



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

# **An Authentication Audit on OpenVMS: An Auditor's Perspective**

Exclusively Cobit Control Objective 5.2:  
Identification, Authentication and Access

© SANS Institute 2000 - 2002, Author retains full rights.

Submitted by Jeff Parker, April 15, 2002  
GSNA Practical v2.0  
*A GIAC Practical applied toward the  
GIAC Systems and Network Auditor (GSNA) certification*

## Table of Contents

### SECTION 1: RESEARCH IN AUDIT, PRACTICE AND CONTROL.....4

<b>Our Scope is a product of the Cobit Framework.....</b>	<b>4</b>
The Two Paragraph Cobit Framework Description.....	4
How Are We Proceeding? .....	4
Auditors Frequently Must Audit Unknown Technology.....	5
<b>Scope.....</b>	<b>6</b>
Business Impact.....	6
System Risk Overview.....	6
Availability.....	7
Integrity.....	7
Confidentiality.....	7
Current State of Practice.....	7
Improvement Potential in Current Practices.....	8
What We Are Auditing.....	8

### SECTION 2: AUDIT CHECKLIST .....9

Checklist Format .....	9
Checklist Items .....	10
Item 1: Minimum password length .....	10
Item 2: Minimum password age.....	10
Item 3: Initial password change .....	11
Item 4: Maximum password age.....	11
Item 5: Bad user/password distinction.....	12
Item 6: Default accounts.....	13
Item 7: Proxy logins.....	13
Item 8: Default user privileges.....	14
Item 9: Account Criterion Override.....	15
Item 10: Sufficient user privileges.....	15
Item 11: “System-level” users by group membership .....	16
Item 12: CTRL-Y During Login.....	17
Item 13: Welcome message.....	17
Item 14: Login (Attempted) Break-in Limit.....	18
Item 15: Password Field Limit Check .....	18
Item 16: Field Service (Provided Account) Disabled.....	19
Item 17: SYSTEST & SYSTEST_CLIG (Provided Accounts) Disabled .....	19
Item 18: Login Timeout Period.....	20
Item 19: Alert of Reading/Altering of UAF file .....	20
Item 20: Prey Accounts.....	21

### SECTION 3: CONDUCTING THE AUDIT .....22

Top 10 Checklist items .....	23
Welcome message (item 13) .....	23
Password Field Limit Check (item 15).....	23
“System-level” users by group membership (item 11).....	23
Field Service (Provided Account) Disabled (item 16).....	24
SYSTEST & SYSTEST_CLIG (Provided Accounts) Disabled (item 17).....	24
Login Timeout Period (item 18).....	24
Sufficient user privileges (item 10).....	26
CTRL-Y During Login (item 12).....	27

Alert of Reading/Altering of the UAF file (item 19).....	27
Minimum Password Length (item 1).....	28

## **SECTION 4: FOLLOW-UP .....30**

### **Executive Summary .....30**

### **Audit Findings .....30**

Findings In Bulleted Form.....	30
I. Password Related.....	31
Password Criteria.....	31
Password Process.....	31
II. Restriction of user privileges.....	31
III. Restricting functionality directly related to the login process.....	31
Findings in Detailed Form.....	32
Password Maximum Age -Passed.....	32
Password Minimum Length -Non-compliant.....	32
Bad user/password distinction -Passed.....	33
Forced Initial Password Change -Passed.....	33
Default user privileges -Passed.....	33
“System-level” users by group membership -Passed.....	33
Sufficient user privileges –Non-compliant.....	34
Default System Accounts –Non-compliant.....	34
Prey Accounts –Non-compliant.....	35
Field Service (Provided Account) Disabled –Non-compliant.....	36
CTRL-Y During Login –Passed.....	37
CTRL-Y for Default User profile –Non-compliant.....	38
Login Timeout Period –Passed.....	38
Password Field Limit Check –Passed.....	39
Welcome message –Non-compliant.....	40

### **Compensating Controls .....41**

Controlling the Factors of Risk.....	41
Graphing Compliance Opposed to Likelihood.....	41

### **Appendix 1: References.....43**

### **Appendix 2: All OpenVMS Privileges .....45**

### **Appendix 3: System Manager Account.....46**

The screenshot of ‘UAF> sh system’.....	46
---	----

## Section 1: Research in Audit, Practice and Control

### Our Scope is a product of the Cobit Framework

Those not familiar with Cobit are strongly encouraged to spend a few afternoons reviewing the six documents available at: <http://www.isaca.org/cobit.htm>. (Free for ISACA members, all but *Audit Guidelines* free for non-members).

### The Two Paragraph Cobit Framework Description

There are six all-encompassing documents, of which *Audit Guidelines* is one. Although the introductory pages of all six documents are similar and capable of explaining to a reader what the 'essence' is of the six documents, the *Executive Summary* does it best. A reader should really browse through all the documents first to gain a general overview, then return to the *Cobit Framework* for a closer look at the outline.

From **the Framework**, the entire realm of IT/process auditing **is divided into four domains**, seen listed below. Those four domains are further divided into 34 **general processes**, each of which may contain several **detailed control objectives**. The processes contain anywhere from 3 to 30 of these control objectives, totaling 318 detailed control objectives for the Framework.

### How Are We Proceeding?

Cobit Framework → Selecting Domain/Control Objectives → Developing Scope

#### Cobit Framework

I have chosen our scope to concentrate on data confidentiality. Therefore, in following the Cobit *Audit Guidelines*, we navigate through the 'three dimensional' orientation of vantage points (IT Domains / IT Resources / Information Criteria):

<u>IT Domains (4)</u>	<u>IT Resources (5)</u>	<u>Information Criteria (7)</u>
Planning and Organization	People	Effectiveness
Acquisition and Implementation	Applications	Efficiency
Delivery and Support	Technology	Confidentiality
Monitoring	Facilities	Integrity
	Data	Availability
		Compliance
		Reliability

Picture the above table as a cube of 4x5x7 divisions, where each division is a number of the detailed Control Objectives described above.

### Selecting Domain/Control Objectives

By following the Cobit Control Objectives Summary Table found in the *Control Objectives* document, we see there are both secondary and primary processes (also called *detailed Control Objectives*). The only primary processes involving both the IT Resource “Data” and the Information Criteria “Confidentiality” are found beneath two general processes under two different domains:

1. Assess Risks under the “Planning and Organization” domain.
2. Ensure Systems Security under the “Delivery and Support” domain.

Each ‘process’ covers several fields of study, so we will narrow this down further.

### Developing Scope<sup>1</sup>

The documents *Audit Guidelines* and the *Control Objectives* provide an approach from which to start an audit. With this approach and a threat/risk understanding unique to each environment, an auditor can assess any infrastructure. It is with this approach that I am starting to audit primary objectives dealing with data confidentiality. Even though this helps narrow the auditing realm to only about 5% of what Cobit could help an auditor deal with, this is still entirely too broad a topic.

For example, narrowing the *Control Objectives* concerned with IT Resource ‘data’ and Information Criteria ‘confidentiality,’ and still under the “Ensure Systems Security” process, still includes all the following subtopics:

- Cryptographic key management
- Non-repudiation
- Data Classification
- Security Surveillance
- User Control of User Accounts
- User Account Management
- ...and 15 other sub-fields!

These topics alone would take too long to review, so we further narrow the focus of this paper as follows:

Our scope involves the following Cobit Framework control objective:

- Identification, Authentication and Access (Control Objective 5.2)

### **Auditors Frequently Must Audit Unknown Technology**

All of the detailed *Control Objectives* are platform independent, so we still need to choose an Operating System. In the interests of truly representing a situation that any auditor could find, I opted to conduct this paper using an OS that I have not even rudimentary knowledge of.

---

<sup>1</sup> This is all laid out clearly so that future GIAC GSNA students can approach OpenVMS data confidentiality ideas and still feel confident about not duplicating efforts.

OpenVMS running on VAX hardware. It is only through utilizing valuable resources such as system manuals, administrators and subject matter experts that the auditor can prepare adequately and focus on the audit itself from the start. This method also yields the most objective (and thus accurate results). It also benefits this paper the most when the audited subject is in a production environment. The use of this VAX hardware follows shortly.

## Scope

I am auditing a VAX 6000-630 system running OpenVMS version 7.1.

The VAX takes on two primary roles:

1. As a client/server interface to a backend VAX cluster, running a corporate proprietary application.
2. As a user storage repository.

Two other, more secondary roles include:

1. Reading of "Notes" - internal and proprietary newsgroup to the organization.
2. Sporadic VAX-mail usage.

## Business Impact

The primary roles are significant to the organization:

Serving as an interface is vital to business needs.

As a storage repository, the data is largely user files and thus can range in importance from less important to near-vital to business needs.

The first role (interface) serves a group of approximately 60 individuals whose primary mission requires this machine, while the second role (storage) serves about 100 people who may access their files only to support their varied job functions.

The two secondary roles, Notes & VAX-mail, have chiefly been replaced by web-conversion and the more common desktop-client mail respectively. Usage in these areas is estimated at 15-20%. Therefore, Notes and VAX-mail will be regarded as particularly minor in respect to the primary roles.

## System Risk Overview

Given the roles described above, a preliminary overview must be drawn regarding risk. This covers three distinct areas:

- Potential problems
- The chances of a potential problem occurring
- the consequences that would result from a problem occurring

## **Availability**

Within the framework of availability, integrity and confidentiality, we begin by hypothesizing what would happen if the system became unavailable. A denial of service is unlikely to occur as this VAXserver is it is one node of a VMS cluster -providing high availability. The redundancy includes the initial connectivity hardware, but that is beyond the scope of this paper. System hardware failing could be considered additional risks, but again, the high availability offered by multiple nodes providing the users service all but nulls the risk of individual boards failing.

The consequences of this node being unavailable would naturally mean the users are unable to perform their job function related to this system. The impact would be great, but the higher reliability of the VMS cluster reduces the likelihood to nearly negligible.

Thus availability risk is very small.

## **Integrity**

The integrity of the system data is also ensured through redundancy. OpenVMS as an operating system provides the functionality to simultaneously shadow (mirror) any volume to more than one other disk. For example a system manager could mirror five sets of a volume. This particular node does employ protection of the volumes by shadowing them, so the probability of harming data integrity by disk failure is greatly reduced. The system boot disks are shadowed (mirrored). The consequences of a failed system disk or any other volume would cause a severely degraded service if not outright denial to the end users.

With reduced probability of such great cost, the integrity risk is thus minor.

## **Confidentiality**

Compared to the above methods protecting the hardware, confidentiality is not as easily maintained. Authentication is singular in scope and weak in transport. Specifically, passwords only are the method and often client/server services such as Telnet are used for management. However, unlike Telnet, the application(s) used by end users do not send their authentication in clear-text. Therefore the authentication process used during normal business is suspected to be of nominal risk. The consequences of end users breaching security, i.e. gaining one another's privileges, are significant due to the probability that some have considerably higher privileges than others.

Thus the confidentiality risk is significant, justifying the basic reason for the audit.

## ***Current State of Practice***

In foraging for established audit guidelines and written programs for auditing VMS, several were found that did overall, generic auditing. A few 'quasi-checklists' were available online, presumably by former DEC employees or field service representatives distributing their experience and knowledge. Those were most useful for having an overview of what to look out for.

I also utilized two subject matter experts (SME) –coworkers who work with OpenVMS on a daily basis. One of the two individuals is a professional consultant who provides both proactive and reactive services for OpenVMS customers. The other



individual is a system manager for the system I chose to audit. Both individuals were extremely useful in helping me develop a reasonable scope. The system manager provided continuous SME support throughout the audit process as well.

