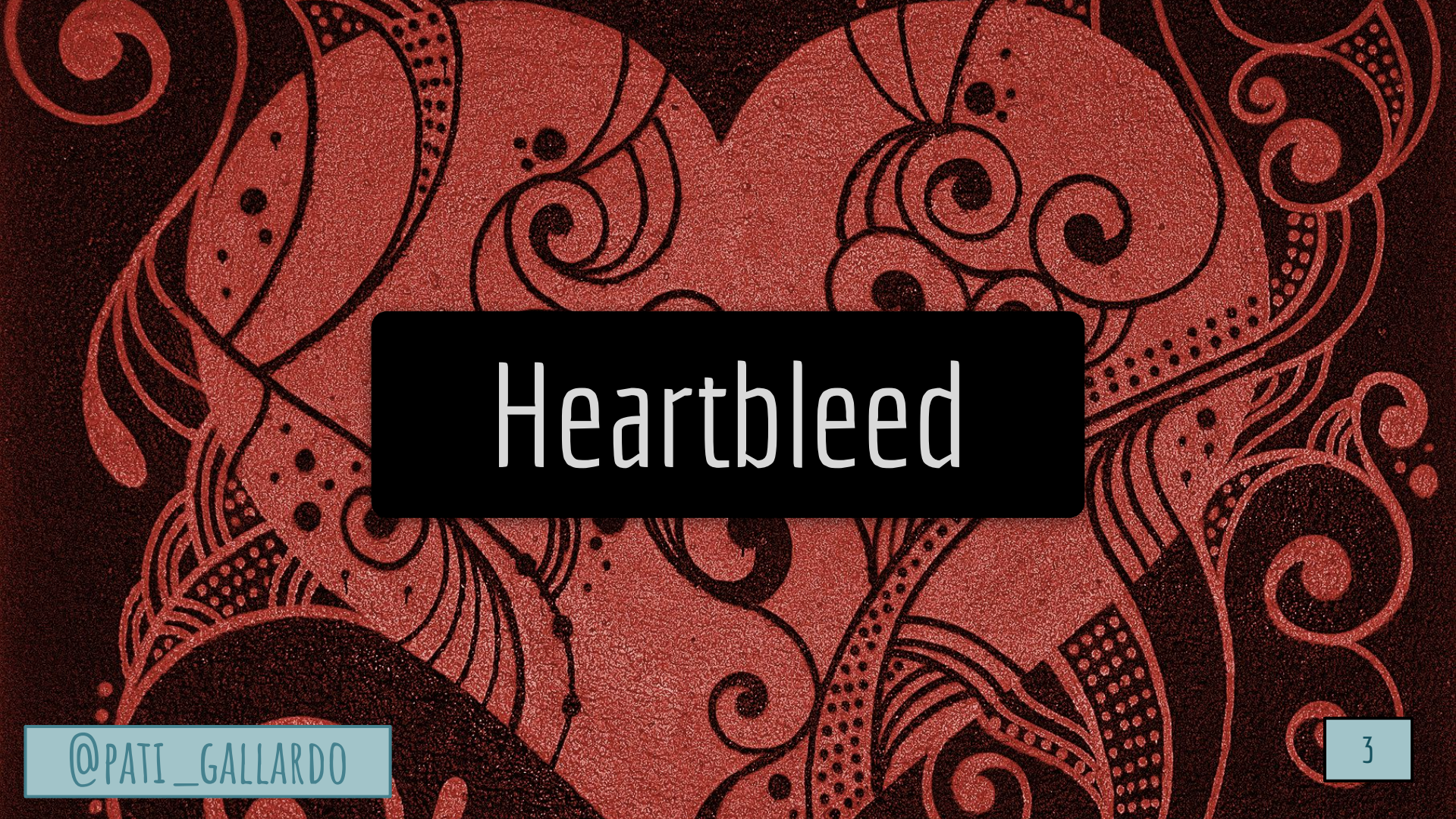

5
amboss
@PATI_GALLARDO
Turtle
Sec

"Basically, an attacker can grab 64K of memory from a server.
The attack leaves no trace, and can be done multiple times
to grab a different random 64K of memory.
This means that anything in memory
-- SSL private keys, user keys, anything -- is vulnerable.
And you have to assume that it is all compromised. All of it.

"Catastrophic" is the right word.
On the scale of 1 to 10, this is an 11."

https://www.schneier.com/blog/archives/2014/04/heartbleed.html


Heartbleed

What was the bug?

- Buffer over-read
- Attacker controlled buffer size

What made it bad?

- Remote attack
- High value memory
- Wide deploy

CVE-2014-0160 Description

TurtleSec

"The (1) TLS and (2) DTLS implementations in OpenSSL 1.0.1 before 1.0.1g **do not properly handle Heartbeat Extension packets**,
which allows remote attackers to **obtain sensitive information from process memory** via crafted packets that trigger a buffer over-read"

Heartbleed is a prime example of an

Information Leak

Heartbleed is famous for how devastating it was
But it also became
the poster child for fuzzing

Introduction to Memory Exploitation

CppEurope 2021

Patricia Aas

@pati_gallardo

Patricia Aas - Trainer & Consultant

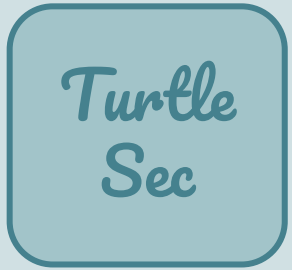
TurtleSec

C++ Programmer, Application Security
Currently : **TurtleSec**

Previously : Vivaldi, Cisco Systems, Knowit, Opera Software
Master in Computer Science

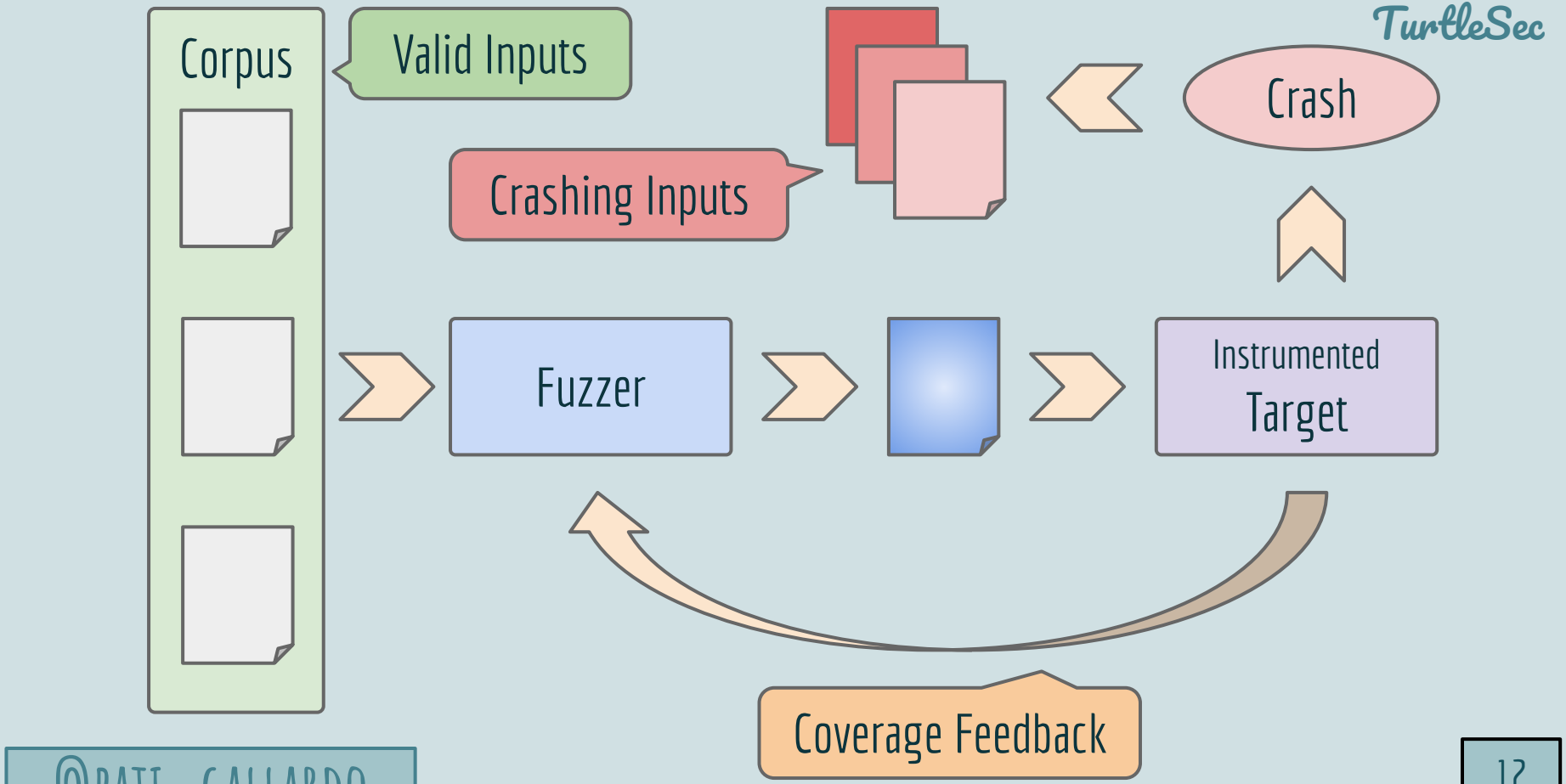
Pronouns: she/they

@PATI_GALLARDO



Fuzzing

@PATI_GALLARDO


Corpus
Valid Inputs
Crashing Inputs
Crash
Fuzzer
Instrumented
Target
Coverage Feedback
TurtleSec
@PATI_GALLARDO

