1. what is the program intended to do (use the man pages if need be to explain *how* it works)?

**The program is intending to take an eos username and use the system() call on it to see what kind of user group that user is part of by executing the shell command: execl("/bin/sh", "sh", "-c", command, (char \*) 0);**

**Where command is groups username.**

1. describe what happened and why it occurred

**The program now prints out a bunch of information about directories, versions, user directories and paths. This occurred because the system call is just executing a shell command and when you tag on**

1. what is some potentially dangerous information that an attacker could learn about the system with this approach?  Give a *specific* example of information that might be useful in a targeted attack?

**Some potentially dangerous information is that the attacker could learn which versions of environments that the system is using from the PATH variable. If one of those has an exploitable part of the program that is known about, then the attacker knows how to exploit it on this system. Especially if this is a simple kiosk machine, then it most likely only has one environment that it is using, which if the attacker knows an flaw in the security of that system, then that system is in jeopardy of being hacked.**

1. what warning does the compiler give? what is the program intended to do (use the man pages if need be to explain *how* it works)?

**You get a warning: ‘gets’ is deprecated (declared at /usr/include/stdio.h:638) [-Wdeprecated-declarations] gets(name);**

1. describe what happened, and *why* it occurred

**The program printed out nothing but was overloaded because the gets() call was deprecated and allowed the user to call string/bin/cal and string/bin/bash. That means that the attacker can now call system commands!**

1. suppose a program like this was running on a kiosk, providing simple information to users in a friendly manner.  Describe how an attacker could use this approach to execute commands on the kiosk system, even if they had no login access.

**An attacker now has access to a shell! That means that they can find out data about the system that is running the program, all the files that the simple system has are viewable and depending on the permissions of the files, they are subject to being changed. They could also make and remove files and compile them and put the whole system in jeopardy.**

(Dulimarta Specific Questions)

1. Describe the output of the script

**It takes a msg string and prints it out repeat number of times, otherwise it gives an error.**

2. Describe the difference between this and the previous output

**It gets cut off after about 20 characters.**