Posthammer

Pervasive Browser-based Rowhammer Attacks
with Postponed Refresh Commands

Finn de Ridder · Patrick Jattke · Kaveh Razavi
Computer Security Group · ETH Zurich

Posthammer

Posthammer

61% of DDR4 chips vulnerable to browser-based Rowhammer

Posthammer

61% of DDR4 chips vulnerable to browser-based Rowhammer

This means

This means

Victim visits web page

This means

Victim visits web page

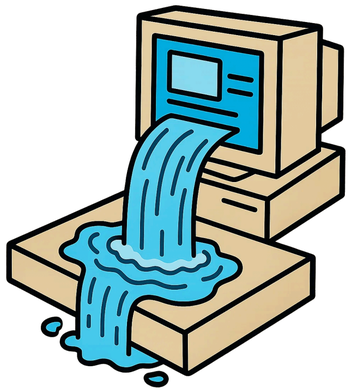
Attacker hammers

This means

Victim visits web page

Attacker hammers

Read-write primitive

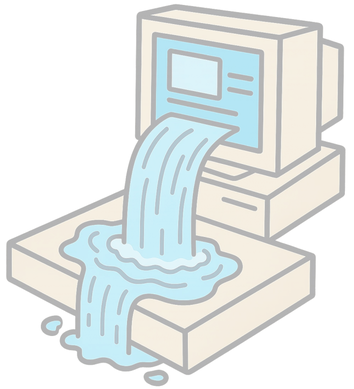


This means

Victim visits web page

Attacker hammers

Read-write primitive



DIMM vulnerable: 17/28

This means

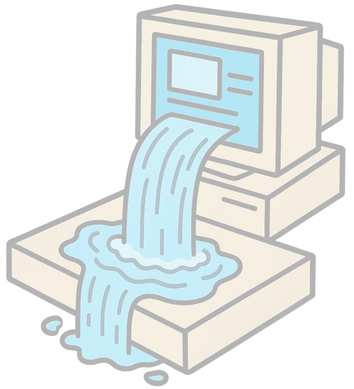
Victim visits web page

DIMM vulnerable: 17/28

Attacker hammers

15 min.

Read-write primitive



This means

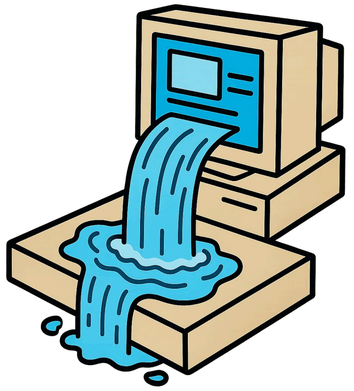
Victim visits web page

DIMM vulnerable: 17/28

Attacker hammers

15 min.

Read-write primitive



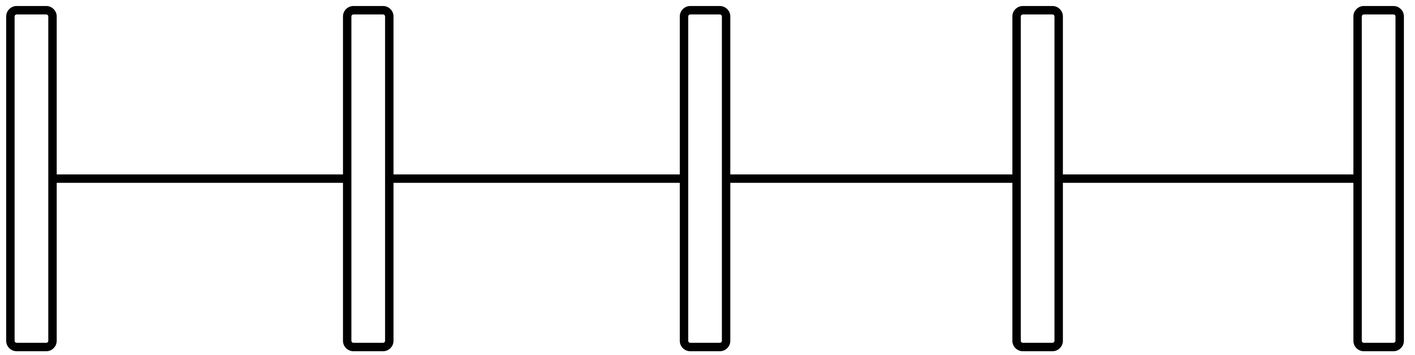
Success rate: 40%

What did we learn?

