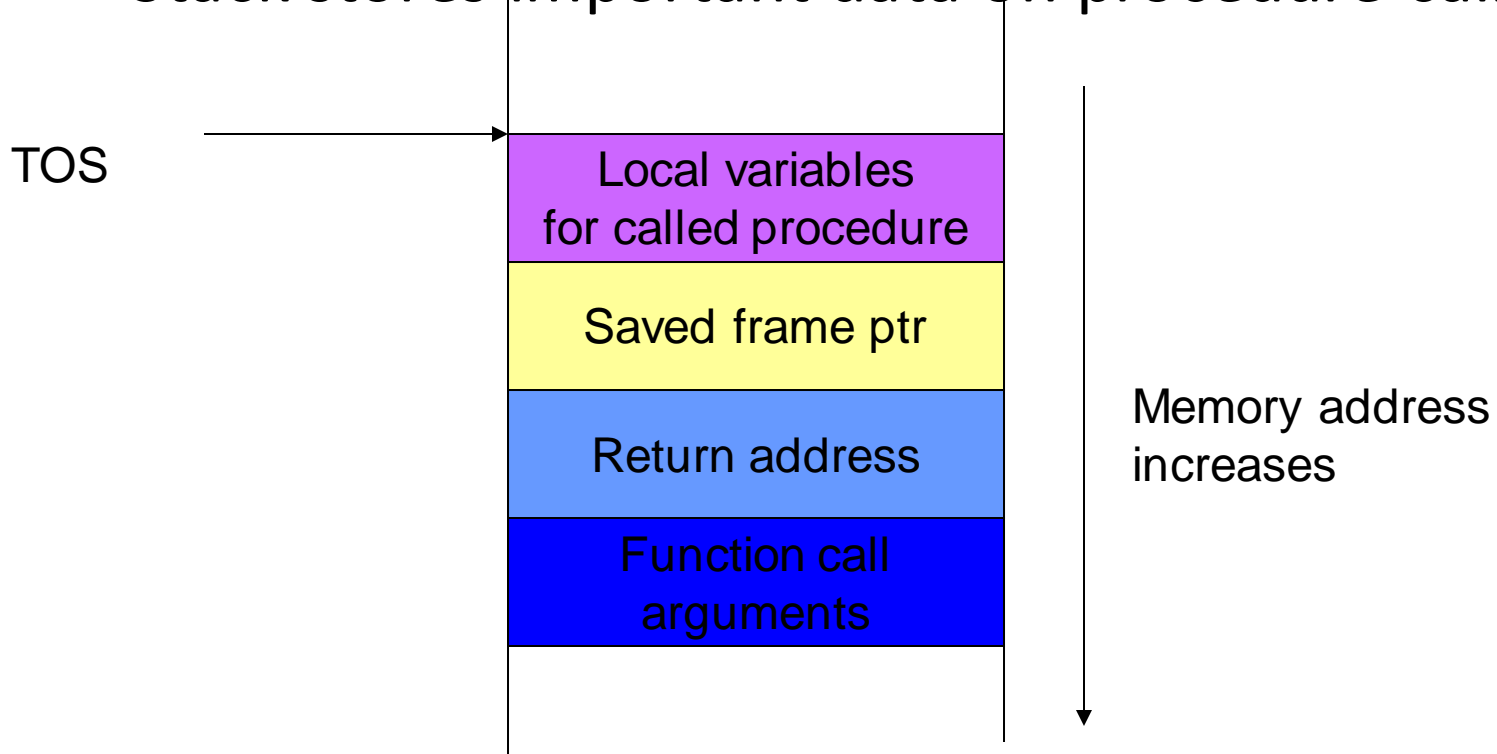


Buffer Overflow Attacks

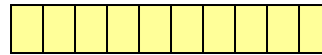
- Stack-based overflow attacks
- Stack stores important data on procedure call



Buffer Overflow Attacks

- Consider a function

```
void sample_function(char* s)
{
    char buffer[10];
    strcpy(buffer, s);
    return;
}
```

