PARIS
LODRON
UNIVERSITÄT
SALZBURG

Software Security

Your memory? Our memory! ☭

Mathias Bögl

Security Objectives

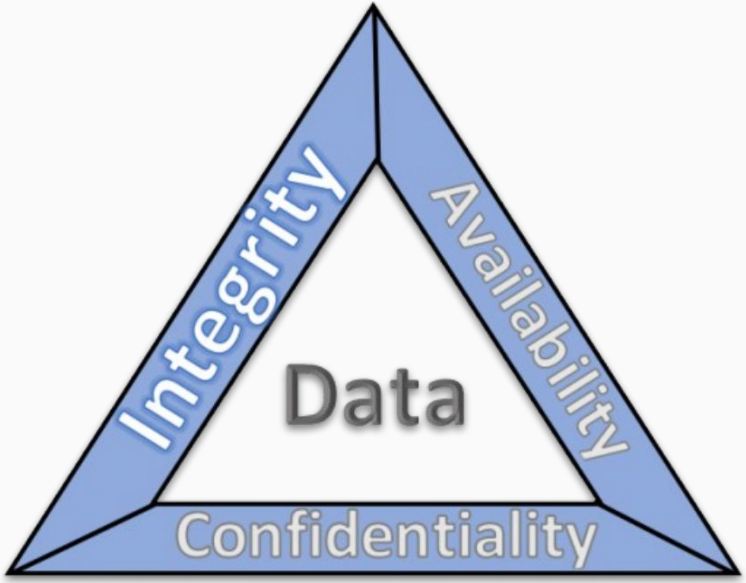


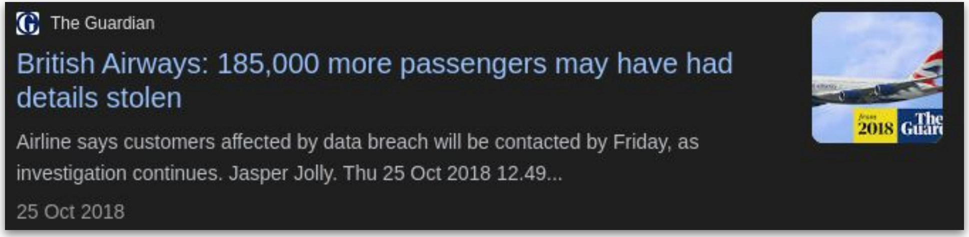
Fig.1: National Cybersecurity Center of Excellence, nccoe.nist.gov

Selection of past vulnerabilities

Web Based

Samy worm

British Airways

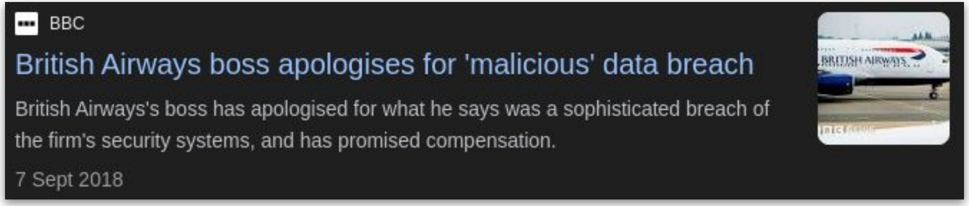


The Guardian

British Airways: 185,000 more passengers may have had details stolen

Airline says customers affected by data breach will be contacted by Friday, as investigation continues. Jasper Jolly. Thu 25 Oct 2018 12.49...

25 Oct 2018

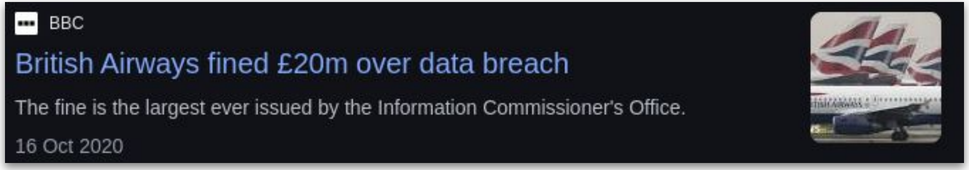


BBC

British Airways boss apologises for 'malicious' data breach

British Airways's boss has apologised for what he says was a sophisticated breach of the firm's security systems, and has promised compensation.

7 Sept 2018



BBC

British Airways fined £20m over data breach

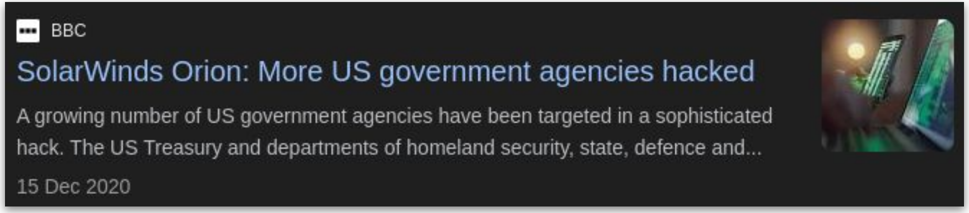
The fine is the largest ever issued by the Information Commissioner's Office.

16 Oct 2020

Supply Chain

SolarWinds

XZ

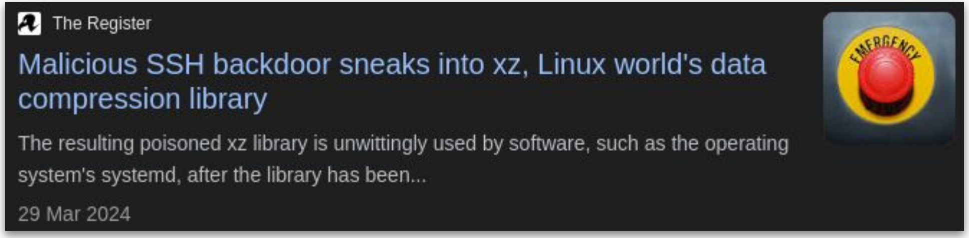


BBC

SolarWinds Orion: More US government agencies hacked

A growing number of US government agencies have been targeted in a sophisticated hack. The US Treasury and departments of homeland security, state, defence and...

15 Dec 2020

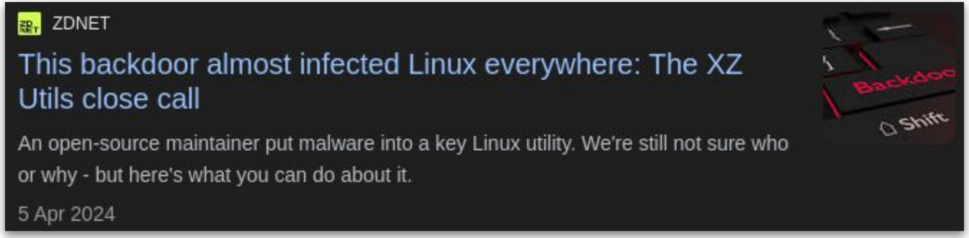


The Register

Malicious SSH backdoor sneaks into xz, Linux world's data compression library

The resulting poisoned xz library is unwittingly used by software, such as the operating system's systemd, after the library has been...

29 Mar 2024



ZDNET

This backdoor almost infected Linux everywhere: The XZ Utils close call

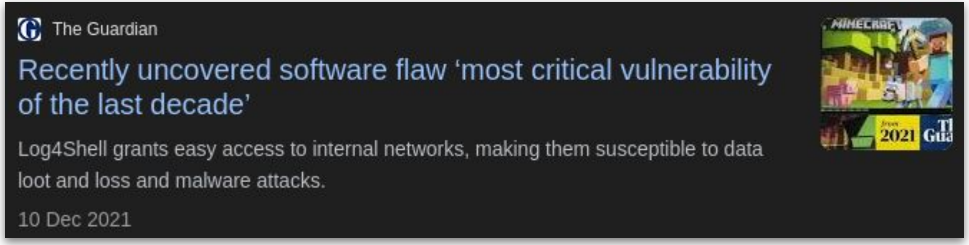
An open-source maintainer put malware into a key Linux utility. We're still not sure who or why - but here's what you can do about it.

5 Apr 2024

ACE/RCE

Log4Shell

CUPS

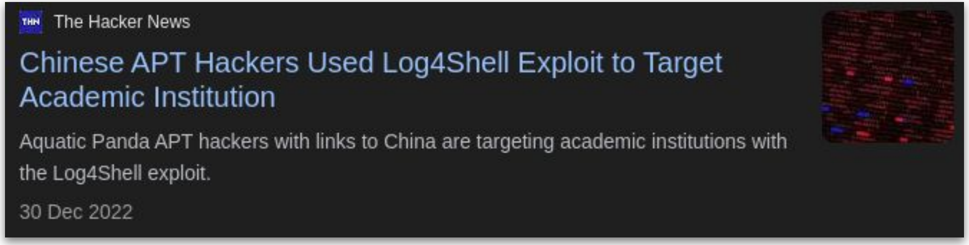


The Guardian

Recently uncovered software flaw 'most critical vulnerability of the last decade'

Log4Shell grants easy access to internal networks, making them susceptible to data loot and loss and malware attacks.

10 Dec 2021

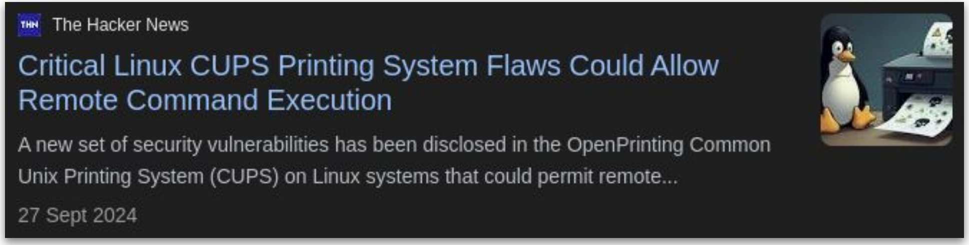


The Hacker News

Chinese APT Hackers Used Log4Shell Exploit to Target Academic Institution

Aquatic Panda APT hackers with links to China are targeting academic institutions with the Log4Shell exploit.

30 Dec 2022



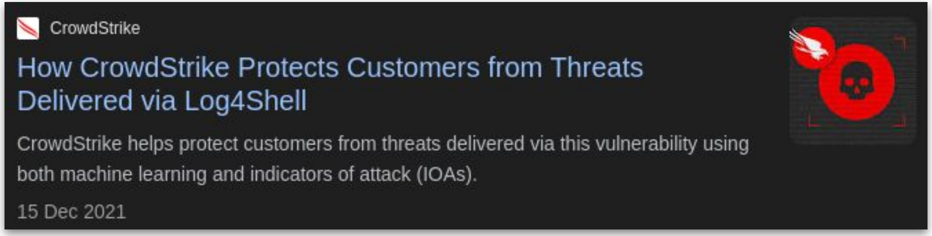
The Hacker News

Critical Linux CUPS Printing System Flaws Could Allow Remote Command Execution

A new set of security vulnerabilities has been disclosed in the OpenPrinting Common Unix Printing System (CUPS) on Linux systems that could permit remote...

