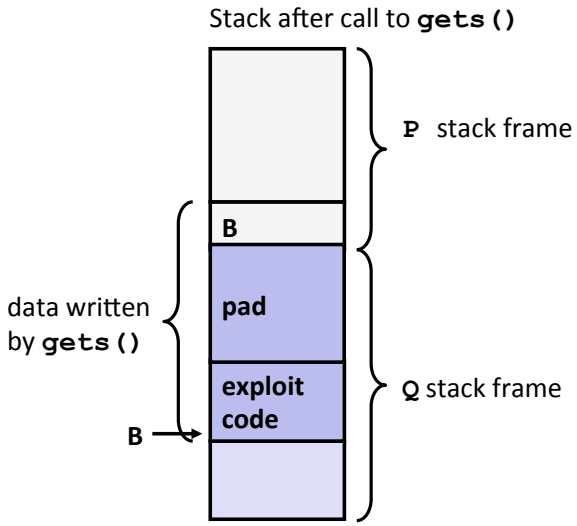
Code Injection Attacks

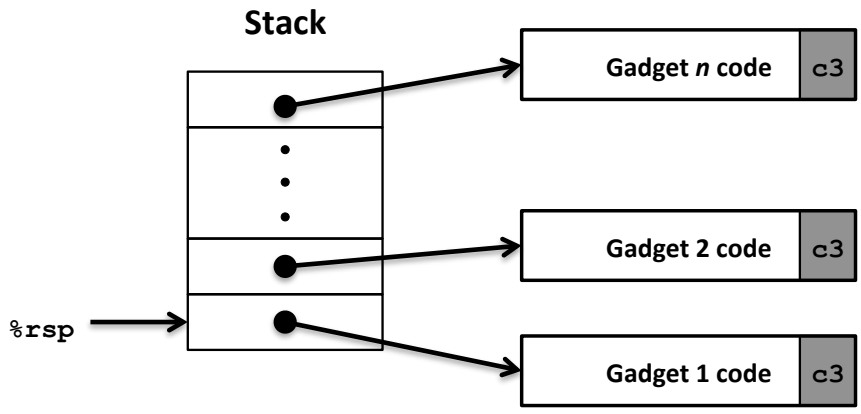
```
void P(){
  Q();
  ...
}
```

return address A

```
int Q() {
  char buf[64];
  gets(buf);
  ...
  return ...;
}
```



Return-Oriented Programming



Finding Gadgets: Function Ends

```
long ab_plus_c(long a, long b, long c) {
   return a * b + c;
}
```

```
00000000004004d0 <ab_plus_c>:
  4004d0:  48 0f af fe  imul %rsi,%rdi
  4004d4:  48 8d 04 17  lea (%rdi,%rdx,1),%rax
  4004d8:  c3           retq
```

rax ← rdi + rdx

Gadget address = 0x4004d4

Finding Gadgets: Repurposing Bytes

```
void setval(unsigned *p) {
    *p = 3347663060u;
}
```

Encodes `movq %rax, %rdi`

```
<setval>:
  4004d9:  c7 07 d4 48 89 c7  movl  $0xc78948d4,(%rdi)
  4004df:  c3                 retq
```

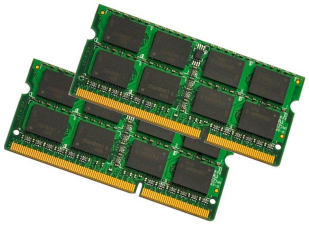
rdi ← rax

Gadget address = 0x4004dc

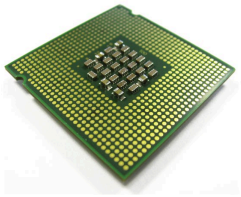
Data Storage

- Disks
  - Hard disk (HDD)
  - Solid state drive (SSD)