I grew to dislike the typical web site containing audit checklists and instead preferred the face-time and interviewing those 'in the know.' Learning happens at a much faster pace when questions can be answered directly as they are asked. I cannot stress enough the value-add given by having subject matter experts close at hand. Their answers in real-time can be given as the need arises, instead of antiquated web pages.

### ***Improvement Potential in Current Practices***

Several of the problems with the checklists found online are a result of the ever-changing playing field. Although the checklists may address areas that were commonplace to the typical system manager back in the 1980's and 1990's, the audit programs and lists lacked a focus that present-day NT administrators or Unix administrators would require or expect. Simply said, attacks that are routine these days did not occur during the heydays of the VAX systems. Even attacks such as Trojan horse programs were more to defend. Specific host-based auditing has improved in recent years.

### ***What We Are Auditing***

As the roles of the machine differ greatly in both usage and function, our scope focuses on what is common to all roles of the machine: the initial login. Namely, we will audit the methods OpenVMS v7.1 running on a VAX employs toward identification, authentication and access methods. This includes the strength of the passwords, password management and default account management. Network security up and including the machine is beyond the scope of the audit.

## Section 2: Audit Checklist

### Checklist Format

Auditor: Checklist items are provided as a table that includes the following fields:

**Risk ( X ):** Given as a function of Likelihood and Consequence. They range from:

Likelihood: Unlikely / Doubtful / Suspect / Probable / Certain

Consequence: Negligible / Slight / Significant / Major / Dire

Numeric total will be 1 to 10 (1 being the lowest, 10 the highest)

Risk will be further explored underneath each item.




**Control Objective:** Brief purpose statement for what we hope this step gains.

**Objective / Subjective:** Only one of the two words will be **BOLD**. Further explanation provided may state ability to verify the results in independently verifiable terms.

**Compliance:** How well the audited system conforms to the above Objective, stated in terms of a yes/no approach or a range of compliance.

**Testing:** Step-by-step on how to test for the above compliance, commands included.

**Reference:** A single icon describes the type of reference. After the icon will be a number to further distinguish it from the general icon. A reference chart in Appendix 1 associates the icon and number to its respective website, manual or book.


Reference	Icon
World Wide Web	
Manuals or books	
Interviews with subject experts	

<b>Example Reference:</b>	 <b>5</b>
---------------------------	--

This cites the 5<sup>th</sup> Internet reference, which is a “VMS Pocket Reference List” located at the URL: <http://www.utexas.edu/cc/docs/ccrl21.html>  
All references, including Internet, books or manuals, may be found in Appendix 1.


## Checklist Items

### Item 1: Minimum password length

<b>Reference:</b>  2	<b>Risk (6)</b> = Likelihood (Suspect) + Consequences (Significant)
<b>Control Objective:</b> Determine if password minimum length measures exist.	
<b>Objective / Subjective:</b> The test is clearly a yes or no to pass. (Binary).	
<b>Compliance:</b> According to a variety of “Best Practices” available on the web, we will state the password length to meet a minimum of 8 characters. Less than eight characters does not meet compliance.	
<b>Testing:</b> To perform this test, have system admin create a new user (ex. ‘GSNA’) on the system. If a password can be changed containing fewer than stated minimum length.  Also, have system manager type at UAF prompt: ‘sh default’ –listing all default user parameters (utilized by system upon new user creation). The “Pwddminimum” parameter shows the minimum number of characters for password entry.	


**Risk:** A long minimum length helps thwart quick attempts to brute force, crack it, or easily guessing. It is somewhat common for admins not to enforce a minimum password length. The consequences of a small password are like a compromised account, it is just a matter of time until it is compromised.

### Item 2: Minimum password age

<b>Reference:</b>  2	<b>Risk (5)</b> = Likelihood (Doubtful) + Consequences (Significant)
<b>Control Objective:</b> Determine if password minimum age exists.	
<b>Objective / Subjective:</b> This is a binary test. Either minimum age exists or not.	
<b>Compliance:</b> A minimum password minimum age should optimally be no less than (max age of password / times possible to guess without tripping bad attempts). To simplify, we shall state compliance is a minimum age of 7 days.	
<b>Testing:</b> To perform this test, have system admin create a new user on the system. Log on as user and change password as prompted for initial login if you haven’t already. Try to change password soon afterward. Does system react and deny change or is new password allowed?	


**Risk:** The inherent risk of having no minimum password age is that a user may opt to change his or her password to appease system requirements, only to change it a second time reverting to an older, familiar password. Consequences are nearly the same as for having no maximum password age, since users may use the same password indefinitely.

### Item 3: Initial password change

<b>Reference:</b> 	<b>Risk (5)</b> = Likelihood (Doubtful) + Consequences (Significant)
<b>Control Objective:</b> Determine if new user upon first login is forced to change initial password.	
<b>Objective / Subjective:</b> It is a binary test, as a first login clearly shows whether or not you must change your password.	
<b>Compliance:</b> Initial or default password cannot be used after first time logging in. Login process should prompt user for new password and initial password may not be simply reentered without the system kicking it back.	
<b>Testing:</b> To perform this test, have the system admin create a new user with a known initial password (either by default or set by the admin). Log out and log in with new user. Is user prompted to change? If so, try entering exact same password. If permitted, test is not a success. If prompted again until new password entered, test is successful.  Also, have system manager type at UAF prompt: <code>'sh default'</code> –listing all default user parameters (utilized by system upon new user creation). If the “Pwchange” parameter shows a setting of “(pre-expired)” (without quotes) –that also verifies the system will force a new user logging on for the first time to change his or her password.	

**Risk:** Default passwords are too common and likely known by anyone who has been given an account. However, new users are likely to log on relatively quickly after a new account has been created, lowering the likelihood of this vulnerability. The consequences of someone successfully exploiting a newly created account with a known default password are not minor, and so the sum risk is given a 5.

### Item 4: Maximum password age


<b>Reference:</b>  3	<b>Risk (6)</b> = Likelihood (Doubtful) + Consequences (Major)
<b>Control Objective:</b> Determine if maximum password age exists.	
<b>Objective / Subjective:</b> Binary test. Either max age exists or not.	
<b>Compliance:</b> According to various “Best Practices” sources, maximum age should be set with consideration of use of machine, difficulty of password cracking, access to password hash file and a number of other factors. For this VMS audit, we shall set a maximum age for passwords at 90 days.	

**Testing:** Have system manager type at UAF prompt: ‘sh default’ –listing all default user parameters (utilized by system upon new user creation). The “Pwdlifetime” parameter shows a setting of X YY:ZZ” where X denotes days, YY = hours and ZZ is minutes. Have system manager then list parameters on your new user “sh GSNA” –and change the Pwdlifetime parameter to have password expire after 5 minutes. After 10 minutes attempt to log on. You should be prompted to immediately change your password.

**Risk:** Obviously the lack of a maximum age on the users’ passwords results in no enforcement to the policy (if it exists) to periodically change one’s password. Most likely such a policy exists and the system is set to enforce it. Typically, the max age is long, but present. Consequences are great if password is guessed and so the risk is increased if the time allowed for those consequences is increased longer than necessary.


Note: A password is strong enough until it can be cracked or guessed. Strength of a password relies equally on its choice of characters and length of time to brute force those characters –directly determining the maximum age.

### Item 5: Bad user/password distinction

	<b>Reference:</b>  <b>Risk (3) = Likelihood (Unlikely) + Consequences (Slight)</b>
<b>Control Objective:</b> Determine if difference exists between attempting to log on using a valid account with bad password and logging on with an account not created.	
<b>Objective / Subjective:</b> This test can yield a range of results, giving some flexibility in judging the results and making the test subjective. However, the auditor may judge a successful audit (no difference) as strictly binary.	
<b>Compliance:</b> If the system replies or responds the same between both actions described in the Control Objective, then the audit of this item can be considered successful. If system responds differently (via different error messages), audit is not successful.	
<b>Testing:</b> To perform this test, have system manager first demonstrate at UAF prompt that a particular user name does not already exist. This can be done at the UAF prompt by typing ‘sh userabc’ and noting the expected ‘no such user’ message. Then the system manager should attempt to log on to the previously created GSNA account with bad password. Note the system response. Have system manager then log on as ‘userabc’ or otherwise unknown account: Note the system response. Responses are identical?	

**Risk:** If a malicious user attempts to gain unauthorized privileges using another person’s account, he will need to verify that the account exists first. Since some operating systems do respond differently to a bad password entry verses a bad account name, the likelihood is great. Still these items’ consequences are relatively low since other factors determine how great is the malicious user’s probability to gaining access even with a known good account. Knowing an account exists should not in itself define weak security (Security through obscurity is bad, mmmkay?).

## Item 6: Default accounts


<b>Reference:</b>  <b>5</b>	<b>Risk (3-8)</b> = Likelihood (Doubtful) + Consequences ( <i>various</i> )
<b>Objective / Subjective:</b> Binary test. Either the accounts are there or not.	
<b>Control Objective:</b> Determine whether various default accounts exist.	
<b>Compliance:</b> There are several accounts installed with OpenVMS by default with default passwords. All of them should have their passwords changed and those unnecessary accounts should be disabled.	
<b>Testing:</b> To perform this test have system admin attempt to log on using the following account and password pairs: guest / guest guest / <no password> default / default decnet / decnet operator / operator operations / operations field / field or service system / system or operator or manager	

**Risk:** Several accounts exist by default in OpenVMS, although the passwords have been quite likely changed to whatever the system manager wanted. While some are not necessary (i.e. guest, operator), others are required for system operation (i.e. system, decnet).

The consequences of those necessary system-level accounts having a default or weak password would be dire. The 'system' account, for example, is what the system manager typically logs in as for day-to-day management.

It also means what it implies: the person has complete control of system resources allocation (CPU, memory, disk quotas, # of processes, etc.). The mischief that can result from the misuse of that account is unlimited.

## Item 7: Proxy logins


<b>Reference:</b>  <b>1</b>	<b>Risk (6)</b> = Likelihood (Doubtful) + Consequences (Major)
<b>Objective / Subjective:</b> This can be considered subjective since, although proxy logins are inherently the more secure method, only the system manager knows best if the use of the system warrants it. They should be used as very underprivileged.	
<b>Control Objective:</b> Determine that Proxy logins are used in place of interactive logins (when applicable.)	

**Compliance:** If Proxy logins are utilized, then ensure they are unprivileged accounts.

**Testing:** To perform this test, have the system manager check Authorize utility for Proxy accounts and type **show known node characteristics** at a command prompt.

**Risk:** During interactive logins, passwords are broadcast across the network to authenticate the user logging in remotely. During Proxy logins, the user may authenticate without having to supply any access control information. The likelihood of passwords being sniffed off of the network increases dramatically when an insider, such as an employee, utilizes tools to do so. Broadcasted passwords become a huge risk to the legitimate account being compromised, granting the “sniffer” privileges potentially very high privileges. The consequences of such a root cause include instant misuse of valid accounts –the highest breach of accountability.

## Item 8: Default user privileges

 <b>Reference:</b>	<b>Risk (6)</b> = Likelihood (Doubtful) + Consequences (Major)
<b>Objective</b> / Subjective: Binary test.	
<b>Control Objective:</b> Determine whether any privileges granted by default are above standard NETMBX and TMPMBX.	
<b>Compliance:</b> These two privileges are the two default privileges granted to default users by the system. Only unless the system manager has altered the default user file DEFAULT.DAT, should it read otherwise. Any additional privileges shown to be given by default would therefore not be a successful audit of this item.	
<b>Testing:</b> To perform this test, have system admin type ‘sh default’ at the UAF prompt. Under “Authorized Privileges” and “Default Privileges” there should only be listed these two: NETMBX & TMPMBX. “Default Privileges” denote those granted immediately upon login, while “Authorized Privileges” are those the user is capable of gaining themselves without any further system manager administration.	