27 Sept 2024

how it started



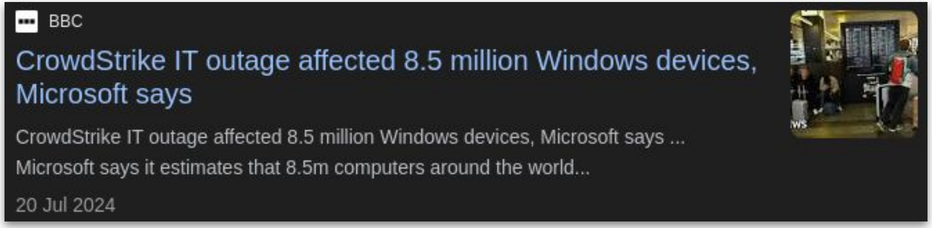
CrowdStrike

How CrowdStrike Protects Customers from Threats Delivered via Log4Shell

CrowdStrike helps protect customers from threats delivered via this vulnerability using both machine learning and indicators of attack (IOAs).

15 Dec 2021

how it’s going



BBC

CrowdStrike IT outage affected 8.5 million Windows devices, Microsoft says

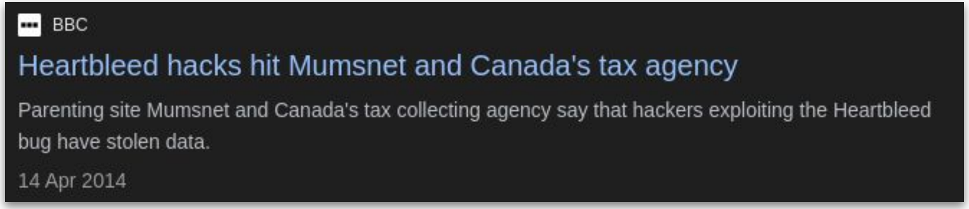
CrowdStrike IT outage affected 8.5 million Windows devices, Microsoft says ... Microsoft says it estimates that 8.5m computers around the world...

20 Jul 2024

Buffer Over-read

Heartbleed

CrowdStrike

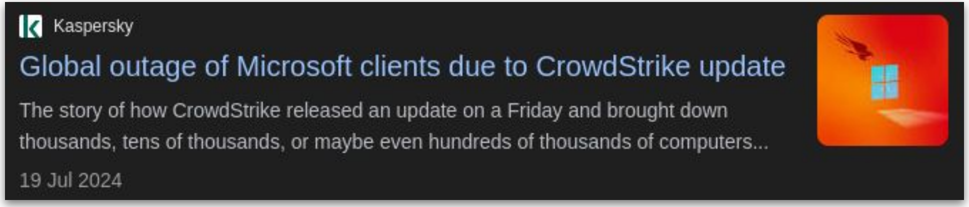


BBC

Heartbleed hacks hit Mumsnet and Canada's tax agency

Parenting site Mumsnet and Canada's tax collecting agency say that hackers exploiting the Heartbleed bug have stolen data.

14 Apr 2014

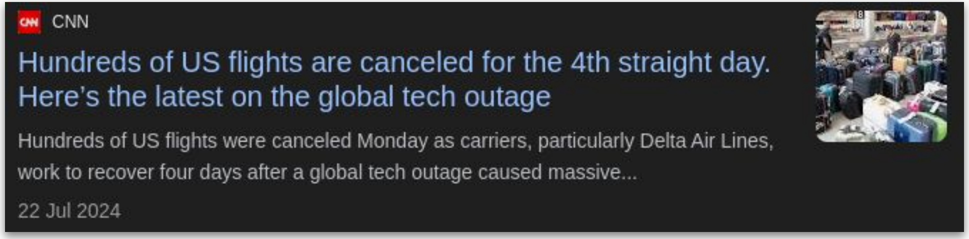


Kaspersky

Global outage of Microsoft clients due to CrowdStrike update

The story of how CrowdStrike released an update on a Friday and brought down thousands, tens of thousands, or maybe even hundreds of thousands of computers...

19 Jul 2024



CNN

Hundreds of US flights are canceled for the 4th straight day. Here’s the latest on the global tech outage

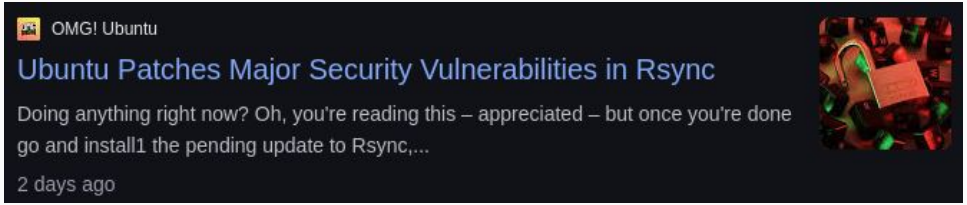
Hundreds of US flights were canceled Monday as carriers, particularly Delta Air Lines, work to recover four days after a global tech outage caused massive...

22 Jul 2024

Buffer Overflow

Rsync

Stuxnet

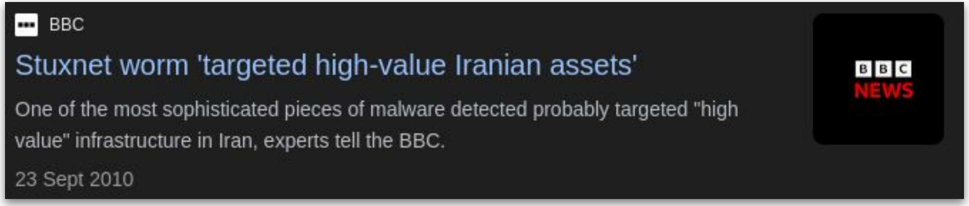


OMG! Ubuntu

Ubuntu Patches Major Security Vulnerabilities in Rsync

Doing anything right now? Oh, you're reading this – appreciated – but once you're done go and install1 the pending update to Rsync,...

2 days ago

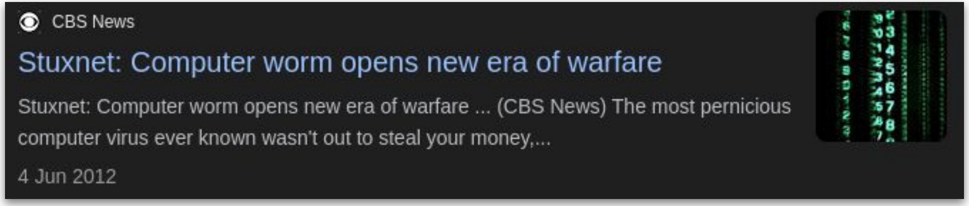


BBC

Stuxnet worm 'targeted high-value Iranian assets'

One of the most sophisticated pieces of malware detected probably targeted "high value" infrastructure in Iran, experts tell the BBC.

23 Sept 2010



CBS News

Stuxnet: Computer worm opens new era of warfare

Stuxnet: Computer worm opens new era of warfare ... (CBS News) The most pernicious computer virus ever known wasn't out to steal your money,...

4 Jun 2012

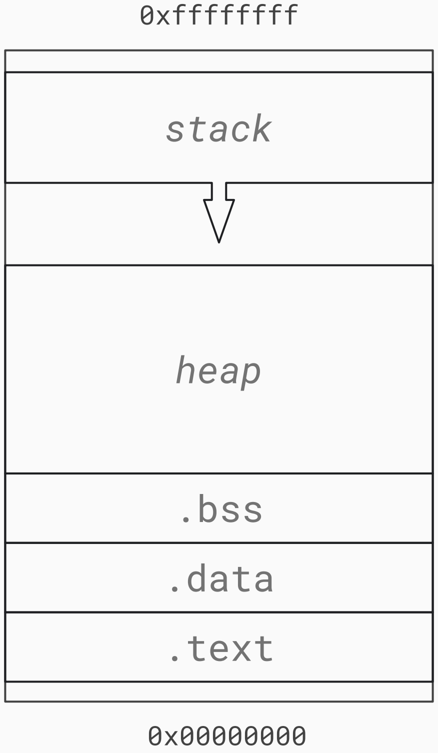
PARIS
LODRON
UNIVERSITÄT
SALZBURG

Memory Safety

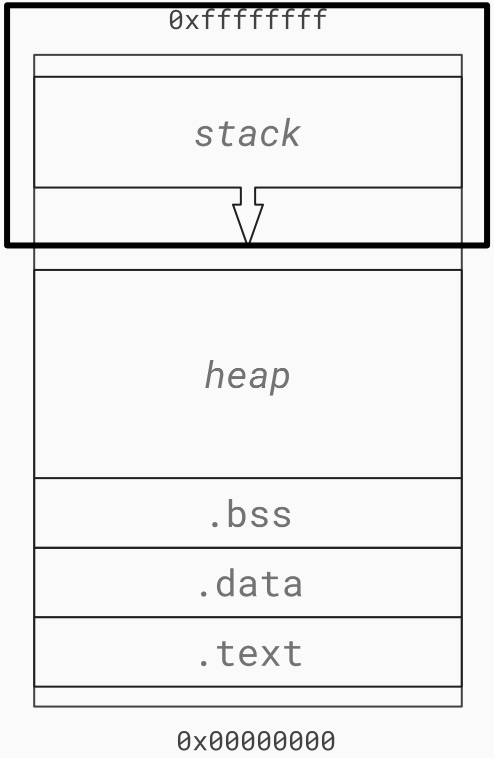
70%

https://www.zdnet.com/article/microsoft-70-percent-of-all-security-bugs-are-memory-safety-issues/

