```
*************************************
*************************************
**                                 **
**        Hacking   : VAX's        **
**                    UNIX         **
**                                 **
*************************************
*************************************
**                                 **
**  Unix is a trademark of BELL LABS **
**  (and you know what *THAT* means) **
**                                 **
*************************************
*************************************
```

Welcome to the Basics of Hacking II: VAX's and UNIX. In this article, we discuss the UNIX system that runs on the various VAX systems. If you are on another UNIX-type system, some commands may differ, but since it is licenced to Bell, they can't make many changes.

****

Hacking onto a UNIX system is very difficult, and in this case, we advise having an inside source, if possible. The reason it is difficult to hack a VAX is this: Many VAX, after you get a carrier from them, respond=>

```
login:
```

They give you no chance to see what the login name format is. Most commonly used are single words, under 8 digits, usually the person's name. There is a way around this: most VAX have an acct. called 'SUGGEST' for people to use to make a suggestion to the system root terminal. This is usually watched by the system operator, but at late he is probably at home sleeping or screwing someone's brains out. So we can write a program to send at the VAX this type of a message:

A screen freeze (Cntrl-S), screen clear (system dependant), about 255 garbage characters, and then a command to create a login acct., after which you clear the screen again, then un-

freeze the terminal. What this does:
When the terminal is frozen, it keeps
a buffer of what is sent. Well, the
buffer is about 127 characters long.
So you overflow it with trash, and then
you send a command line to create an
acct. (system dependant). After this
you clear the buffer and screen again,
then unfreeze the terminal. This is
a bad way to do it, and it is much
nicer if you just send a command to
the terminal to shut the system down,
or whatever you are after...
There is always, *ALWAYS* an acct.
called ROOT, the most powerful acct.
to be on, since it has all of the
system files on it. Uf you hack your
way onto this one, then everything is
easy from here on...
On the UNIX system, the abort key is
the Cntrl-D key. Watch how many times
you hit this, since it is also a way to
log off the system!
A little about UNIX architechture:
The root directory, called ROOT, is
where the system resides. After this
come a few 'sub' root directories,
usually to group things (stats here,
priv stuff here, the user log here...).
Under this comes the superuser (the
operator of the system), and then
finally the normal users. In the UNIX
'shell' everything is treated the same.
By this we mean: You can access a
program the same way you access a user
directory, and so on. The way the UNIX
system was written, everything, users
included, are just programs belonging
to the root directory. Those of you
who hacked onto the ROOT, smile, since
you can screw everything...
The main level (exec level) prompt on
the unix system is the $, and if you
are on the root, you have a # (super-
user prompt).
Ok, a few basics for the system...
To see where you are, and what paths
are active in reguards to your user

account, then type => pwd
This shows your acct. seperated by
a slash with another pathname (acct.),
possibly many times.
To connect through to another path,
or many paths, you would type:
YOU=> path1/path2/path3
and then you are connected all the
way from path1 to path3. You can
run the programs on all the paths
you are connected to. If it does
not allow you to connect to a path,
then you have insufficient privs, or
the path is closed and archived onto
tape. You can run programs this way
also:
YOU=> path1/path2/path3/program-name
UNIX treats everything as a program,
and thus there a few commands to
learn...
To see what you have access to in the
end path, type=> LS
for list. This show the programs
you can run. You can connect to
the root directory and run it's
programs with=>
/ROOT
By the way, most UNIX systems have
their log file on the root, so you
can set up a watch on the file, waiting
for people to log in and snatch their
password as it passes thru the file.
To connect to a directory, use the
command: => cd pathname
This allows you to do what you want
with that directory. You may be asked
for a password, but this is a good
way of finding other user names to
hack onto.
The wildcard character in UNIX, if
you want to search down a path for
a game or such, is the *.
=> ls /*
Should show you what you can access.
the file types are the same as they
are on a DEC, so refer to that section
when examining file. To see what is
in a file, use the => pr filename

command, for print file.

We advise playing with pathnames to get the hang of the concept. There is on-line help available on most systems with a 'HELP' or a '?'.

We advise you look thru the help files and pay attention to anything they give you on pathnames, or the commands for the system.

You can, as a user, create or destroy directories on the tree beneath you. This means that root can kill everything but root, and you can kill any that are below you. These are the

```
=> mkdir pathname
=> rmdir pathname
```

commands.

Once again, you are not alone on the system... type=> WHO

To see what other users are logged in to the system at the time. If you want to talk to them=> write username will allow you to chat at the same time, without having to worry about the parser. To send mail to a user, say

```
=> mail
```

and enter the mail sub-system.

To send a message to all the users on the system, say => wall

which stands for 'write all'

By the way, on a few systems, all you have to do is hit the <return> key to end the message, but on others you must hit the Cntrl-D key.

To send a single message to a user, say => write username

this is very handy again! If you send the sequence of characters discussed at the very beginning of this article, you can have the super-user terminal do tricks for you again.

PRIVS:

If you want super-user privs, you can either log in as root, or edit your acct. so it can say => su

this now gives you the # prompt, and allows you to completely by-pass the protection. The wonderful security conscious developers at bell made it very difficult to do much without privs, but once you have them, there is absolutely nothing stopping you from doing anything you want to.

To bring down a UNIX system:

```
=> chdir /bin
=> rm *
```

This wipes out the pathname bin, where all the system maintenance files are.

Or try:

```
=> r -r
```

This recursively removes everything from the system except the remove command itself.

Or try:

```
=> kill -1,1
=> sync
```

This wipes out the system devices from operation.

When you are finally sick and tired from hacking on the VAX systems, just hit your Cntrl-D and repeat key, and you will eventually be logged out.

****

The reason this file seems to be very sketchy is the fact that bell has 7 licenced versions of UNIX out in the public domain, and these commands are those common to all of them. We recommend you hack onto the root or bin directory, since they have the highest levels of privs, and there is really not much you can do (Except develope software) without them.

****

Next to Come:
The Basics of Hacking III: Data General

****
****

This article written by:
The Knights of Shadow

****
****

More points of note for non-UNIX systems

On VAX's try Username: FIELD with same password (FIELD)
On CMS try these usernames with same passwords:

```
.  PVM   SMART   ADMIN   IFS   IPCS   OSPACKS
.  DIRMAINT      MAINT   ISMAINT      ISPVM
.  ISPVM1        MVS
```