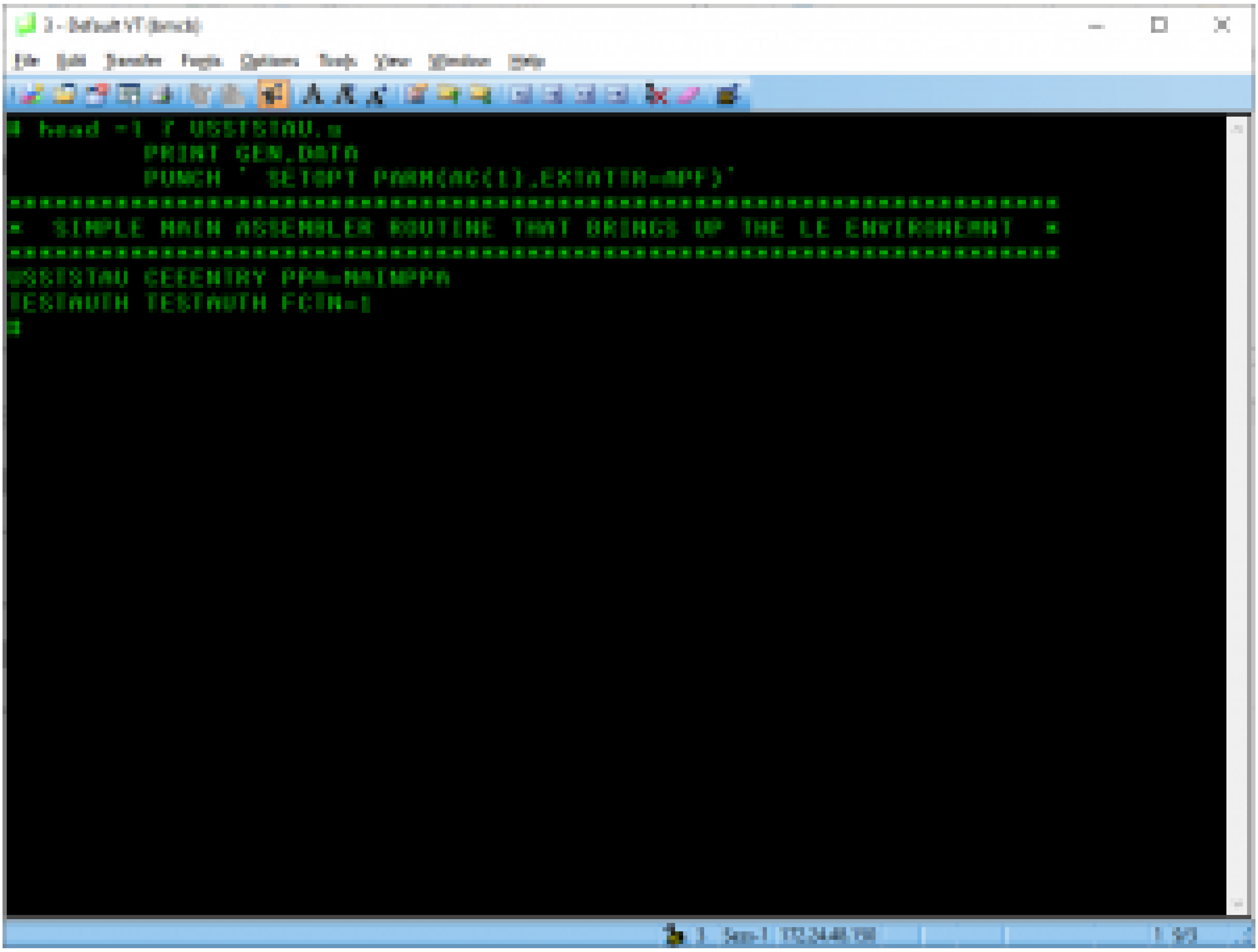
For this installment, we will focus on privilege escalation—the act of getting into a supervisor state without having the expressed permission to do so. As most developers know, one of the most common ways hackers exploit this is through an authorized program facility (APF), which has working exploit code available on Github. To run in an authorized state, a program must meet two criteria: 1) be link-edited with authorization code 1 (AC=1), and 2) the resulting executable load module must be stored in in an APF-authorized library. This can be an existing library, or one that is added dynamically.