Revisited: Memory Layout



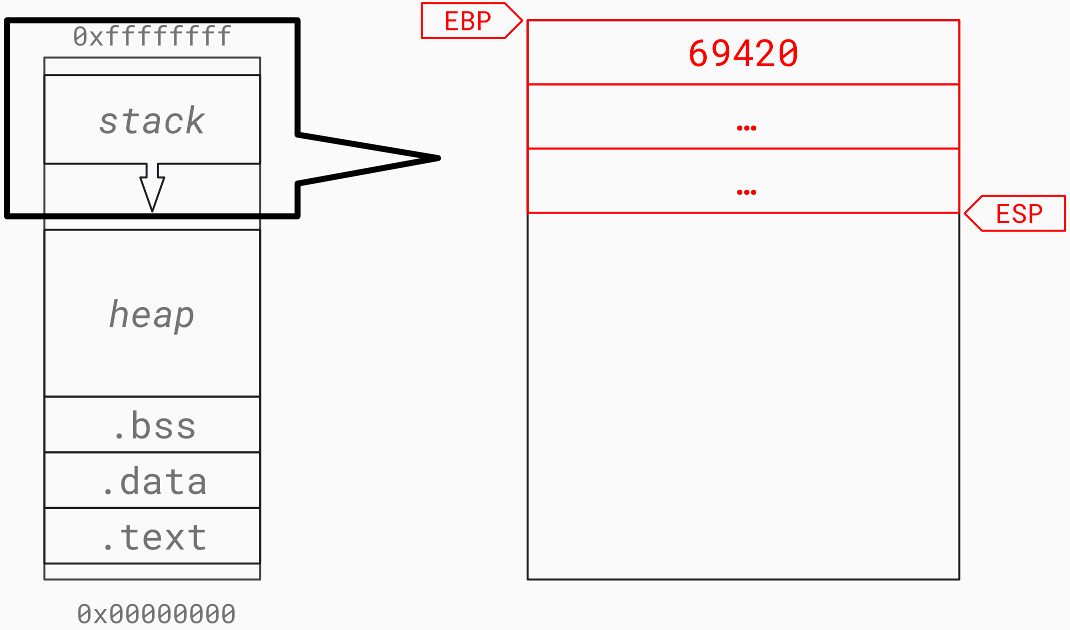
Revisited: Stack



Revisited: Stack

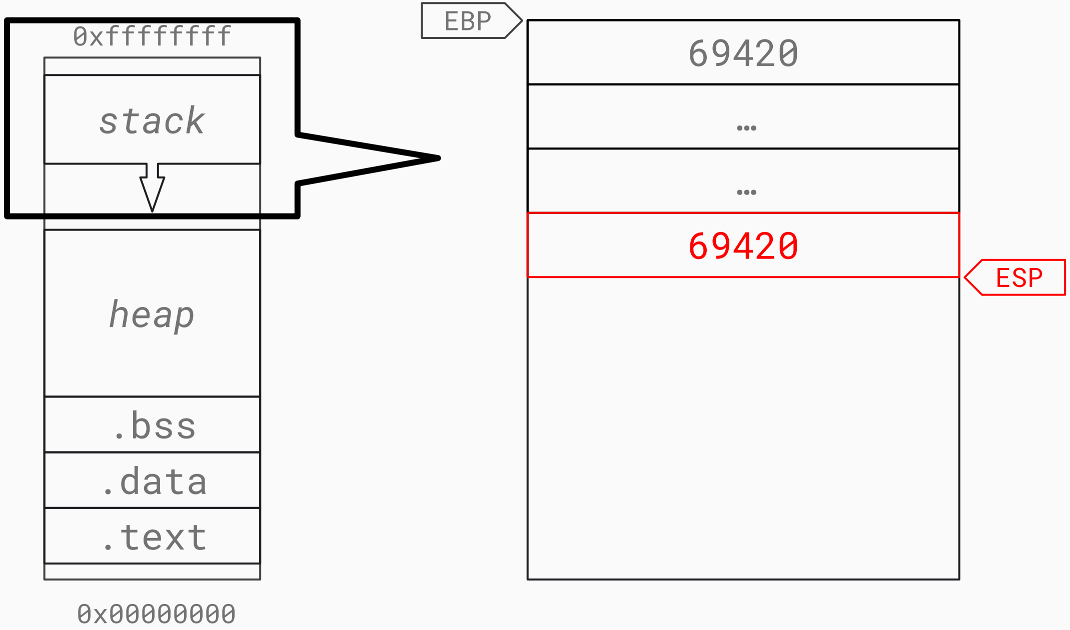
- LIFO stack in RAM
- special instructions (push/pop)
- special registers (esp/ebp)
- size of individual items known at compile time
- local variables, function params, state before function call, …

Revisited: Stack



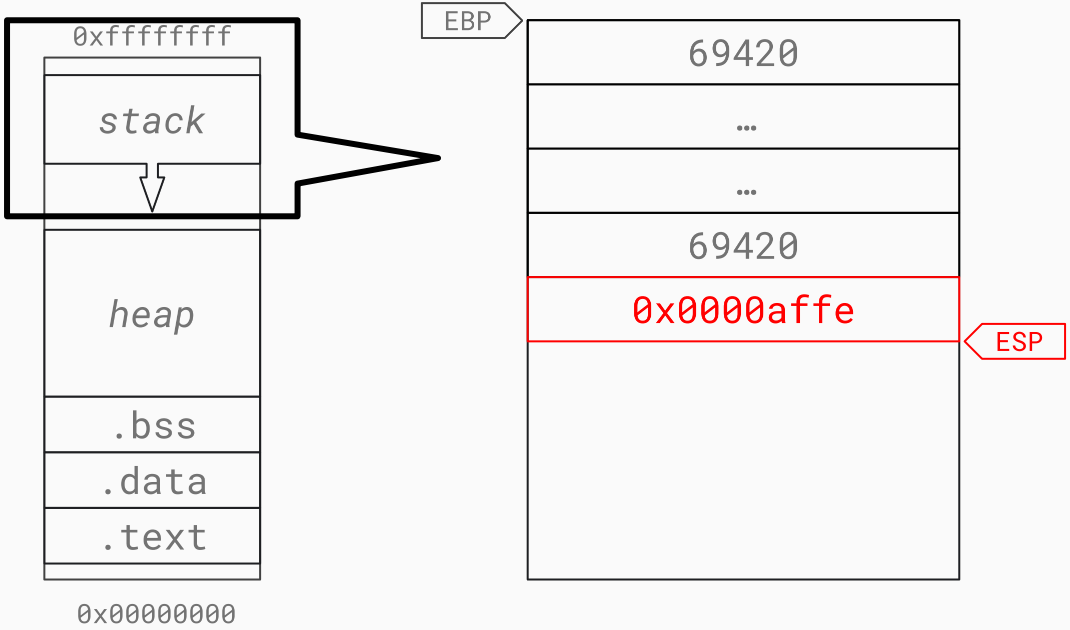
```
foo:
         …
EIP      push  -4(%ebp)
         call  bar
         add   $8, %esp
         …
bar:
         push  %ebp
         mov   %esp, %ebp
         …
         pop   %ebp
         ret
```

Revisited: Stack



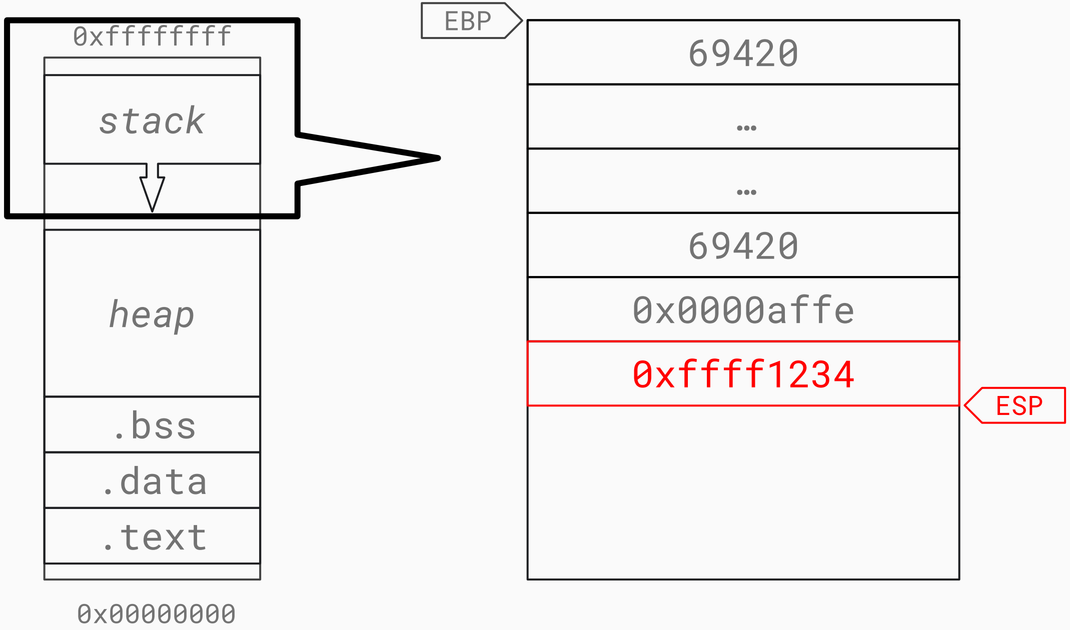
```
foo:
        …
        push  -4(%ebp)
EIP     call  bar
        add   $8, %esp
        …
bar:
        push  %ebp
        mov   %esp, %ebp
        …
        pop   %ebp
        ret
```

Revisited: Stack



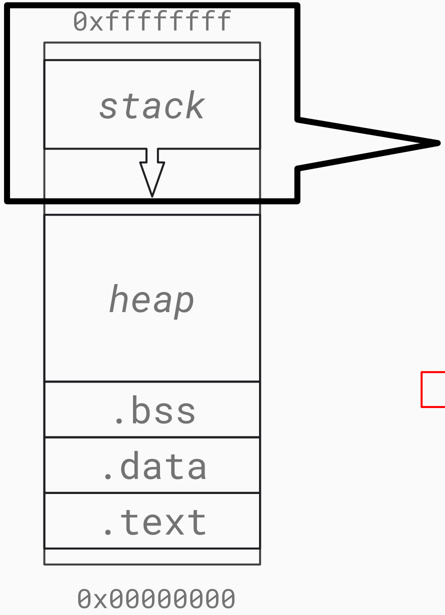
```
foo:
          …
          push  -4(%ebp)
          call  bar
    EIP   add   $8, %esp
          …
bar:
          push  %ebp
          mov   %esp, %ebp
          …
          pop   %ebp
          ret
```

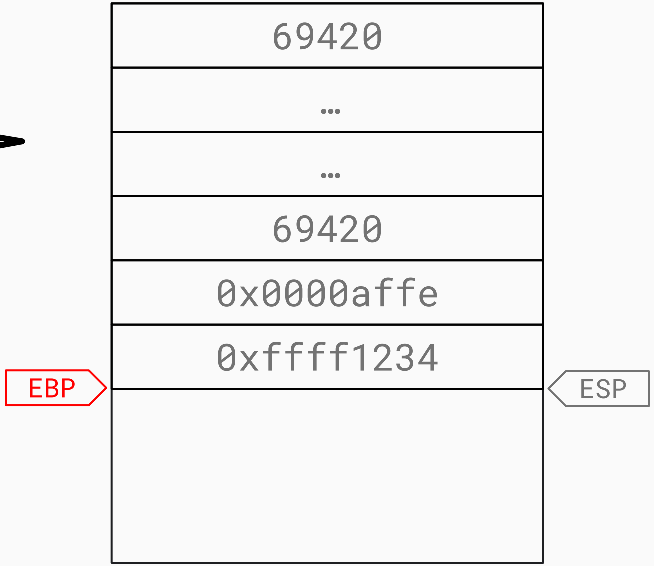
Revisited: Stack



```
foo:
      …
      push  -4(%ebp)
      call  bar
      add   $8, %esp
      …
bar:
      push  %ebp
EIP   mov   %esp, %ebp
      …
      pop   %ebp
      ret
```