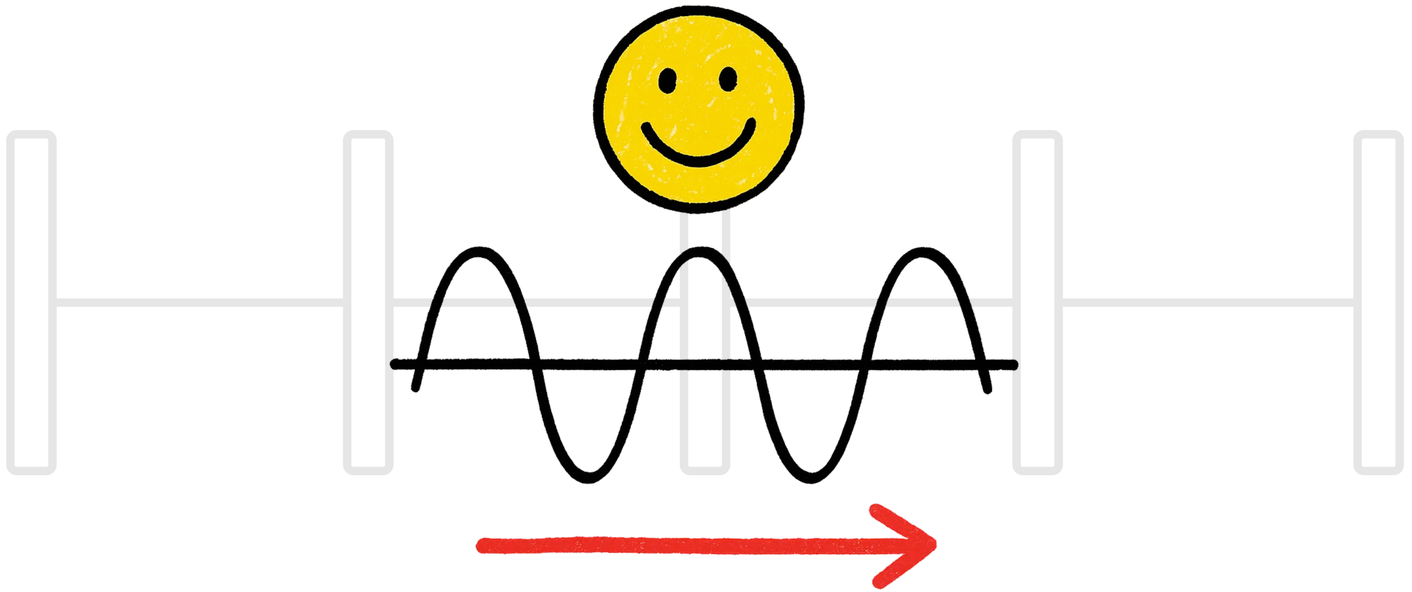
Insight 1

Refresh commands

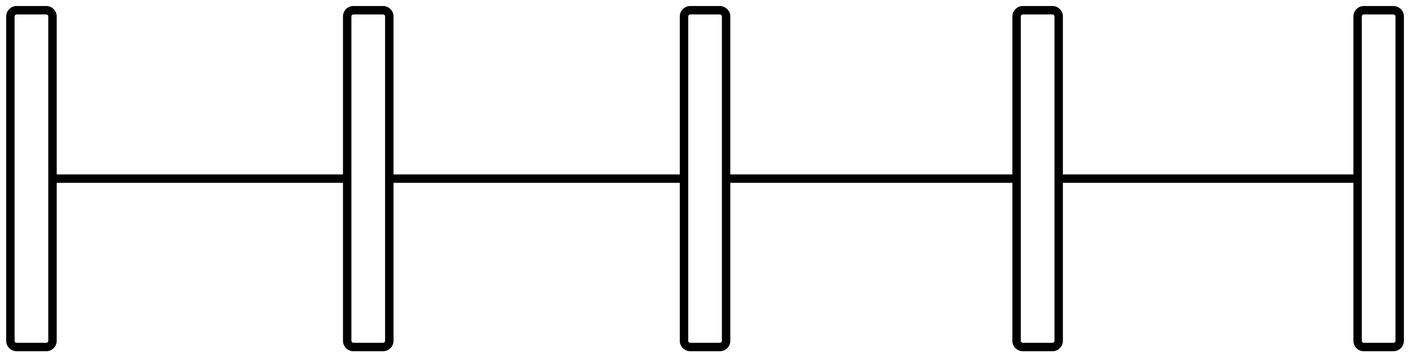
Refresh commands



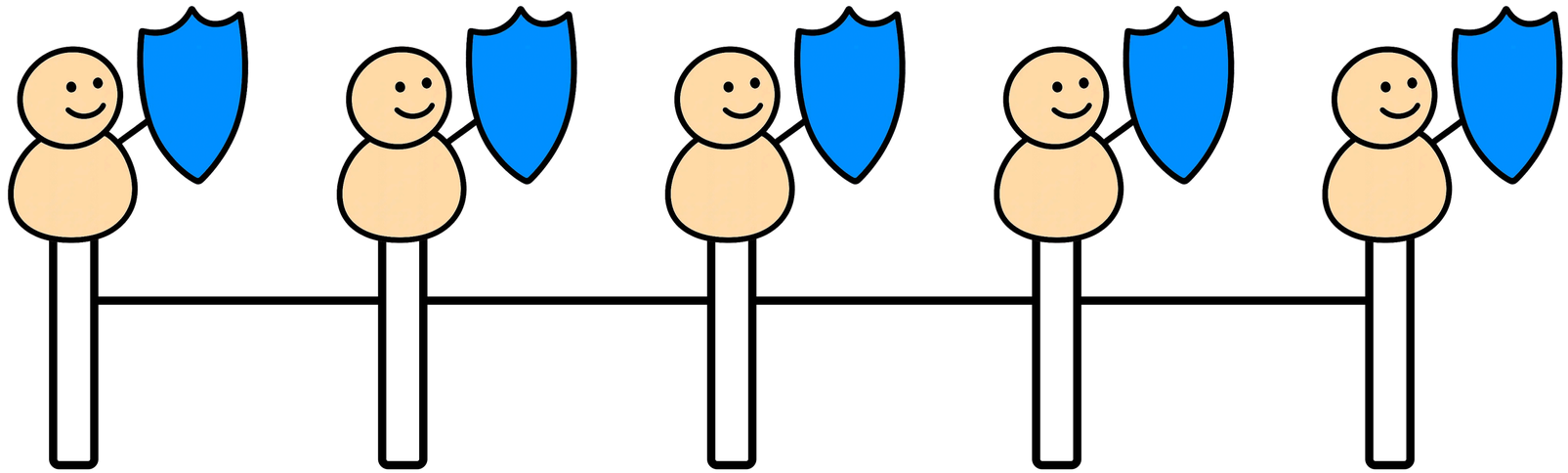
Refresh commands



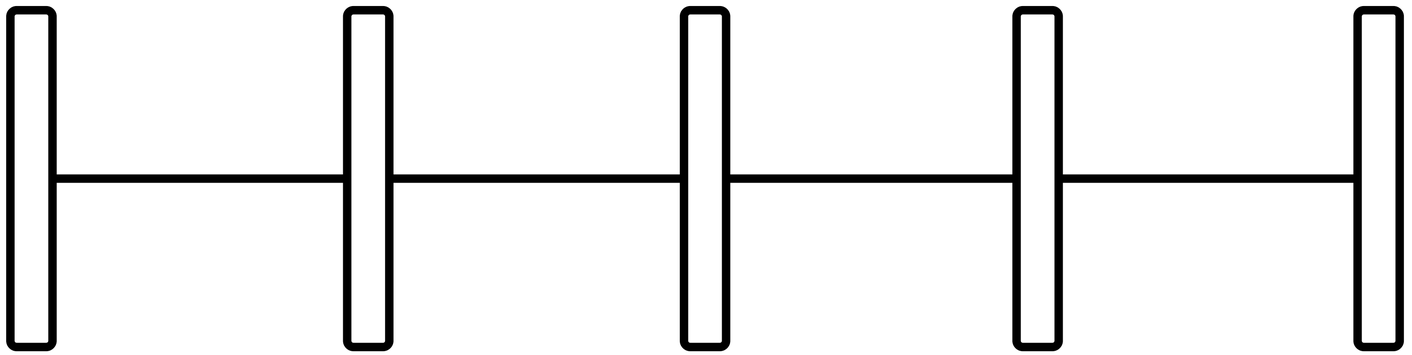
Refresh commands



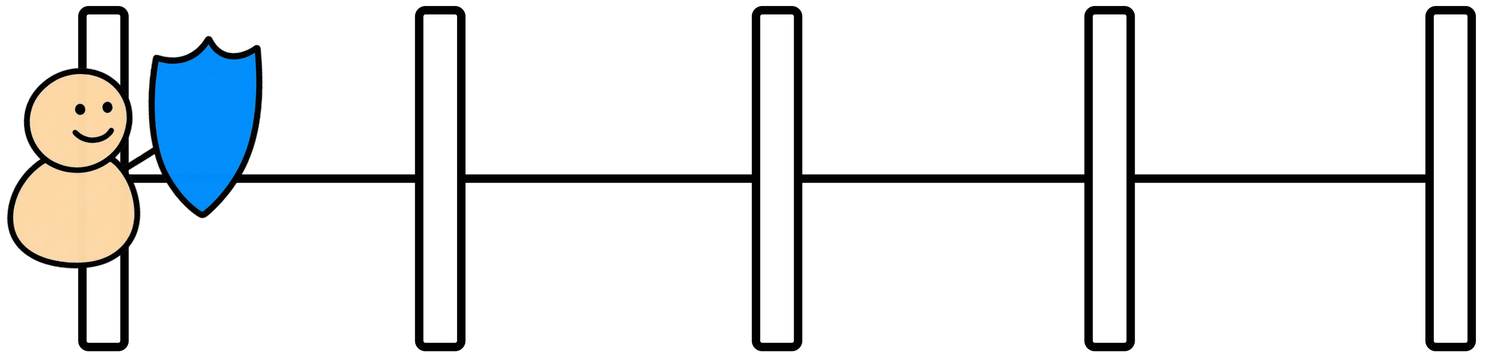
Refresh commands



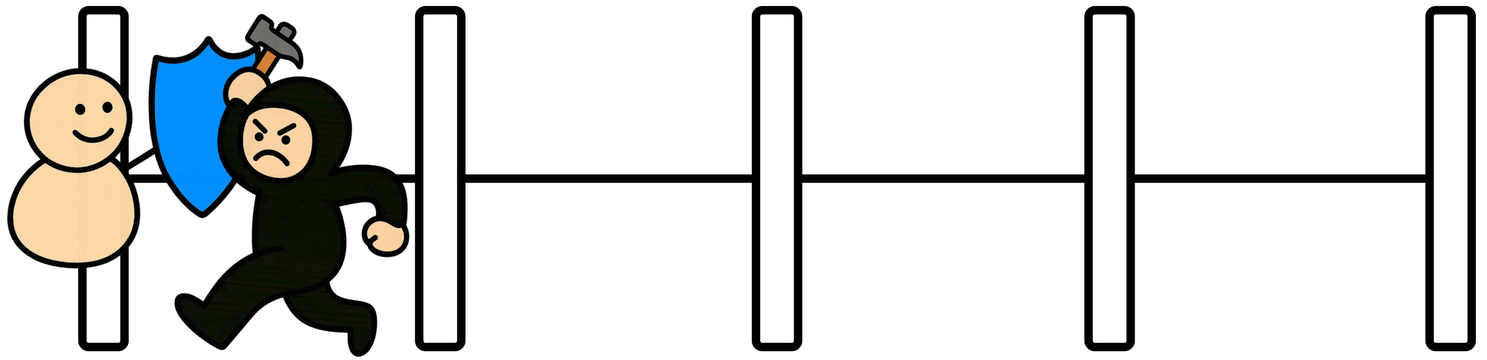
Refresh commands



