# uprobe optimization and override

jiri olsa / isovalent @ cisco

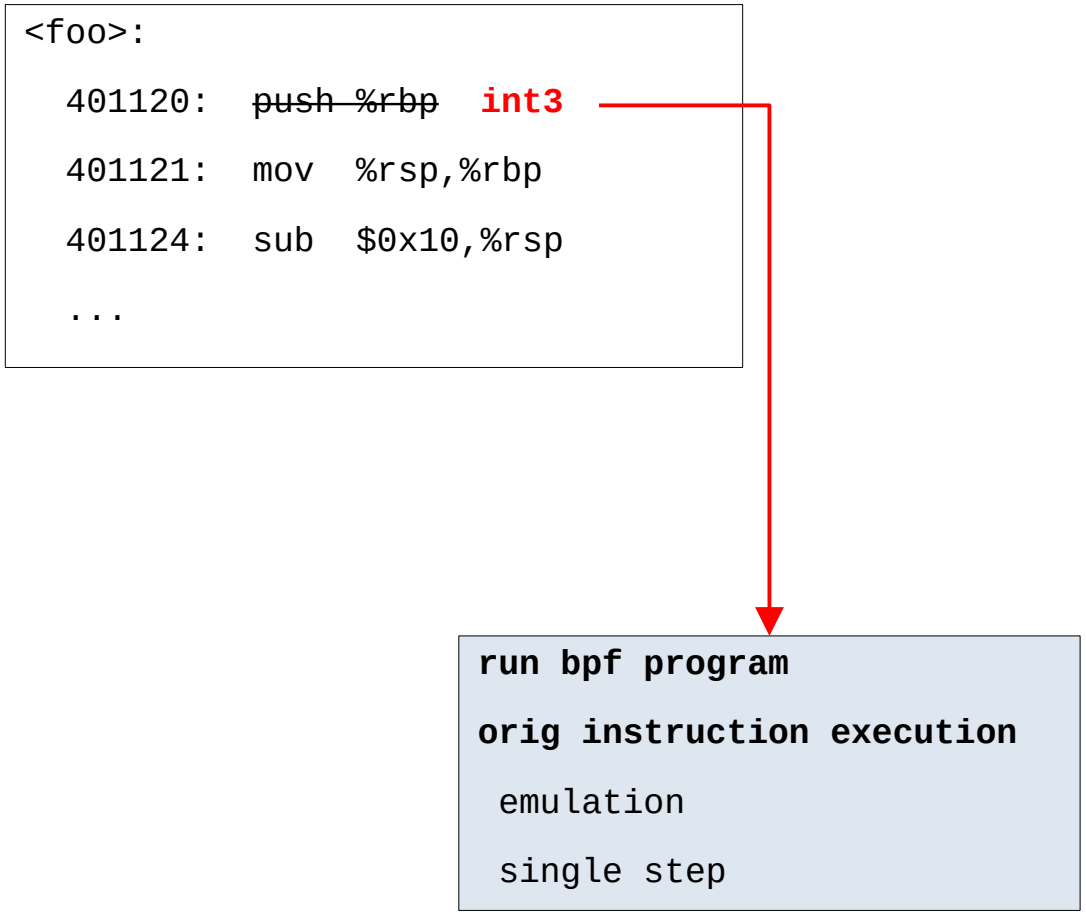
# UPROBE OPTIMIZATION

- on top of nop5 and beyond
- on x86_64

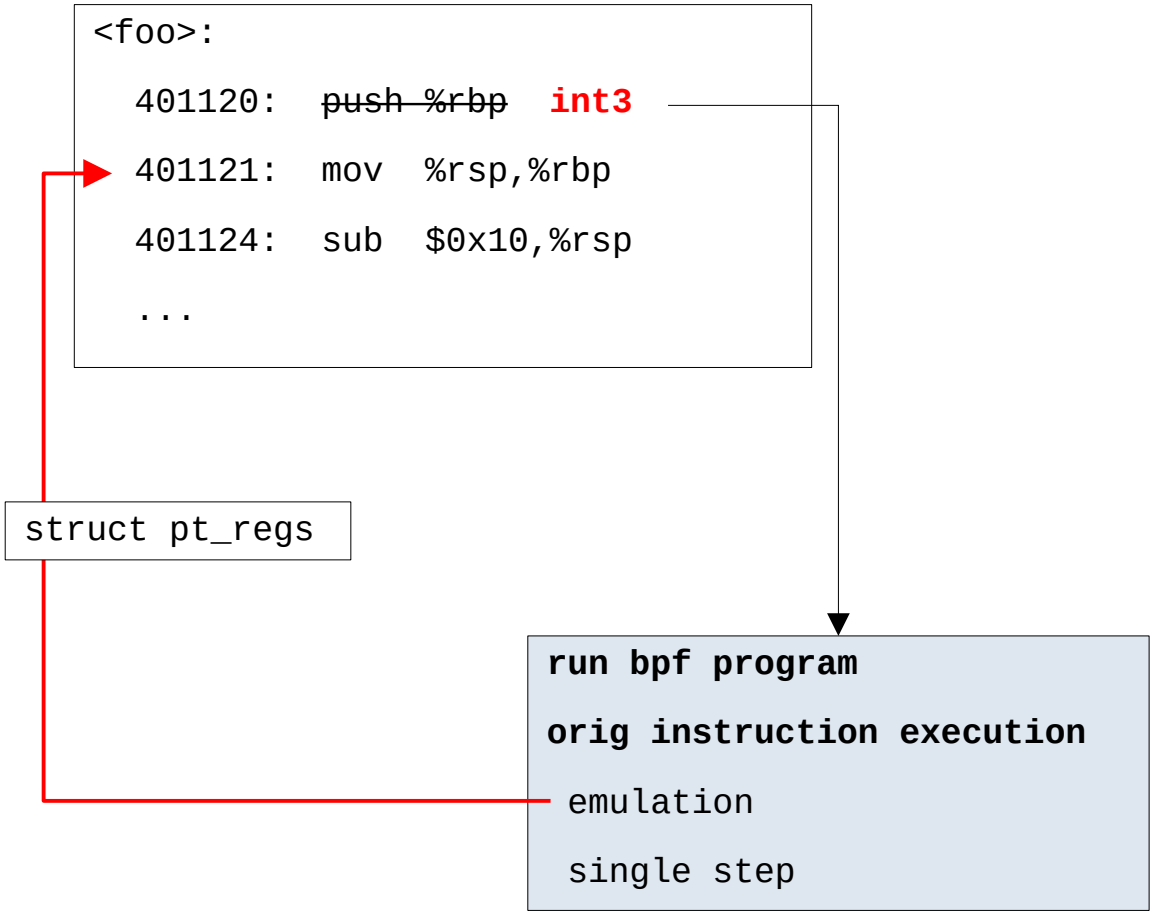
# UPROBE QUICK COURSE

```
<foo>:
  401120:  push %rbp
  401121:  mov  %rsp,%rbp
  401124:  sub  $0x10,%rsp
  ...
```