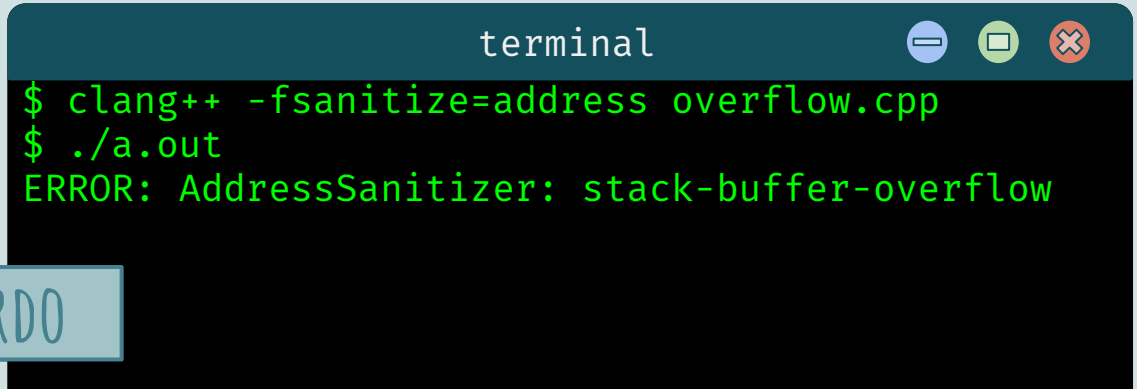
Now that's is all nice and good
But most memory errors don't cause us to crash
At least not right away

