typical buffer overflow pattern

cause program to write past the end of a buffer

that somehow causes different code to run

(usually code the attacker wrote)

why buffer overflows?

for a long time, most common vulnerability

common results in arbitrary code execution

related to other memory-management vulnerabilities

- which usually also result in arbitrary code execution

network worms and overflows

worms that connect to vulnerable servers:

Morris worm included some buffer overflow exploits

- Morris worm: first self-replicating malware
- in mail servers, user info servers

2001: Code Red worm that spread to web servers (running Microsoft IIS)

overflows without servers

bugs dealing with corrupt files:

Adobe Flash (web browser plugin)

PDF readers

web browser JavaScript engines

image viewers

movie viewers

decompression programs

...

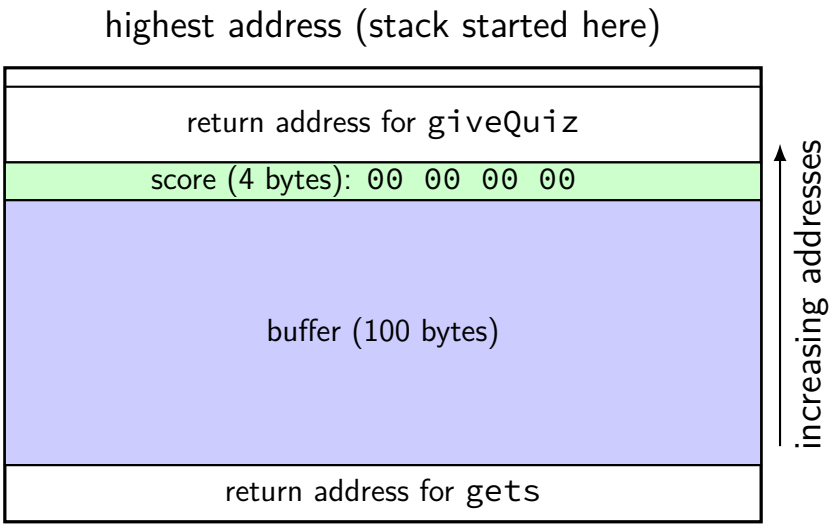
simpler overflow

```
struct QuizQuestion questions[NUM_QUESTIONS];
int giveQuiz() {
    int score = 0;
    char buffer[100];
    for (int i = 0; i < NUM_QUESTIONS; ++i) {
        gets(buffer);
        if (checkAnswer(buffer, &questions[i])) {
            score += 1;
        }
    }
    return score;
}
```

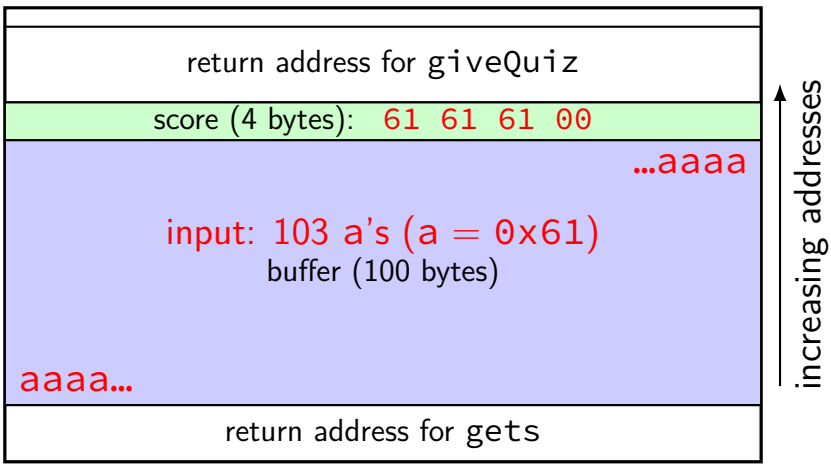
simpler overflow

```
struct QuizQuestion questions[NUM_QUESTIONS];
int giveQuiz() {
    int score = 0;
    char buffer[100];
    for (int i = 0; i < NUM_QUESTIONS; ++i) {
        gets(buffer);
        if (checkAnswer(buffer, &questions[i])) {
            score += 1;
        }
    }
    return score;
}
```

simpler overflow: stack



simpler overflow: stack



exercise: stack layout

```
GradeAssignment:
  pushq   %rbp
  pushq   %rbx
  xorl    %ebx, %ebx
  subq    $72, %rsp
  leaq    8(%rsp), %rbp
for_loop:
  movq    %rbp, %rdi
  call    gets
  movl    %ebx, %esi
  movq    %rbp, %rdi
  call    GradeAnswer
  leaq    24(%rsp), %rdi
  movl    %eax, (%rdi,%rbx,4)
  incq    %rbx
  cmpq    $10, %rbx
  jne     for_loop
  call    Process
```

```
int GradeAssignment(FILE *in) {
  int scores[10]; char buffer[16];
  for (int i = 0; i < 10; ++i) {
    gets(buffer);
    scores[i] =
        GradeAnswer(buffer, i);
  }
  Process(scores);
}
```

exercise: how many bytes after `buffer[0]` is the first byte of `scores[0]`?