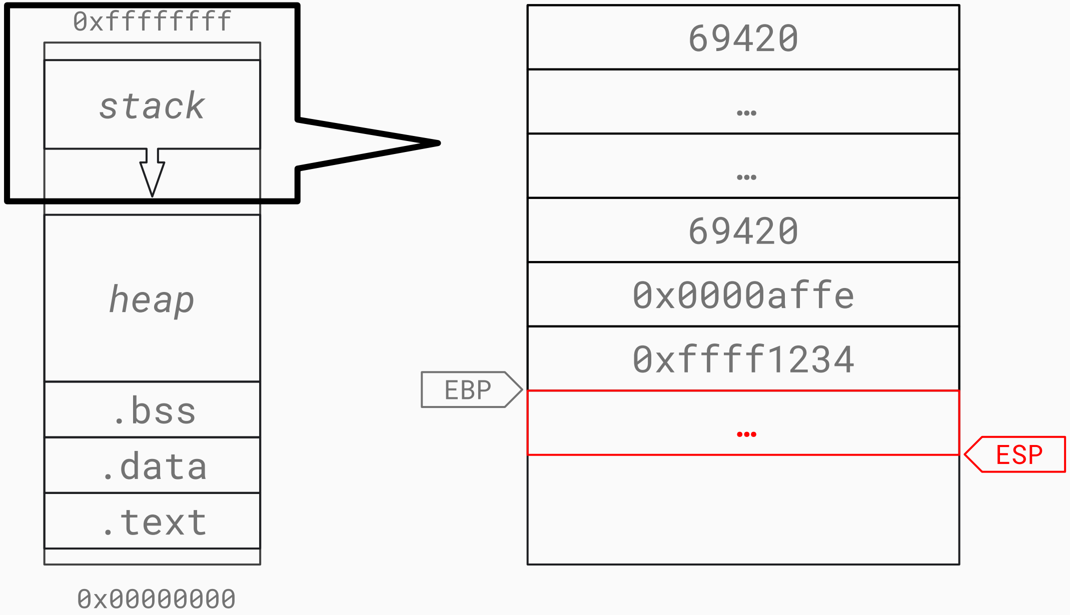
Revisited: Stack





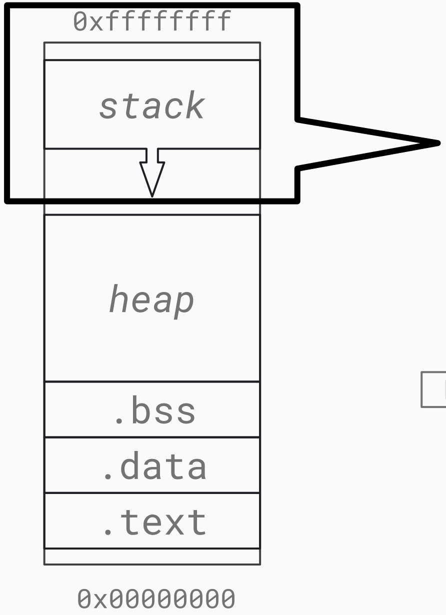
```
foo:
        …
        push  -4(%ebp)
        call  bar
        add   $8, %esp
        …
bar:
        push  %ebp
        mov   %esp, %ebp
EIP >   …
        pop   %ebp
        ret
```

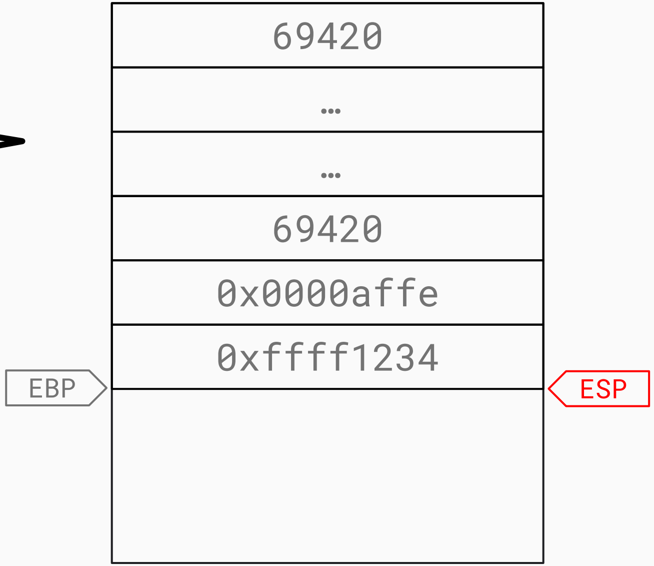
Revisited: Stack



```
foo:
        …
        push  -4(%ebp)
        call  bar
        add   $8, %esp
        …
bar:
        push  %ebp
        mov   %esp, %ebp
EIP     …
        pop   %ebp
        ret
```

Revisited: Stack

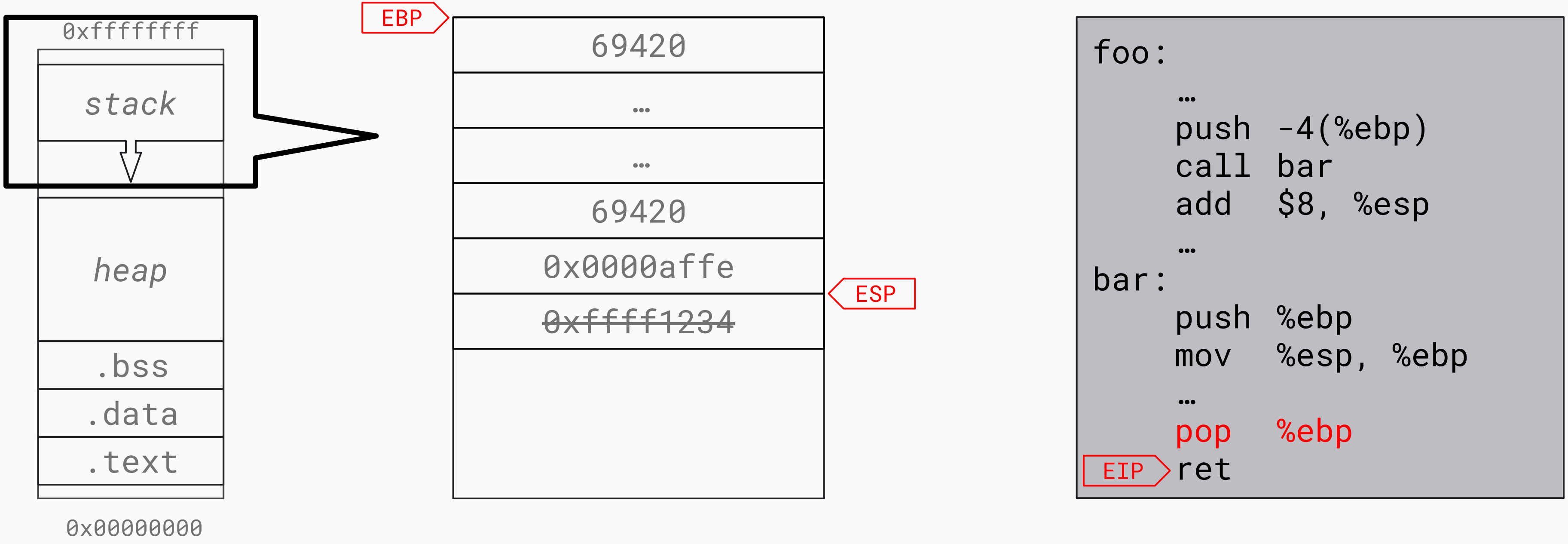




69420
…
…
69420
0x0000affe
0xffff1234

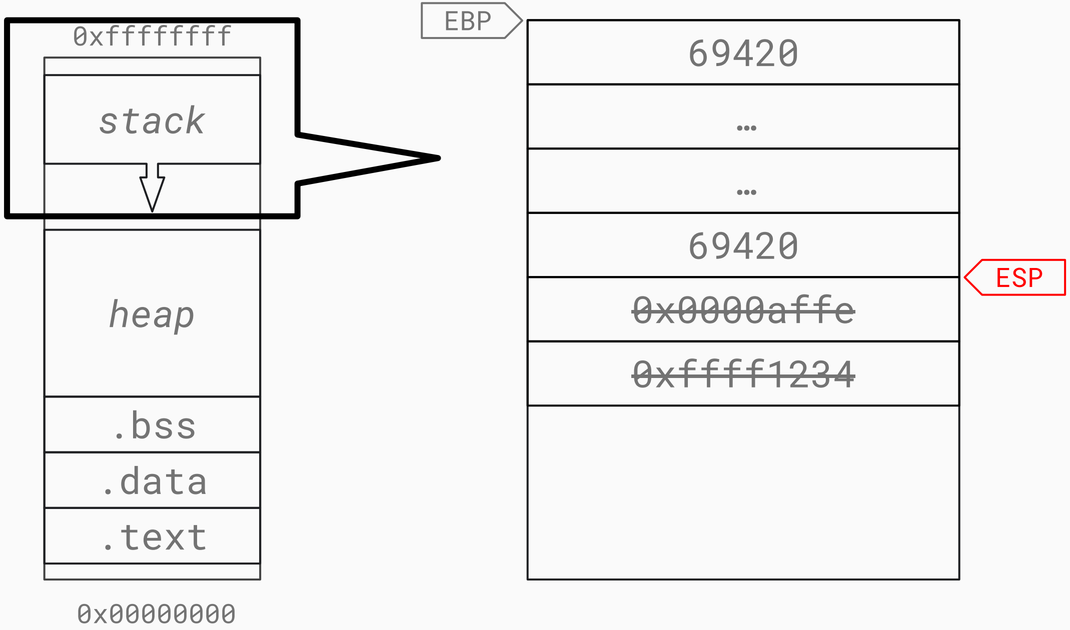
```
foo:
          …
          push  -4(%ebp)
          call  bar
          add   $8, %esp
          …
bar:
          push  %ebp
          mov   %esp, %ebp
          …
    EIP   pop   %ebp
          ret
```

Revisited: Stack



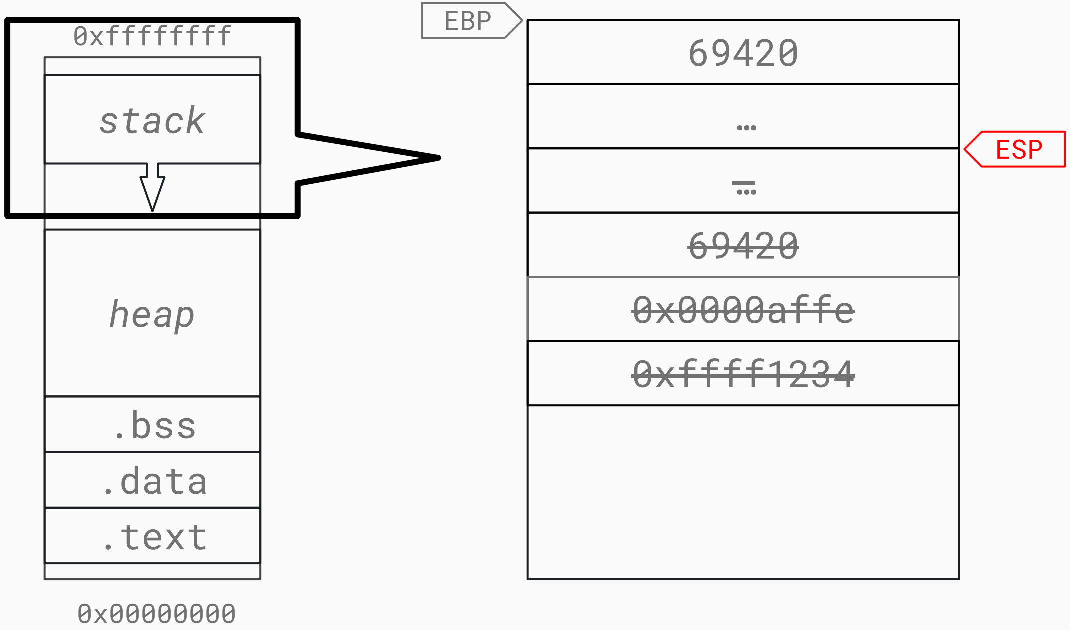
```
foo:
      …
      push -4(%ebp)
      call bar
      add  $8, %esp
      …
bar:
      push %ebp
      mov  %esp, %ebp
      …
      pop  %ebp
      ret
```