Sanitizers

Address Sanitizer

TurtleSec

compiler instrumentation
run-time library



VS

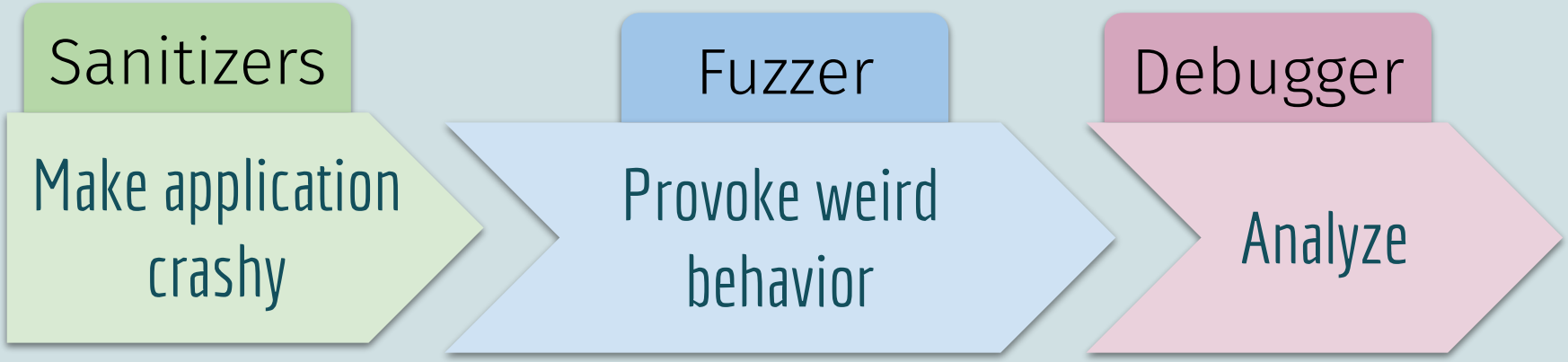
GCC

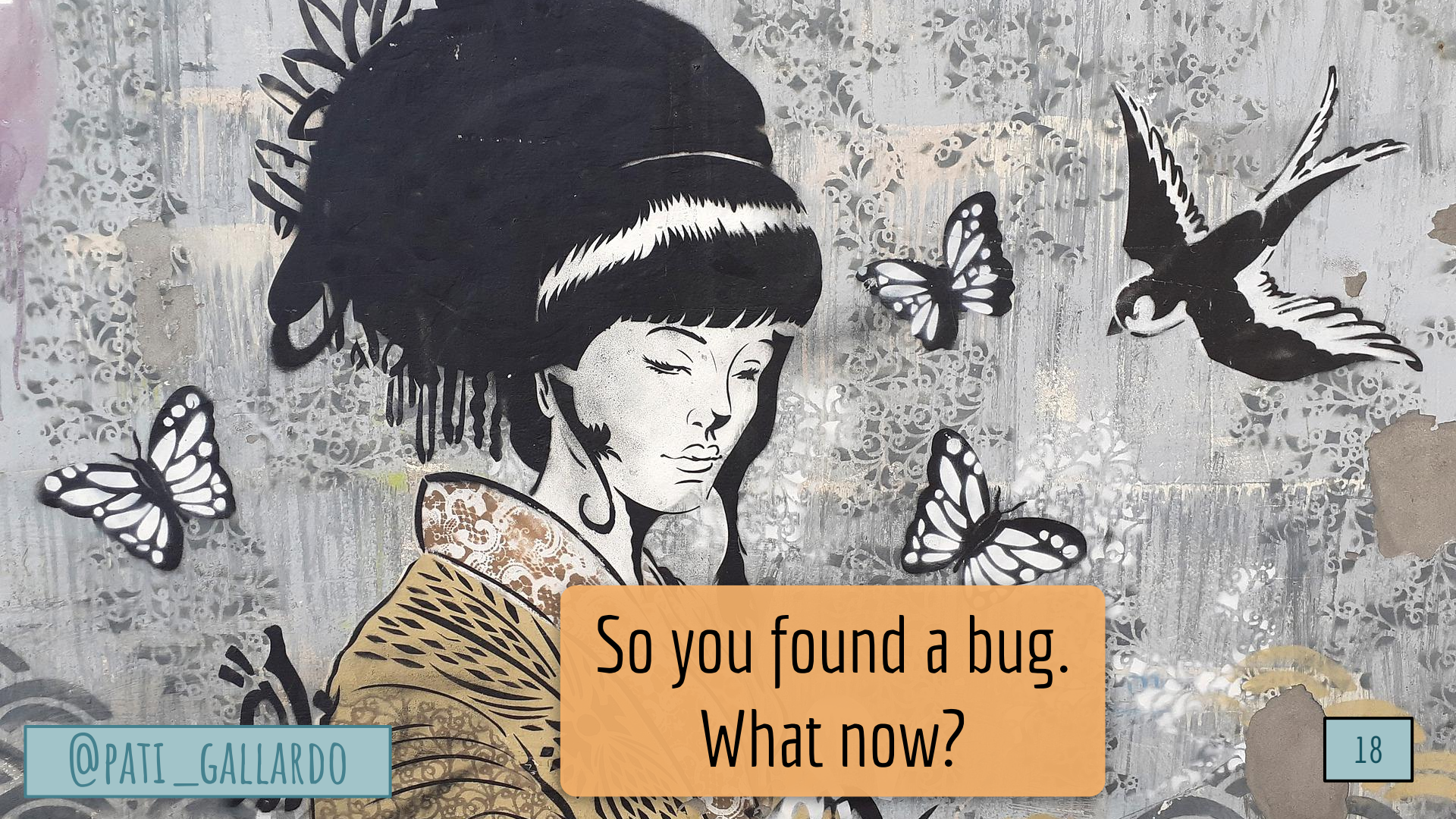
Clang

Address Sanitizer provokes crash-like behavior for many memory bugs

Supercharges fuzzing

Makes it possible to find “hidden” bugs

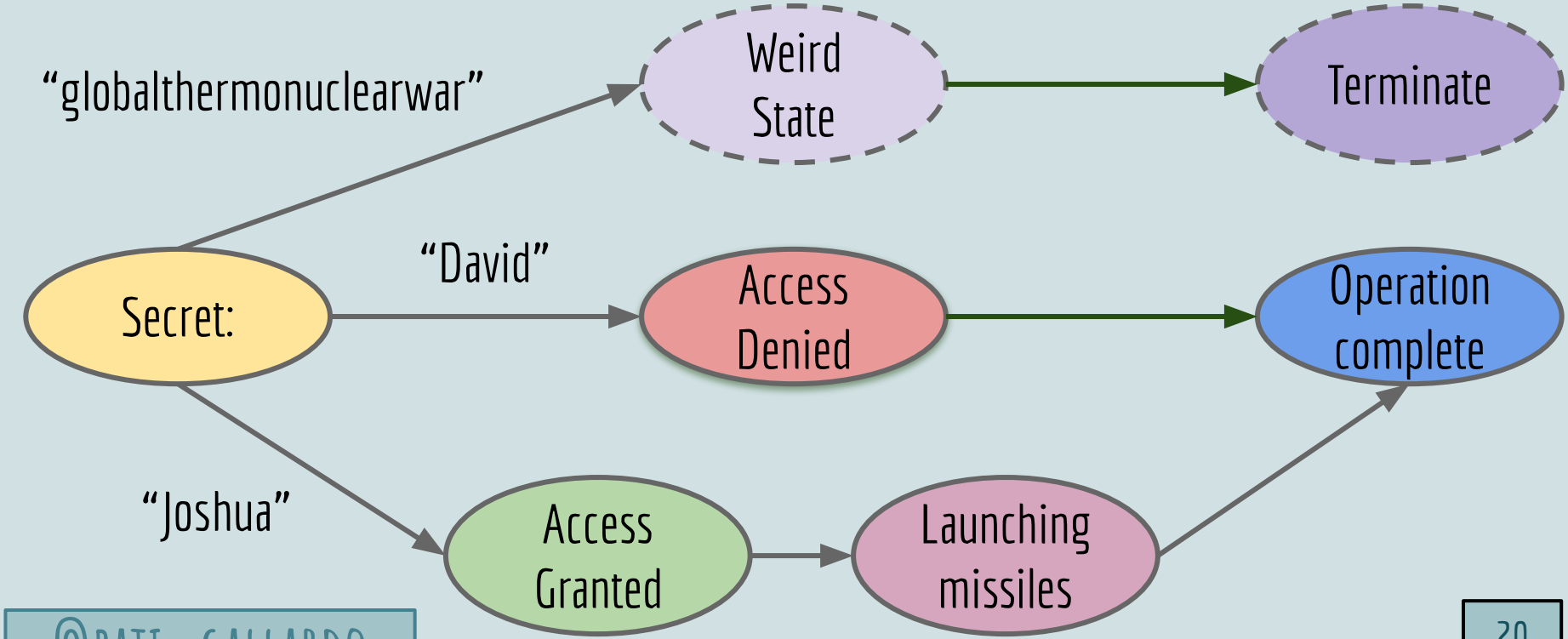

Sanitizers
Make application crashy
Fuzzer
Provoke weird behavior
Debugger
Analyze


So you found a bug.
What now?

Exploitation

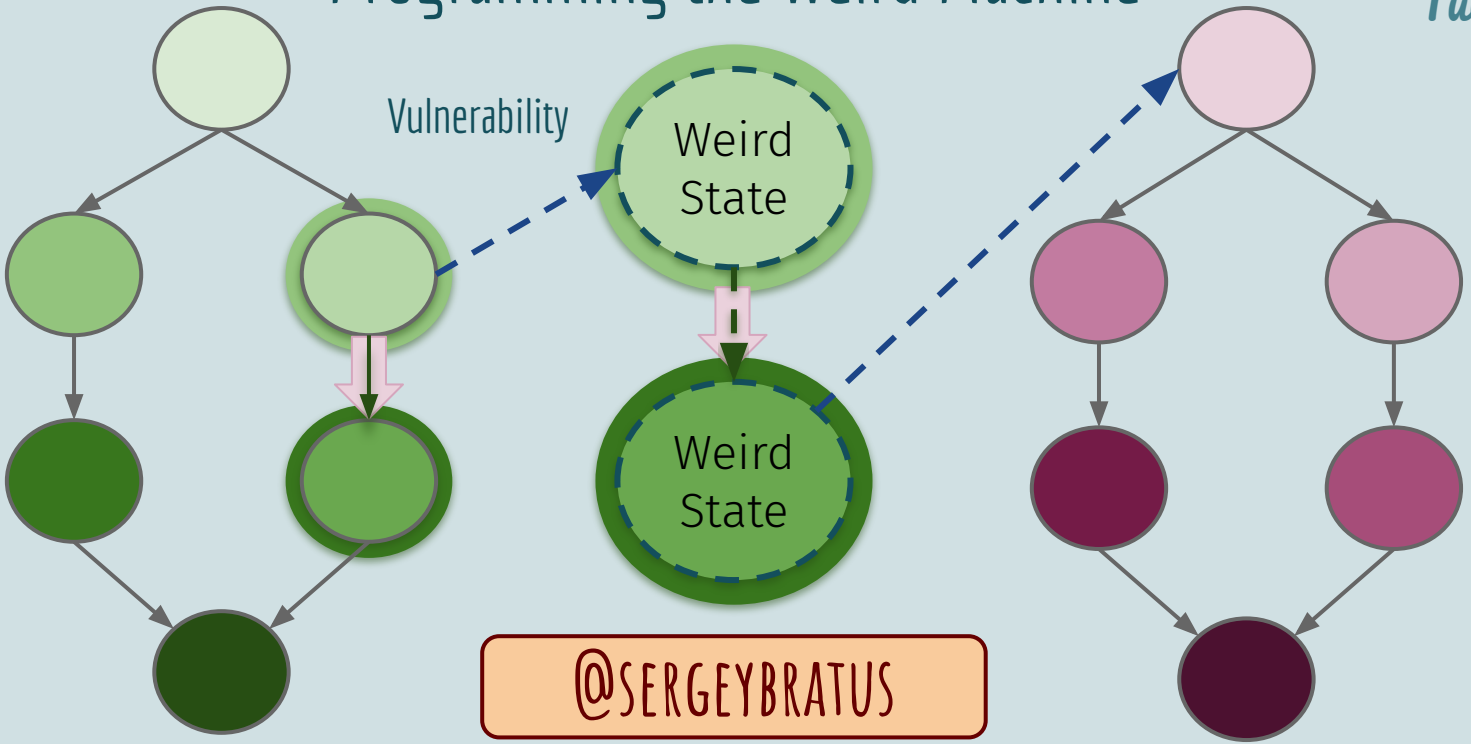
The Programmers Mental State Machine

TurtleSec



Programming the Weird Machine

TurtleSec



The Target

The Shellcode

@SERGEYBRATUS

@HALVARFLAKE

@PATI_GALLARDO

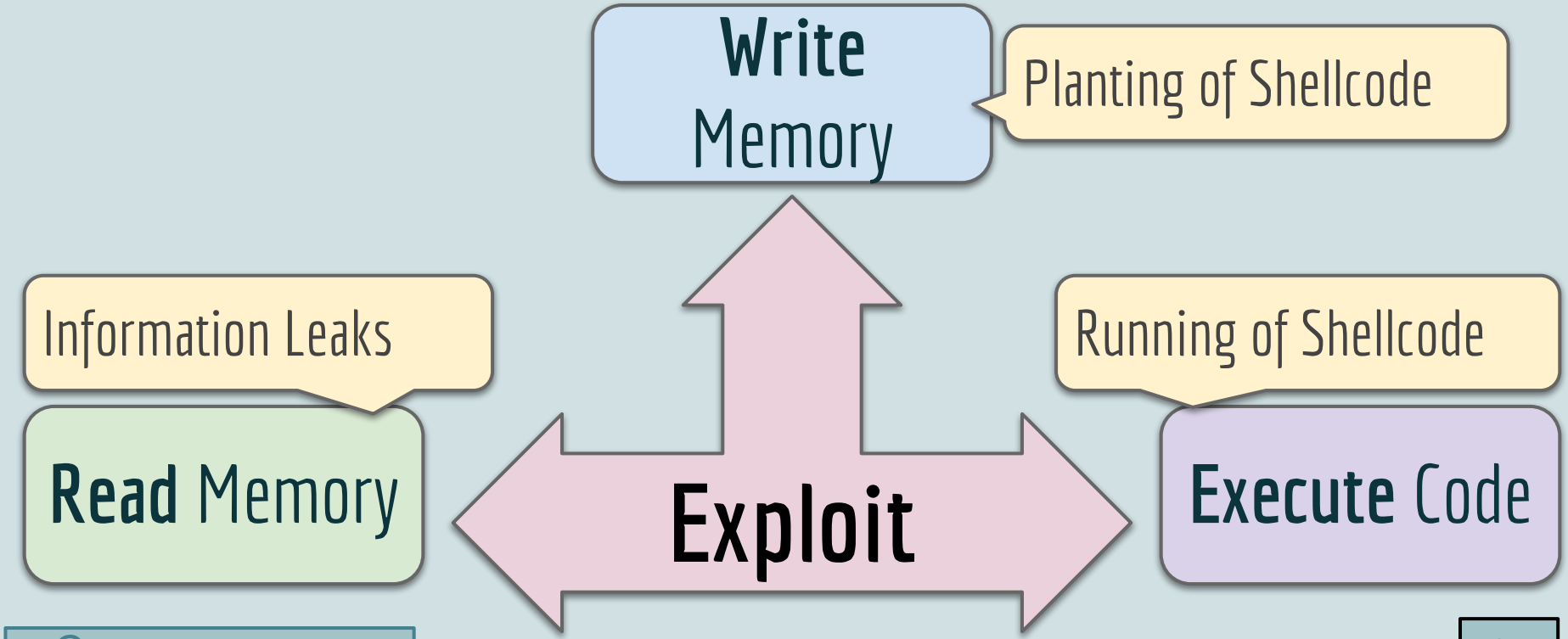
Shellcode

TurtleSec

Piece of code, typically in machine code,
that is delivered and executed as a part of an exploit.
Called “shellcode” because a **traditional use was to start a shell**, for example sh.
In real exploits it will **deliver some kind of mechanism for further (remote) compromise** of the system.