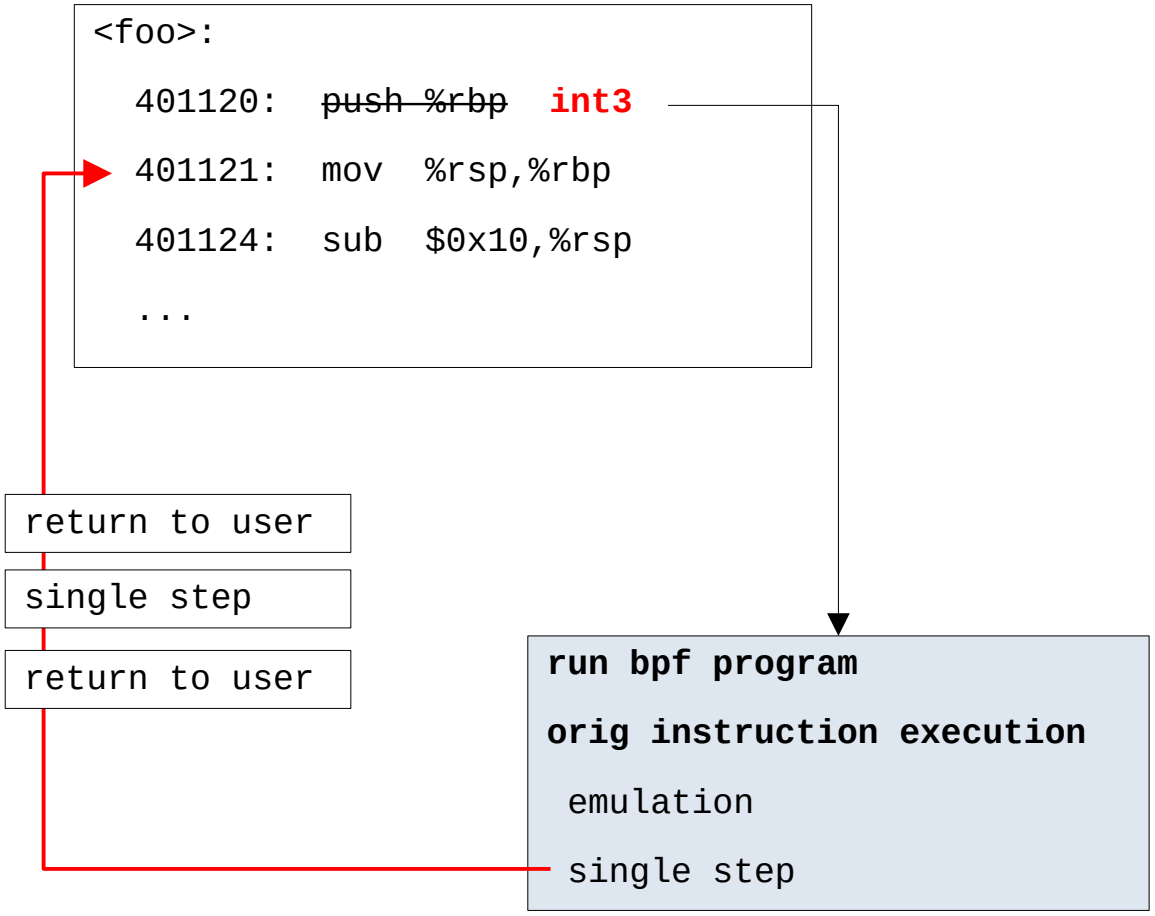
# UPROBE QUICK COURSE



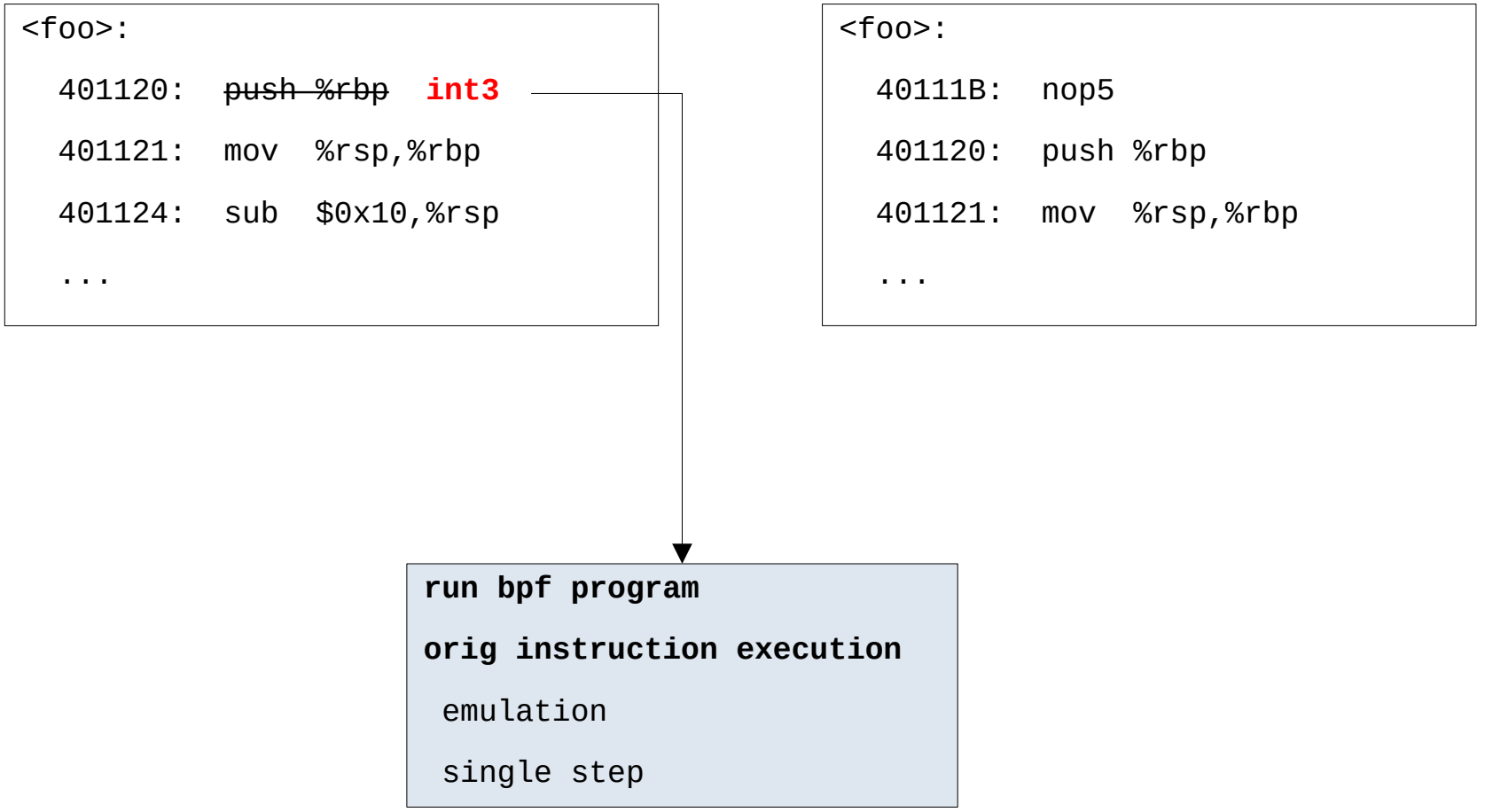
# UPROBE QUICK COURSE



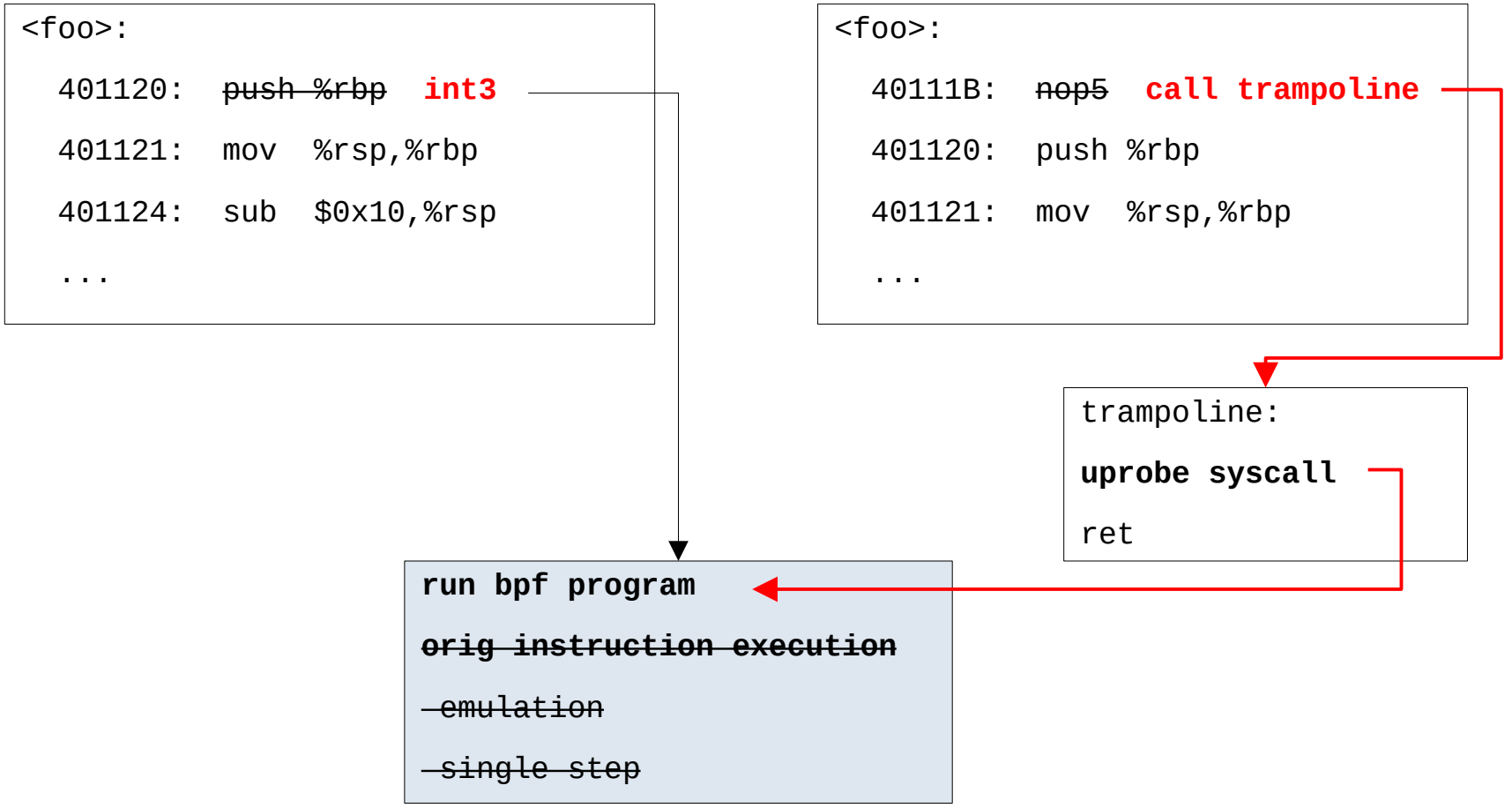