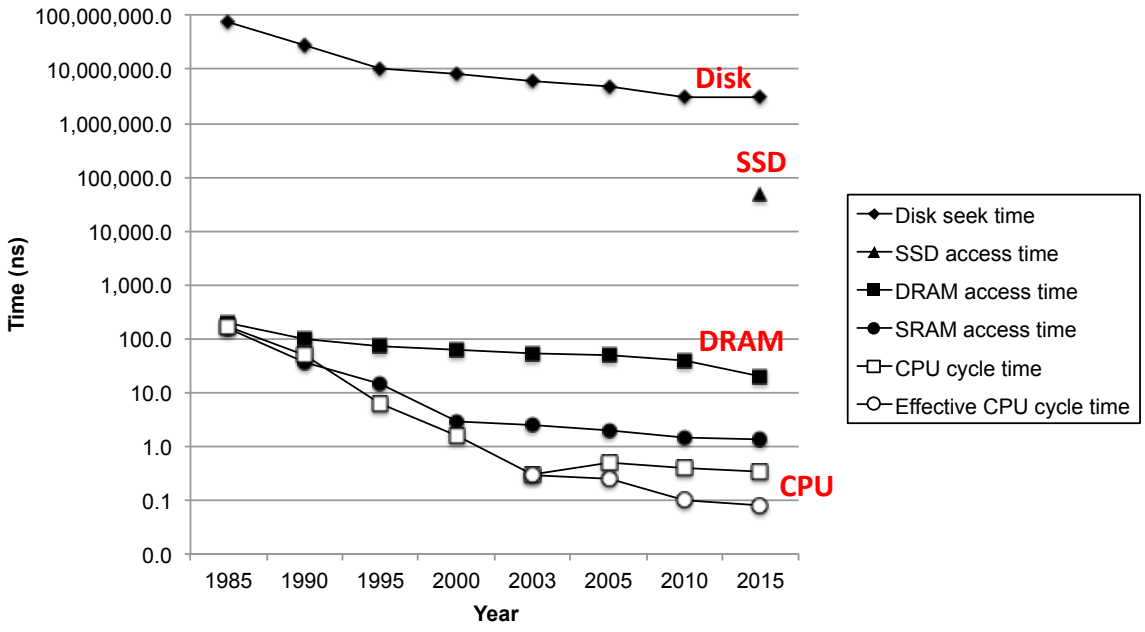
- Random Access Memory
  - Dynamic RAM (DRAM)
  - Static RAM (SRAM)



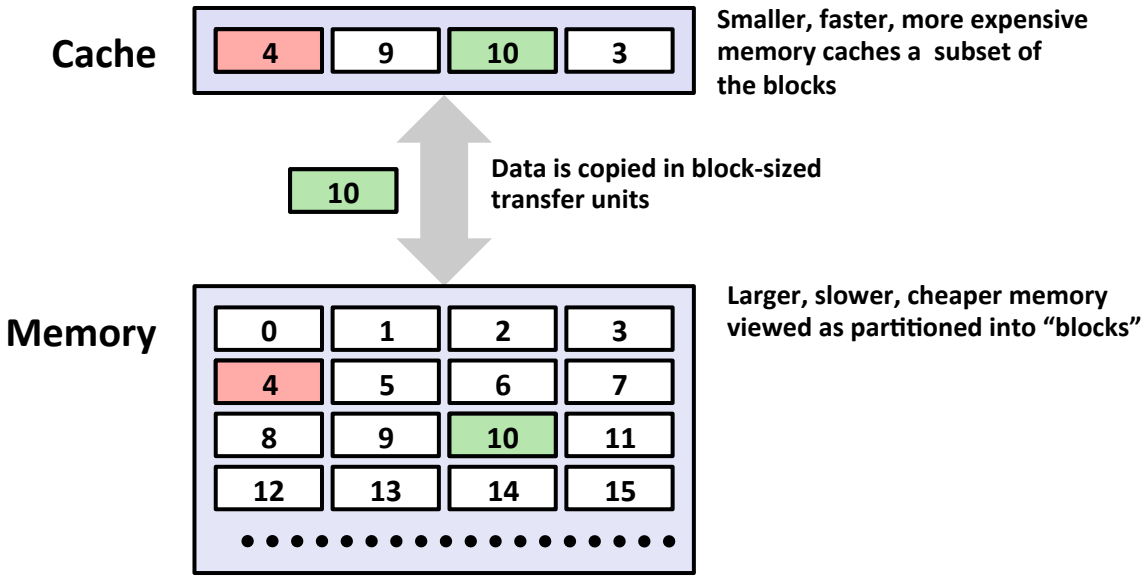
- Registers
  - %rax, %rbx, ...



