Lesson 2.2: Programming Explicitly

# SECURITY VULNERABILITIES IN C/C++ PROGRAMMING

## Programming Explicitly

**Matt Bishop, Ph.D.**
Professor of Computer Science,
UC Davis

UCDAVIS
Continuing and Professional Education

Slide 1: Programming Explicitly

# Programming Explicitly

In source code

| Source Code | Description |
|---|---|
| `char *getenv(char *name)` | Return value of *name*; don't tamper with value! |
| `int putenv(char *str)` | Insert str into the environment |
| `int setenv(char *name, char *value, int overwrite)` | If the variable *name* is not present, insert it with the given value; if it is, and overwrite is non-zero, change it; otherwise, do nothing |
| `int putenv(char *name)` | Delete *name* from the environment |

Slide 2: Programming Explicitly

# Programming Explicitly

| | |
|---|---|
| Third argument to *main*: | `main(int argc, char **argv, char **envp);` |
| Global externally defined variable; ends with **NULL** | `char **environ;` |
| Third argument to *execve*, other *exec* functions passed on implicitly by all unless reset explicitly | `execvp(char *prog, char *a1, ..., NULL);` |

Slide 3: Example Use (Direct)

# Example Use (Direct)

This changes the process' idea of root to the value of HOME

```
if ((p = getenv("HOME")) == NULL)
        p = "/tmp";
if (chroot(p) < 0){
        perror(p);
        return(-1);
}
...
```

Slide 4: Example Use (Indirect)

# Example Use (Indirect)

Through a library function that uses them:

```
system("echo -n 'Today is '; date")
```

*system* is a library function that calls

- "/bin/sh", a shell that uses
- PATH to locate commands without "/" in name

It also sets:

- SHELL to "/bin/sh"
- HOME to user's home directory