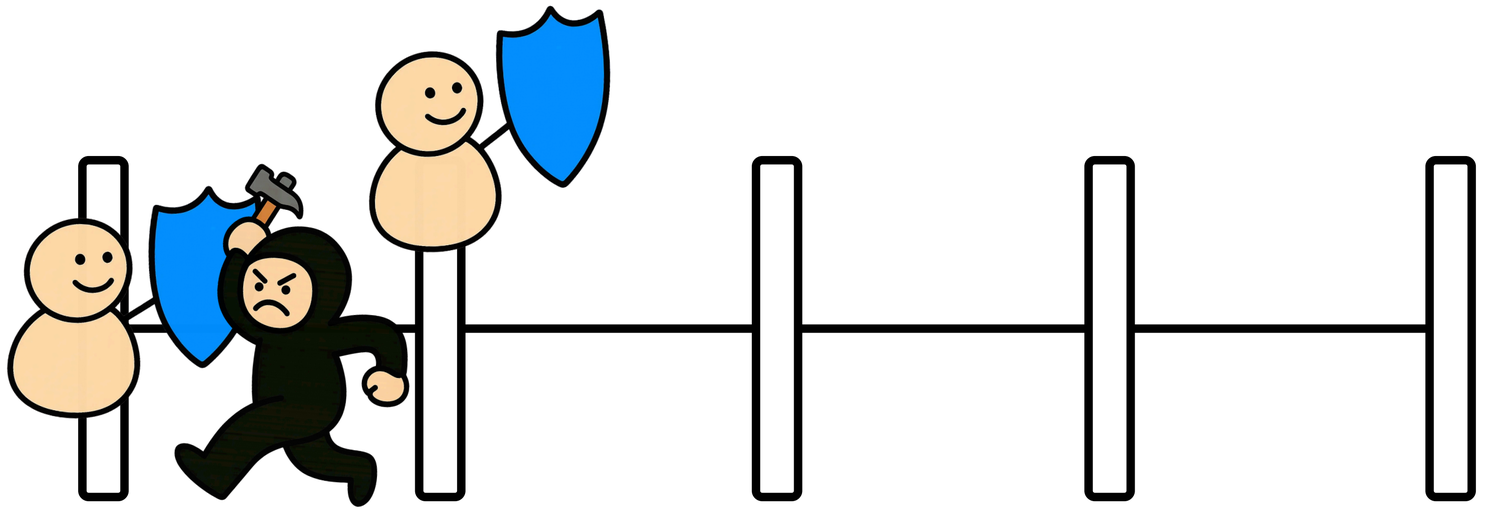
Refresh commands



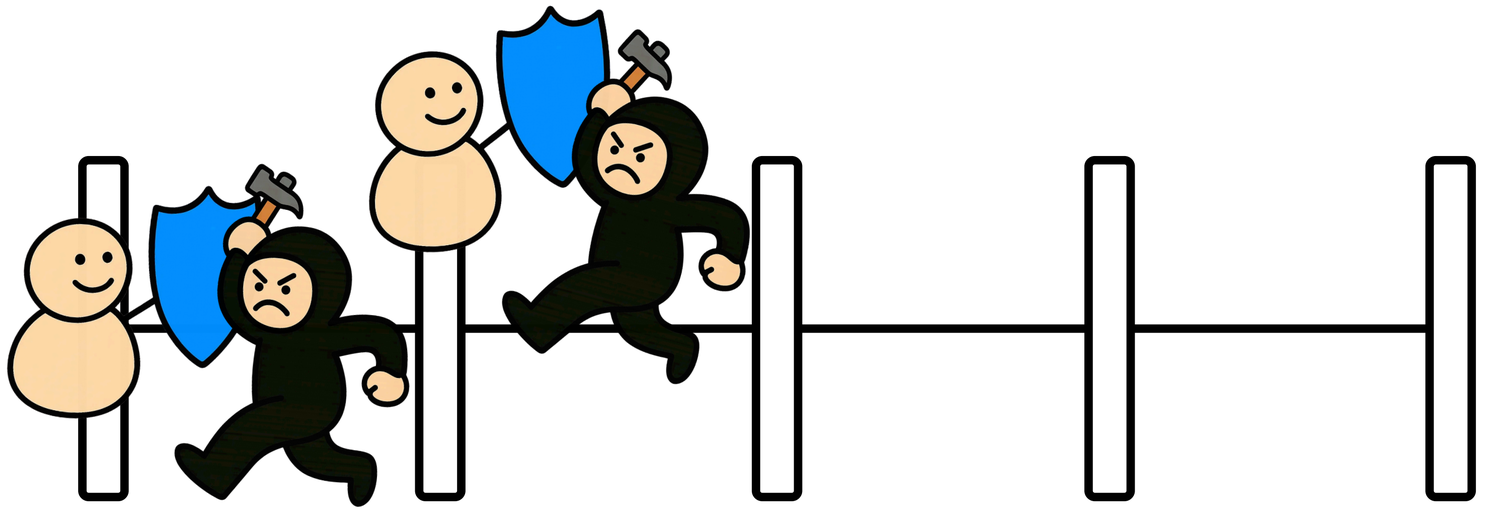
Refresh commands



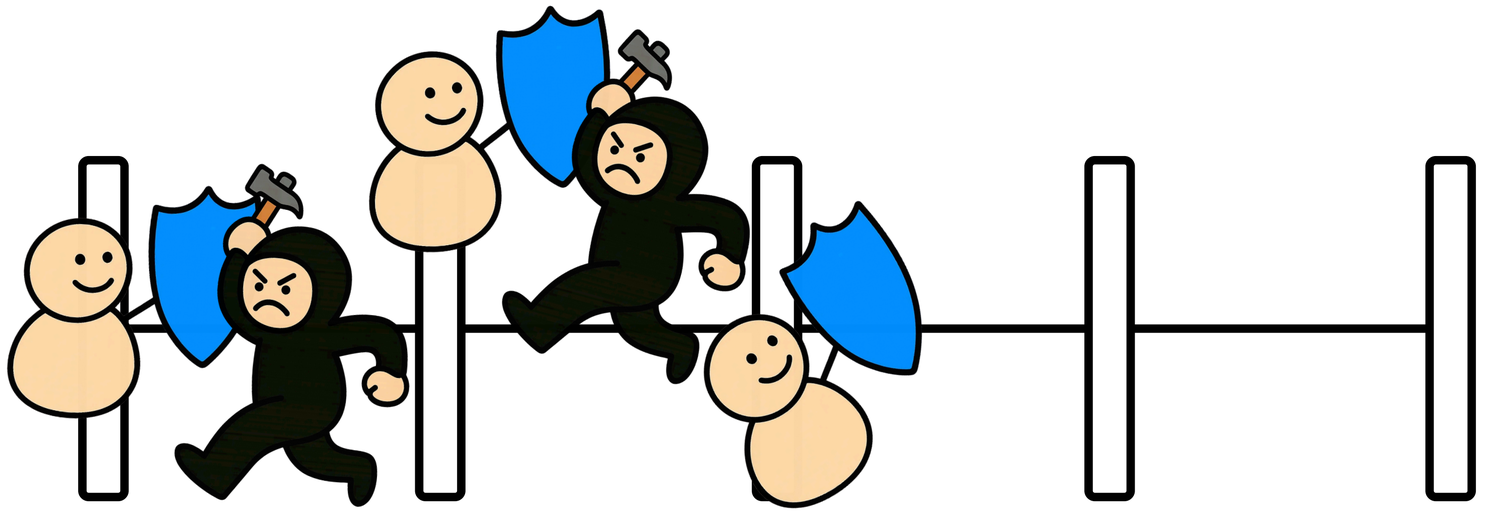
Refresh commands



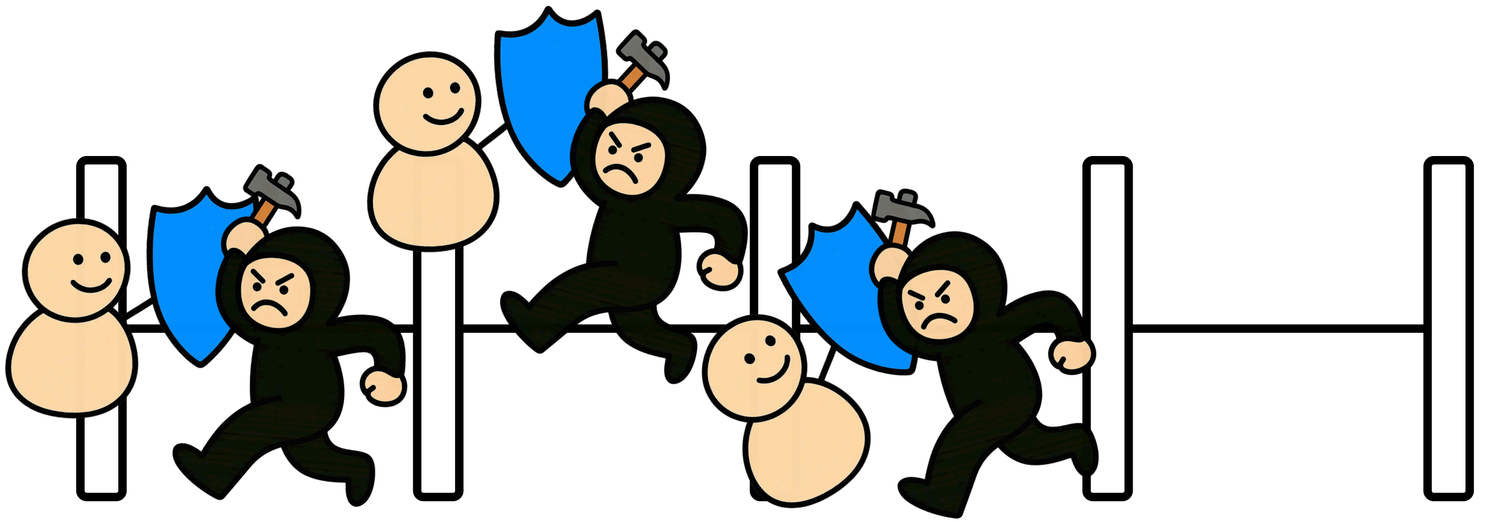
Refresh commands



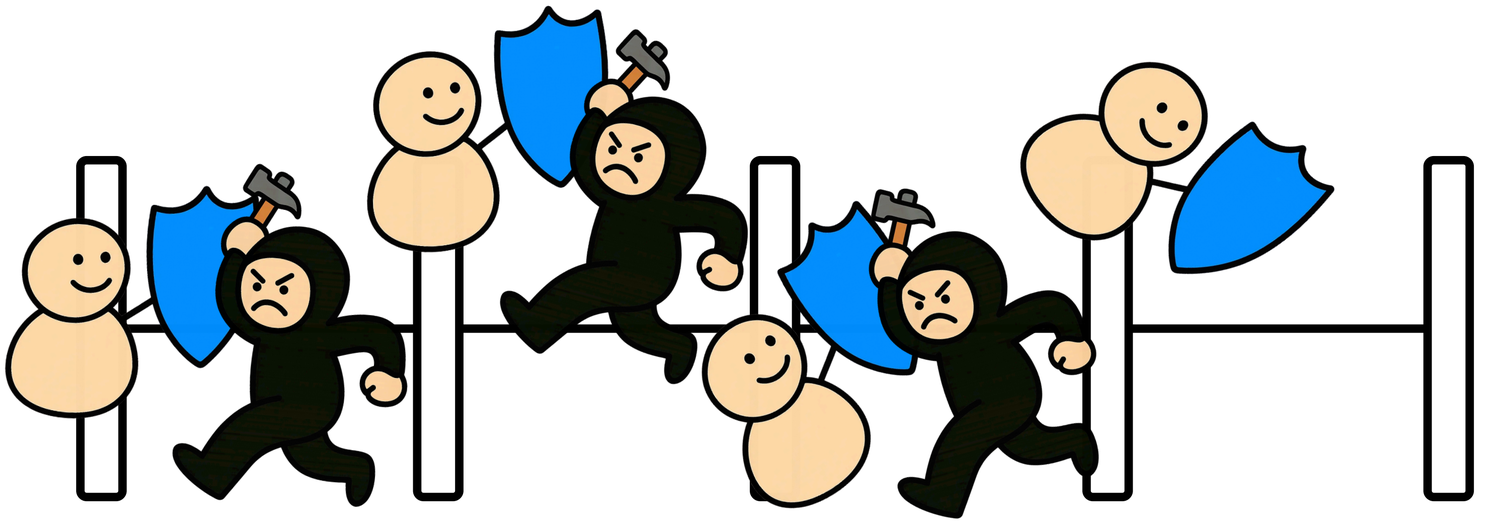
Refresh commands



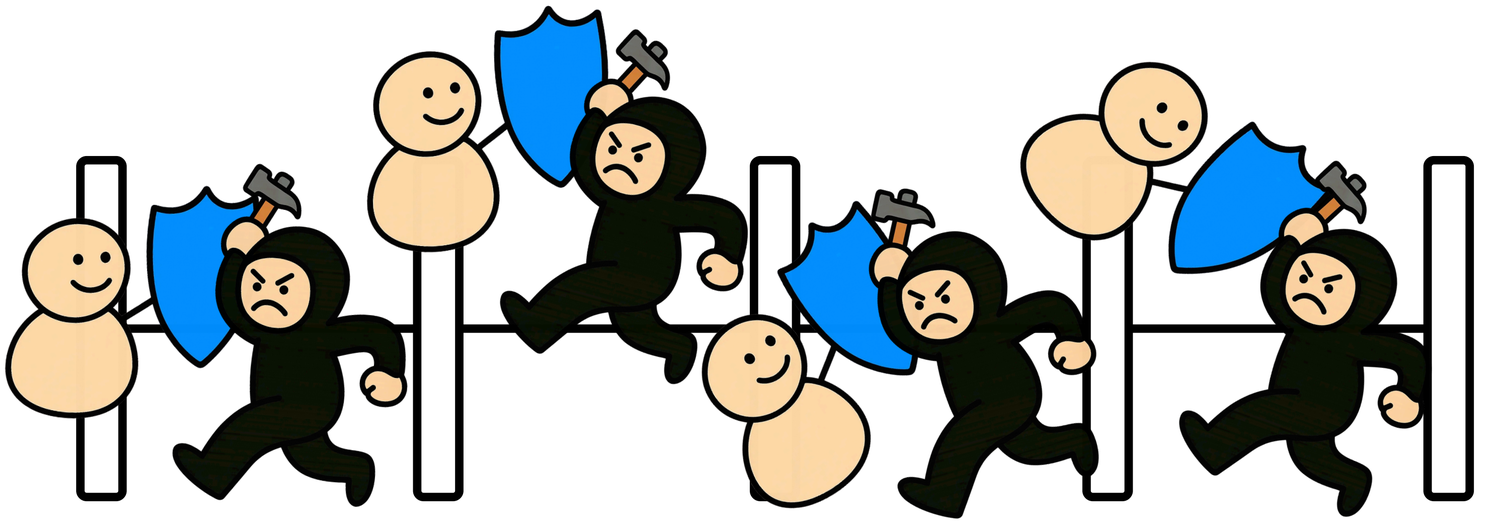
Refresh commands