# UPROBE QUICK COURSE



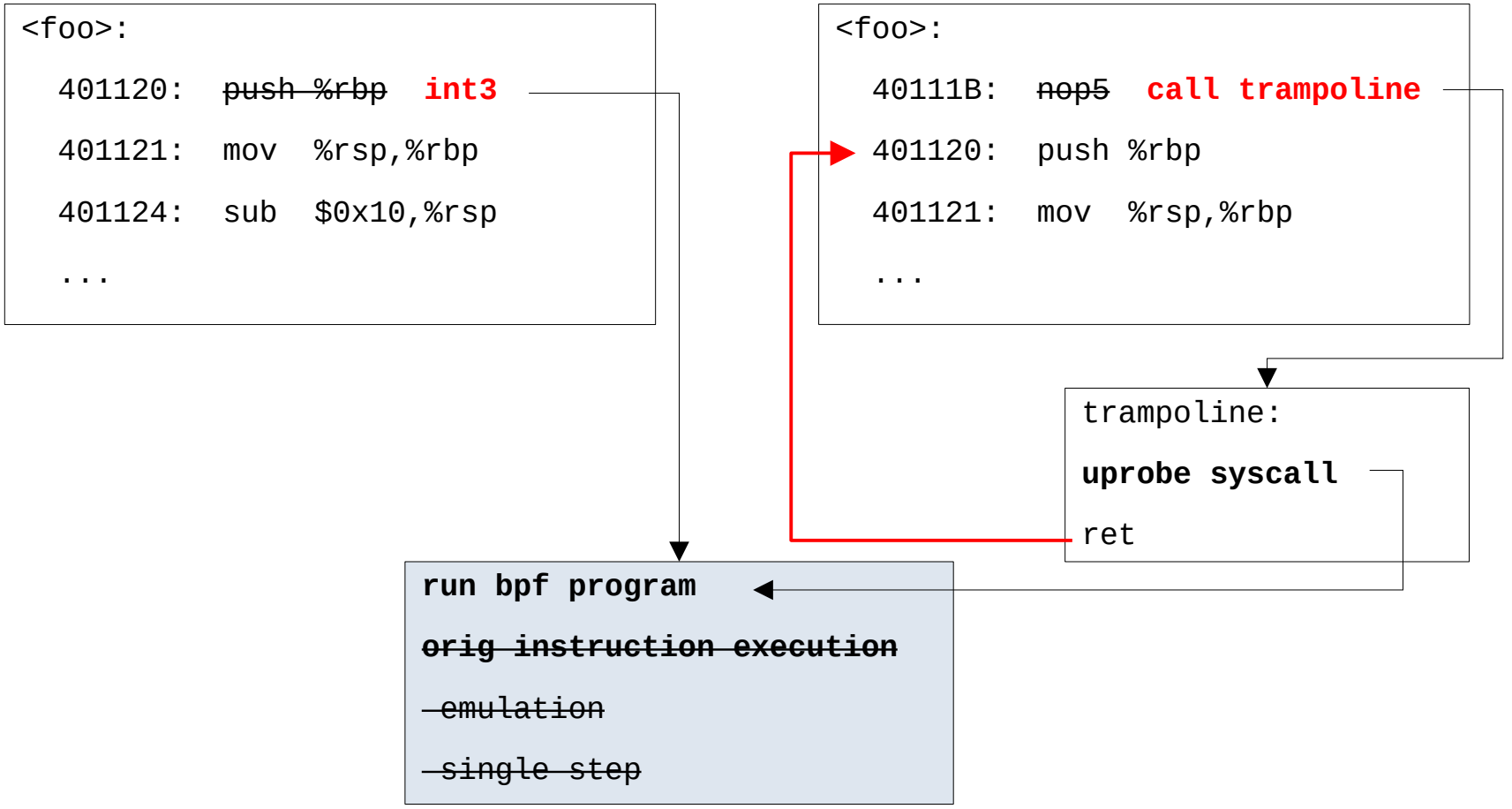
# UPROBE QUICK COURSE



# UPROBE QUICK COURSE



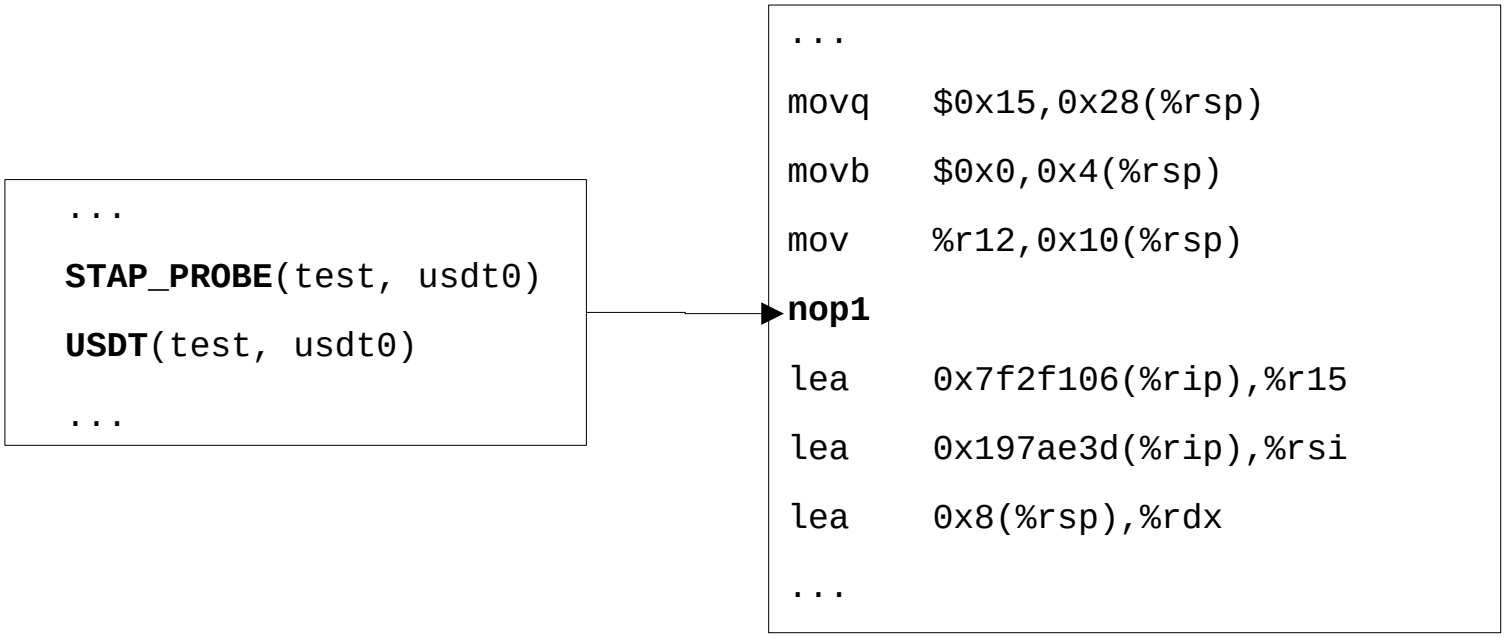