The Anatomy of an Exploit

TurtleSec



Code Execution

TurtleSec

To run your shellcode you need the instruction pointer to jump to your shellcode.
The instruction pointer jumps in many different scenarios - goal here is to control where it jumps to, examples:

return from a function
virtual function call
function pointer

“Primitives”

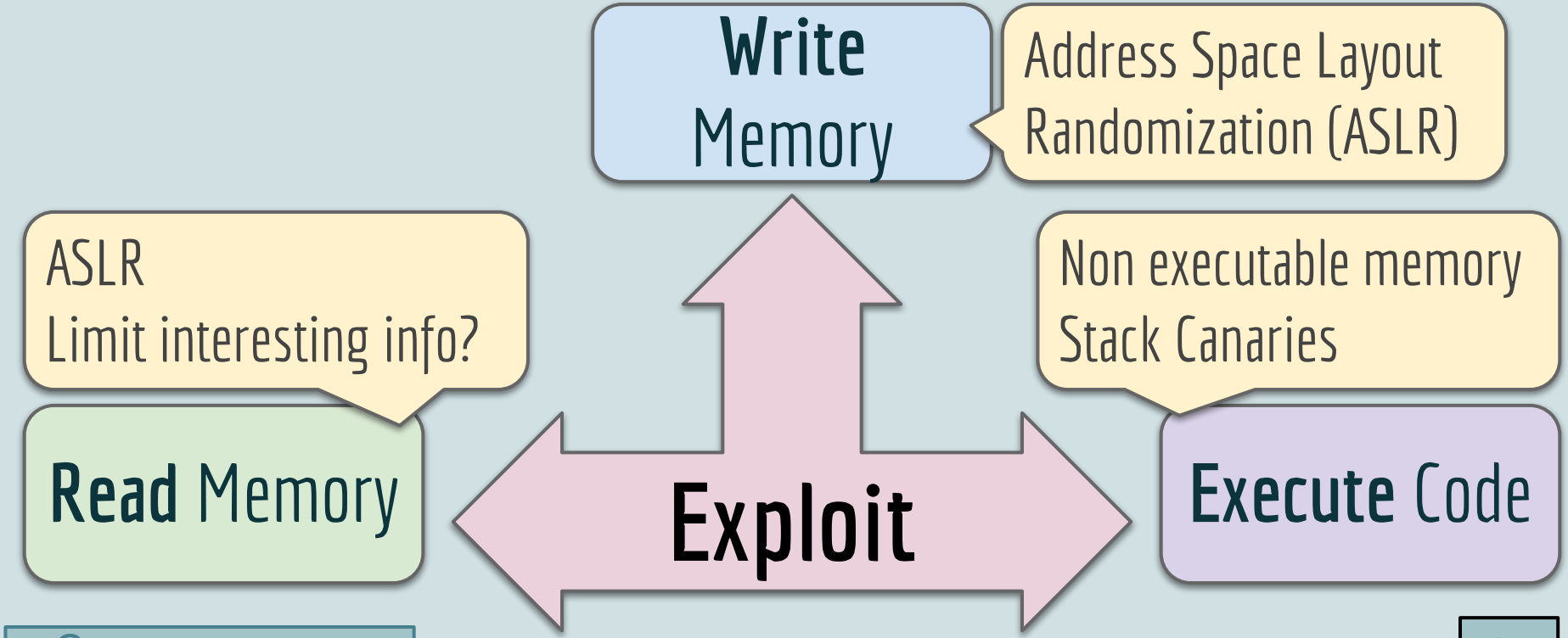
A vulnerability or a capability in the application that can be used as a part of a wider exploit is often referred to as a “*primitive*”- examples:

Arbitrary Read Primitive
Write-What-Where Primitive
Read-Where Primitive

Mitigations

Platform and Compiler Mitigations

TurtleSec



Cleaning Memory?

Dead Store Elimination

The compiler is allowed to optimize away stores that cannot be detected
Meaning memset'ing of memory that is never read can be removed