Refresh commands

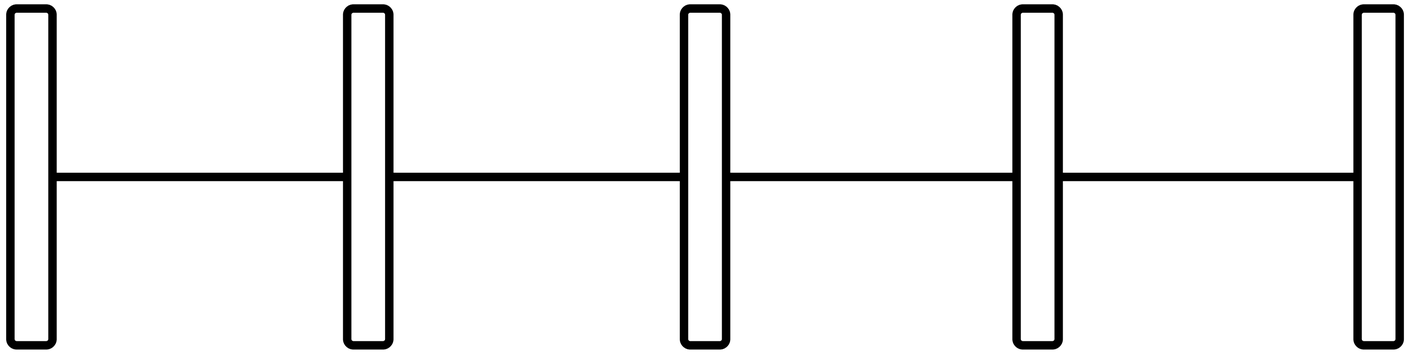


Refresh commands

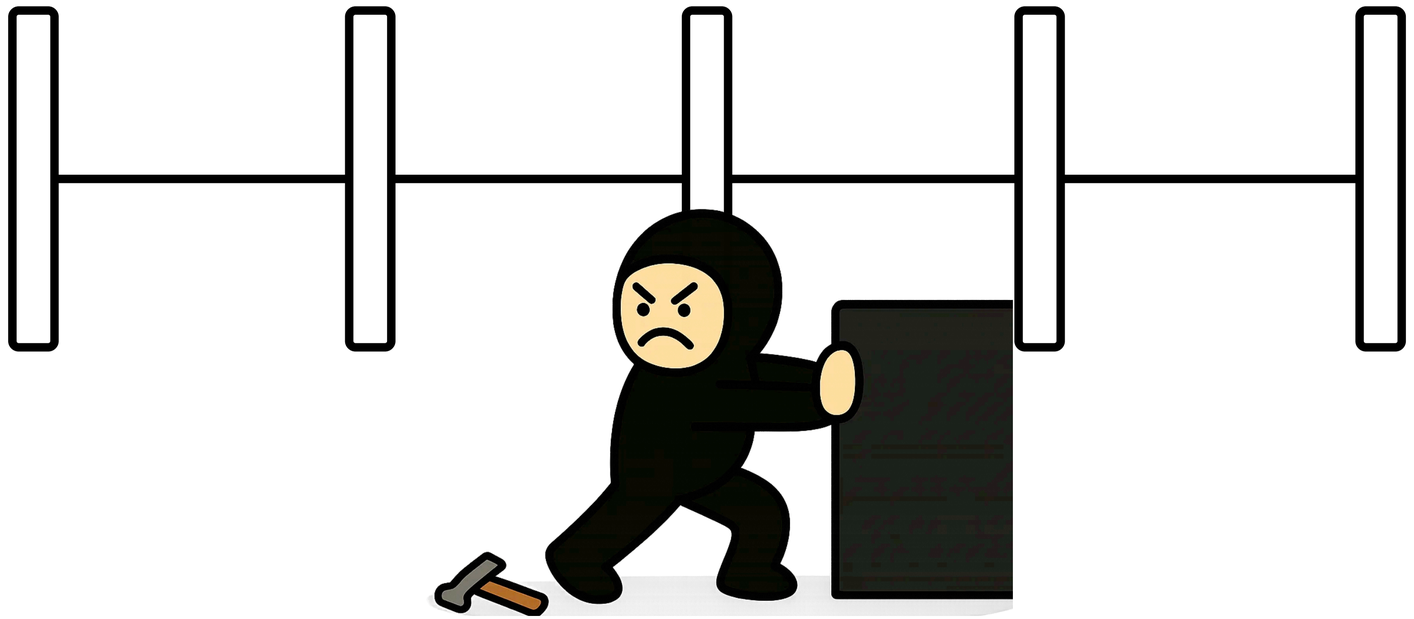


Refresh postponement

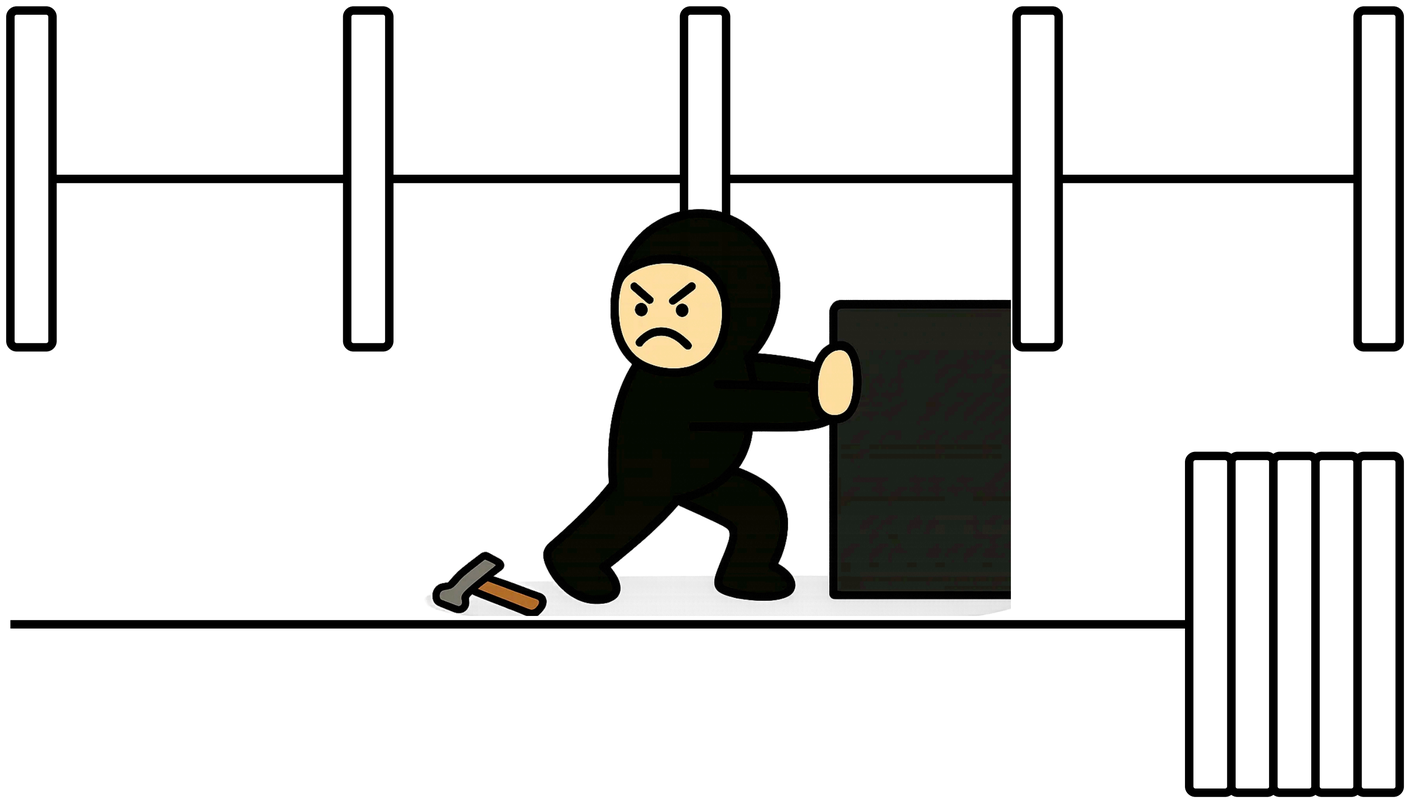
Refresh postponement



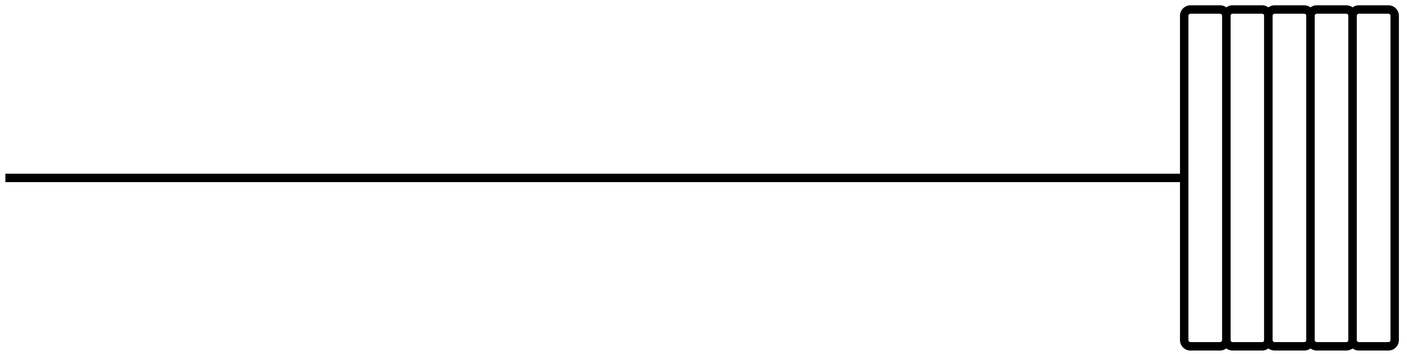
Refresh postponement



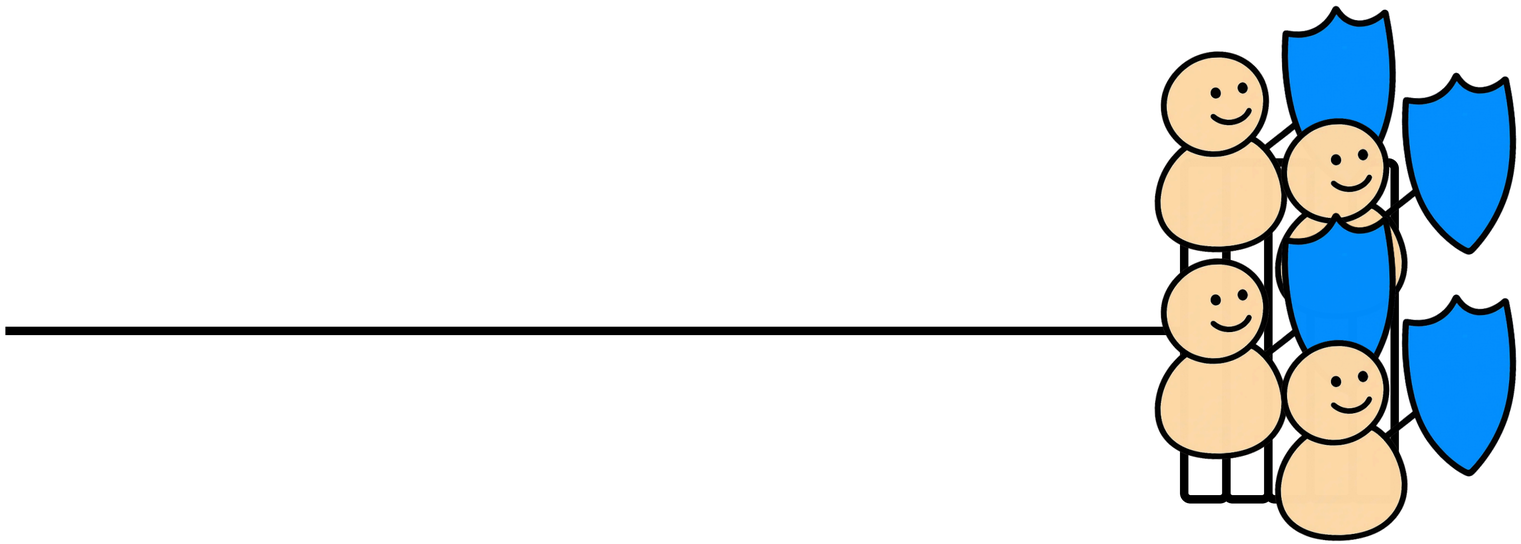
Refresh postponement



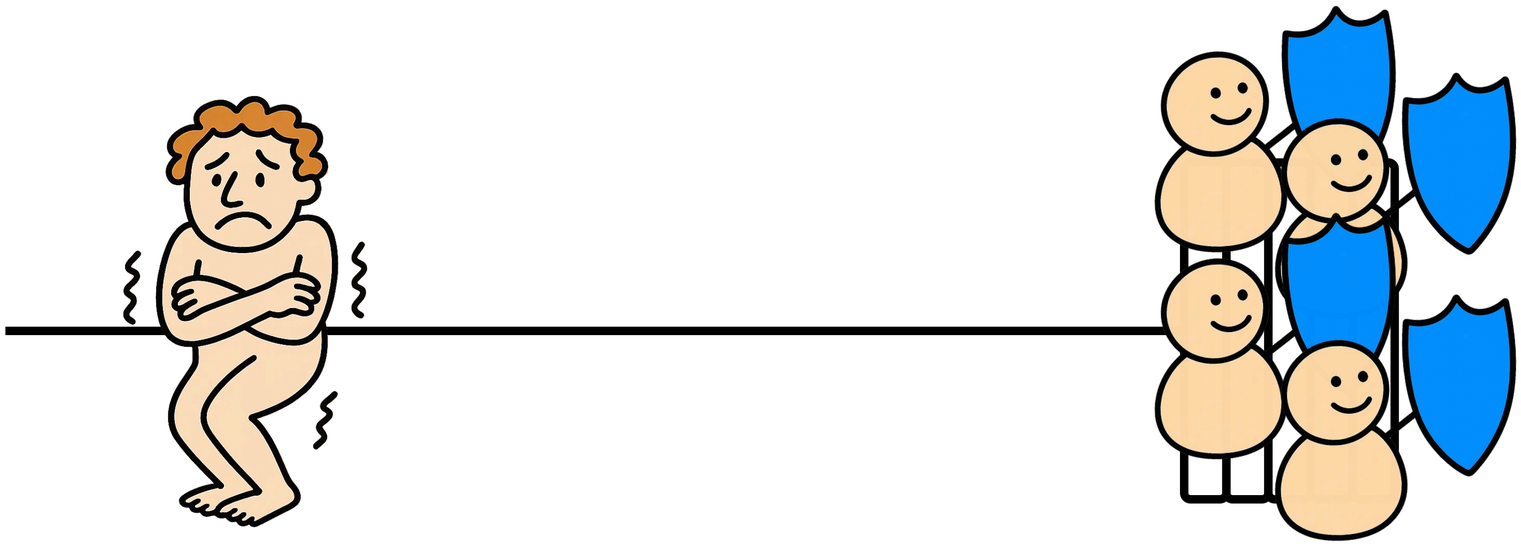