# UPROBE QUICK COURSE



# USDT OPTIMIZATION

## make usdt macros emit nop5

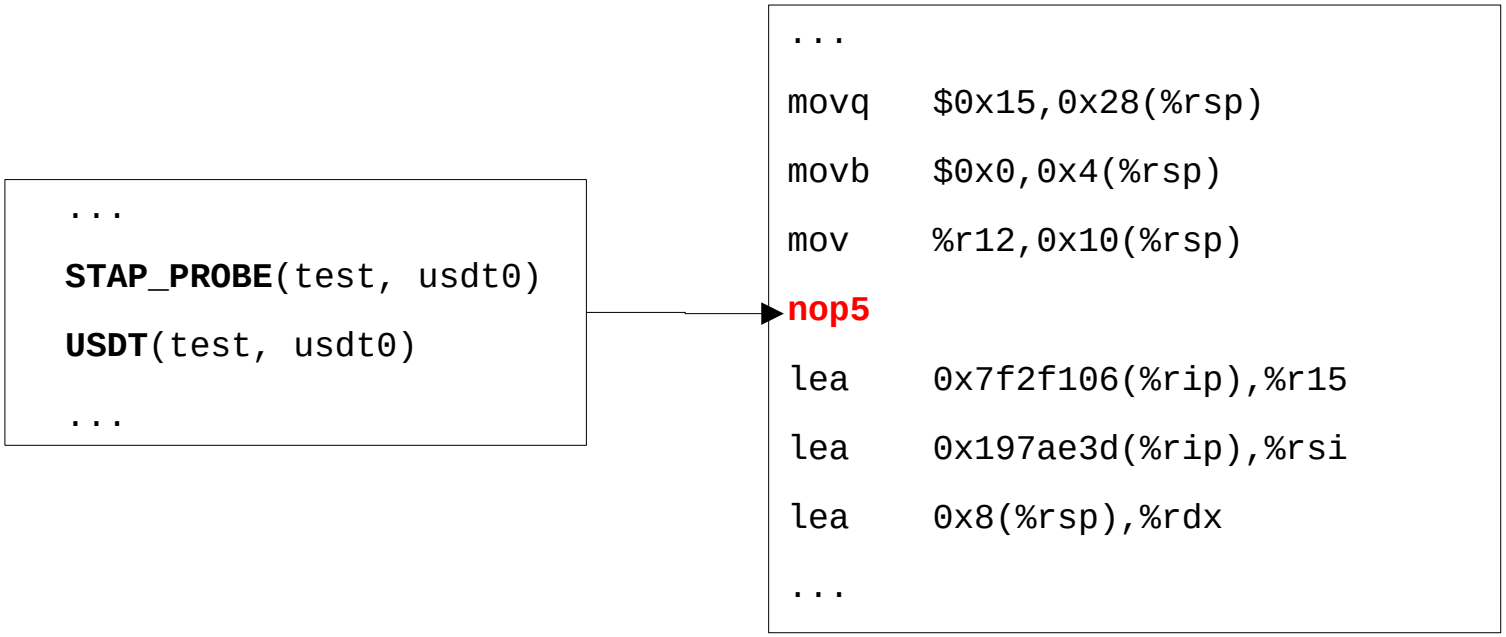
## backward compatibility with old kernels?