**Risk:** OpenVMS is unique among operating systems in that its privileges empower users by functions more than by distinct file rights or object rights.

For example, some of 35 or so user/group privileges include:

ALLSPOOL -allocate spooled devices.

NETMBX -create network connections.

TMPMBX -create temporary mailboxes.

PRMMBX -create permanent mailboxes.

GRPPRV -able to grant self the highest among the group’s members.

SETPRV -able to grant self any privilege –Any privilege.

*All privileges are documented for reference sake in Appendix 2.*


So, you can see some privileges are much more empowering than others. The likelihood of someone getting privileges they’re not entitled to depends wholly on the privileges they’ve got presently. In other words, the ability to ‘get’ depends on what



they've 'got.' The likelihood the system manager has their default user file set up to give more than the standard two (TMPMBX & NETMBX) is doubtful to unlikely.


Similarly, the consequences are dependent on which permissions are gained. The potential for abuse is great (if security is not tight to begin with). So, those currently with system-level privileges must be trusted not to abuse their authority.

### Item 9: Account Criterion Override

<b>Reference:</b> 	<b>Risk (5)</b> = Likelihood (Doubtful) + Consequences (Significant)
<b>Control Objective:</b> Determine if user creation follows restrictions set forth by default user criteria.	
<b>Objective / Subjective:</b> Binary test.	
<b>Compliance:</b> This item is fulfilled (pass) if the new user (by way of command line switches) cannot be created when attempting to override Default User criteria.	
<b>Testing:</b> To perform this test, have the system manager create a user using command line switches, attempting to override the minimal/maximum criteria set by the Default User file.	

**Risk:** The risk associated with being able to create a new user that bends the rules of the Default User depends on your point of view. Obviously the system manager can and should be able to override limits or system restrictions to an extent. And the likelihood of a system manager bypassing the parameters for no good reason is unlikely; they will do so when necessary. The consequences, though, are perhaps lessened security. Therefore the risk is relatively low to medium – a 5 out of 10.

### Item 10: Sufficient user privileges


<b>Reference:</b> 	<b>Risk (6)</b> = Likelihood (Suspect) + Consequences (Significant)
<b>Control Objective:</b> Determine if users are granted privileges (as shown above) higher than necessary to perform daily job functions.	
<b>Objective / Subjective:</b> People will challenge any removal of authority as a degrading of their work potential. This will become subjective with the system manager as the ultimate moderator.	
<b>Compliance:</b> A deliberate major misappropriation of privileges can define the item as noncompliant. Otherwise it is up to the auditor to evaluate the user's elevated abilities unless the system manager can demonstrate otherwise.	



**Testing:** To perform this test, have system manager list his operators, backup operators, group managers and other IT staff. Go over the individuals with the system manager to oversee their privileges, keeping in mind their duties. For evaluating all the users not on the system manger's list, proceed this way: Have the system manager run the Authorize utility. Then at UAF prompt, type: `sh [*,*]/page=save/search=SYSPRV` (instead of SYSPRV, you may insert any privilege you wish to search for) Evaluate all users not on that list with privileges other than the two standard (NETMBX & TMPMBX) –especially be watchful of those with SYSPRV, BYPASS, DEVOUR or GRPPRV privileges.

**Risk:** Privileges granted in OpenVMS, like in all operating systems, give the system managers (root, admins, etc) ultimate authority and control. That demands trust on the side of management to believe that control will not be abused. Assuming trust is given correctly, the users who should not be granted that control are the main target of this audit item. Likelihood is suspect since privileges granted for a one-time occurrence may not have been removed at a later date. The consequences vary in part due to the different privileges, but on average are significant. They range from acquiring unauthorized system resources to achieving system-level privileges.


#### Item 11: “System-level” users by group membership

<b>Reference:</b>  <b>1</b>	<b>Risk (7) = Likelihood (Doubtful) + Consequences (Dire)</b>
<b>Control Objective:</b> Determine users within System group.	
<b>Objective / Subjective:</b> An objective test based only by group number membership.	
<b>Compliance:</b> Any user who is not directly responsible for the system's administration should not be a member of a system group. System group is defined by the first of two UIC	
<b>Testing:</b> To perform this test, have system manager run the sysgen utility by typing: <code>\$ mcr sysgen or run sysgen</code> <code>SYSGEN&gt;set/out=sysgen_audit.lis</code> <code>SYSGEN&gt;show/all</code> <code>SYSGEN&gt;EXIT</code> Then, you may view the sysgen_audit.lis text file using the edit utility of choice. Search for the parameter “MAXSYSGROUP=x”. That number represents the highest group number (within the UIC number) that represents automatic system manager privileges. Any user who is created in some group within 1 to x empowers them to system manger. To determine if anyone falls within that limit, have the system manager type the following command: <code>show [1,*]/full</code> <code>show [2,*]/full</code> <code>show [3,*]/full</code> ...and so on until reaching the MAXSYSGROUP number.	

**Risk:** The SYSGEN utility displays several parameters. Quite possibly most have not been altered since installation, but since many login parameters directly impact how the

process handles unauthorized persons as well as how it defines persons by their group membership, it is important to review the relevant SYSGEN login parameters. The parameter applicable for this particular test is the MAXSYSGROUP=x, where x represents the highest group number in which membership equates the user to system manager. Typical values are between 1 and 10. Consequences of an ordinary user belonging to such a low group number spells potential trouble.


### Item 12: CTRL-Y During Login

<b>Reference:</b>  <b>4</b>	<b>Risk (6)</b> = Likelihood (Suspect) + Consequences (Significant)
<b>Control Objective:</b> Determine if user may break the login session using CTRL-Y.	
<b>Objective / Subjective:</b> A binary test to execute.	
<b>Compliance:</b> If a user may press CTRL-Y (break) during the login session and be placed at a command line, then the system is not compliant.	
<b>Testing:</b> To perform this test, have system manager return to login prompt by typing at prompt "SET HOST 0" (that's zero, not the letter 'o'). They may also shorthand type "SET H 0" Upon login, have system manager type any existing user name –then press CTRL-Y. System may simply report "Interrupt" in reverse text.	

**Risk:** The keyboard combination Control-Y is a hard break to the process quoted in research material as a method of breaking to a command line: "Aborts current command or program immediately (interrupt system)." Interviews confirm this definition; it is a violent way of interrupting the process at hand and could be used maliciously.

The likelihood of this working 'as advertised' may depend on the version of VAX VMS installed and running. The consequence of someone being able to break out of the process is that the person may be able to then execute DCL (Digital Command Line) scripts without the restrictions normally placed by permissions.

### Item 13: Welcome message

<b>Reference:</b>  <b>3</b>	<b>Risk (2)</b> = Likelihood (Unlikely) + Consequences (Negligible)
<b>Control Objective:</b> Determine the proper use of a SYS\$WELCOME message.	
<b>Objective / Subjective:</b> The test is binary, pending the existence of the message.	
<b>Compliance:</b> This test is if the message appears when first logging in. The test can be considered a pass if the message states at least minimal warning to unauthorized persons. Depending on importance or use it additionally may be stating to users they're are subject to monitoring or legal consequences of unauthorized use.	


**Testing:** To perform this test, have system manager return to login prompt by typing at prompt “SET HOST 0” (that’s zero, not the letter ‘o’). They may also shorthand type “SET H 0”

The text appearing before the **Username:** prompt is the SYS\$WELCOME message.

**Risk:** Until recently the welcome message has been just that: a welcome message. Overtime it has become one of many necessary first lines of defense. Even the subject of legal battles, an inviting welcome message has unfortunately become what the defense requires for a victory.

On new systems the chances of the welcome message being altered is increasingly likely, but still on older systems, it remains a lower priority item that rarely gets looked at twice. The consequence of such a benign welcome message is equally subtle, but as mentioned above in setting precedent legal cases for the defense, it’s important enough to change. Although it is probable some hacker has specifically targeted a machine with a hospitable welcome message, it is highly unlikely that the hacker received any greater success due to the message alone. All told, minor consequence with decreasing likelihood make the total risk minimal but worth noting.

#### Item 14: Login (Attempted) Break-in Limit


<b>Reference:</b>  <b>1</b>	<b>Risk (6)</b> = Likelihood (Doubtful) + Consequences (Major)
<b>Control Objective:</b> Determine the value for LGI_BRK_LIM (nominal $\leq 5$ attempts).	
<b>Objective / Subjective:</b> Binary test.	
<b>Compliance:</b> If the limit to number of attempted break-ins is equal to five or less, then compliance should be met.	
<b>Testing:</b> To perform this test, have the system manager take a new user and verify that the user has not logged on unsuccessfully in the amount of time necessary to clear bad attempts (LGI_BRK_TMO). Log on unsuccessfully enough times as to note in the Intrusion logs (via a different terminal). This number of times should also be verified by checking the SYSGEN utility for the value of LGI_BRK_LIM.	

**Risk:** The combination of a high number of allowed bad attempts before taking action and a short time-out period can make it difficult to monitor unsuccessful logins. The default values are 5 attempts and 300 seconds, but should be verified. The likelihood of someone being opportunistic depends directly on the particular group of users. System managers’ previous bad experiences can yield lower tolerances for those values.

The consequences of setting these values improperly are obvious: increased opportunities for intruders.

The intrusion logs can be checked at a command prompt by typing: show intru


#### Item 15: Password Field Limit Check

<b>Reference:</b>  <b>4</b>	<b>Risk (5)</b> = Likelihood (Unlikely) + Consequences (Major)
---	--

<b>Control Objective:</b> Determine if password field is 'limit checked.'
<b>Objective / Subjective:</b> Subjective as the number of characters to test as an excess is variable.
<b>Compliance:</b> System should, at the least not allow or, balk the input of an excess of characters for the password field.
<b>Testing:</b> To perform this test, enter a valid account for the Username: prompt. For the Password: prompt, attempt to enter no less than 257 characters.

**Risk:** While this may not be a legitimately structured 'buffer overflow' attack, it does represent one of many foolish actions taken by users. Hence, the likelihood of such an 'attack' happening is relatively low, but the consequences of an unwelcome system response could carry major costs. Therefore, the total risk is a low to medium.


#### Item 16: Field Service (Provided Account) Disabled

<b>Reference:</b> 	<b>Risk (7) = Likelihood (Suspect) + Consequences (Major)</b>
<b>Control Objective:</b> Determine if the 'Field Service' account, provided upon installation has been disabled or at least had vendor supplied password changed.	
<b>Objective / Subjective:</b> Test is objective.	
<b>Compliance:</b> System provided 'Field' account should minimally have password changed from default ('service'). Also, in order to consider compliant or pass, the account should be disabled unless system manager contends account is used often.	
<b>Testing:</b> To perform this test, have system manager at UAF prompt type: sh field The account should show the word 'Disuser' underneath the owner or account name. Despite what the setting shows, the system manager should still attempt to log on using the proper credentials for the Field user.	

**Risk:** The Field Service account is one of the best known default accounts on any OpenVMS system. Coincidentally, the Field Service account is one of the ways to differentiate between VAX VMS and VMS on Alpha hardware. By default, on a VAX system, the Field service account's username is "field," while on an Alpha, the Field Service account's username is specified by the system manager, i.e. Joe, Sarah, etc.

The probability of the Field Service account being disabled is unknown across all VMS systems, but the consequences of having such a popular and powerful account available for abuse are severe. The risk for this account being available is relatively high.


#### Item 17: SYSTEST & SYSTEST\_CLIG (Provided Accounts) Disabled

<b>Reference:</b> 	<b>Risk (6) = Likelihood (Doubtful) + Consequences (Major)</b>
<b>Control Objective:</b> Determine if the 'SYSTEST' accounts, provided upon installation has been disabled or at least had vendor supplied password changed.	