Refresh postponement



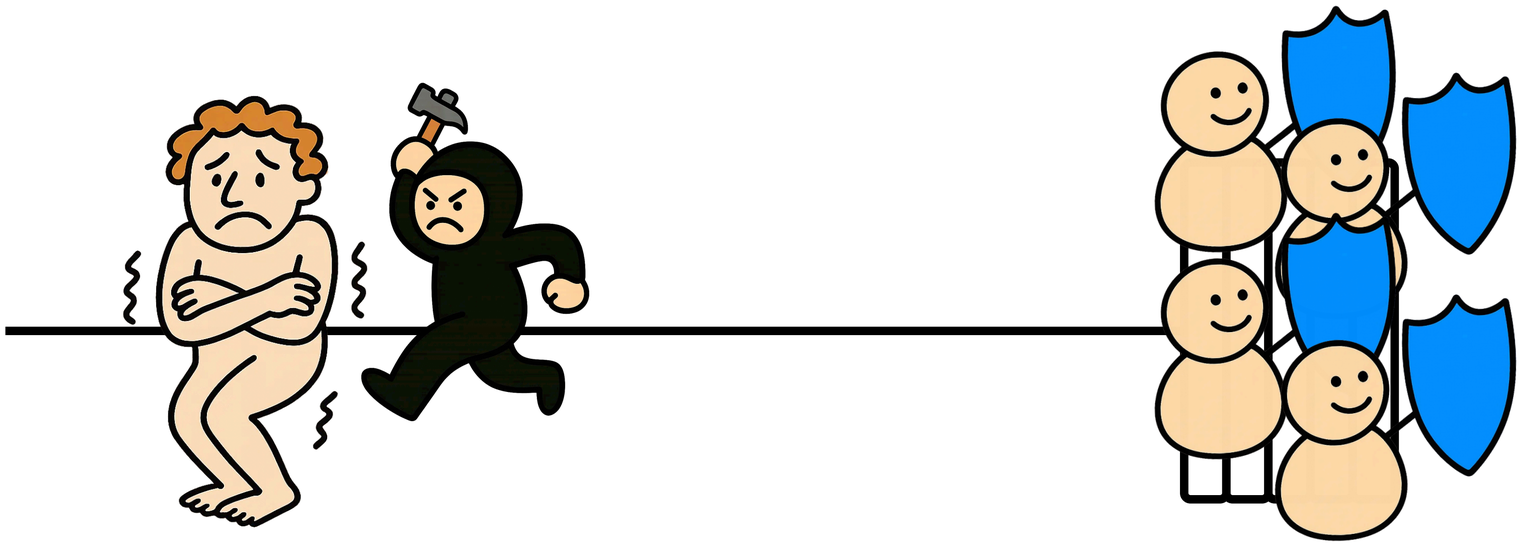
Refresh postponement



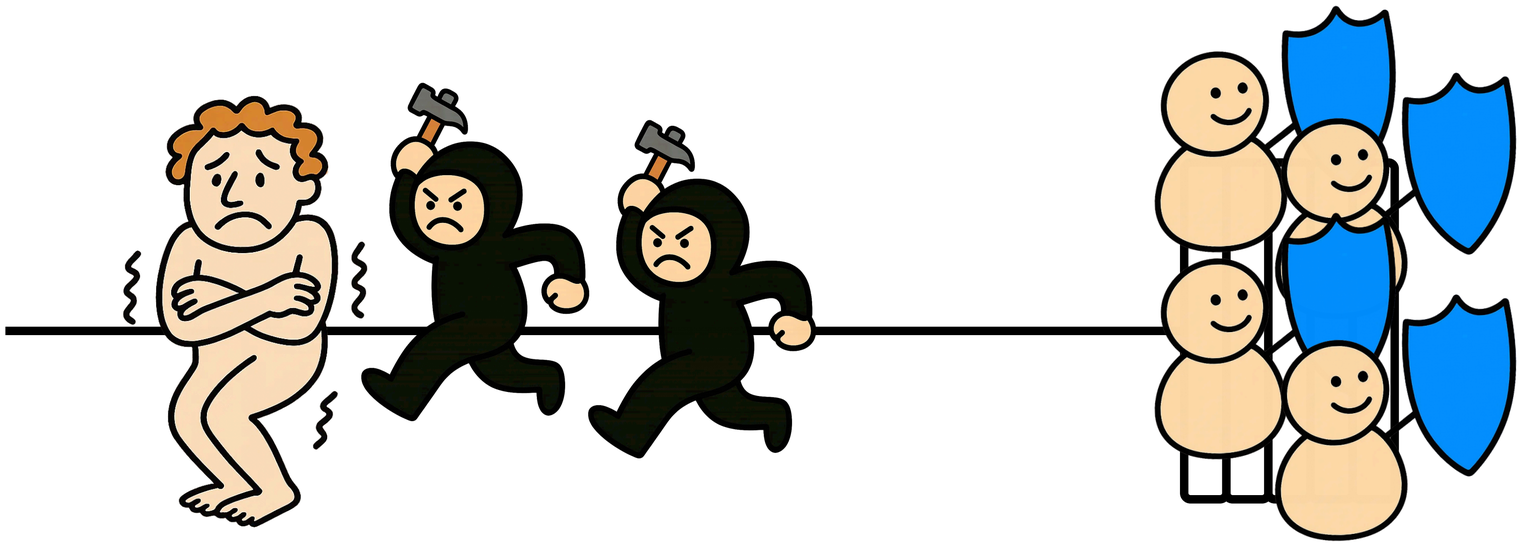
Refresh postponement



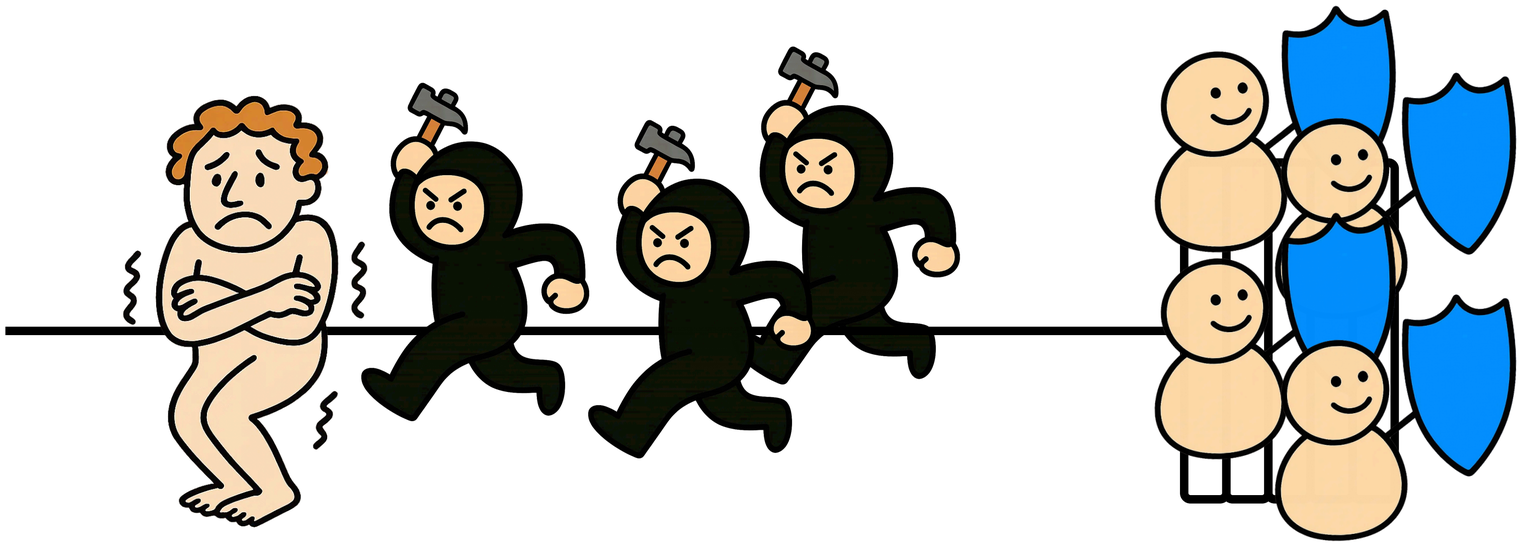
Refresh postponement



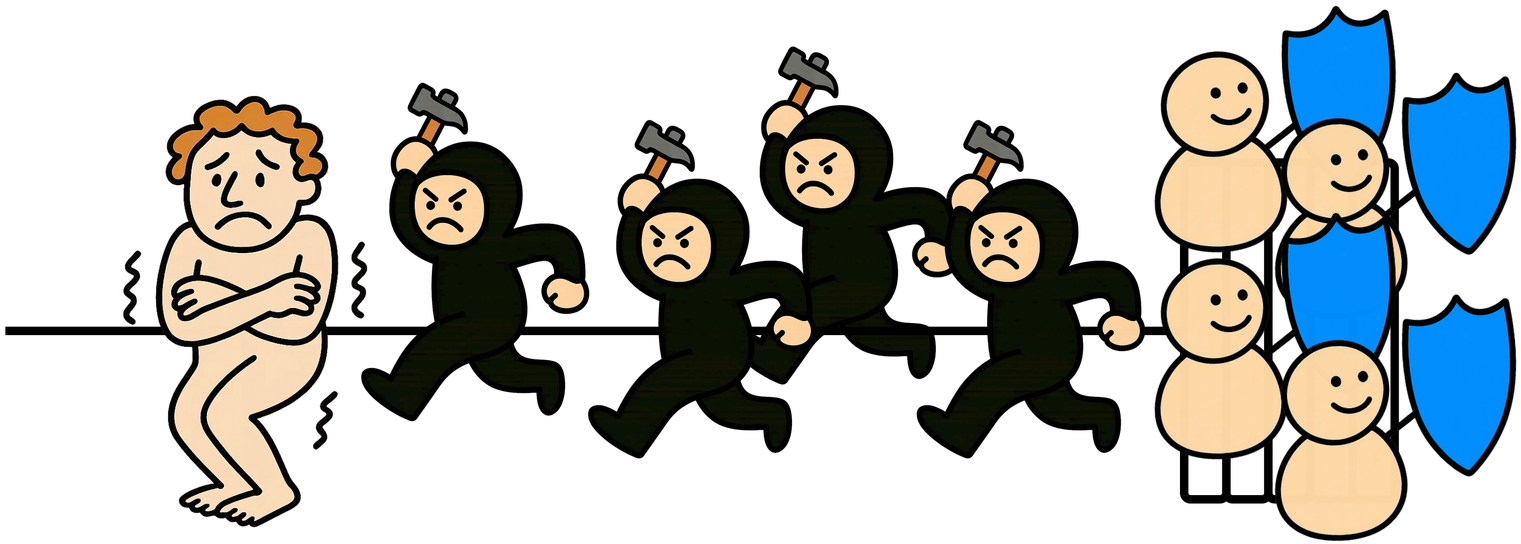
Refresh postponement



Refresh postponement



Refresh postponement



Insight 2

Uniform vs. non-uniform patterns