# USDT OPTIMIZATION

## make usdt macros emit nop5

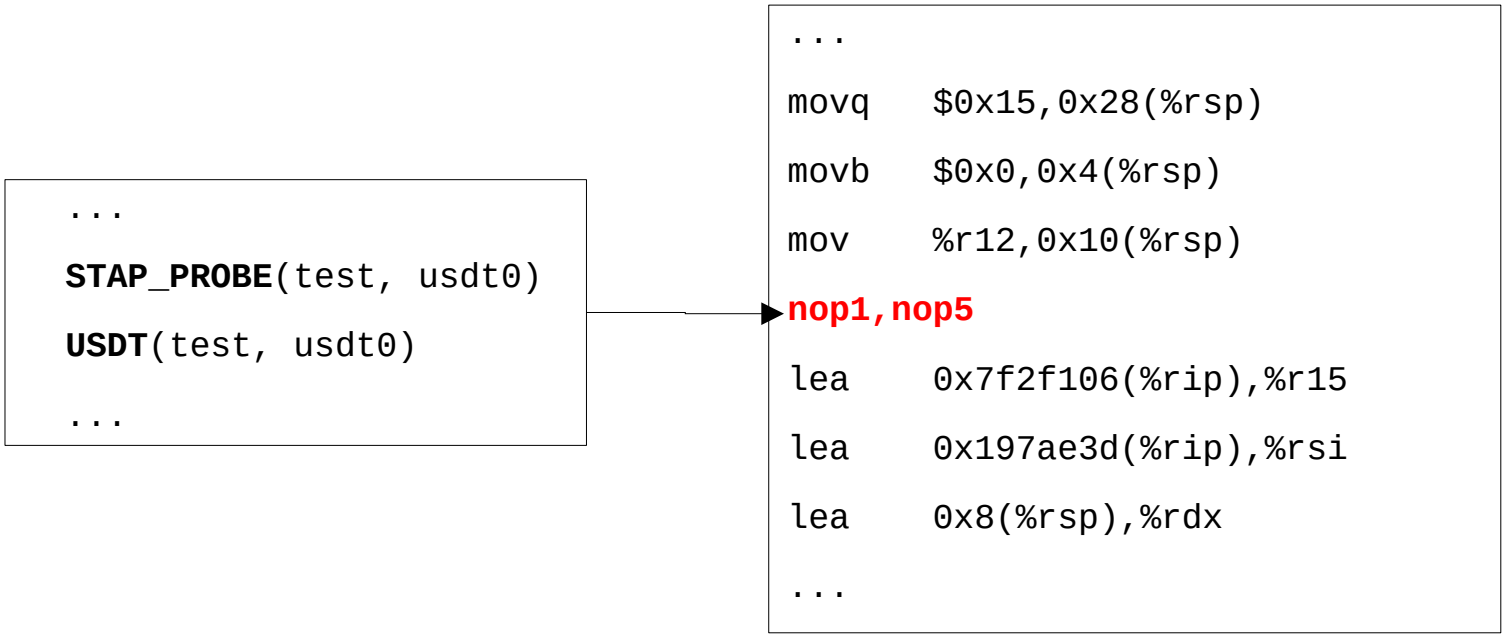
## backward compatibility with old kernels?