<b>Objective / Subjective:</b> Test is objective.
<b>Compliance:</b> System provided 'SYSTEST' accounts should minimally have password changed from default passwords. Also, in order to consider compliant or pass, the accounts should be disabled unless explicitly required.
<b>Testing:</b> To perform this test, have system manager at UAF prompt type: sh [*,*]/brief The accounts should be near the top and show the word 'Disuser' underneath the owners.

**Risk:** Unlike the Field and the System (system manager) account, the SYSTEST accounts are not installed by default on just any OpenVMS system. This is a VAX VMS specific vulnerability. The probability of the accounts being disabled is probably slim. The consequences of having such powerful accounts available for abuse are major. The risk for this accounts being available is relatively high.

### Item 18: Login Timeout Period

<b>Reference:</b>  <b>1</b>	<b>Risk (6)</b> = Likelihood (Doubtful) + Consequences (Major)
<b>Control Objective:</b> Determine if value for LGI_PWD_TMO is reasonable.	
<b>Objective / Subjective:</b> Binary test considering quantitative value for compliance.	
<b>Compliance:</b> The value for timing out a login session should not be long at all considering a user is either entering in their password or not. There should not be a "gray area" of too long. Therefore, by "best practices" the value shall be set for 30 seconds.	
<b>Testing:</b> To perform this test, have system manager run the sysgen utility by typing: \$ mcr sysgen or run sysgen SYSGEN>set/out=sysgen_audit.lis SYSGEN>show/all SYSGEN>EXIT Then, you may view the sysgen_audit.lis text file using the edit utility of choice. Search for the parameter "LGI_PWD_TMO=y" The number in place of 'y' is the value in seconds that the login process will wait before timing out.	

### Item 19: Alert of Reading/Altering of UAF file


<b>Reference:</b>  <b>2</b>	<b>Risk (6)</b> = Likelihood (Doubtful) + Consequences (Major)
<b>Control Objective:</b> Determine if the administrator is made aware of any modification to the UAF file. This critical file is the equivalent to the NT SAM database or the password shadow file in *nix.	
<b>Objective / Subjective:</b> Binary test.	

**Compliance:** If simply “touching” the file may provide any alert –both immediately to the console and to a log, that conforms as a pass.

**Testing:** To perform this test, have system manager interact with the file in the least intrusive way. The file is least intruded upon -but still accessed- by simply typing: “sh (any user)” at the command prompt.  
Be aware of console messages soon following.

**Risk:** Without immediate alerts or logged entries, access to arguably the most important file on a VMS system would be untraceable and go unnoticed. The likelihood of anyone touching these files as a result of trying to touch any other file is extremely high. The consequences are, of course, great if privileges allow the file to be modified. Therefore the risk to a specific file necessitates that any access or attempts to access it be logged and available to the system managers.

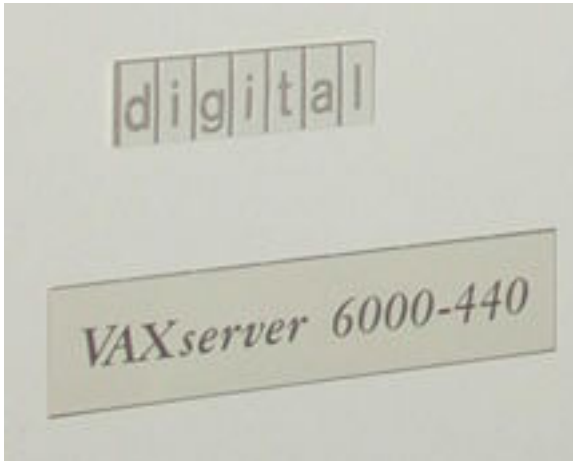
## Item 20: Prey Accounts

<b>Reference:</b>  <b>7</b>	<b>Risk (7)</b> = Likelihood (Suspect) + Consequences (Major)
<b>Control Objective:</b> Determine if accounts have gone on for ‘too long’ without use.	
<b>Objective / Subjective:</b> This subjective test requires the objective honesty of the system manager.	
<b>Compliance:</b> Complying with this control objective relies on the ability to understand that would be hackers are wise not to create new accounts but rather are prone to simply modify existing ones.	
<b>Testing:</b> To perform this test, have the system manager print out full output for all users. Typing sh [*,*]/full is the command to display all user output. The important value is check is the <b>Last Login</b> seen on the left margin. An example can be seen in the SYSTEM MANAGER account screenshot in Appendix 3.	

**Risk:** The ‘prey account’ is an account that could be used undetected. This is because it is already a legitimate account. The likelihood that someone wishing to gain an account would use an abandoned or rarely used account rather than create a new one is very high indeed. The consequences, of course, include the delayed notification or the obscurity that the account’s audit trail leaves behind. The risk of letting old accounts go without disabling or deleting can be high. This is especially true with VMS, since the account is the key to gaining system resources, and it seems all ‘blackhat’ instructive texts the author researched pointed toward such accounts.



## Section 3: Conducting The Audit



The subject of our audit was originally a Digital VAXserver 6000-440

Since its purchase, estimated in 1989, the 6000-440 has been upgraded to a VAXserver 6000-630 (630 represents the faster '600-class' CPU boards with 3 CPUs) CPU board upgrade estimated in 1995.



## **Top 10 Checklist items**

After conducting the audit along with the system manager, we identified the ten most noteworthy items audited.

### **Welcome message (item 13)**

Upon completing connection to the cluster node, we are greeted with the message:

Welcome to OpenVMS (TM) VAX Operating System, Version V7.1

Immediately afterward the Username: prompt appears. The SYSS\$WELCOME message is therefore lacking any sort of notice or forewarning of monitoring. Perhaps an improved message could be placed to ‘scare off the honest folks’ as one might say. Certainly the best alternative is a message stating that the user can assume he or she may be subject to monitoring and perhaps that any unauthorized actions can and will lead to legal action, resulting in jail terms to the fullest extent of the law –or something along those lines.

### **Password Field Limit Check (item 15)**

At the system console, we break out of an existing session with the command “set h 0” (remember: this is perfectly acceptable shorthand for set host 0) –which is asking the machine to ‘go out to the network and return again,’ setting us up for a new session.

MSE4> set h 0

Welcome to OpenVMS (TM) VAX Operating System, Version V7.1

Username: system

Password:

User authorization failure

Username: system

Password:

Error reading command input

Terminator not seen

%REM-S-END, control returned to node MSE4::

MSE4>

You see from the screen output that we tried entering the incorrect password for the system account twice. The first time was a ‘premature’ carriage return and so we were prompted again. The second time the node responded with error messages mainly stating there was no terminator (carriage return) when reaching the limit of the field.

### **“System-level” users by group membership (item 11)**

The data below is a slightly edited and reformatted output of all users within UIC groups 1 through 8. The number 8 was picked since it is the value of MAXSYSGROUP, representing the system-level users.



As you can see from the output, five such accounts exist, which is relatively good. The system manager accounts must remain active for obvious reasons. The troubling part involves the three remaining accounts –all installed by default. Only one of the three is disabled (noted by the Flag “Disuser”). The other two, the default Field Service account and the basic SYSTEST account could still be disabled to prevent potential abuse. That leads us into the next two items of note.

SYSTEM MANAGER	SYSTEM	[1,4]	SYSTEM	All	4	SYS\$SYSROOT:[SYSMGR]
SYSTEM MANAGER	PAULA	[1,4]	SYSTEM	All	4	SYS\$SYSROOT:[SYSMGR]
SYSTEST-UETP	SYSTEST	[1,7]	SYSTEST	All	4	SYS\$SYSROOT:[SYSTEST]
SYSTEST-UETP	SYSTEST_CLIG	[1,7]	SYSTEST	All	4	Disuser
FIELD SERVICE	FIELD	[1,10]	FIELD	All	4	SYS\$SYSROOT:[SYSMAINT]

### Field Service (Provided Account) Disabled (item 16)

The system manager and I first logged into the system with his privileges. He then typed in the command `sh field` at the command prompt.

You can see from the data below that the user “field” has a UIC group of 1. From previous audit items, we know that is within the MAXSYSGROUP value of 1 – 10. Therefore the account is fully functional as a system manager account. This is expected and warranted as the Field Service technicians often require such privileges. But typically the account is not required until external maintenance is needed and a call is logged. In the meantime, the account should be disabled. In OpenVMS, a disabled user is noted by the ‘Disuser’ flag. That flag may also be seen if an account has been tried too often unsuccessfully.

Owner	Username	UIC	Account	Privs	Pri	Directory
FIELD SERVICE	FIELD	[1,10]	FIELD	All	4	SYS\$SYSROOT:[SYSMAINT]

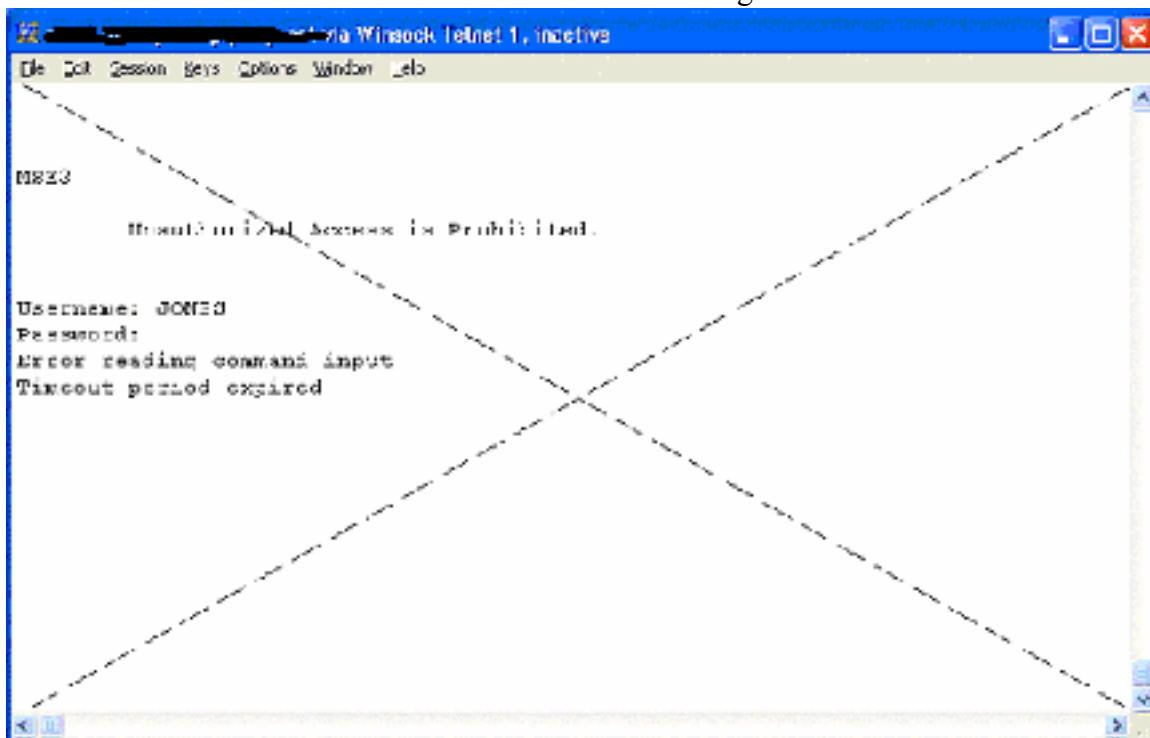
### SYSTEST & SYSTEST\_CLIG (Provided Accounts) Disabled (item 17)

The SYSTEST accounts create load on a system to simulate users and they are only created and used on VAX hardware. That means they are best used typically during initial installation or major hardware/software changes. From the output below we see that one of these accounts has been disabled. For maximum safety, both of these and other unused default accounts should have the flag “Disuser” set.