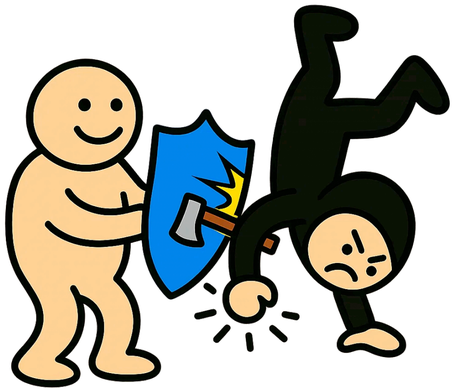
Uniform patterns

Uniform patterns

Uniform patterns

Uniform patterns

Uniform patterns



Non-uniform patterns

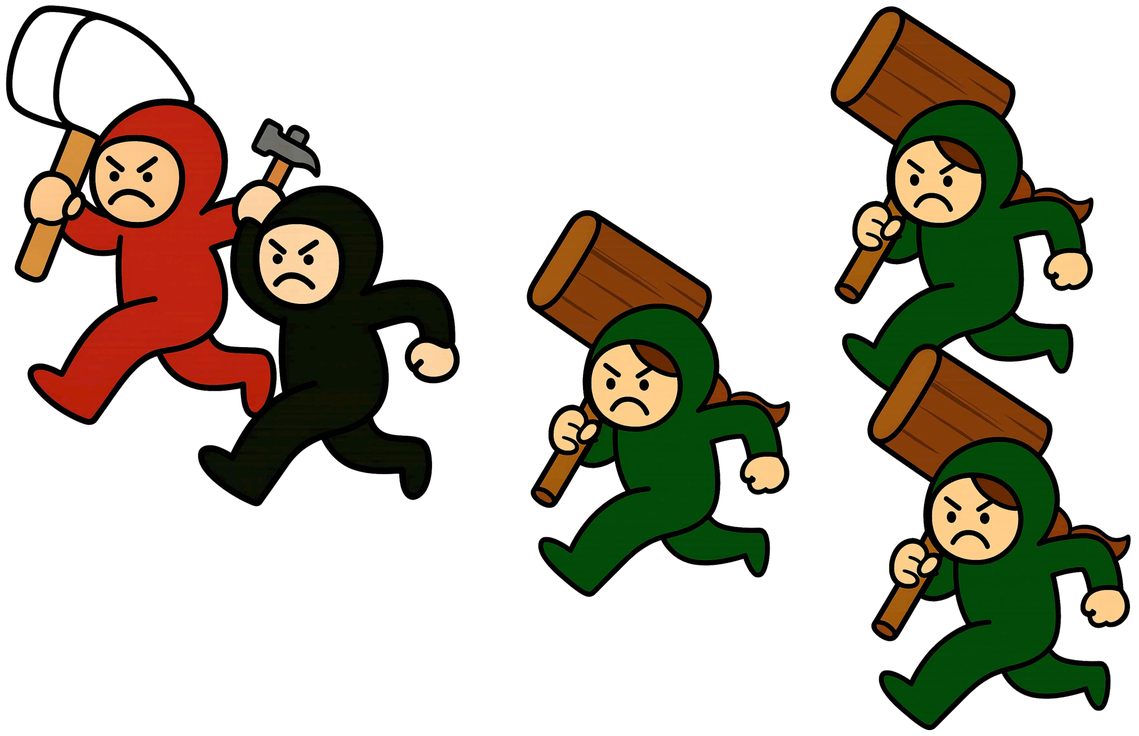
Non-uniform patterns

Non-uniform patterns

Non-uniform patterns

Non-uniform patterns

Non-uniform patterns



Non-uniform patterns

Non-uniform patterns