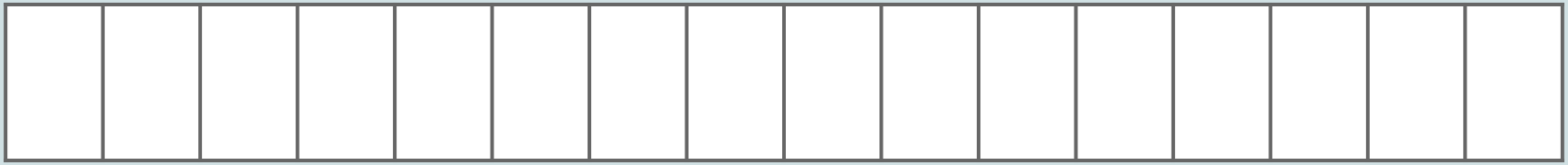
The Case Of The Disappearing Memset

The Heap

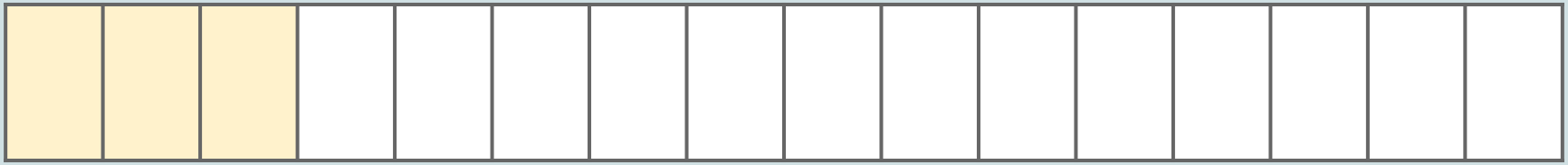
Allocators

@PATI_GALLARDO

Simple Pool Allocator

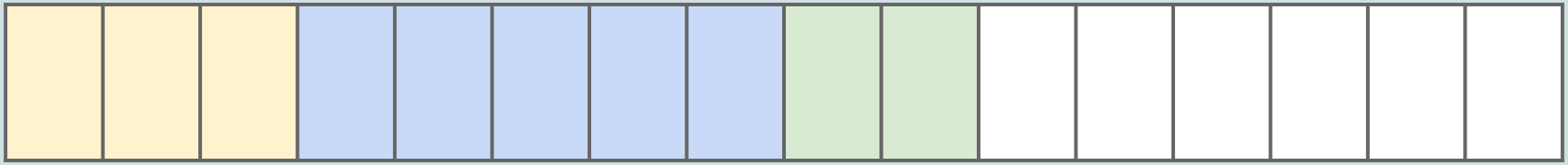


Empty Pool

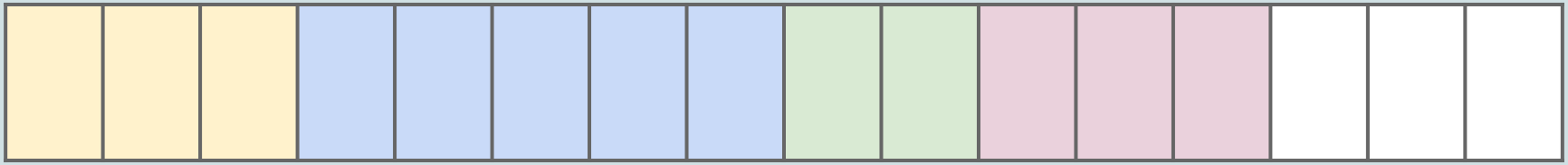


Initial allocations

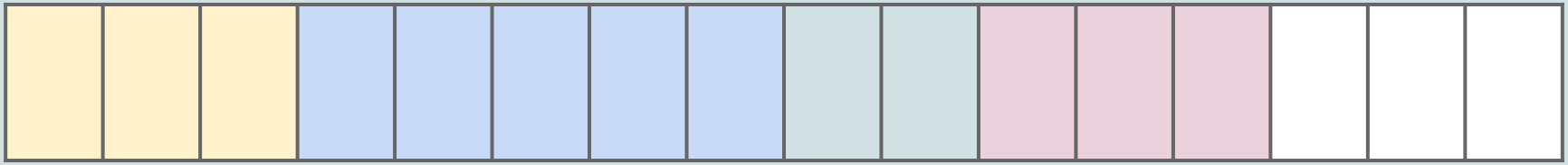
Initial allocations



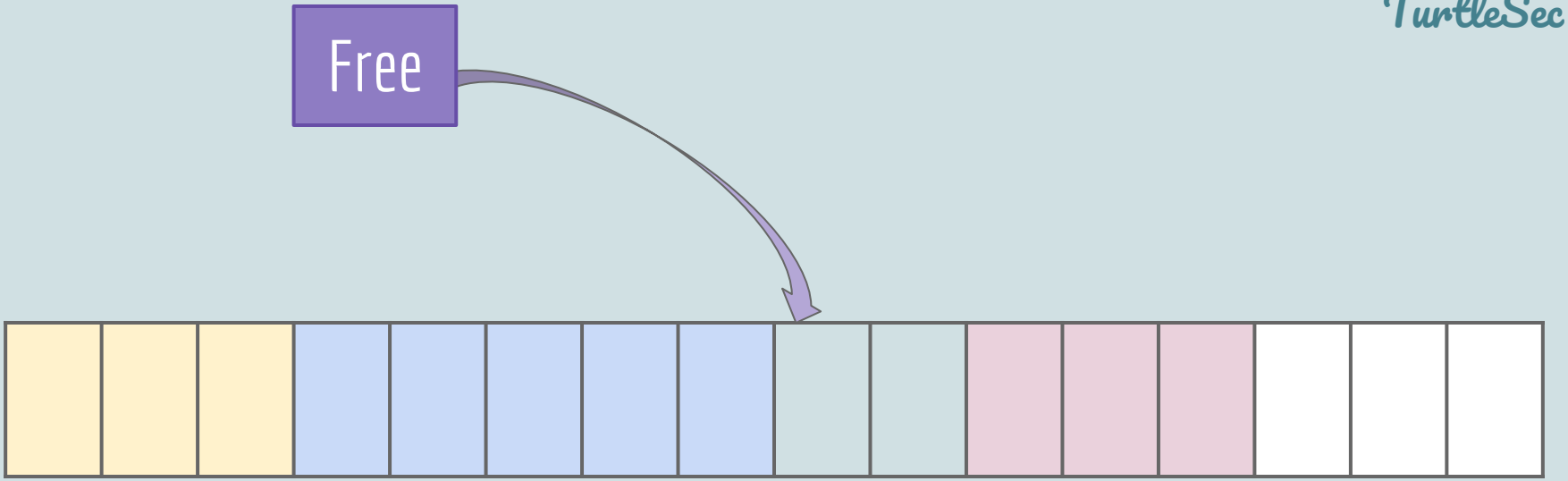
Initial allocations



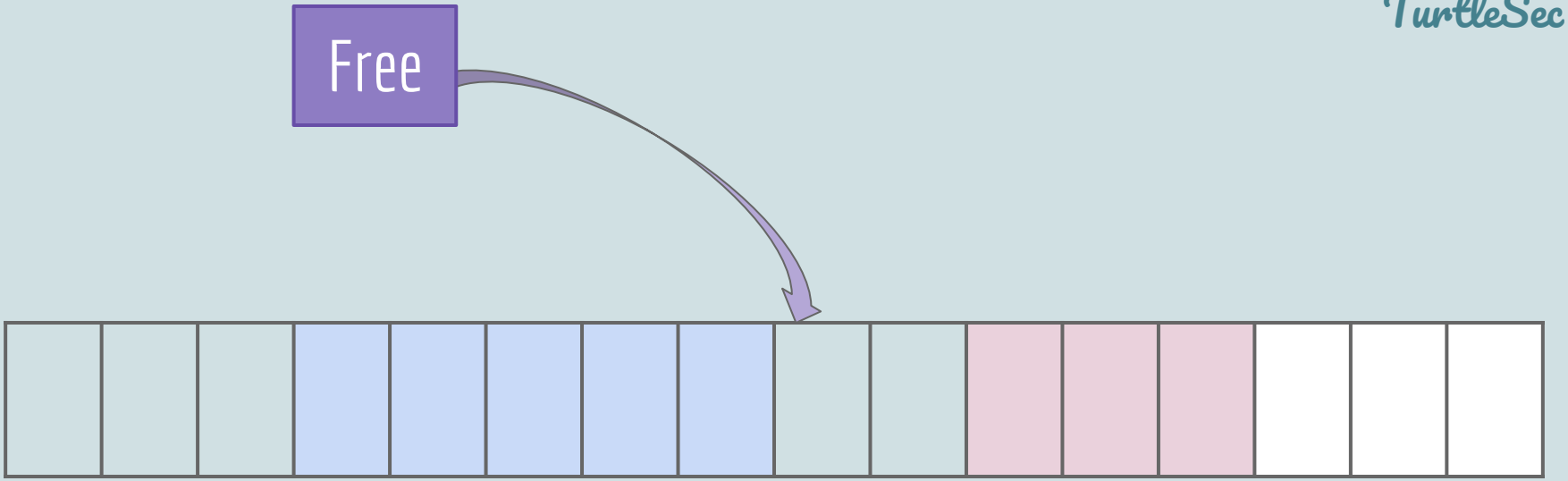
Initial allocations



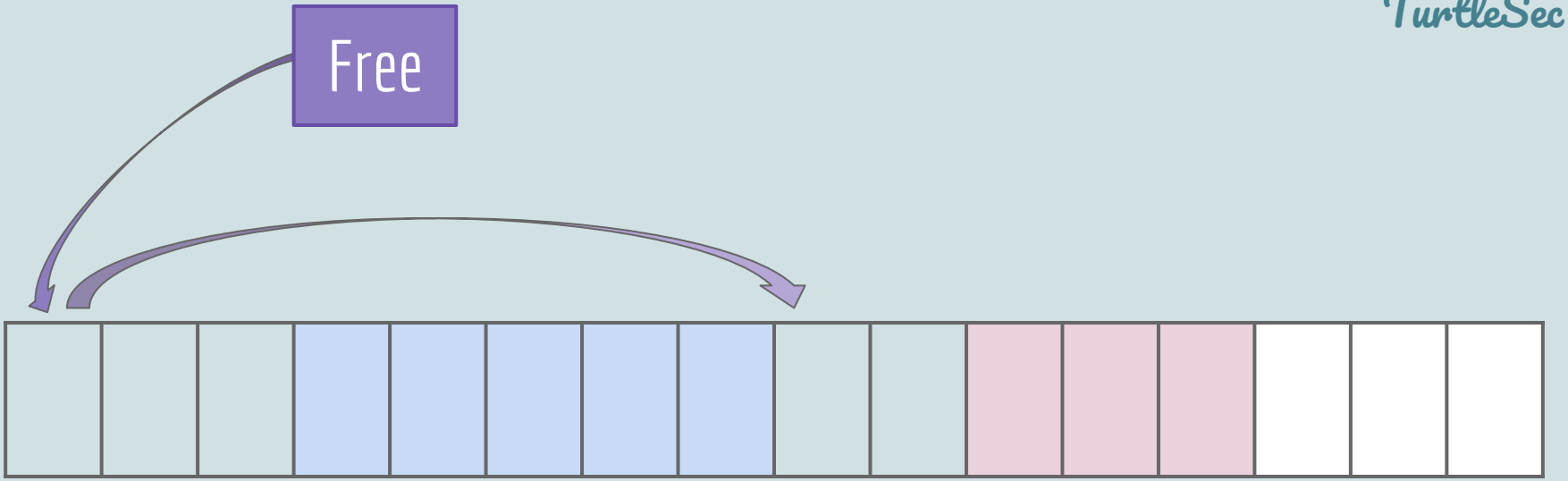
An allocation is freed - what now?



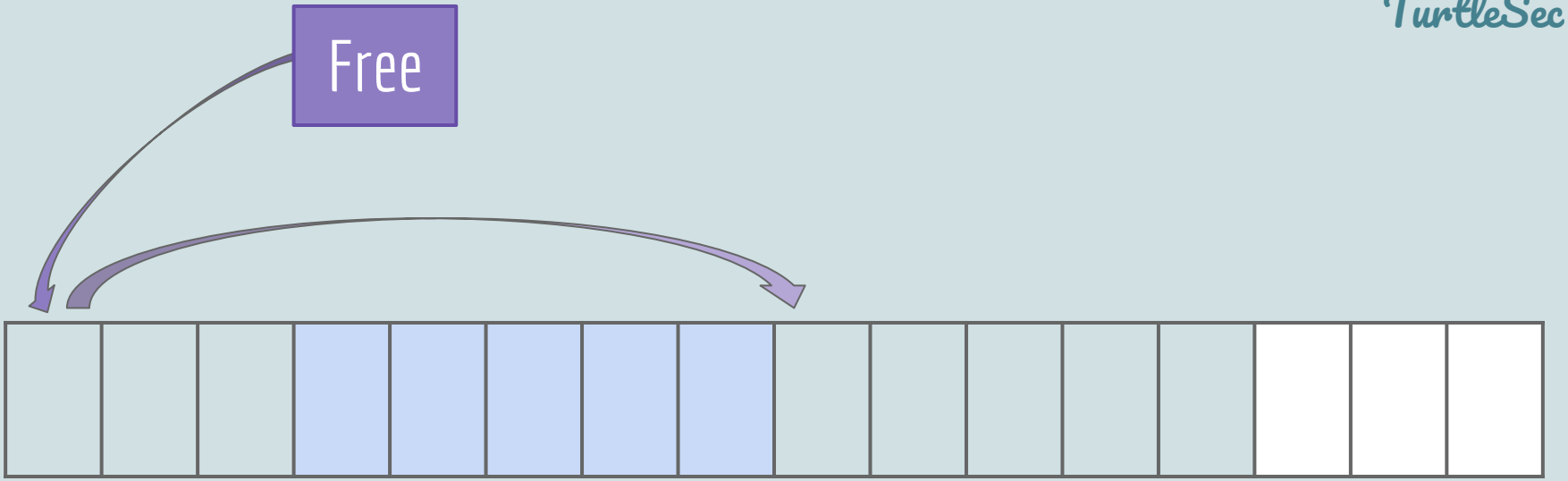
An allocation is freed - what now?