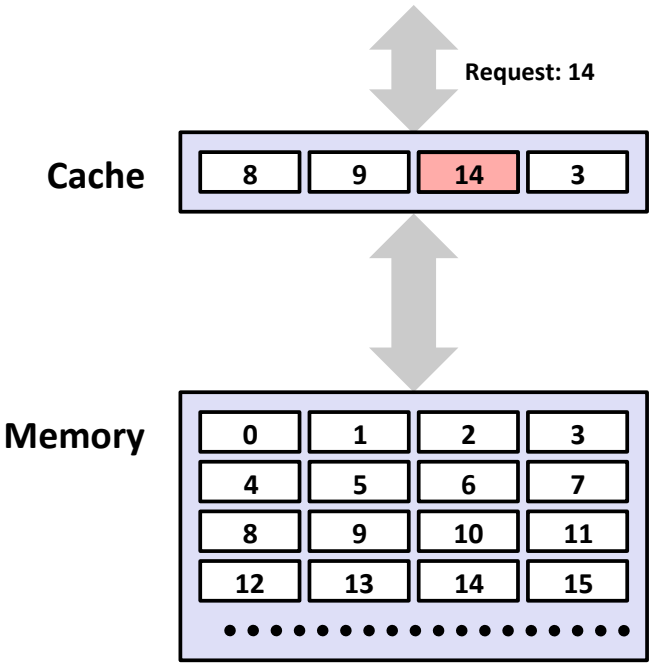
The CPU-Memory Gap



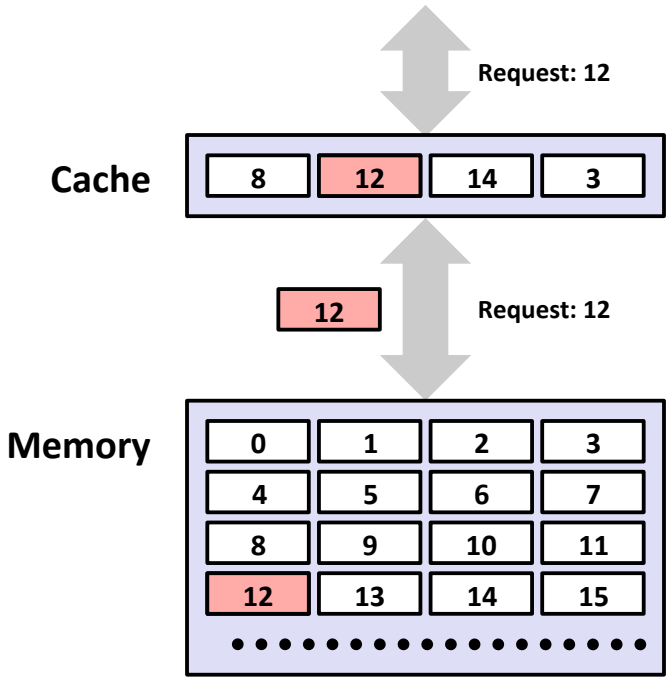
Caching



Cache Hit

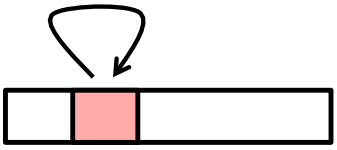


Cache Miss

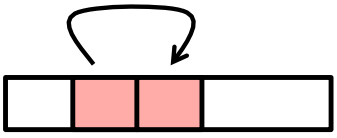


Locality

- **Temporal locality:**



- **Spatial locality:**



Locality Example (1)

```
sum = 0;
for (i = 0; i < n; i++)
    sum += a[i];
return sum;
```


Locality Example (2)

```
int sum_array_rows(int a[M][N]) {
    int i, j, sum = 0;

    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            sum += a[i][j];
    return sum;
}
```

Locality Example (3)

```
int sum_array_cols(int a[M][N]) {
    int i, j, sum = 0;

    for (j = 0; j < N; j++)
        for (i = 0; i < M; i++)
            sum += a[i][j];
    return sum;
}
```


The Memory Hierarchy