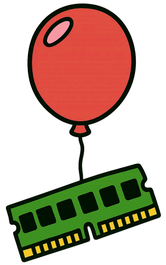
Non-uniform patterns in the browser

Non-uniform patterns in the browser

Challenge

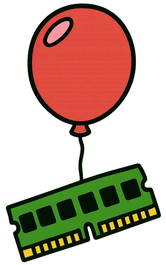
Reaching DRAM

Reaching DRAM

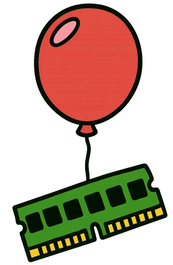


Cache line flush (native)

Reaching DRAM



Cache line flush (native)



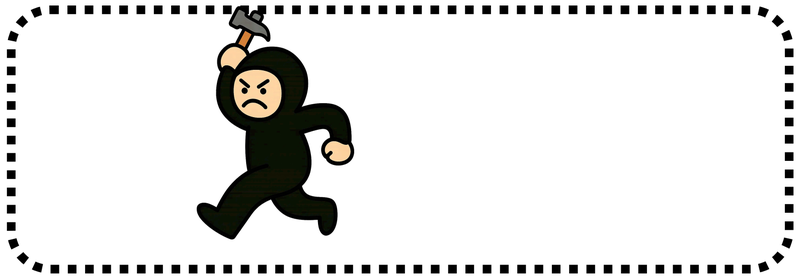
Eviction set (browser)