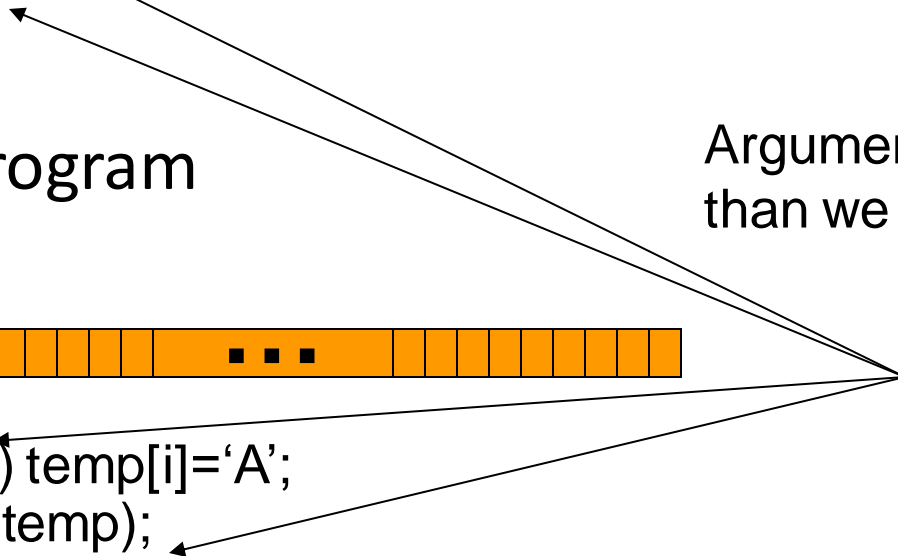


- And a main program

```
void main()
{
    int i;
    char temp[200];
    for(i=0; i<200; i++) temp[i]='A';
    sample_function(temp);
    return;
}
```

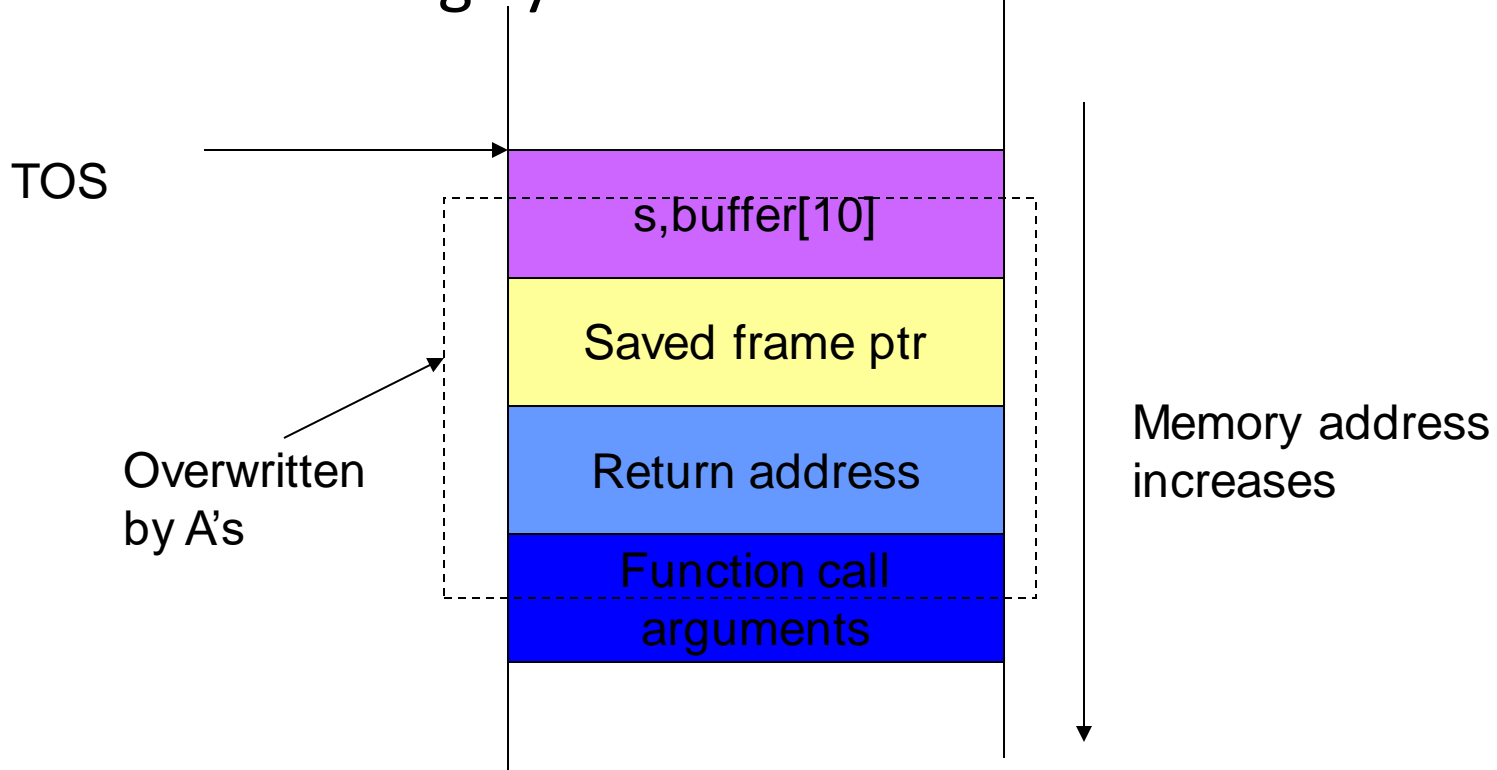


Argument is larger
than we expected



Buffer Overflow Attacks

- Large input will be stored on the stack, overwriting system information



Buffer Overflow Attacks

- Attacker overwrites return address to point somewhere else
 - “Local variables” portion of the stack
 - Places attack code in machine language at that portion
 - Since it is difficult to know exact address of the portion, pads attack code with NOPs before and after