Owner	Username	UIC	Account	Privs	Pri	Directory
SYSTEST-UETP	SYSTEST	[1,7]	SYSTEST	All	4	SYS\$SYSROOT:[SYSTEST]
SYSTEST-UETP	SYSTEST_CLIG	[1,7]	SYSTEST	All	4	Disuser

### Login Timeout Period (item 18)

Below is a screenshot of the terminal window after having waited for 30 seconds:



You can see from the terminal screen that VMS assumes nothing but lack of input:

“Error reading command input / Timeout period expired”

Also note that of the two error messages, the first one “Error reading command input” is the same as the first error message when trying to give a 257 character password. Then only the second error gives indication to why the user was unable to complete the input.

© SANS Institute

## Sufficient user privileges (item 10)

Upon testing item 9 with the command lines given, it was discovered that the users had various privilege levels. Some of the output can be seen below:

Owner	Username	UIC	Account	Privs	Pri	Directory
Dick Brown	BROWN	[23,12]	CC_Y2K	All	4	USRD1:[BROWN]
Richard CHINO	CHINO	[23,14]	CC_Y2K	All	4	USRD5:[CHINO]
Bob LAMB	LAMB	[23,15]	CC_Y2K	Normal	4	USRD6:[LAMB]
George Holme	HOLME	[23,24]	CC_Y2K	All	4	USRD1:[HOLME]
Corinne Noore	NOORE	[23,112]	CC_Y2K	Normal	4	USRD3:[NOORE]
Mike Alum	ALUM	[23,116]	CC_Y2K	Normal	4	USRD6:[ALUM]
Claus Vest	VEST	[23,122]	CC_Y2K	Normal	4	USRD5:[VEST]
Laurence Oman	OMAN	[23,124]	CC_Y2K	Normal	4	USRD5:[OMAN]
Ken Camp	CAMP	[11,104]	CC_Y2K	Normal	4	USRD3:[CAMP]
Dom Dell	DELL	[11,106]	CC_Y2K	Normal	4	USRD3:[DELL]
Susan Hogen	HOGEN	[21,3]	CC_Y2K	Normal	4	USRD4:[HOGEN]
Lee Munge	MUNGE	[21,4]	CC_Y2K	Normal	4	USRD1:[MUNGE]
Lee Melvin	MELVIN	[21,6]	CC_Y2K	Normal	4	USRD4:[MELVIN]

The 13 users in this screen output is a decent 10% representation of the total collection of accounts on the system. Please note there are three displayed with “All” privileges as opposed to ‘Normal.’ Actually, exactly 19 out of the 114 total accounts have ‘All’ displayed here. ‘All’ indicates that they have the SETPRV privilege. The reference in Appendix 2 demonstrates that the SETPRV privilege enables a user to grant himself any privilege. So, in short it is the ultimate right. The users showing ‘All’ may have it as an Authorized Privilege (not set by default), but it that is only a single command away becoming active. All other privileges would return as ‘Normal.’

The display above was shown by the command: `sh [*,*]/br`

The display screen to see all privileges per user is done by: `sh [*,*]/full`

The above information would occupy dozens of pages, so we keep it concise (/br=brief) distinguishing only those who have the SETPRV privilege, granting them any privilege and establishing them as system-level users. An example of a full user profile display can be seen with the System Manager’s account in Appendix 3.

## CTRL-Y During Login (item 12)

The keystroke combination CTRL & 'Y' is an aggressive way to break a process and/or session. Both the research and interviews confirm that there is a specific flag set on a per user basis to disable the CTRL-Y functionality.

Upon inspection of the Default User account, we see that it is kept enabled by default for each user, unless it is revoked by the system manager at a later time. I found that all of the users inspected still had the ability. It does indeed provide an ability to stop a process, providing some debugging ability if the need arose. However, we needed to find out if it could do equal interruption to the login process, as many online 'hack' forums and audit checklists cautioned against.

The system manager brought up a login screen and following is what ensued:

```
MSE4> set h 0
```

Welcome to OpenVMS (TM) VAX Operating System, Version V7.1

Username: system

Password:

**Interrupt**

User authorization failure

Username:

**Interrupt**

Username:

**Interrupt**

Are you repeating ^Y to abort the remote session on node 0?

So, after we tried to interrupt the process three times, the system balked and asked the above question. Yet VMS did not convey the system is ready to release command of the process. Nonetheless, the Flag "DisCtlY" should be set to comply with Objective 12.

## Alert of Reading/Altering of the UAF file (item 19)

I noticed that a number of different messages come up when we add a user, alter a current user or even so much as list details of users. All messages, though different in naming seem to indicate activity to a single file: SYSUAF.DAT (the User Authorization File file). This is the file that contains all user profile information: quotas, passwords, group identity, etc...)

Our activities are recreated here:

### Upon adding a user

To add a user, we first check to make sure the user does not already exist:

```
UAF> show jones
%UAF-W-BADSPC, no user matches specification
```

As you can see, we are currently in the UAF utility. Upon checking for the user, the UAF file reports 'no user match'

We want to add a user by using the UIC account values 255 & 1, so we check for those already existing:

```
UAF> show [255,1]
%UAF-W-BADSPC, no user matches specification
```

No such user exists. We then go ahead and add a dummy user named Jones:

```
UAF> Add jones/owner="Jeff Parker"/uic=[255,1]/device=usrd6: -
/dir=[jones]/account=y2j/pass=snafu/expiration=30-apr-2002
%UAF-I-PWDLESSMIN, new password is shorter than minimum password length
%UAF-I-ADDMSG, user record successfully added
%UAF-I-RDBADMSGU, identifier JONES value [000255,000001] added to rights
database
```

This is where it got interesting with the error messages! Notice that three distinct alerts come up:

1. Our password was too short.
2. Our user was successfully added (soon to bring up the next audit item)
3. Our user Jones was added to the database.

## Minimum Password Length (item 1)

As you can see from our example, we found that the UAF file was updated with a new user, UIC and password –despite the fact that the password did not meet minimum password length criterion. Again, it can be recreated by adding a different user:

```
UAF> Add smithers/owner="Jeff Parker"/uic=[255,5]/device=usrd6: -
/dir=[smithers]/account=y2j/pass=five/expiration=30-apr-2002
%UAF-I-PWDLESSMIN, new password is shorter than minimum password length
%UAF-I-ADDMSG, user record successfully added
%UAF-I-RDBADMSGU, identifier SMITHERS value [000255,000005] added to rights
database
```

This user ("Smithers") has been created with a password of four characters, while the minimum password length is six. And although the Default User sets these parameters to enforce the minimums, the user is still created and its record is added to the rights database file (SYSUAF.DAT = the UAF file).

We should note that the user, upon logging on, will be forced to change his or her password and at that time the user will not be permitted to continue without a password meeting minimum length.

```

net via Winsock Telnet
[?] Edit Session Keys Options Window Help
CAT>
CAT>
CAT>
CAT> sh default

Username: DEFAULT                                Owner:
Account:                                          UID:  [300,200]  (100_YES,DEFAULT)
CLI:      CLI                                     Period:  DCLCABLES
Default:  CSRD4:[USER]
ISICMD:
Flags:
Primary days:  Mon Tue Wed Thu Fri
Secondary days:                               Sat Sun
No access restrictions
Expiration:      (none)                      Badminimum:  6      Login Fails:    0
Inadlifetime:   90 01:00                     Watchdog:      (pre-expired)
Last login:     (none) (interactive),         (none) (non-interactive)
Maxjobs:        0  Fillin:  100  Bypass:  66000
Maxacctjobs:    0  shrfillin:  20  Bypass:  0
Maxdetach:      0  UOlim:  10000  UQuotes:  4056
Wclm:          10  UOlim:  100  Wdel:  4056
Pils:          4  ASYlin:  325  MQuo:  512
Quoprio:        0  ICLim:  200  Mextents: 16384
CPU:            (none) Enclm:  200  Bypass:  5000
Authorized Privileges:
  NETMXX  TMEEXX
Default Privileges:
  NETMXX  TMEEXX
CAT>

```

© SANS Institute 2000

---

## Section 4: Follow-Up

### Executive Summary

This audit targeted the authentication methods used by VAX OpenVMS. The audit was conducted on a VAXserver 6000-630 running OpenVMS 7.1. System roles were defined to the auditor as follows:

- interface running a corporate application, gateway to backend VAX
- storage repository for approximately 100 users
- a small number of secondary uses, including native mail and newsgroup

The audit process objectives were specifically created from a framework commonly used for auditing. While the framework seeks to encompass all that is auditable, covering the broadest range from procurement to monitoring, our goal here was specifically dealing with aspects of the system's security: authentication, access and identification.

Given those facets of security to audit, our aim was to verify those security components operate with the characteristics intended by the original developers and maintainers.

It is believed that all three aspects and the above aims were sufficiently met and with clear results. In addition to the results, a checklist is also submitted for future audits to contrast with the baseline now done. Items found to be satisfactory are commented on and clarified as to why they met expectations. Those items found not to be in compliance were described in terms of both the reasoning behind the grade and the recommendations given to achieve an improved grade at a later appraisal.

### Audit Findings

#### *Findings In Bulleted Form*

The audit findings are first given as pass/failure and their associated risk. Risk is determined as a function of the possible consequences and the probability of those consequences coming to realization. Risk is expressed numerically from 1 to 10, with '1' representing the lowest and '10' the highest. The numeral is in parentheses after the audit item. For example: Password minimum length (6) = a risk rating of 6 out of 10.

The findings are again given in more detail following the abridged summary.

The Authentication audit shall be broken down into the following sub-areas:

- I. Password related
- II. Restriction of user privileges
- III. Restricting functionality directly related to the login process

## I. Password Related

The first criteria are related to strength and endurance, and then we approached the authentication process from a procedural standpoint.

### Password Criteria

The following shows what criteria were audited and how they fared:

- minimum age (5) *-non-compliant (n/a on OpenVMS)*
- maximum age (6) *=pass (equal or less than 90 days)*
- minimum length (6) *-non-compliant (six characters  $\leq$  eight)*

### Password Process

The following shows what criteria were audited and how they fared:

- bad user/password distinction(3) *=pass (no difference in messages)*
- forced initial password change(5)*=pass (password must change on 1<sup>st</sup> login)*

## II. Restriction of user privileges

How users' privileges were maintained and managed from creation through lifetime of account was also of strong interest in this audit.

The following shows what criteria were audited and how they fared:

- default user privileges (6) *=pass ('default user' account exists with low privs)*
- system-level user group (7) *=pass (no ordinary users with UIC group  $\leq$  8)*
- sufficient user privileges (6) *-non-compliant (several with elevated privileges)*
- default system accounts (3-8) *-non-compliant (default unused accounts active)*
- 'prey' accounts (7) *-non-compliant (dormant accounts exist enabled)*
- Field Service acct disabled (7) *-non-compliant (FS not disabled outside of use)*

## III. Restricting functionality directly related to the login process

How the system was configured to handle potentially harmful actions or intentions was the third focus area of the audit.

- CTRL-Y During Login (6) *=pass (no break to command line)*
- CTRL-Y disabled for default accounts(6)*-non-compliant (no DisCtlY in default user)*
- login timeout period (6) *=pass (timeout is  $\leq$  30 seconds)*
- proxy logins (6) *=pass (non-privileged proxy accounts exist)*
- password field limit check (5) *=pass (system stays in control)*
- welcome message (2) *-non-compliant (message needs update)*
- login attempted break-in limit (6) *=pass ( $\leq$  5 attempts; system locks account)*



## ***Findings in Detailed Form***

### **Password Maximum Age -Passed**

The maximum age setting for this system was found to be 90 days or less, complying with this particular audit item.

