Buffer Overflows

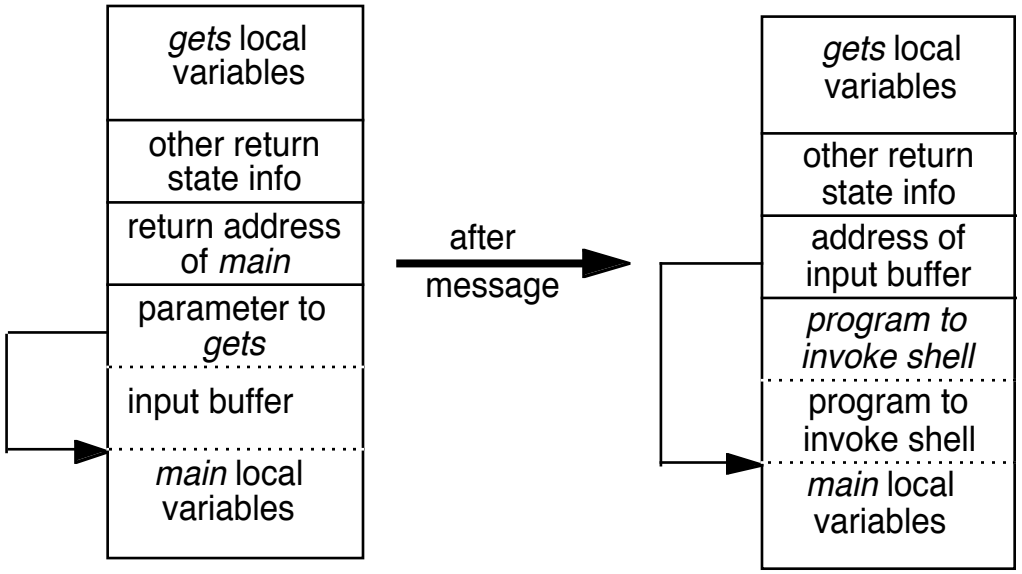
ECS 153 Spring Quarter 2021

Help for Lab 2

Buffer Overflow Problem

- Data is loaded into an array (buffer)
- The data is larger than the array, and so overflows it
- As a result, program may violate security policy
  - Results in attacker being able to execute something it shouldn’t
  - A break-in

Example: *fingerd* Buffer Overflow

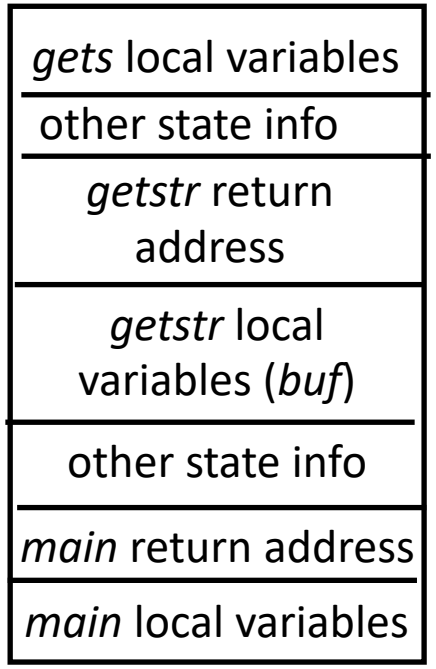


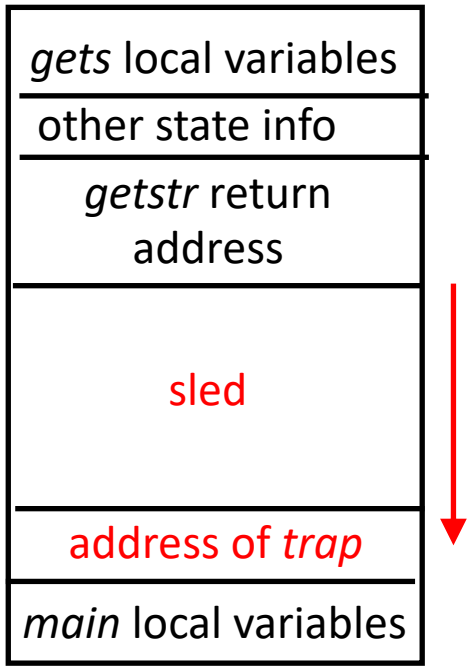
- Input put onto stack without checking length
- If input too long, overwrites PSL and return address
- Load your favorite machine code into the buffer, and overflow, setting return address to address of buffer

How to Enable Execution with *gcc* and Linux

- Linux turns off execute permission for the stack, so you can load your program but not execute it
  - Give the flag `–z execstack` to allow this

For Lab 2



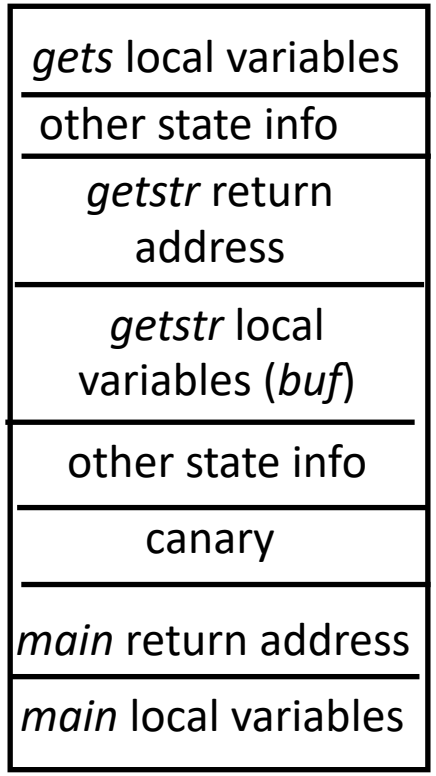


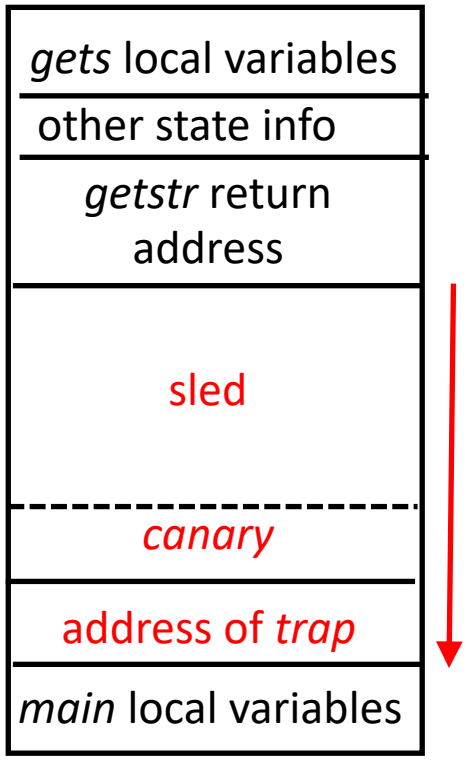
- Just as before, but the "return address" is now the address of trap

How to Do This with *gcc* and Linux

- On entry to *getstr*(), Linux places a "canary" (random number, basically) between *buf* and what follows it (here, the "other state info") and saves the value
- If the value of the canary is different when *getstr*() returns, the program is stopped
  - Give the flag `–fno-stack-protector` to turn this off

More Lab 2


gets local variables
other state info
getstr return address
getstr local variables (*buf*)
other state info
canary
main return address
main local variables


gets local variables
other state info
getstr return address
sled
canary
address of *trap*
main local variables