Accordingly, any addition of a program to an existing APF library, or the addition of a new library to the APF list, needs to be reported to the security operation center (SOC), as this represents a significant security threat to the mainframe. To help with this, BMC AMI Security provides dynamic enrichment for any action in an APF library or a dynamic APF command that would not be available in traditional logs. These are then processed into real-time alerts that can be sent directly to an enterprise security information and event monitoring (SIEM) solution or a security orchestration, automation, and response (SOAR) solution to trigger an incident response.

This is all well and good. However, our mainframe experts decided to take it one step further and highlight methods of running in an authorized state that **do not involve the use of z/OS APF libraries** at all.

UNIX System Services (USS) Directory Method

One of these methods uses a **USS directory location** to store programs with the extended-attribute APF, which will allow them to execute in an authorized state. It is easy to assemble and link-edit such a program, even while strictly running in a USS shell (i.e., without submitting a batch job). Here is sample assembler source that exploits this method.



```
# head -n 7 USSTSTAU.s
         PRINT GEN,DATA
         PUNCH ' SETOPT PARM(AC(1),EXTATTR=APF)'
***********************************************************************
*  SIMPLE MAIN ASSEMBLER ROUTINE THAT BRINGS UP THE LE ENVIRONMENT    *
***********************************************************************
USSTSTAU CEEENTRY PPA=MAINPPA
TESTAUTH TESTAUTH FCTN=1
#
```

There are a couple of ways that the extended attribute can be assigned to a program instead of assembling it. For example, shell commands *extattr* and *untar* support the extended-APF attributes, as well as PGM=BPXCOPY and ISPF option 3.17 (using line command *mx*). Needless to say, each needs to be locked down.

This method is slightly more concerning because many organizations fail to consider USS security the same way they consider standard z/OS permissions, and our mainframe penetration testers have found it to be one of the most successful avenues to escalate their privileges while attacking some of the largest mainframe shops in the world.

The good news is, that regardless of the manner used to set the APF attribute, it is both securable and auditable. BMC AMI Security provides the same level of enrichment and out-of-the-box detection to provide real-time alerts for any anomalous or malicious activity in USS. In addition, it provides an automated audit capability that will alert your security team to any open USS permissions that may have been overlooked.

Multiple Virtual Storage (MVS) Command Method

The last method we will discuss in this series is the use of MVS commands. As one of our consultants said, "**Give me access to run MVS commands and I own the mainframe.**"

First, some context. Before it can be executed, a z/OS program must be fetched and loaded into storage. For performance reasons, z/OS dedicates an area of storage for pre-fetching and loading programs during an initial program load (IPL). This portion of storage is known as the link pack area (LPA). These programs can be executed without specifying a library (JOBLIB/STEPLIB) and are found before LINKLIST. They are available to all address spaces, and because they reside in the LPA, they can run as authorized as long as they were link-edited with AC=1.

To avoid an IPL, LPA is not normally changed, except for emergency maintenance. In these emergency situations, an MVS command can be used to add a program to the LPA on the fly and it will run as authorized, even it is added from a non-APF library.

This command:

SETPROG LPA,ADD,MODNAME=*mybypass*,DSNAME=*prefix.non.apf.listed*

dynamically adds program *mybypass* to the LPA.

So, what does this mean? With an MVS command, you can quickly add any program that gives you full control over the mainframe to the LPA in an authorized state. Or, more succinctly, any user with MVS command authorization should be considered a fully privileged user.

Fortunately, the use of MVS commands is also securable and auditable. BMC AMI Security real-time detection and alerting is enriched to report on the use of any MVS command or any suspicious activity or indicator of compromise (IOC). In the case of adding a program to the LPA, this needs to be sent to a SOC with the same urgency as the addition of an APF library, since it indicates the potential misuse of authorization for a privilege escalation. For most organizations, if you are not actively monitoring for this type of activity in real-time, you will not know there is a potential catastrophic event until it is too late.

The singular takeaway from the plethora of ways to abuse privileges is that **you won't know there is abuse unless you are looking.** Our mainframe hackers have a near-perfect record of finding privilege escalation methods at major organizations that truly felt their environments were perfectly secured. Because they were never tested adversarially, they developed a false sense of security that comes from decades of not being attacked in the same capacity as internet-facing Windows and Linux servers.

With BMC AMI Security, you can leverage automated audit capabilities to lock down and harden your environments from external or insider threats. Our threat detection and response will then provide you with real-time monitoring and alerting capabilities integrated directly into your 24x7 SOC so that you can avoid the existential threat of ransomware on the mainframe that has crippled the first victims of mainframe malware. BMC AMI Security is built by hackers to stop hackers. Learn more at bmc.com/ami-security.