The age can be found by typing at the UAF command prompt:

UAF> sh default/full

The value for the parameter PWDLIFETIME is represented as XX YY:ZZ, where XX is in days, YY:ZZ is in minutes:seconds. Our VMS system audited showed a value of 90 00:00 –in other words, 90 days.

### **Password Minimum Length -Non-compliant**

As you can see from our example, we found that the UAF file was updated with a new user, UIC and password –despite the fact that the password did not meet minimum password length criterion. Again, it can be recreated by adding a different user:

```
UAF> Add smithers1
/owner="Jeff Parker"/uic=[255,5]/device=usrd6: -
/dir=[smithers]/account=y2j/pass=five/expiration=30-apr-2002
%UAF-I-PWDLESSMIN, new password is shorter than minimum password length
%UAF-I-ADDMSG, user record successfully added
%UAF-I-RDBADDMSGU, identifier SMITHERS value [000255,000005] added to rights
database
```

This user (“Smithers”) has been created with a password of four characters, while the minimum password length is six. Although the Default User sets these parameters to enforce the minimums, the user is still created and its record is added to the rights database file (SYSUAF.DAT = the UAF file).

We should note that the user upon logging on will be forced to change his or her password and at that time the user will not be permitted to continue without a password meeting minimum length.

**RISK:** The ability to ‘crack’ or overcome a password is directly related to its strength. The weaker the password is, the faster the unauthorized user’s time to overcome it. If an unauthorized user does manage to make use of a valid account, then their use of that account is only limited by the restrictions placed on the legitimate employee. There should be a huge difference placed on access between employees and outsiders. Unfortunately, that difference lies in the strength of the password.

### **AUDIT RECOMMENDATION:**

Simply increase the password minimum length by 2 characters. It is a relatively simply and fast procedure that the system manager can execute in a few minutes.

Most important is the recommended user training your company should implement. Users made to understand *why* a longer password is being enforced are much more likely to adhere to and respect the slight inconvenience it places on them.

## **Bad user/password distinction -Passed**

This test was completed by successfully carrying out two steps:

We first demonstrated at the UAF prompt that a particular user name did not already exist. This can be done at the UAF prompt by typing (for example) 'sh Johnson' and noting the expected 'no such usr' message. Then we attempted to log on to the previously created Johnson account with a bad password. We then note the system response. Second, we log on as 'userabc' or an otherwise unknown account: Note the system response –if the system responds with exactly alike messages, then no 'intelligence' is passed on to a potential intruder. The intruder can not tell whether or not he has a valid account but a bad password, or an invalid account. This increases the amount of time the intruder spends logging in.

## **Forced Initial Password Change -Passed**

In performing this test, we created a new user with a known initial password (set by the system manager). We logged out and logged in as a new user. As the new user, we were prompted to change. We tried entering the exact same initial password, but the system would not allow it. We were prompted again until we entered a new password entered, therefore the test was successful.

Also the UAF prompt, the system manager typed: 'sh default' –listing all default user parameters (utilized by system upon new user creation). If the "Pwdchange" parameter shows a setting of "(pre-expired)" (without quotes) –that also verified that the system would force a new user logging on for the first time to change his or her password.

## **Default user privileges -Passed**

The test was to determine whether any privileges granted by default were above standard NETMBX and TMPMBX. By typing 'sh default' at the UAF prompt, we were able to see that the default user account (used as a template for creating new users) had only the two standard, low-level privileges assigned to it: NETMBX & TMPMBX. Those should be both under "Authorized Privileges" and "Default Privileges"

## **"System-level" users by group membership -Passed**

In performing this test we ran the SYSGEN utility by typing:

```
$ mcr sysgen or run sysgen
SYSGEN>set/out=sysgen_audit.lis
SYSGEN>show/all
SYSGEN>EXIT
```

We then viewed the sysgen\_audit.lis text file using an edit utility.

Upon finding the parameter "MAXSYSGROUP", we were able to determine the number representing the highest group number (within the UIC number) that represents automatic system manager privileges. Any user who is created in some group within 1 to x

empowers them to system manger. We determined if anyone falls within that limit, by typing the following commands:

```
show [1,*]/full
```

```
show [2,*]/full
```

```
show [3,*]/full
```

...and so on until reaching the MAXSYSGROUP number (8 in our case).

We found that only five accounts are created within this set of groups (1-8). All accounts were system created and so no end users reside within these groups.

## **Sufficient user privileges –Non-compliant**

This test was to determine whether users are granted privileges higher than necessary to perform daily job functions.

We performed this test by listing all the operators, backup operators, group managers and other IT staff. We went over the individuals with the system manager to oversee their privileges, keeping in mind their duties.

The second list was for evaluating all the users not on the “system manger-level” Running the Authorize utility we typed: `sh [*,*]/page=save/search=SYSPRV` (instead of SYSPRV, you may insert any privilege you wish to search for)

We evaluated all users not on the system-level list and found several with privileges other than the two standard privileges (NETMBX & TMPMBX). Especially of interest are the users with the SETPRV privilege, which allows them to grant themselves any privilege they wish. Obviously this takes a great deal of trust on the system manager’s part and can be slightly political if attempting to manage too tightly.

**RISK:** Privileges granted in OpenVMS, like in all operating systems, give the system managers (root, admins, etc) ultimate authority and control. That demands trust on the side of management to believe that control will not be abused. Assuming trust is given correctly, the users who should not be granted that control are the main target of this audit item. Likelihood is difficult to estimate since privileges granted for a one-time occurrence may or may not have been removed at a later date. The consequences vary in part due to the different privileges, but on average are significant. They range from acquiring unauthorized system resources to achieving system-level privileges.

**AUDIT RECOMMENDATIONS:** We recommend that the system manager reevaluate all employees with system-level privileges and determine whether such privileges are absolutely necessary for their job functions. If still granted, control management of any changes to the system must be more closely monitored and documented. If system-level privileges are necessary, then risk could still be reduced by host-based auditing to track specific use of such privileges. Perhaps other measures should be explored to reduce the consequences, thereby reducing the overall risk.

## **Default System Accounts –Non-compliant**

We had set out to check/verify that the following system-generated accounts were either necessary or disabled:

guest / guest  
guest / <no password>  
default / default  
decnet / decnet  
operator / operator  
operations / operations  
field / field or service  
system / system or operator or manager

While many accounts were either essential (system, field, decent) or disabled (guest, operator, support, system\_clig) there are some that are still enabled and dormant –some beyond this list.

**RISK:** Several accounts exist by default in OpenVMS, though the passwords have been quite likely changed to whatever the system manager wanted. While some are not necessary (i.e. guest, operator), others are required for system operation (i.e. system, decnet).

The consequences of those necessary system-level accounts having a default or weak password would be dire. The ‘system’ account, for example, is what the system manager typically logs in as for day-to-day management. Being ‘system’ also means what it implies: the person has complete control of system resources allocation (CPU, memory, disk quotas, # of processes, etc.). There are no limits to misuse of that account.

**AUDIT RECOMMENDATIONS:** We strongly recommend that those accounts be reevaluated by the system manager for applications or connectivity requirements. If deemed unnecessary, then they should be disabled. If they are necessary, risk could be further reduced by mitigating the consequences should an account be compromised.

**COST:** There is no cost to simply disable dormant accounts.

## **Prey Accounts –Non-compliant**

By having the system manager print out a full output for all users, we were able to note if there is an imbalance of active/inactive users. By typing `sh [*,*]/full` you can display this information. The important value to check is the **Last Login** seen on the left margin. An example can be seen in the SYSTEM MANAGER account screenshot in Appendix 3.

**RISK:** As opposed to dormant system accounts, the dormant user accounts include the human factor which may make it more difficult to distinguish ‘normal’ activity. The ‘prey account’ is an account that could fly under the radar screen if used. This is because it is already a legitimate account. The likelihood that someone wishing to gain an account would use an abandoned or rarely used account verses create a new one is very high indeed. The consequences, of course, include the delayed heads-up or the obscurity that the account’s audit trail leaves behind. The risk of letting old accounts go without disabling or deleting can be high. This is especially true with VMS, since the account is

the key to gaining system resources and it seems all 'blackhat' instructive texts the author researched pointed their readers toward such accounts.

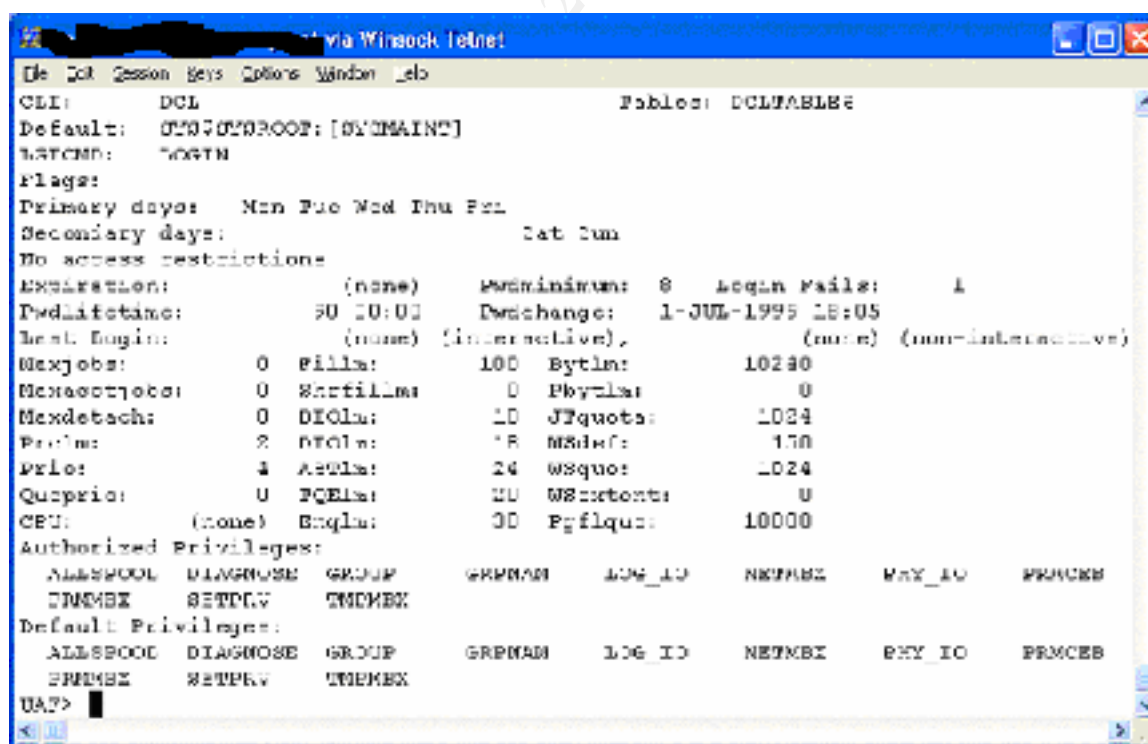
**AUDIT RECOMMENDATIONS:** We recommend approaching users who have not logged on to their VMS account in the past 180 days and discussing with them about the impact disabling their account would have on them. User awareness would greatly benefit the internal community as well in this task.

**COST:** There is a slight cost to implementing user training, but it has shown in many instances to be a well-spent area for improving security. Employees are the best first line of defense.

### Field Service (Provided Account) Disabled –Non-compliant

The Field Service account is probably the most well-known account and therefore may be the most frequently attempted by unauthorized users. Unfortunately, the default password from installation is also well known, leaving a popular vulnerability often exposed. In the case of this VMS system, the password is maintained well. However, the audit item is flagged as non-compliant since 'Best Practices' recommends that the account be disabled when not in use.