exercise: stack layout

```
GradeAssignment:
  pushq   %rbp
  pushq   %rbx
  xorl    %ebx, %ebx
  subq    $72, %rsp
  leaq    8(%rsp), %rbp
for_loop:
  movq    %rbp, %rdi
  call    gets
  movl    %ebx, %esi
  movq    %rbp, %rdi
  call    GradeAnswer
  leaq    24(%rsp), %rdi
  movl    %eax, (%rdi,%rbx,4)
  incq    %rbx
  cmpq    $10, %rbx
  jne     for_loop
  call    Process
```

```
int GradeAssignment(FILE *in) {
  int scores[10]; char buffer[16];
  for (int i = 0; i < 10; ++i) {
    gets(buffer);
    scores[i] =
        GradeAnswer(buffer, i);
  }
  Process(scores);
}
```

exercise: how many bytes after `buffer[0]` is the first byte of `scores[0]`? answer: 16

exercise: overflow?

```
GradeAssignment:
  pushq   %rbp
  pushq   %rbx
  xorl    %ebx, %ebx
  subq    $72, %rsp
  leaq    8(%rsp), %rbp
for_loop:
  movq    %rbp, %rdi
  call    gets
  movl    %ebx, %esi
  movq    %rbp, %rdi
  call    GradeAnswer
  leaq    24(%rsp), %rdi
  movl    %eax, (%rdi,%rbx,4)
  incq    %rbx
  cmpq    $10, %rbx
  jne     for_loop
  call    Process
```

```
int GradeAssignment(FILE *in) {
  int scores[10]; char buffer[16];
  for (int i = 0; i < 10; ++i) {
    gets(buffer);
    scores[i] =
        GradeAnswer(buffer, i);
  }
  Process(scores);
}
```

exercise: if input into buffer is 50 copies of the character `'1'` what is value of `scores[0]`?

exercise: overflow?

```
GradeAssignment:
  pushq   %rbp
  pushq   %rbx
  xorl    %ebx, %ebx
  subq    $72, %rsp
  leaq    8(%rsp), %rbp
for_loop:
  movq    %rbp, %rdi
  call    gets
  movl    %ebx, %esi
  movq    %rbp, %rdi
  call    GradeAnswer
  leaq    24(%rsp), %rdi
  movl    %eax, (%rdi,%rbx,4)
  incq    %rbx
  cmpq    $10, %rbx
  jne     for_loop
  call    Process
```

```
int GradeAssignment(FILE *in) {
  int scores[10]; char buffer[16];
  for (int i = 0; i < 10; ++i) {
    gets(buffer);
    scores[i] =
        GradeAnswer(buffer, i);
  }
  Process(scores);
}
```

exercise: if input into buffer is 50 copies of the character `'1'` what is value of `scores[0]`?
answer: 0x31313131

backup slides

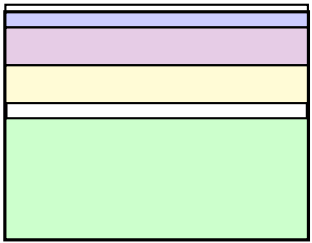
exercise: stack layout?

```
void vulnerable() {
  unsigned char needed_hash[32] = {...};
  unsigned char actual_hash[32];
  char buffer[100];
  gets(buffer);
  ComputeSHA256(buffer, actual_hash);
  if (memcmp(needed_hash,actual_hash,32)==0) {
    Interesting();
  }
}
```

```
vulnerable:
  endbr64
  pushq %rbp
  pushq %rbx
  subq  $184, %rsp
    ...
  movq  %rsp, %rbp
  movq  %rbp, %rdi
  call  gets@PLT
  leaq  112(%rsp), %rbx
  movq  %rbx, %rsi
  movq  %rbp, %rdi
  call  ComputeSHA256@PLT
  leaq  144(%rsp), %rdi
  movl  $32, %edx
  movq  %rbx, %rsi
  call  memcmp@PLT
  testl %eax, %eax
  je  .L4
.L1:
  addq  $184, %rsp
  popq  %rbx
  popq  %rbp
  ret
```

exercise: stack layout?

```
void vulnerable() {
  unsigned char needed_hash[32] = {...};
  unsigned char actual_hash[32];
  char buffer[100];
  gets(buffer);
  ComputeSHA256(buffer, actual_hash);
  if (memcmp(needed_hash,actual_hash,32)==0) {
    Interesting();
  }
}
```




```
vulnerable:
  endbr64
  pushq %rbp
  pushq %rbx
  subq  $184, %rsp
    ...
  movq  %rsp, %rbp
  movq  %rbp, %rdi
  call  gets@PLT
  leaq  112(%rsp), %rbx
  movq  %rbx, %rsi
  movq  %rbp, %rdi
  call  ComputeSHA256@PLT
  leaq  144(%rsp), %rdi
  movl  $32, %edx
  movq  %rbx, %rsi
  call  memcmp@PLT
  testl %eax, %eax
  je  .L4
.L1:
  addq  $184, %rsp
  popq  %rbx
  popq  %rbp
  ret
```