Revisited: Stack



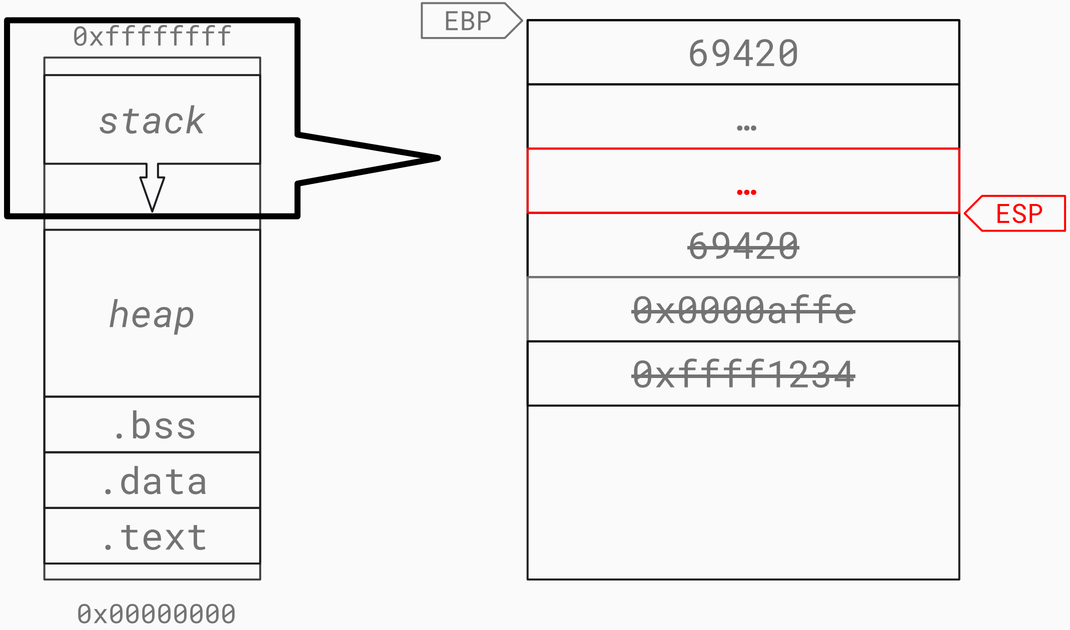
```
foo:
         …
         push  -4(%ebp)
         call  bar
 EIP     add   $8, %esp
         …
bar:
         push  %ebp
         mov   %esp, %ebp
         …
         pop   %ebp
         ret
```

Revisited: Stack



```
foo:
        …
        push  -4(%ebp)
        call  bar
        add   $8, %esp
  EIP   …
bar:
        push  %ebp
        mov   %esp, %ebp
        …
        pop   %ebp
        ret
```

Revisited: Stack

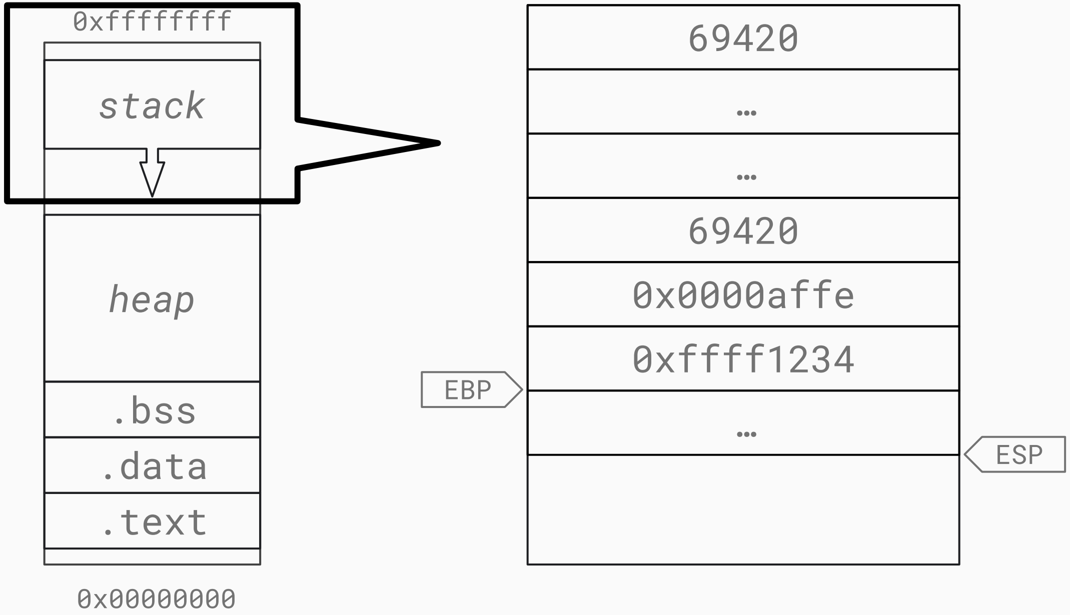


```
foo:
        …
        push  -4(%ebp)
        call  bar
        add   $8, %esp
  EIP   …
bar:
        push  %ebp
        mov   %esp, %ebp
        …
        pop   %ebp
        ret
```

Exploited: Stack

1. corrupt metadata like %RIP on stack
2. wait for function to return
3. function now “returns” to malicious location
4. success $$$

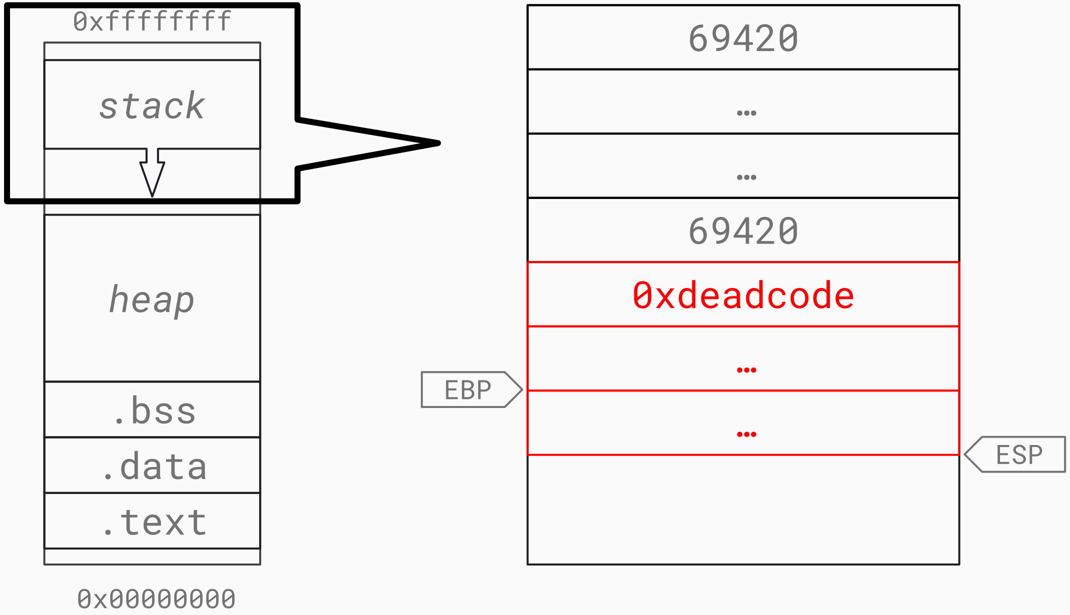
Exploited: Stack



```
malicious:
      …
      mov   $0xb, %al
      int   $0x80

bar:
      push  %ebp
      mov   %esp, %ebp
EIP   …
      pop   %ebp
      ret
```

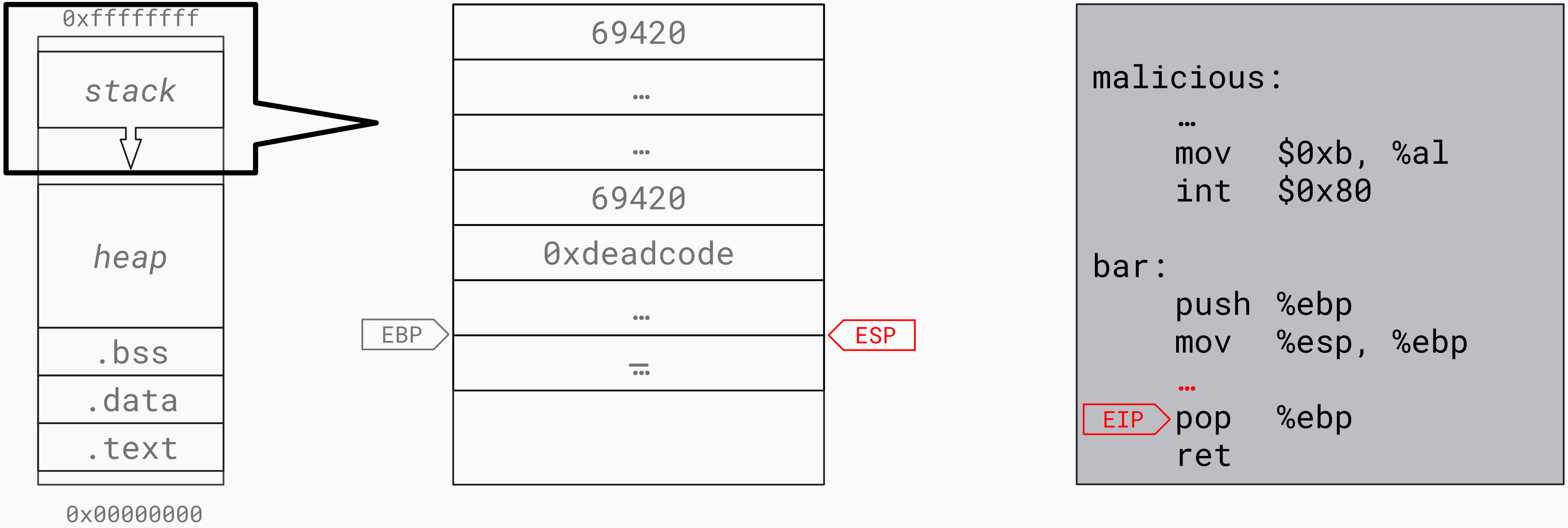
Exploited: Stack



```
malicious:
        …
        mov   $0xb, %al
        int   $0x80

bar:
        push  %ebp
        mov   %esp, %ebp
EIP     …
        pop   %ebp
        ret
```