Determining if the account is disabled or active can be done by typing 'sh field' at the UAF prompt. A screen shot of the Field account on this server is shown below:

A screenshot of a VMS system window titled "via Winsock telnet". The window shows the command prompt "UAF>" and the output of the "sh field" command. The output displays various account settings for the "field" user, including "Primary days", "Secondary days", "No access restrictions", "Expiration", "Pwldlifetime", "Last Login", "Maxjobs", "Maxdetachs", "Prio", "Quoprio", "CPU", "Authorized Privileges", and "Default Privileges". The "field" user is shown as active, with a "Pwldlifetime" of 30 days and a "Last Login" of 1-JUL-1995 18:05. The "Authorized Privileges" section lists "ALLSPool", "DIAGNOSE", "GROUP", "GRNAM", "LOG\_ID", "NETMBZ", "PRY\_IO", and "PRMCEB". The "Default Privileges" section lists "ALLSPool", "DIAGNOSE", "GROUP", "GRNAM", "LOG\_ID", "NETMBZ", "PRY\_IO", and "PRMCEB". The "UAF>" prompt is visible at the bottom of the window.

The lack of a DisUser Flag demonstrates that the account is indeed active.

The above information was also seen by showing the default Field Service account directly:

Owner	Username	UIC	Account	Privs	Pri	Directory
FIELD SERVICE	FIELD	[1,10]	FIELD	All	4	SYSS\$SYSROOT:[SYSMAINT]

**RISK:** The SETPRV default privilege (and, of course, authorized privilege) must be given to the Field Service account to do their job. That privilege is also the main point of inherent risk of leaving this account enabled.

**AUDIT RECOMMENDATIONS:** We would recommend that this account be simply disabled until it is needed.

**COST:** There is no cost to disabling this account.

## CTRL-Y During Login –Passed

We tried to perform this test by attempting to break out of the login process utilizing the keyboard combination Control & ‘Y’.

The keystroke combination CTRL & ‘Y’ is an aggressive way to break a process or session. Both the research found and interviews confirm that there is a specific flag set on a per user basis to disable the CTRL-Y functionality.

However, we needed to find out if CTRL-Y could also interrupt the login process, as many online ‘hack’ forums and audit checklists cautioned against.

The system manager brought up a login screen, which returned the following display:

```
MSE4> set h 0
```

```
Welcome to OpenVMS (TM) VAX Operating System, Version V7.1
```

```
Username: system
```

```
Password:
```

```
Interrupt
```

```
User authorization failure
```

```
Username:
```

```
Interrupt
```

```
Username:
```

```
Interrupt
```

```
Are you repeating ^Y to abort the remote session on node 0?
```

So, after three times trying to interrupt the process, the system balked and asked the above question: not a result suggesting the system is ready to release command of the process. Nonetheless, the Flag “DisCtlY” should be set to comply with Objective 12.

### **CTRL-Y for Default User profile –Non-compliant**

An inspection of the Default User account, reveals that it is kept enabled by default. Consequently, each user retains the ability unless it is revoked by the system manager at a later time. I found that all of the users inspected still had the ability. CTRL-Y does indeed provide an ability to stop a process, providing some debugging ability if the need arose.

Although the break-session key combination fails to break the login process, this function is still viewed as a liability relative to “best practice.”

**RISK:** The ability to break a process is viewed as a liability. If someone were able to reach a command line without being hampered by the restrictions normally placed on his user account, he would then be able to perform actions unmonitored and unlimited.

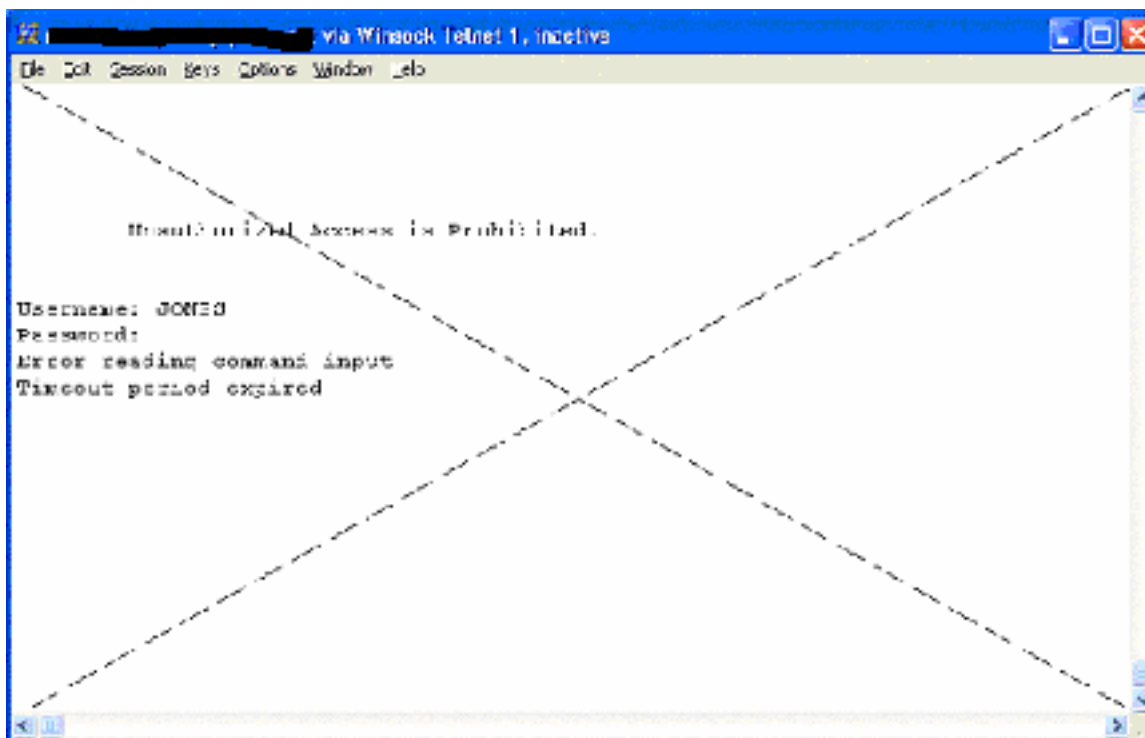
**AUDIT RECOMMENDATIONS:** We recommend having the system manager place the DisCtrlY flag on all users in order to restrict this “breaking” ability from the typical end user. If the user demonstrates the need for debugging or other purposes, then that is an issue for the system manager to decide.

**COST:** There is no additional cost to retracting this feature other than taking away the ability to break out of an abnormal process.

### **Login Timeout Period –Passed**

Below is a screenshot of the terminal window after having waited for 30 seconds:

© SANS Institute 2000 - 2002, Author retains full rights.



You can see from the terminal screen that VMS assumes nothing but lack of input: “Error reading command input / Timeout period expired”

Also note that of the two error messages, the first one “Error reading command input” is the same as the first error message that appeared when we earlier tried to exceed the password field limit. Only the second error gives indication to why the user was unable to complete the input.

### Password Field Limit Check –Passed

At the system console, we break out of an existing session with the command “set h 0” (remember: this is perfectly acceptable shorthand for set host 0) –which is asking the machine to ‘go out to the network and return again,’ setting us up for a new session.

MSE4> set h 0

Welcome to OpenVMS (TM) VAX Operating System, Version V7.1

Username: system  
Password:  
User authorization failure  
Username: system  
Password:  
Error reading command input  
Terminator not seen



```
%REM-S-END, control returned to node MSE4::  
MSE4>
```

You see from the screen output that we tried entering the incorrect password for the system account twice. The first time was a 'premature' carriage return and so we were prompted again. The second time the node responded with error messages mainly stating there was no terminator (carriage return) when reaching the limit of the field.

## Welcome message –Non-compliant

The following welcome message is typical and in an ideal world could be considered acceptable.

Welcome to OpenVMS (TM) VAX Operating System, Version V7.1

Unfortunately, by today's standards this is inadequate. Especially since legal precedents have been set that release defendants from criminal prosecution, it has become more important to include in any Welcome message statements declaring the system to be off-limits to unauthorized personnel.

Further wording should state that the user can assume he or she may be subject to monitoring and perhaps that any unauthorized actions can and will lead to legal action, resulting in jail terms to the fullest extent of the law –or something along those lines. Therefore, we recommended the SYSSWELCOME message be something such as:

"This is a <your company here> computer system. This computer system, including all related equipment, networks, and network devices (specifically including Internet access) are provided only for authorized personnel. These computer systems may be monitored for all lawful purposes, including ensuring that their use is authorized. During monitoring, information may be examined, recorded, copied, and used for authorized purposes. Use of this computer system, authorized or unauthorized, constitutes consent to monitoring of this system. Unauthorized use may subject you to criminal prosecution."

Of course, that is somewhat extreme, but still only half of what, for example, the government utilizes for their initial banner.

**RISK:** Until recently the welcome message has been just what its name implies: a message welcoming users. Even the subject of legal battles, an inviting welcome message has unfortunately become what the defense requires for a victory.

On new systems the chances of the welcome message being altered is increasingly likely, but still on older systems, it remains a lower priority item that rarely gets looked at twice. The consequence of such a benign welcome message is equally subtle, but as mentioned above in setting precedent legal cases for the defense, it's important enough to change. Although it is probable some hacker has specifically targeted a machine with a hospitable welcome message, it is highly unlikely that the hacker received any greater success due to the message alone. All told, minor consequence with decreasing likelihood make the total risk minimal but worth noting.

**AUDIT RECOMMENDATIONS:** We recommend that the banner at least convey a sense of “official use only” and “subject to monitoring.” With these two messages coming across to anyone wishing to access the machine, whether authorized or unauthorized, they will understand that they may not go unnoticed.

**COST:** The only cost involved is incurred when the message can be used against the company in a legal battle when a hacker uses it to demonstrate that he was “Welcomed” into the system.

## Compensating Controls

This section where reviews strategies to mitigate risk if the estimated costs were too high. However, in this environment, we find that there are very few areas that could not be remedied with little to no cost.

## Controlling the Factors of Risk

There is a different approach we would like to explore with compensating controls. In the best interest of lowering **Risk**, we need to explore what components determine risk. Risk is a function of the likelihood something can go wrong and the consequences of what will come about when it does go wrong.

We have noticed that the vast majority of these audit failures carry relatively less likelihood and bear heavier consequences than may be encountered at a different environment.

For example, of the X items that were non-compliant, the average value of Risk was 5.6. In preparation for any audit we calculate Risk from the numeric values we assign each item for both Likelihood and Consequence.

For this audit we found the averages for those non-compliant items were:

Likelihood=2.6

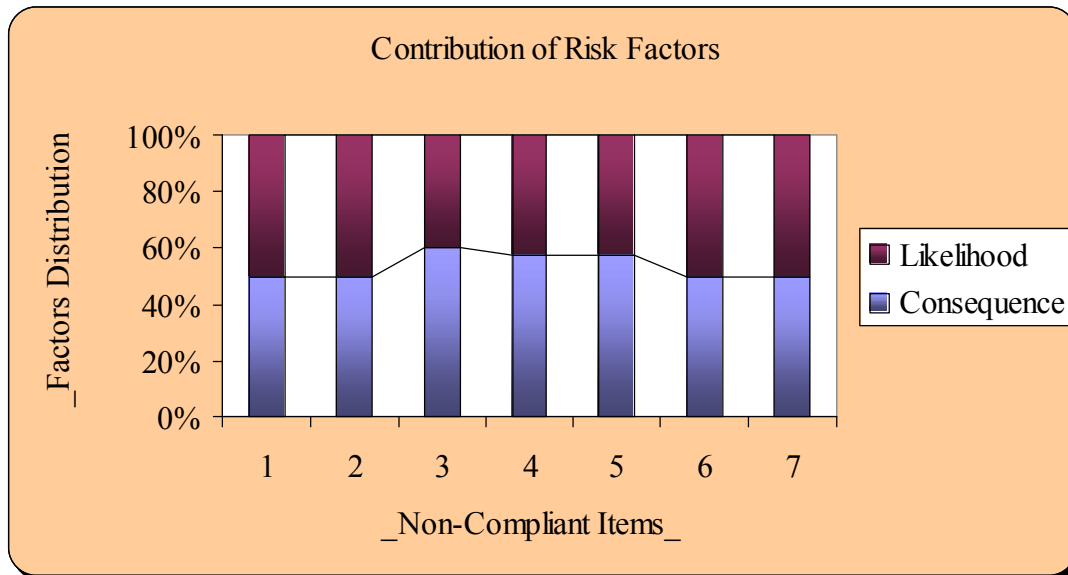
Consequence=3.0

## Graphing Compliance Opposed to Likelihood

It was observed that in this audit, items of average to high risk (4 - 8) were more due to relatively high consequence as opposed to the likelihood factor.