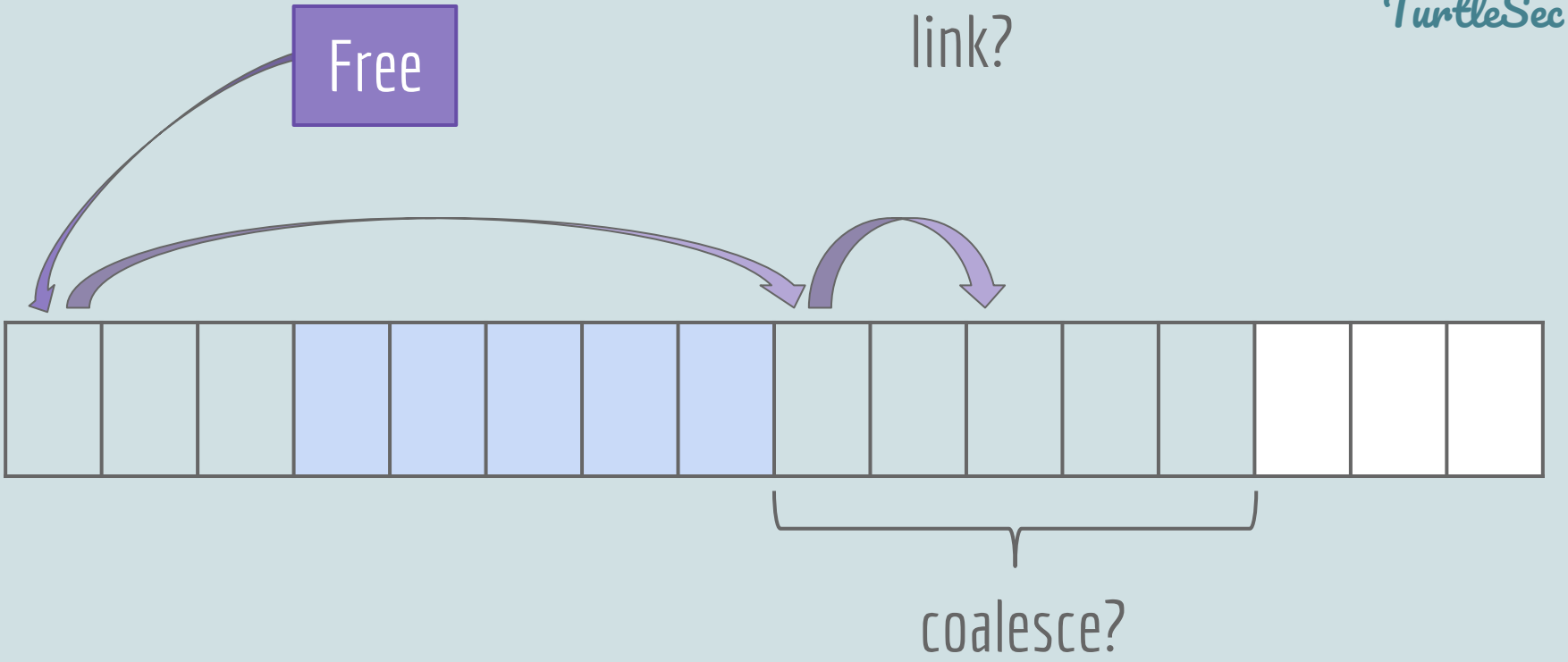
Another allocation is freed - what now?



Another allocation is freed - what now?



Another allocation is freed - what now?


Free
link?
coalesce?

So... how can we exploit this behavior?
We can allocate!

Heap Spraying

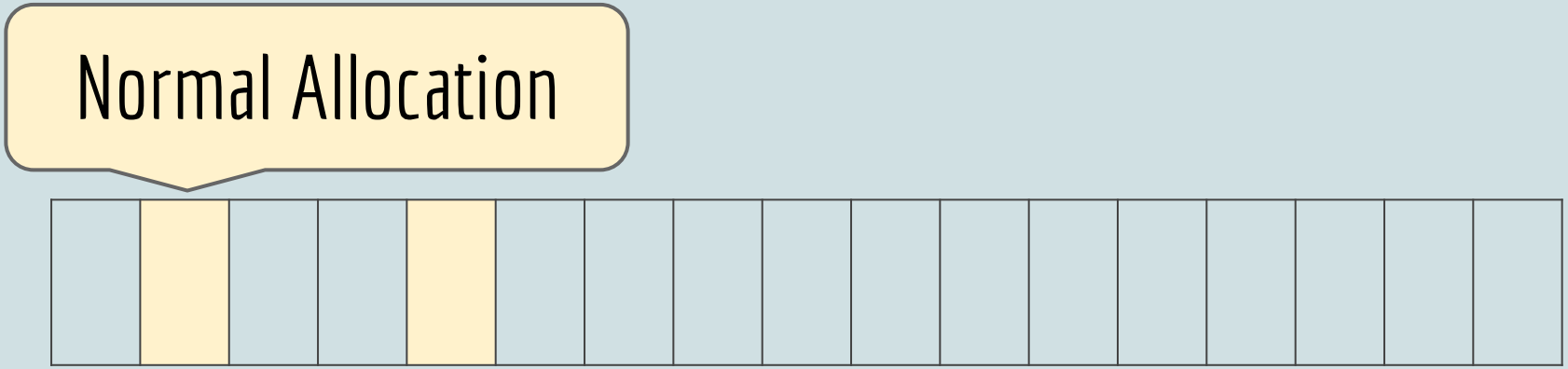
Heap Spraying

TurtleSec

Fill memory with a certain byte sequence
possibly shellcode
so that a “random” jump might hit it

Heap Spraying

TurtleSec

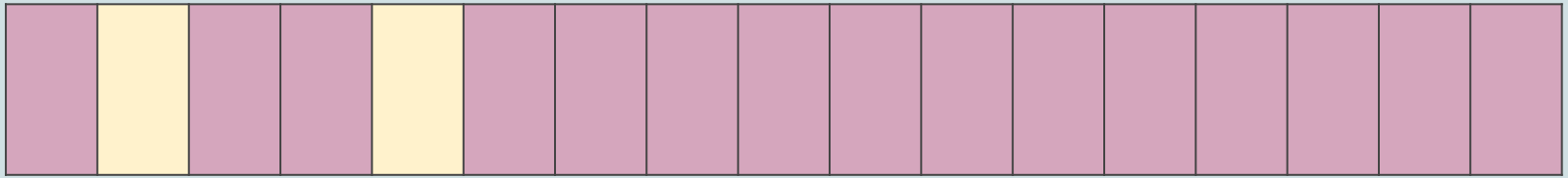


Initial state

Heap Spraying

TurtleSec

Normal Allocation



Fill memory with shellcode

Shellcode

This is a bit scattershot
Can we have more control?