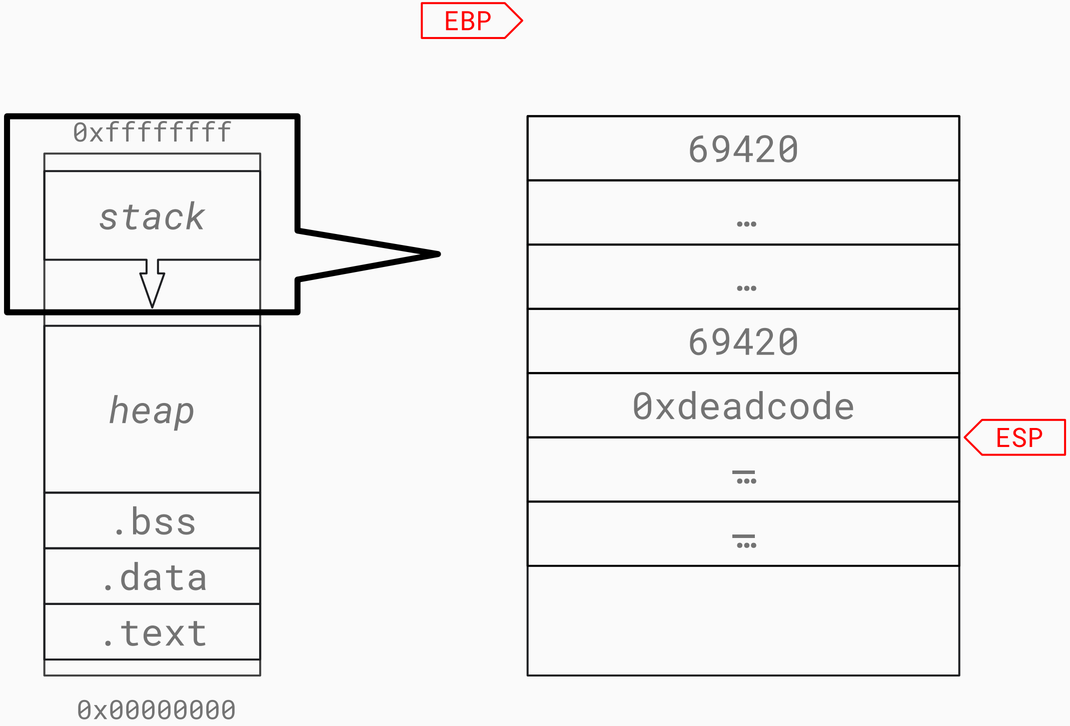
Exploited: Stack



```
malicious:
      …
      mov   $0xb, %al
      int   $0x80

bar:
      push  %ebp
      mov   %esp, %ebp
      …
EIP > pop   %ebp
      ret
```

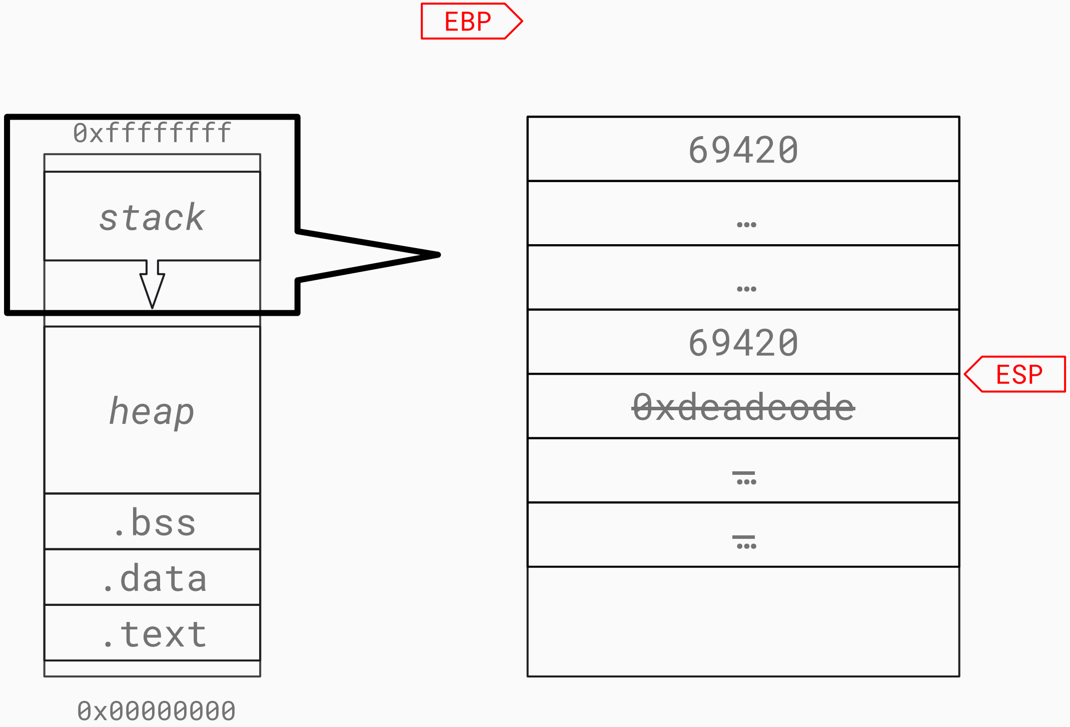
Exploited: Stack



```
malicious:
          …
          mov   $0xb, %al
          int   $0x80

bar:
          push  %ebp
          mov   %esp, %ebp
          …
          pop   %ebp
 EIP      ret
```

Exploited: Stack



```
malicious:
EIP   …
      mov   $0xb, %al
      int   $0x80

bar:
      push  %ebp
      mov   %esp, %ebp
      …
      pop   %ebp
      ret
```

Live Demo

Mitigations: Stack Canaries

- adds “magic” canary in function prologue
- value is validated before returning
- per default only for some vulnerable functions (gcc)
- **“-fstack-protector” (gcc)**

Mitigations: NX Bit

- marks memory regions as non executable (in MMU)
- does NOT prevent function pointer override
- on per default when using mmap
- set “prot” param to PROT_EXEC to disable

Mitigations:
RELRO

- marks (all) parts of the GOT readonly
- partial per default, only against global var overflows
- full can drastically increase load times of exe
- **“-z relro” + “-z now” for full ro-mode (gcc)**

Mitigations: ASLR

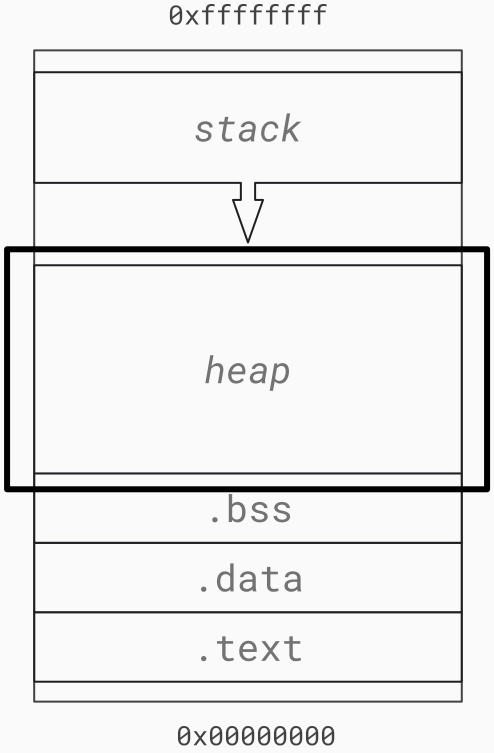
- randomizes virtual memory addresses
- usually just for stack, heap and shared libs
- depends on OS and configuration
- **“-fPIE” (gcc) AND “sysctl kernel.randomize_va_space=2” (UNIX-like)**

Software Security

Your memory? Our memory! ☭

Mathias Bögl

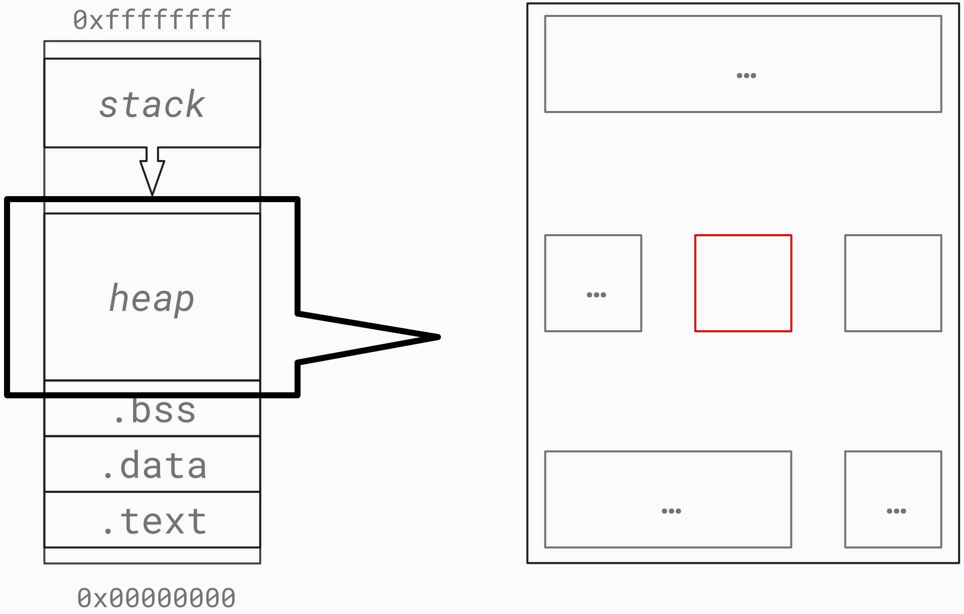
Revisited: Heap



Revisited: Heap

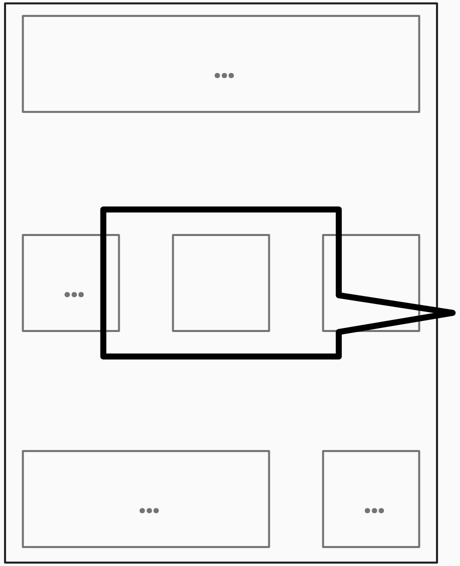
- managed by runtime (e.g. glibc)
- OS supplies chunk of (RAM) memory (mmap/sbrk)
- tree or linked list to track blocks in chunk
- size of individual items known at runtime
- big and/or long living variables (via malloc/new and free/delete)

Revisited: Heap



```
int main()
{
   …
   uint8_t *buf;
   buf = malloc(1024);
   memset(buf,'A',1024);
   …
   free(buf);
   …
}
```