This can be seen more easily if we graph ‘Likelihood’ and ‘Consequence’ individually to determine how each factor contribute to overall risk.

Here we graphed the Risk taken from the seven non-compliant items:



We can see more clearly now that the consequences of something going wrong typically contribute more toward overall risk than the likelihood of it happening. Comparing this to an environment plagued with an abundance of risks –where most are of minor consequence yet are more prone to happening- our VMS environment seems to be more under control but of heavier concern. By graphing our data in such a way that we may see when a pattern exists, we are in a better position to identify “root cause” of an environment’s underlying issues –this could be ranging from poor procedure, training, a trend of ignoring ‘the minor things for major fires.’ All play a part in improving the overall quality of security (or the audit in general) of any location.

## Appendix 1: References



### Internet References

1. Lazurus, C. (2000, August 29). *Digital VAX/VMS Audit Program*. [WWW Document] URL: [http://www.auditnet.org/docs/vax\\_vms2.txt](http://www.auditnet.org/docs/vax_vms2.txt)
2. Oringel, J.L. (1993). *VAX/VMS Audit Program* [WWW Document]. URL: [http://www.auditnet.org/docs/vax\\_vms.txt](http://www.auditnet.org/docs/vax_vms.txt)
3. Beaver, T. (1995, August 18). *VMS HACK FAQ* [WWW Document]. URL: <http://web.textfiles.com/hacking/vms.txt>
4. Academic Computing and Instructional Technology Services. (1996, September). *VMS Pocket Reference List* [WWW Document]. URL: <http://www.utexas.edu/cc/docs/ccrl21.html>
5. Entity. (1989, September 12). *Beginners Guide to VAX/VMS Hacking* [WWW Document]. URL: <http://membres.lycos.fr/wir/vax-beg.txt>
6. Phrack Magazine. (2001, April 10). *Some Helpful VAX/VMS Utilities* (i45, p45-15) [WWW Document]. URL: <http://www.phrack.com/phrack/45/P45-15>



### Books, Magazines or Dissertations

1. Compaq OpenVMS Documentation (1997, October 2) Available at URL: <http://www.openvms.compaq.com:8000/ssb71/6346/6346p020.htm> (sec.12.2.2)
2. McMillan, R. *A Practical Exercise in Securing an OpenVMS System*. Queensland, Australia: Prentice Center. Also available as WWW Document URL: <http://nsi.org/Library/Compsec/openvms.txt>
3. SANS GSEC course material (Online)
4. SANS GCIH course material (Patriot SANS, Boston, MA 2000)

© SANS Institute 2000 - 2002, Author retains full rights.

## Appendix 2: All OpenVMS Privileges

### NORMAL PRIVS

MOUNT      Execute mount volume QIO  
NETMBX      Create network connections  
TMPMBX      Create temporary mailbox

### GROUP PRIVS

GROUP      Control processes in the same group  
GRPPRV      Group access through SYSTEM protection field

### DEVOUR PRIVS

ACNT      Disable accounting  
ALLSPOOL      Allocate spooled devices  
BUGCHK      Make bugcheck error log entries  
EXQUOTA      Exceed disk quotas  
GRPNAM      Insert group logical names in the name table  
PRMCB      Create/delete permanent common event flag clusters  
PRMGBL      Create permanent global sections  
PRMMBX      Create permanent mailboxes  
SHMEM      Create/delete structures in shared memory

### SYSTEM PRIVS

ALTPRI      Set base priority higher than allotment  
OPER      Perform operator functions  
PSWAPM      Change process swap mode  
WORLD      Control any process  
SECURITY      Perform security related functions  
SHARE      Access devices allocated to other users  
SYSLCK      Lock system-wide resources

### FILES PRIVS

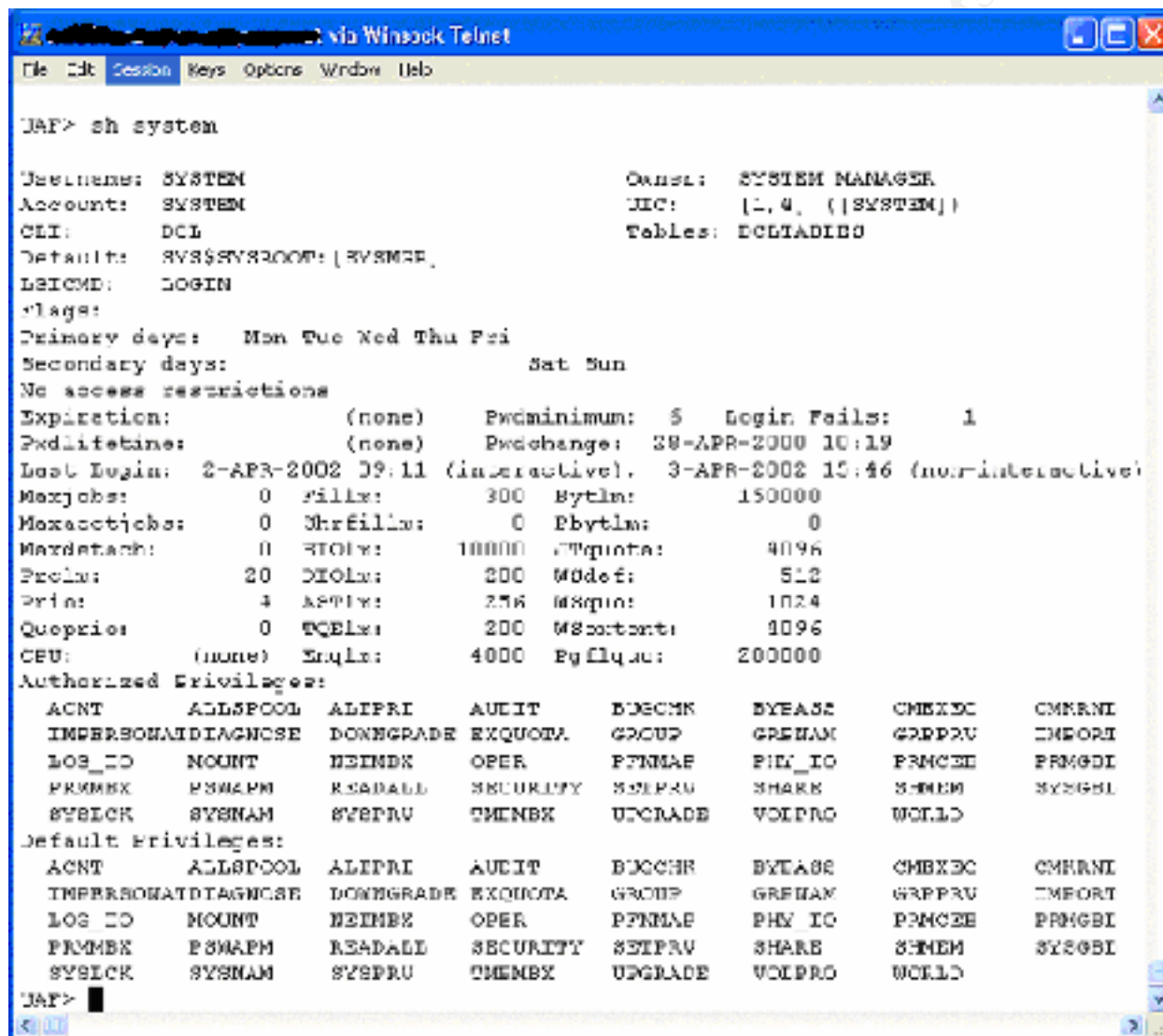
DIAGNOSE      Diagnose devices  
SYSGBL      Create system wide global sections  
VOLPRO      Override volume protection

### ALL PRIVS

BYPASS      Disregard protection  
CMEXEC      Change to executive mode  
CMKRNL      Change to kernel mode  
DETACH      Create detached processes of arbitrary UIC  
LOG\_IO      Issue logical I/O requests  
PFNMAP      Map to specific physical pages  
PHY\_IO      Issue physical I/O requests  
READALL      Possess read access to everything  
SETPRV      \*\*\* ENABLE ALL PRIVILEGES! \*\*\*  
SYSNAM      Insert system logical names in the name table  
SYSPRV      Access objects through SYSTEM protection field

## Appendix 3: System Manager Account

The screenshot of 'UAF> sh system'



```
UAF> sh system

Username: SYSTEM                               Owner: SYSTEM MANAGER
Account: SYSTEM                               UIC: [1, 0, 0] ([SYSTEM])
CLI: DCL                                       Tables: DCLTABLES
Default: SYS$SYSEXEC: [SYSEXEC,
LSICMD: LOGIN
Flags:
Primary days: Mon Tue Wed Thu Fri
Secondary days:                               Sat Sun
No access restrictions
Expiration: (none)                            Pwdminimum: 5      Login Fails: 1
Pwdlifetime: (none)                          Pwdchange: 28-APR-2000 10:19
Last Login: 2-APR-2002 09:11 (interactive), 3-APR-2002 15:46 (non-interactive)
Maxjobs: 0      Filix: 300      Bxflm: 150000
Maxactjobs: 0   Jhrfilix: 0     Pbyflm: 0
Maxdatch: 0     RTOix: 10000    Jtqquota: 4096
Procm: 20       DIOix: 200     W0def: 512
Prin: 4         APPIx: 256     W8qpin: 1024
Queprio: 0      TQELx: 200     W8scotnt: 1096
CPU: (none)     Enqlx: 4000    Pqflqac: 200000

Authorized Privileges:
ACNT      ALLSPOOL  ALIPRI     AUDIT      BDCCHK     BYEASZ     CMEXEC     CMHANI
IMPERSONATE DIAGNOSE  DOWNGRADE  EXQUOTA    GROUP      GRHAM      GRPPRV     IMPORT
LOG IO    MOUNT      REIMBX     OPER       PTNMAP     PHV IO     PRNCEE     PRNGBI
PRMMBX    PSWAPM     READALL    SECURITY    SETPRV     SHARE      SHMEM     SYSGBI
SYSICK    SYSNAM     SYSPRV     TMENBX     UPGRADE    VOIDPRO    WCLD

Default Privileges:
ACNT      ALLSPOOL  ALIPRI     AUDIT      BDCCHK     BYEASZ     CMEXEC     CMHANI
IMPERSONATE DIAGNOSE  DOWNGRADE  EXQUOTA    GROUP      GRHAM      GRPPRV     IMPORT
LOG IO    MOUNT      REIMBX     OPER       PTNMAP     PHV IO     PRNCEE     PRNGBI
PRMMBX    PSWAPM     READALL    SECURITY    SETPRV     SHARE      SHMEM     SYSGBI
SYSICK    SYSNAM     SYSPRV     TMENBX     UPGRADE    VOIDPRO    WCLD

UAF>
```