Heap Grooming

(Heap Feng Shui)

@PATI_GALLARDO

Create predictable memory patterns

Trick the allocator to allocate a specific chunk

A chunk you can control

Let's see it in action

Putting it all together

“The Shadow Brokers”

Hacking group behind a leak in 2016-17

The leaked exploits and tools are believed to be NSAs

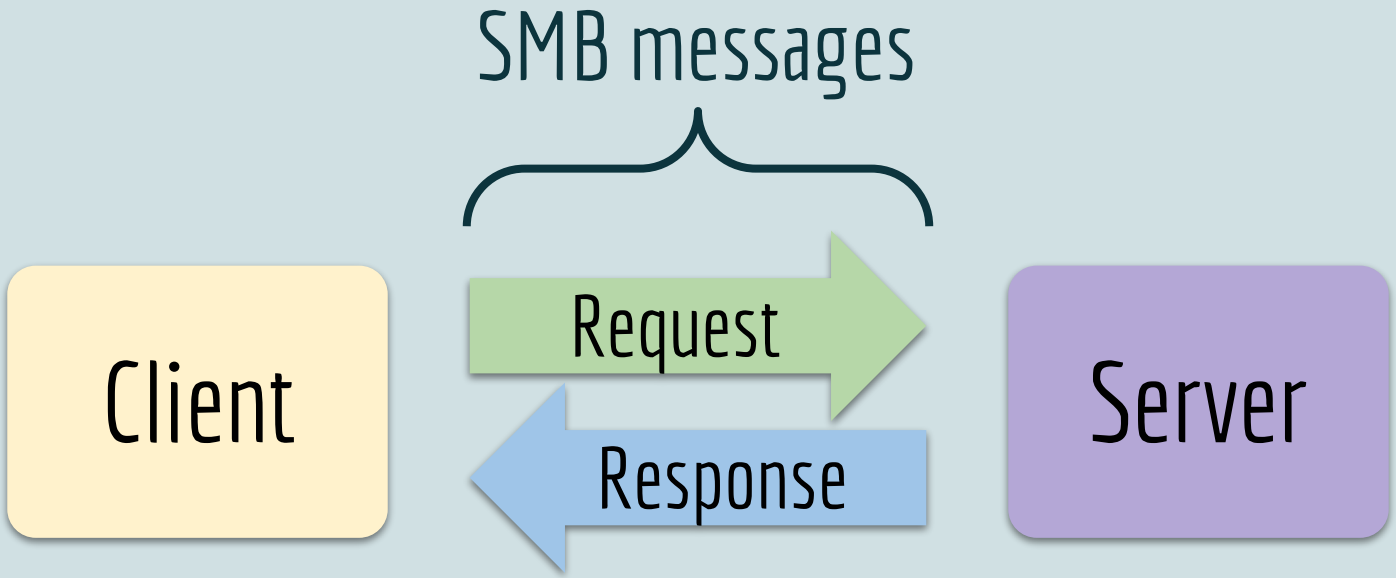
The Shadow Brokers are suspected to be Russian

The leak was done in several batches

Most famous is the **Eternal Blue** exploit

Very Light Background: Windows SMBv1

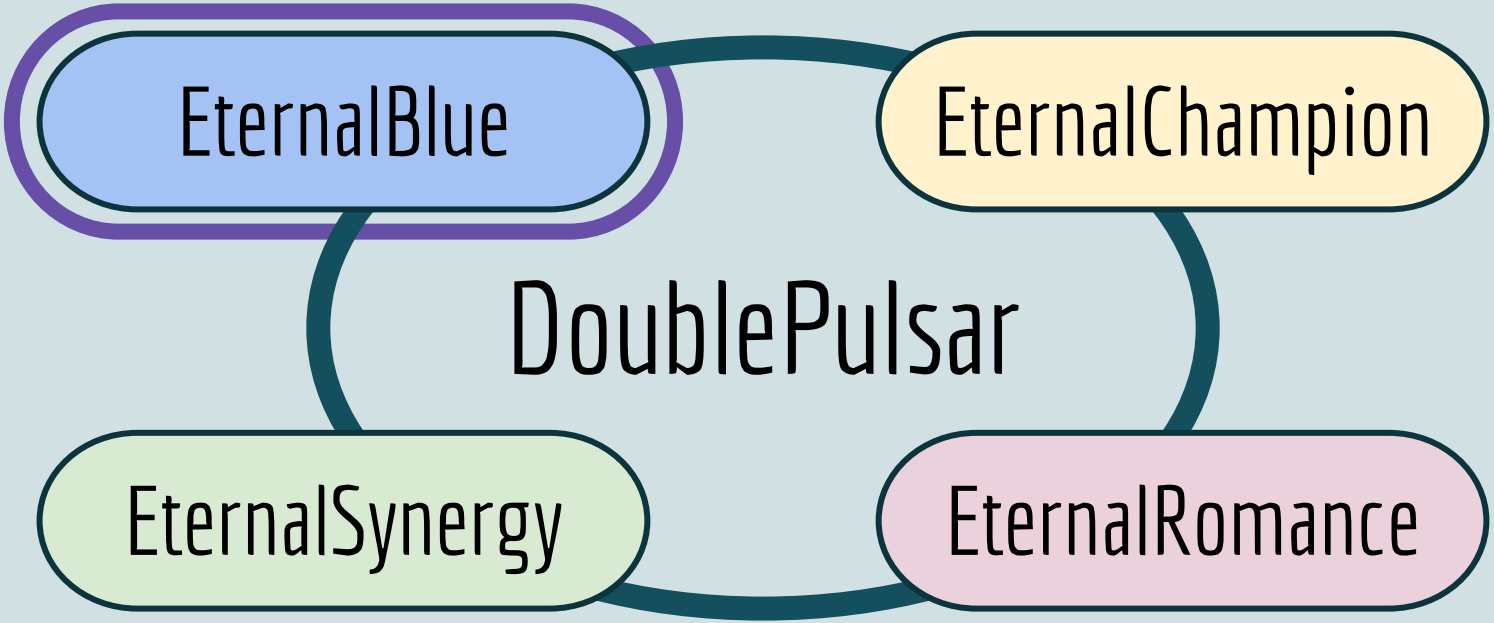
TurtleSec



Aside: This is the diagram of all things computer

Eternal Exploits

EternalBlue


EternalBlue
EternalChampion
DoublePulsar
EternalSynergy
EternalRomance

Write-What-Where Primitive and Remote Code Execution

EternalBlue

Linear Buffer Overrun, Heap Spray / Heap Grooming

Main bug

“When updating the length of the list,
the size is written to as if it were a 16-bit ushort,
when it is actually a 32-bit ulong.
This means that the upper 16-bits are not updated
when the list gets truncated.”

Microsoft Defender Security Research Team

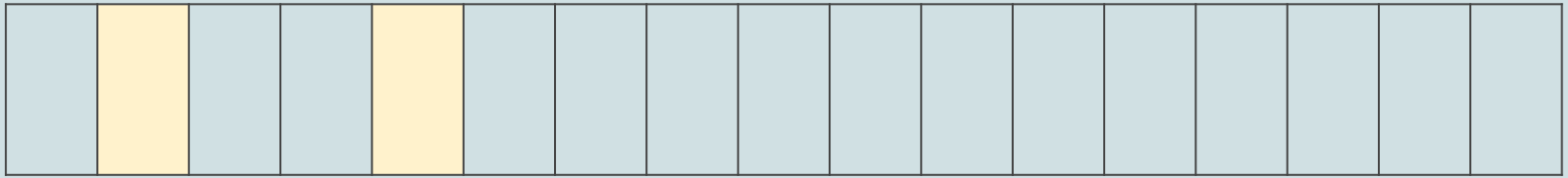
Heap Grooming and Spray

TurtleSec

- Primes the heap
- Fills with blocks ready for shellcode
- Makes room for buffer that will overrun
- Overrun will prepare code execution
- Hopes to overrun into one of the prepared blocks