Revisited: Heap



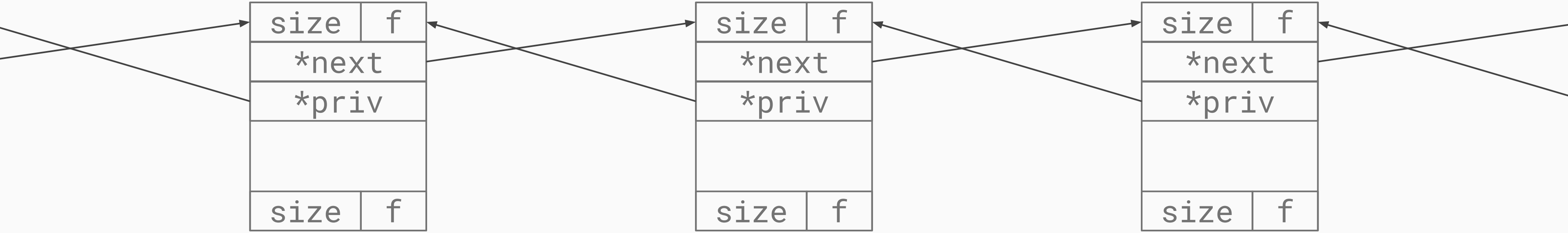
Allocated

size	flags
data	
size	flags

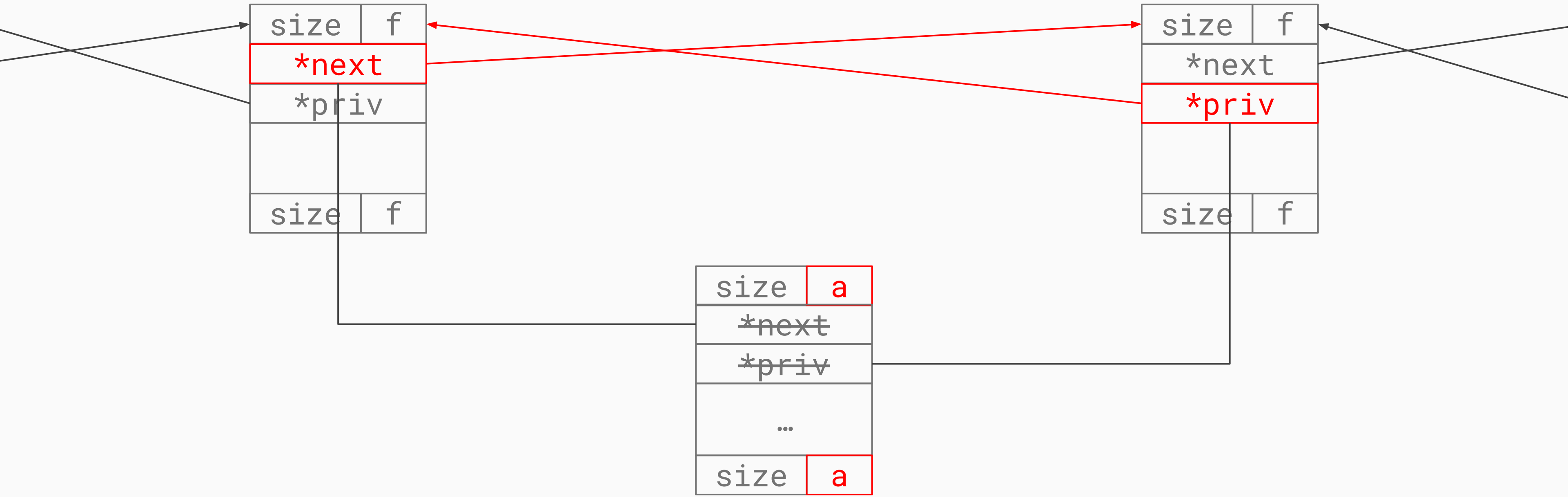
Free

size	flags
*next	
*priv	
size	flags

Revisited: Heap



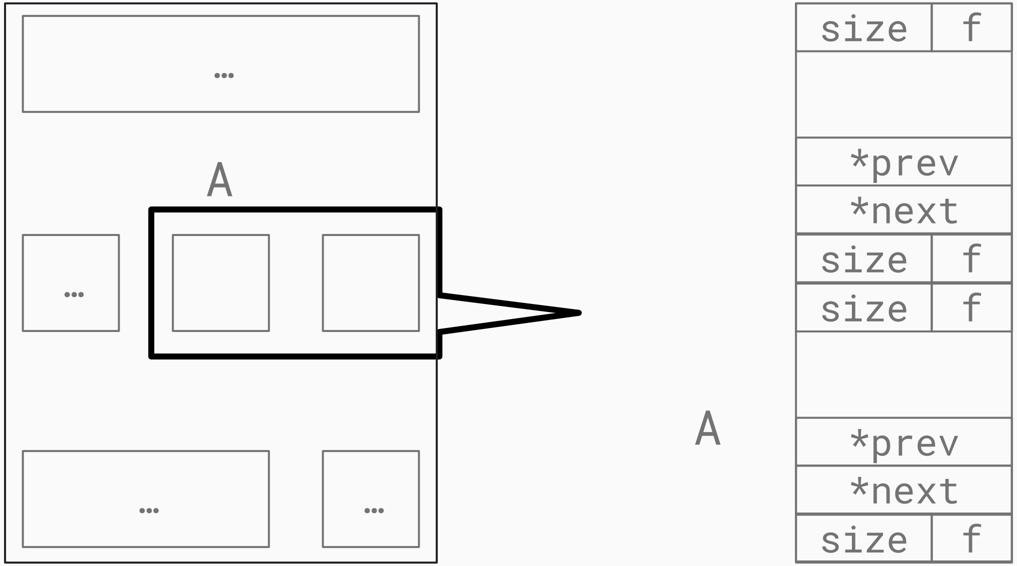
Revisited: Heap



Exploited: Heap

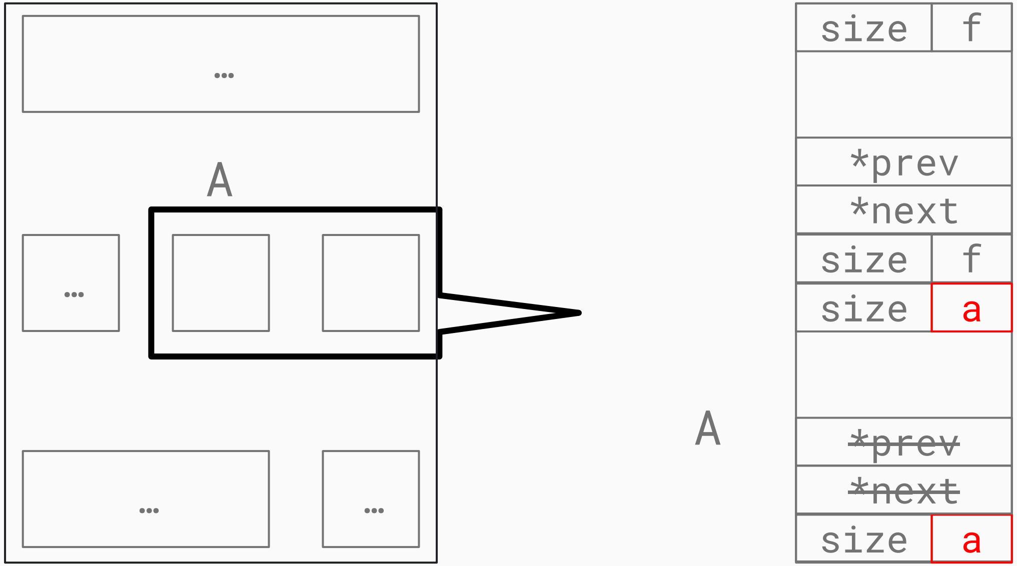
1. corrupt metadata of the next block (size, pointers, …)
2. wait for allocator to run on corrupted block
3. allocator now writes new “metadata” at malicious location
4. wait for malicious location to be used (e.g. function pointer in GOT)
5. success $$$

Exploited: Heap



```
int main()
{
  …
  A = malloc(…);
  …
  gets(&A);
  …
  X = malloc(…);
  …
}
```

Exploited: Heap



size f

*prev

*next

size f

size a

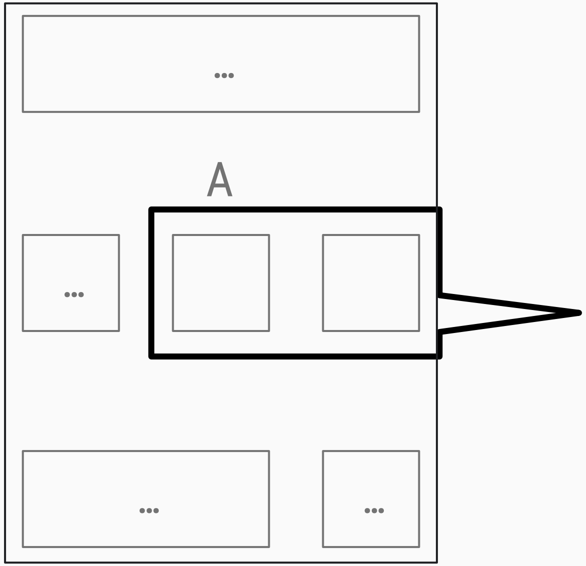
~~*prev~~

~~*next~~

size a

```
int main()
{
   …
   A = malloc(…);
   …
   gets(&A);
   …
   X = malloc(…);
   …
}
```