# USDT OPTIMIZATION

**make usdt macros emit ~~nop5~~ *nop1,nop5***

**backward compatibility with old kernels!**



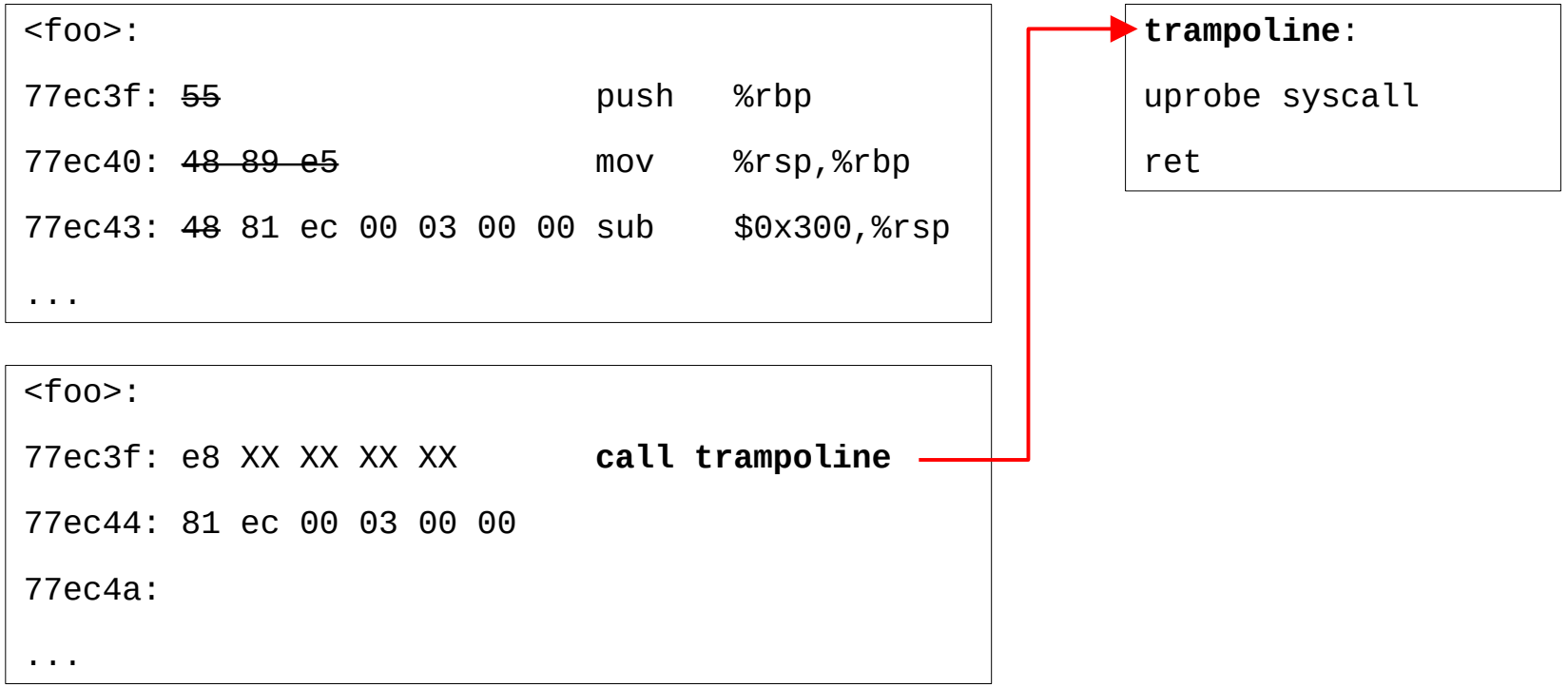
# UPROBE OPTIMIZATION

- cover more sites/prologues
- on top of emulated instructions

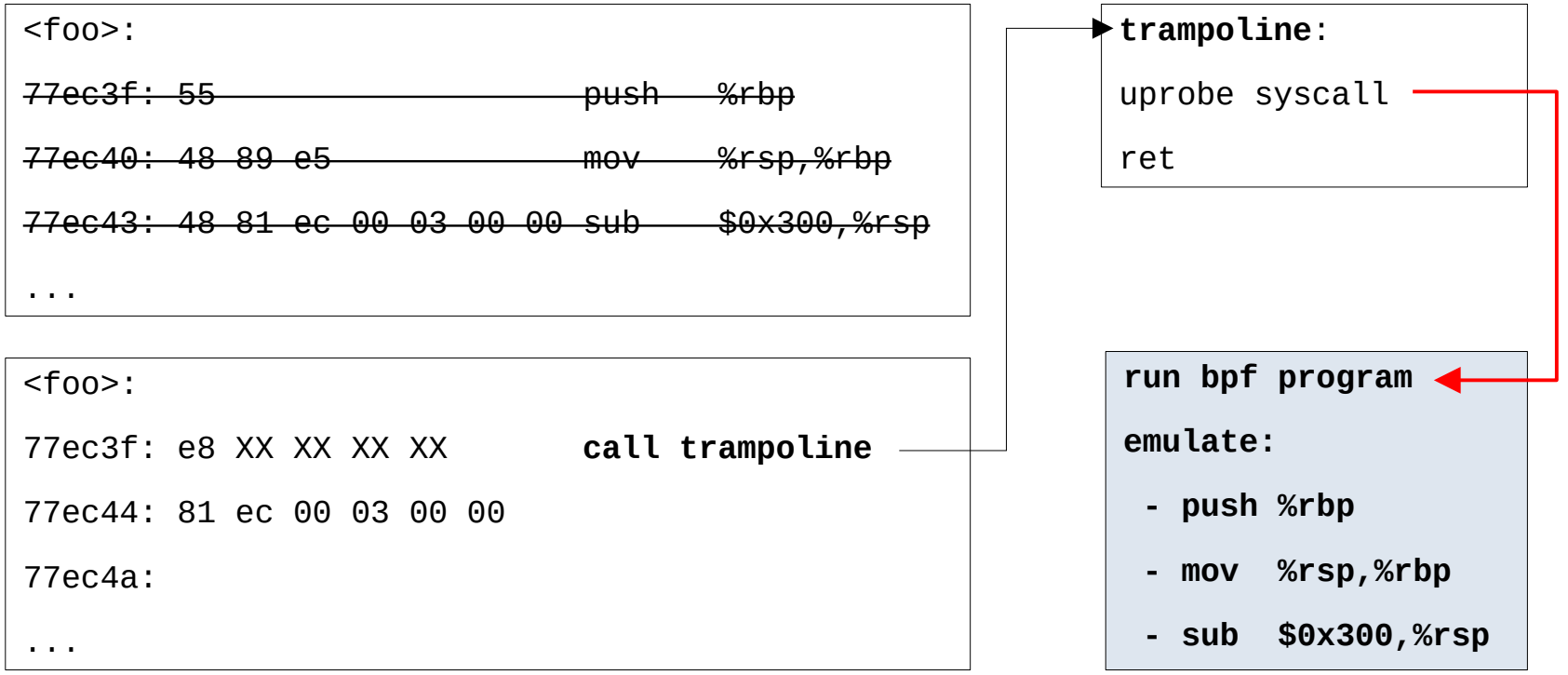
# UPROBE OPTIMIZATION

```
<foo>:
77ec3f: 55                    push   %rbp
77ec40: 48 89 e5              mov    %rsp,%rbp
77ec43: 48 81 ec 00 03 00 00 sub    $0x300,%rsp
...
```