Heap Grooming

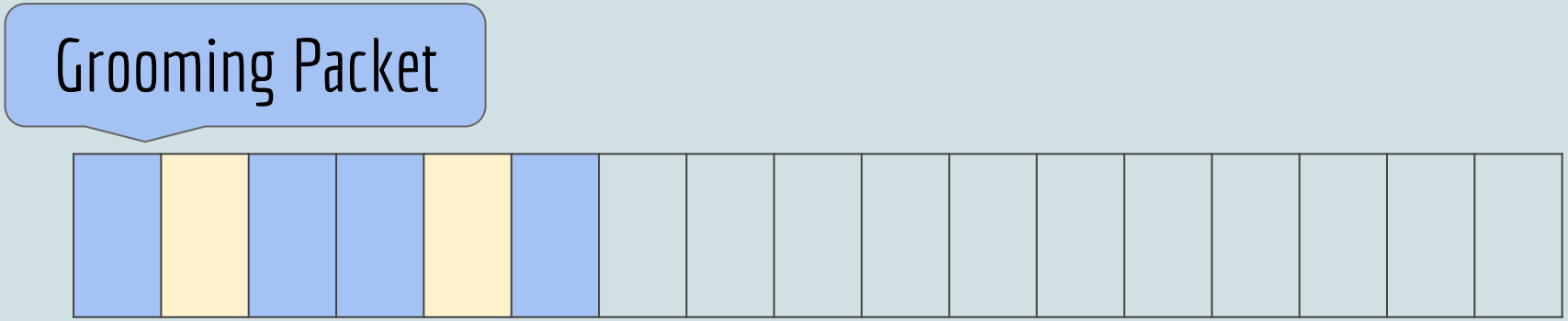
TurtleSec



Initial state

Heap Grooming

TurtleSec



Filling gaps to make allocations predictable

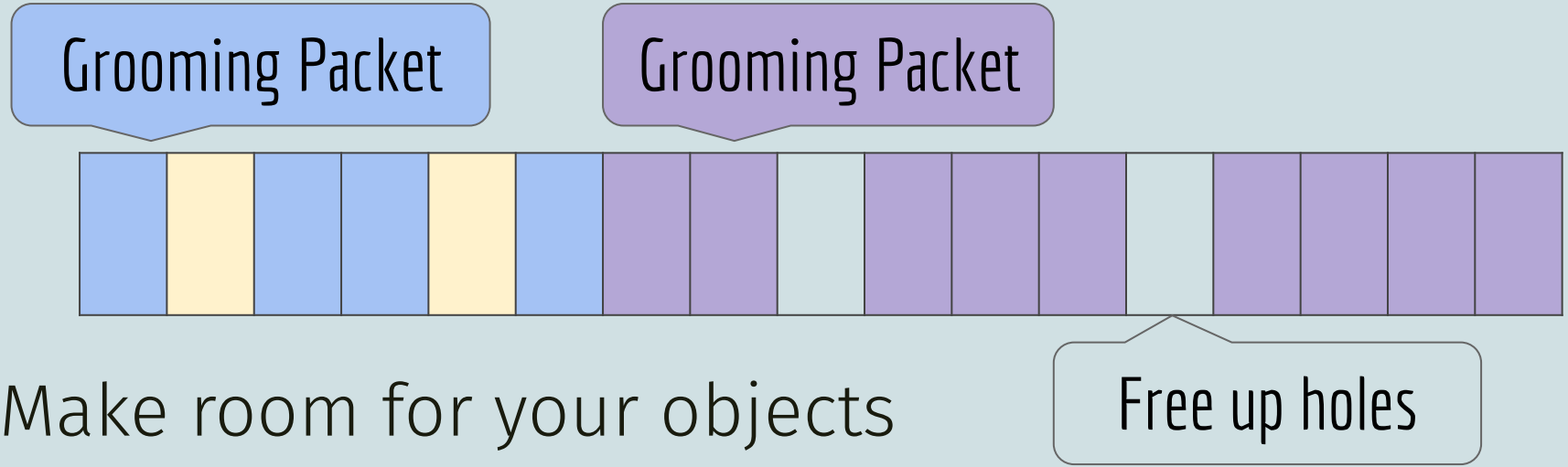
Heap Grooming

TurtleSec

Prefill before making pattern

Heap Grooming

TurtleSec



Make room for your objects

Heap Grooming

TurtleSec

Pattern: Fish in a barrel

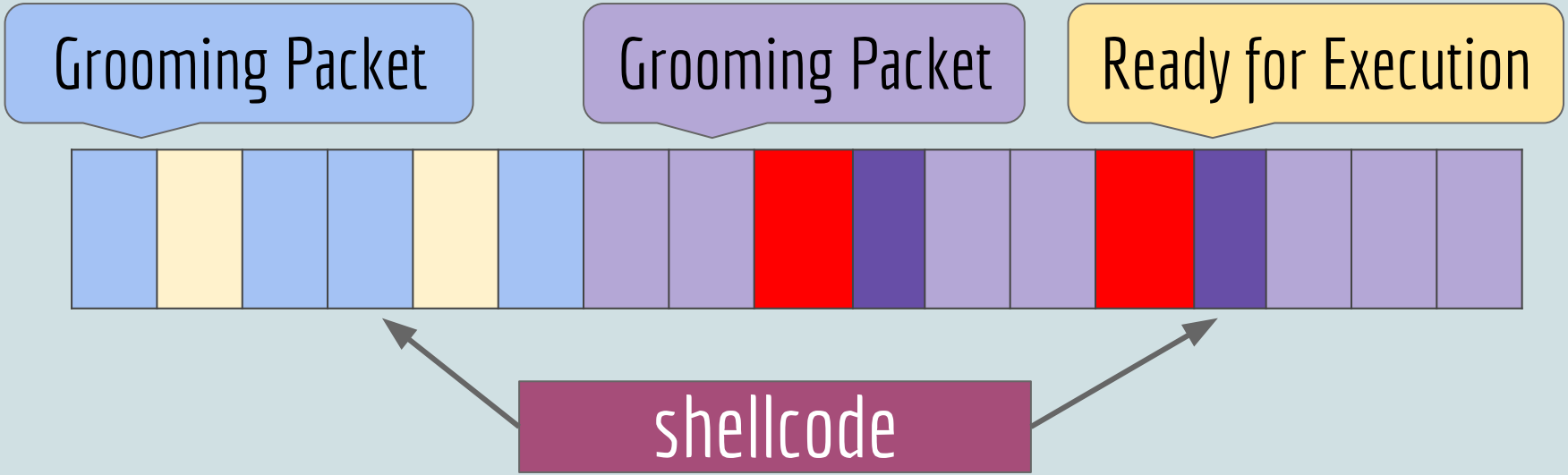
Heap Grooming

TurtleSec

Pattern: Fish in a barrel

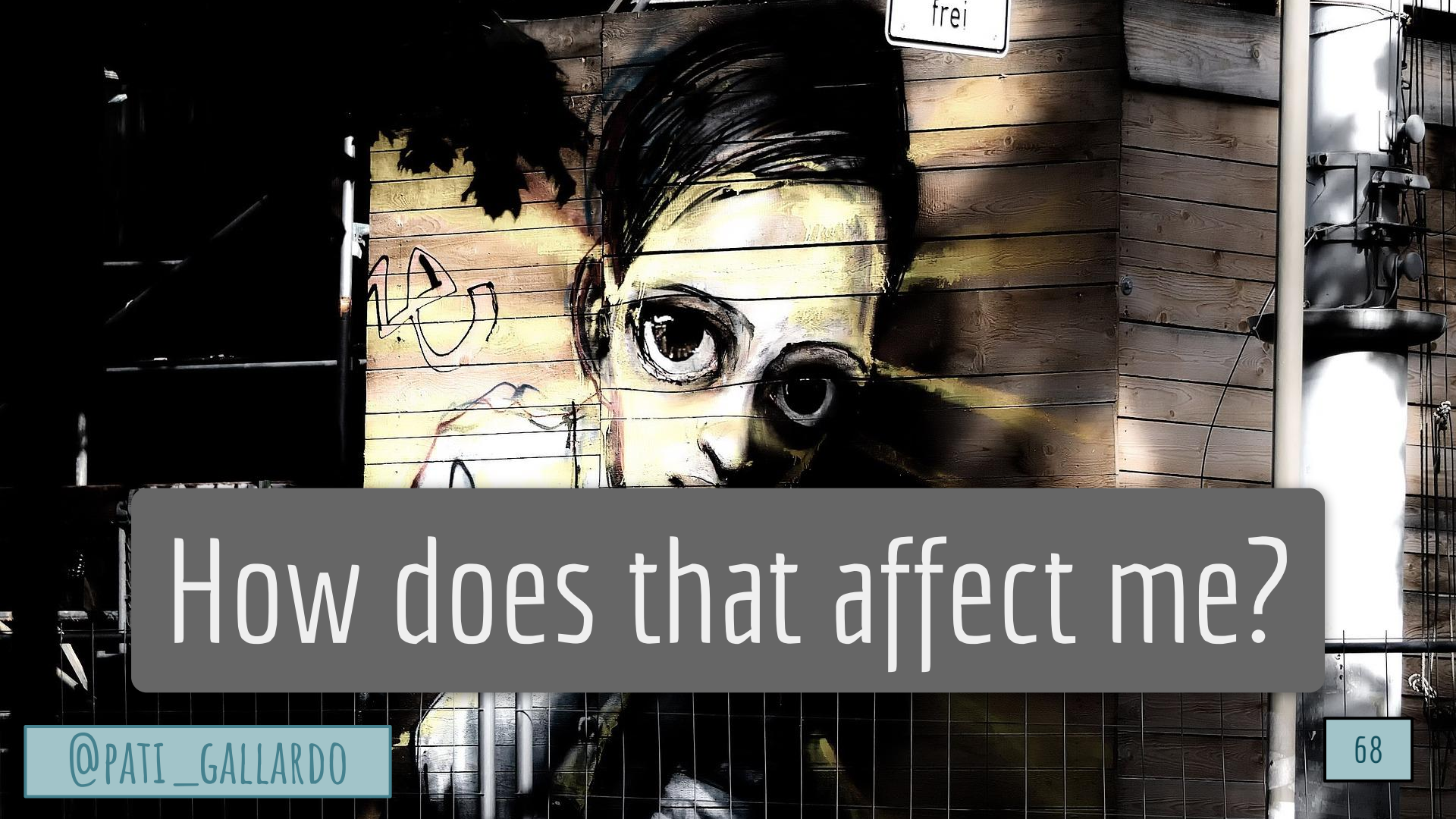
Heap Grooming

TurtleSec



Code Execution

When connection is closed
the shellcode is executed
in the block(s) that have been overrun
Installs the DoublePulsar backdoor implant



How does that affect me?

There is no magic here

These are bugs you can find

The tools *they* use are tools *you* can use

Basically: Fix Bugs


5
amboss
@PATI_GALLARDO
Turtle
Sec