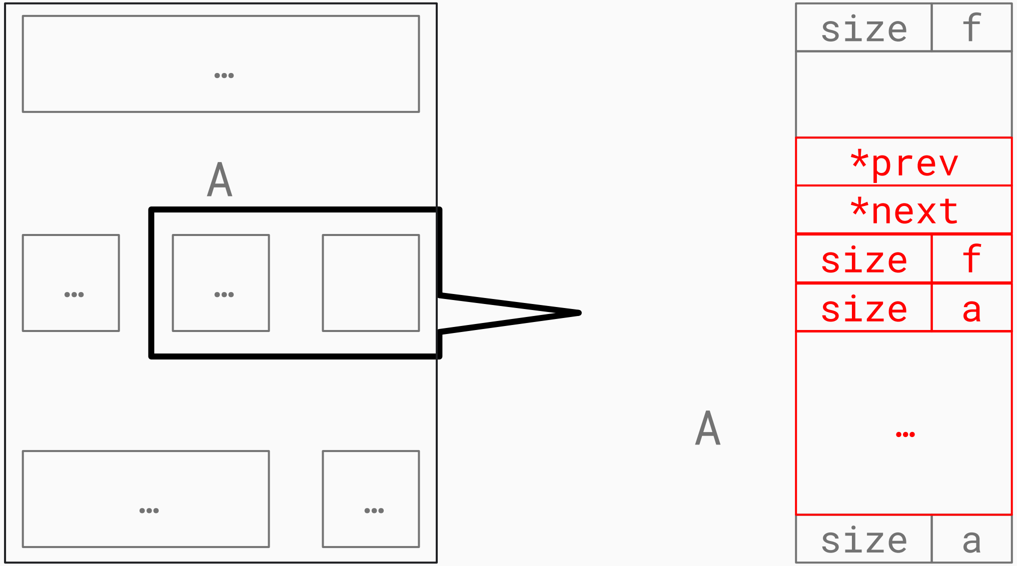
Exploited: Heap



	size	f
	*prev	
	*next	
	size	f
	size	a
A	~~*prev~~	
	~~*next~~	
	size	a

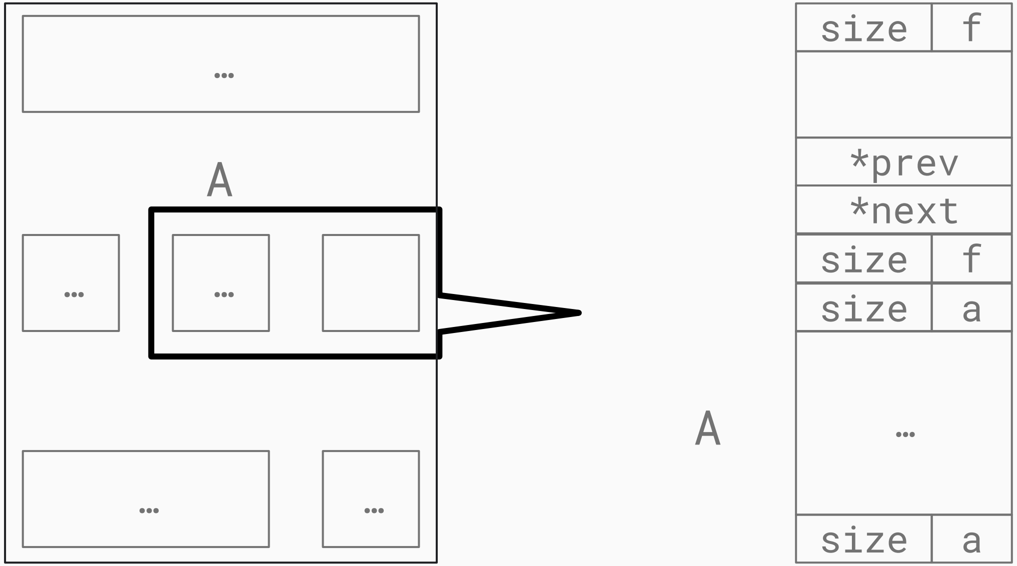
```
int main()
{
   …
   A = malloc(…);
   …
   gets(&A);
   …
   X = malloc(…);
   …
}
```

Exploited: Heap



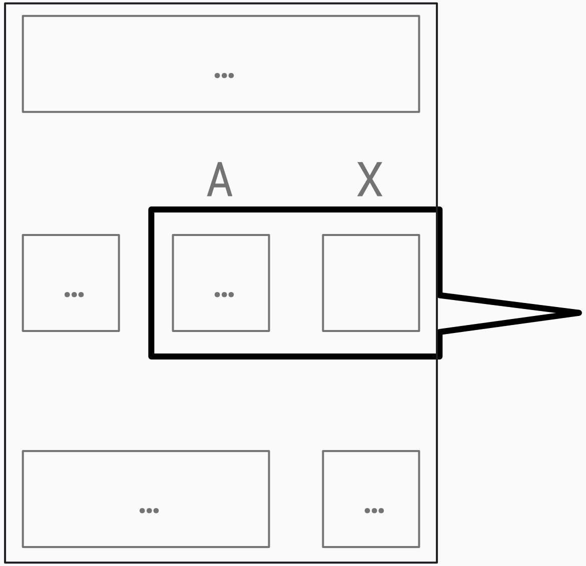
```
int main()
{
   …
   A = malloc(…);
   …
   gets(&A);
   …
   X = malloc(…);
   …
}
```

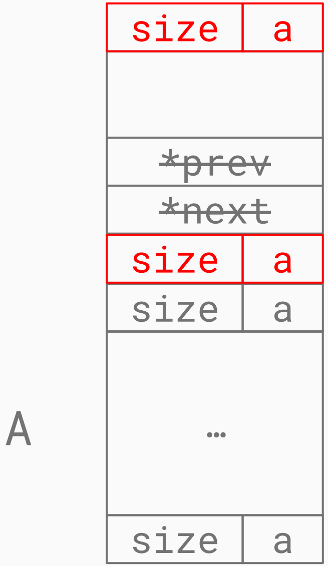
Exploited: Heap



```
int main()
{
  …
  A = malloc(…);
  …
  gets(&A);
  …
  X = malloc(…);
  …
}
```

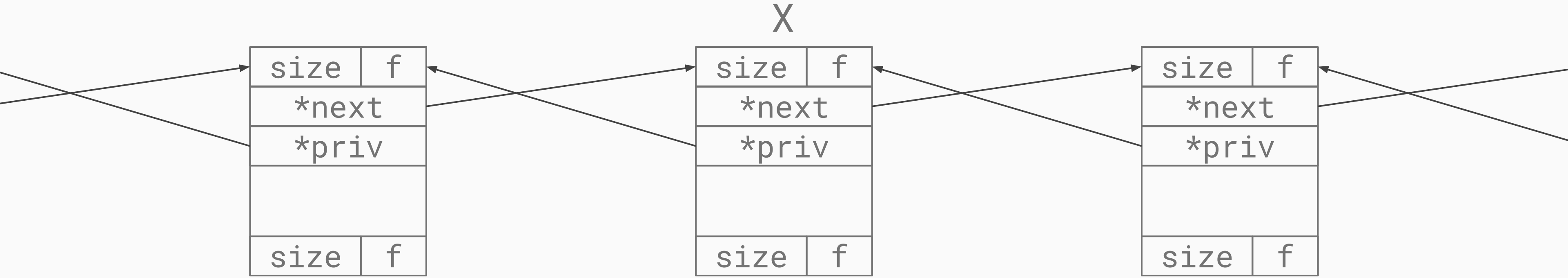
Exploited: Heap



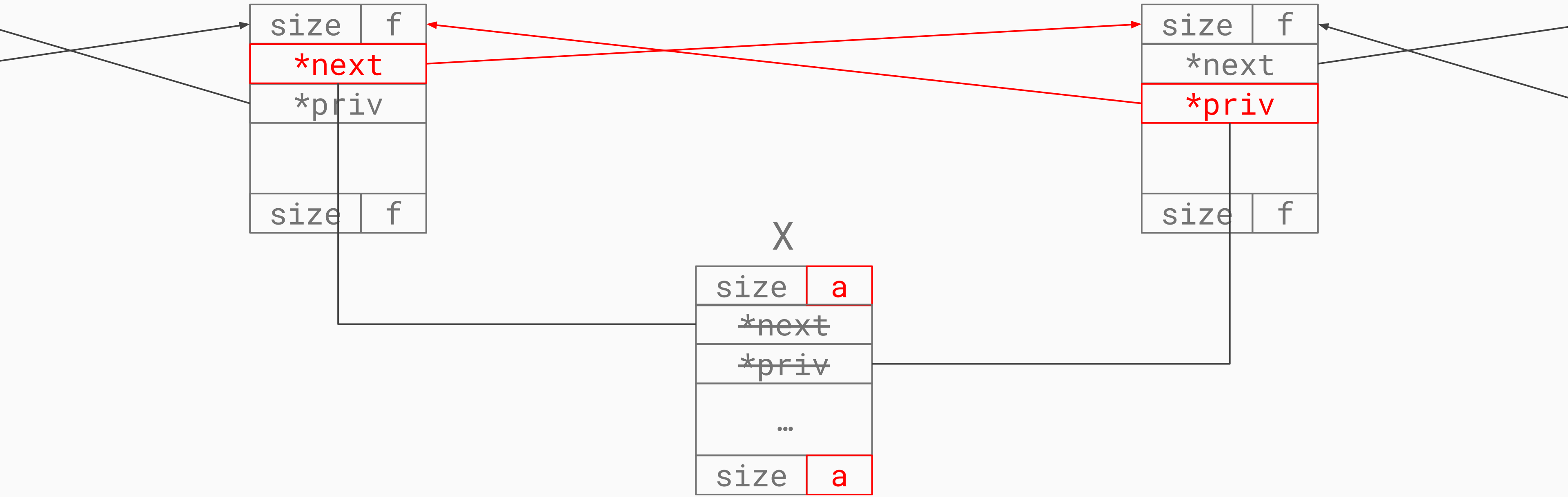


```
int main()
{
   …
   A = malloc(…);
   …
   gets(&A);
   …
   X = malloc(…);
   …
}
```

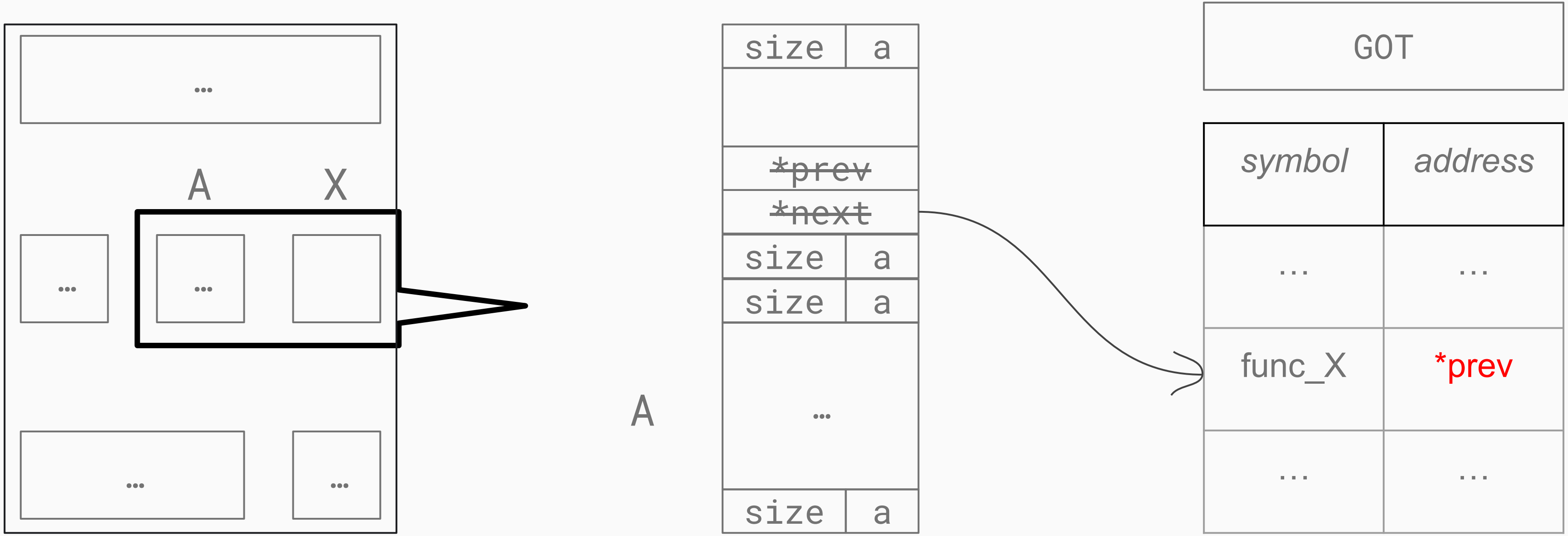
Revisited: Heap



Revisited: Heap



Exploited: Heap



GOT

symbol	*address*
…	…
func_X	*prev
…	…

PARIS
LODRON
UNIVERSITÄT
SALZBURG

Software Security

Your memory? Our memory! ☭

Mathias Bögl