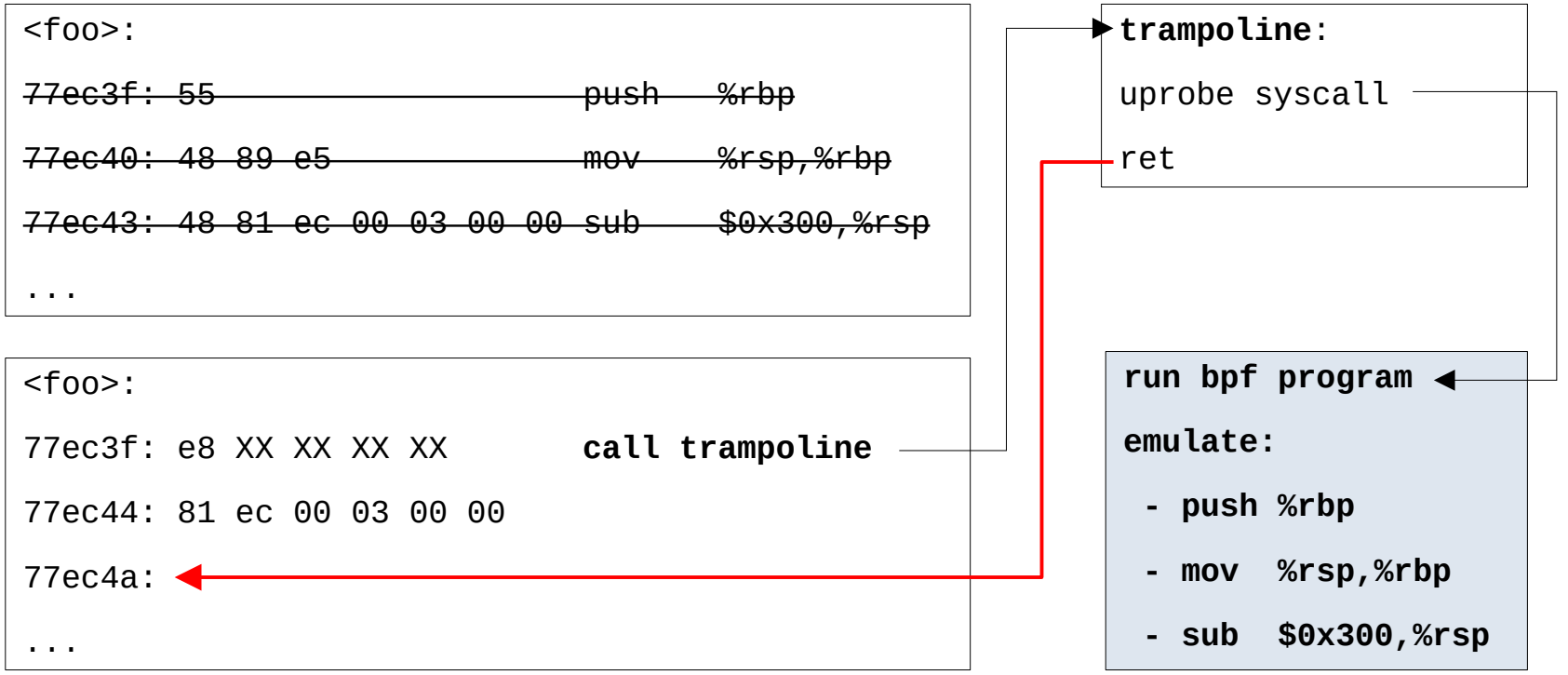
# UPROBE OPTIMIZATION



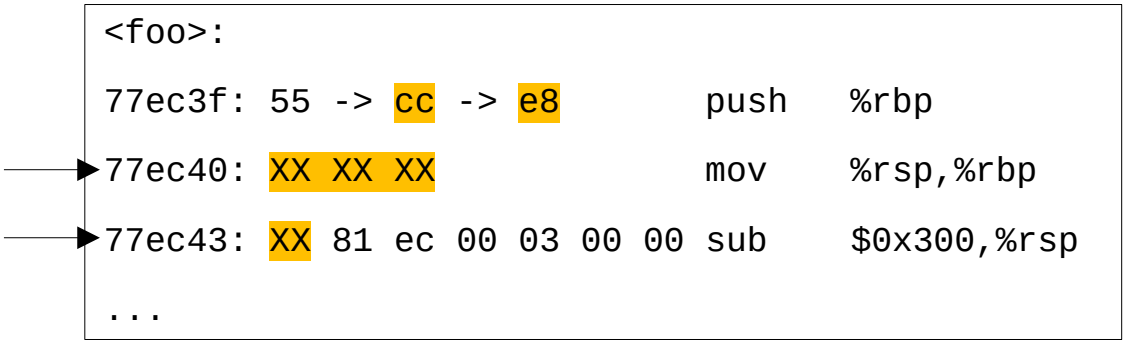
# UPROBE OPTIMIZATION



# UPROBE OPTIMIZATION



# HUM.. call update



# HUM.. call update

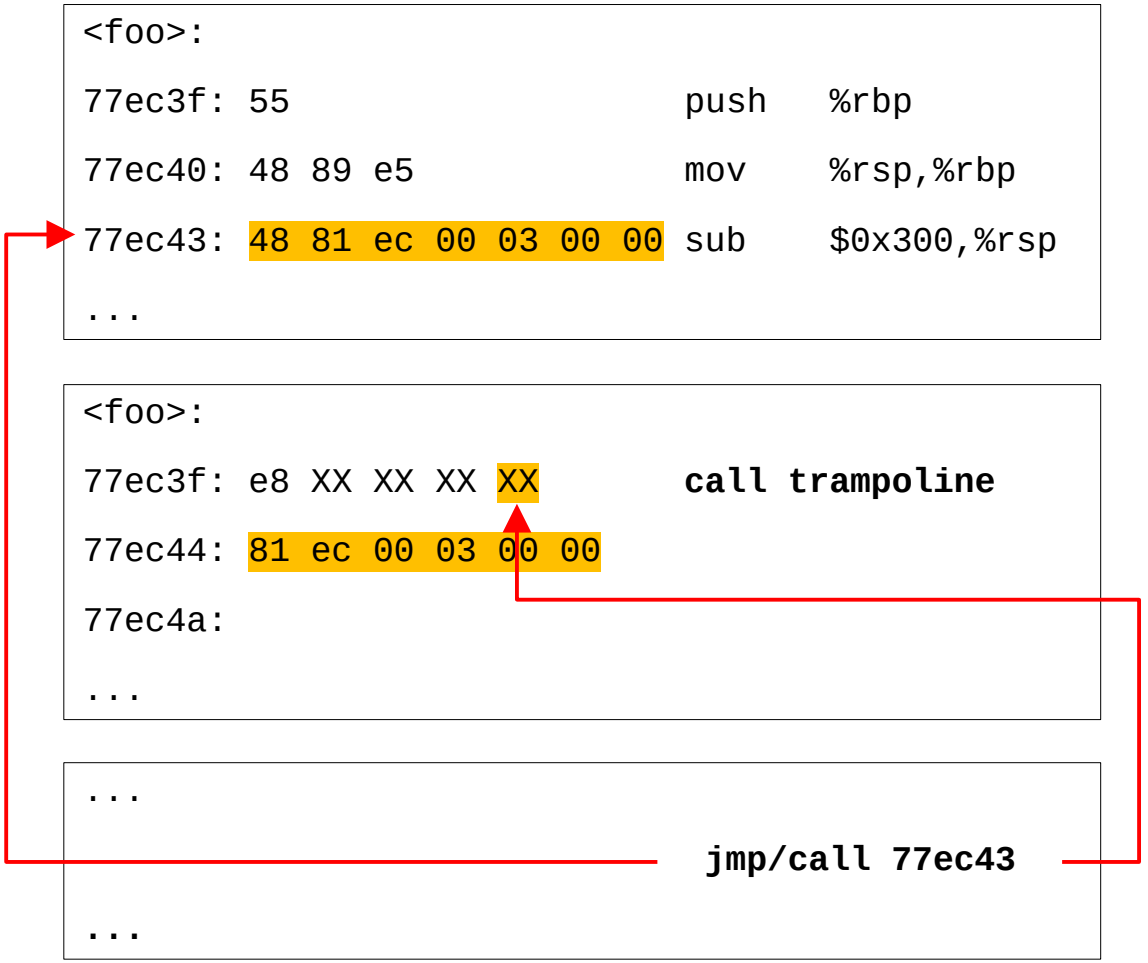
```
<foo>:
77ec3f: 55 -> cc -> e8        push   %rbp
77ec40: XX XX XX              mov    %rsp,%rbp
77ec43: XX 81 ec 00 03 00 00 sub    $0x300,%rsp
...
```

```
<foo>:
77ec3f: 55 -> cc -> e8        push
77ec40: cc XX XX              mov
77ec43: cc 81 ec 00 03 00 00 sub
...
```

**HUM.. call update**

- **figure out early optimization**
- **stop process**

# HUM.. need sysctl knob? ;-)



# UPROBE OVERRIDE

- control user space execution
- guard app against CVEs

# UPROBE OVERRIDE

## uprobe program can write to context (v6.18)

## ip change does not execute original instruction

```
SEC("uprobe")
int BPF_UPROBE(test)
{
  ctx->ax = 0xdead;
  ctx->ip = new_ip;
  return 0;
}
```

# UPROBE OVERRIDE

## function override like in kernel

```
<foo>:
push   %rbp
mov    %rsp,%rbp
sub    $0x300,%rsp
mov    %edi,%eax
mov    %al,-0x2f4(%rbp)
lea    -0x160(%rbp),%rax
mov    $0x30,%edx
mov    $0x0,%esi
mov    %rax,%rdi
```

**rip=(%rsp)**

**rsp=%rsp + 8**

# UPROBE OVERRIDE

## deep in the function

```
<foo>:
push   %rbp
mov    %rsp,%rbp
sub    $0x300,%rsp
mov    %edi,%eax
mov    %al,-0x2f4(%rbp)
lea    -0x160(%rbp),%rax
mov    $0x30,%edx
mov    $0x0,%esi
mov    %rax,%rdi
```

**rip=?**

**rsp=?**

**restore other callee saved registers**

**thanks, questions?**