Eviction sets

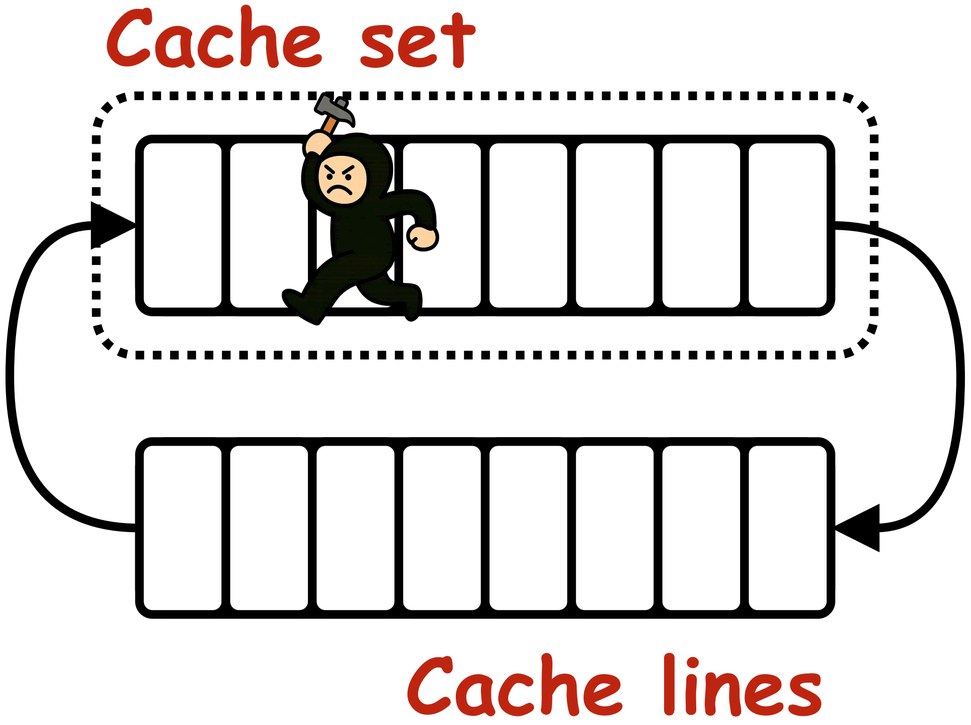
Cache set

Eviction sets

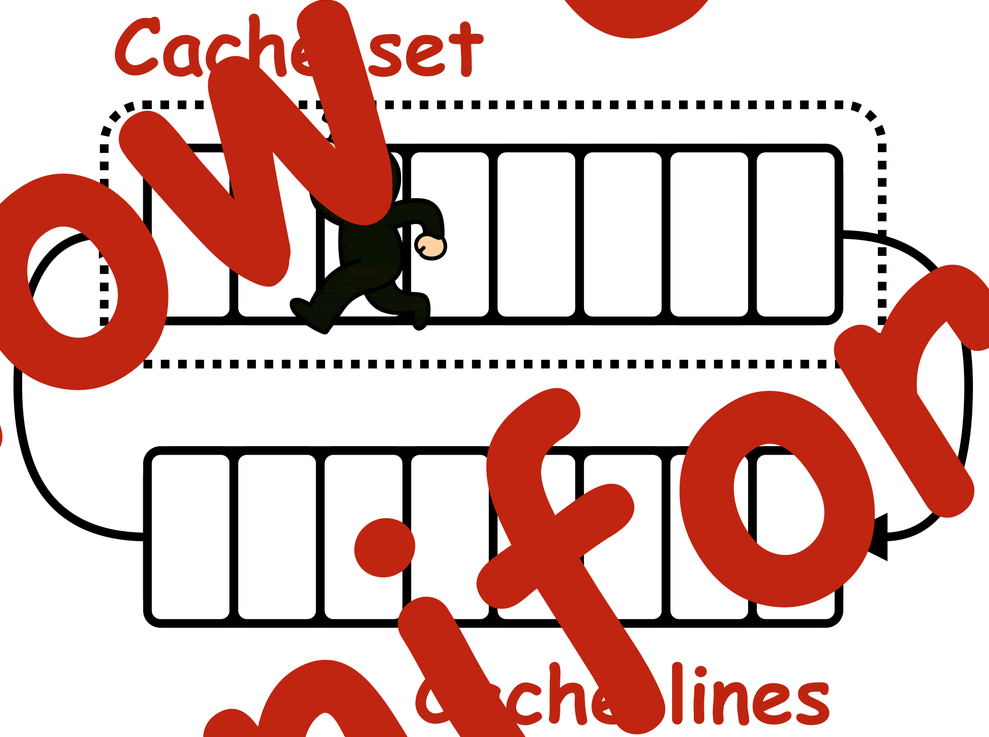
Cache set



Eviction sets

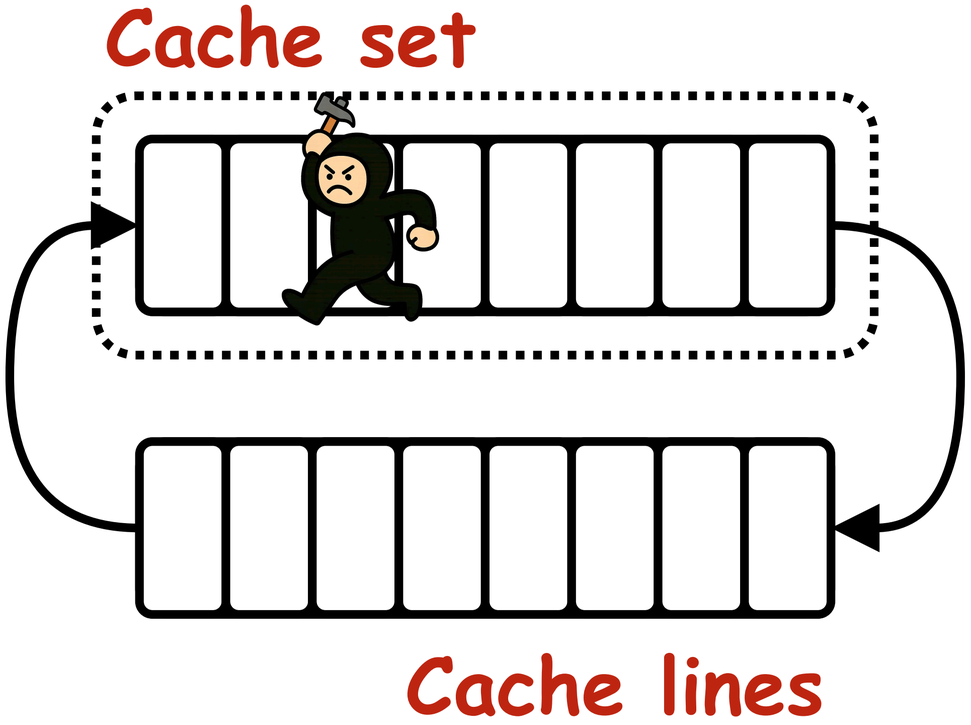


Eviction sets

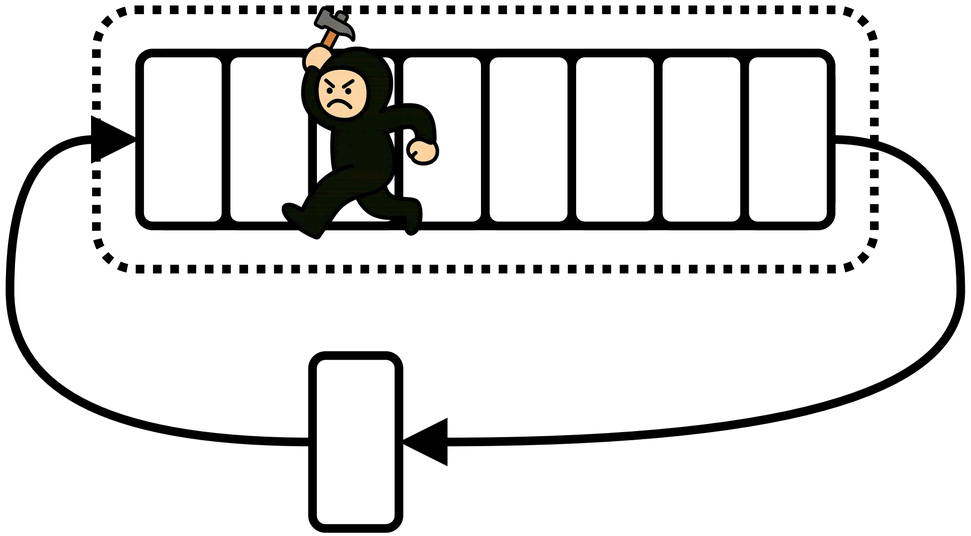


Slow and uniform

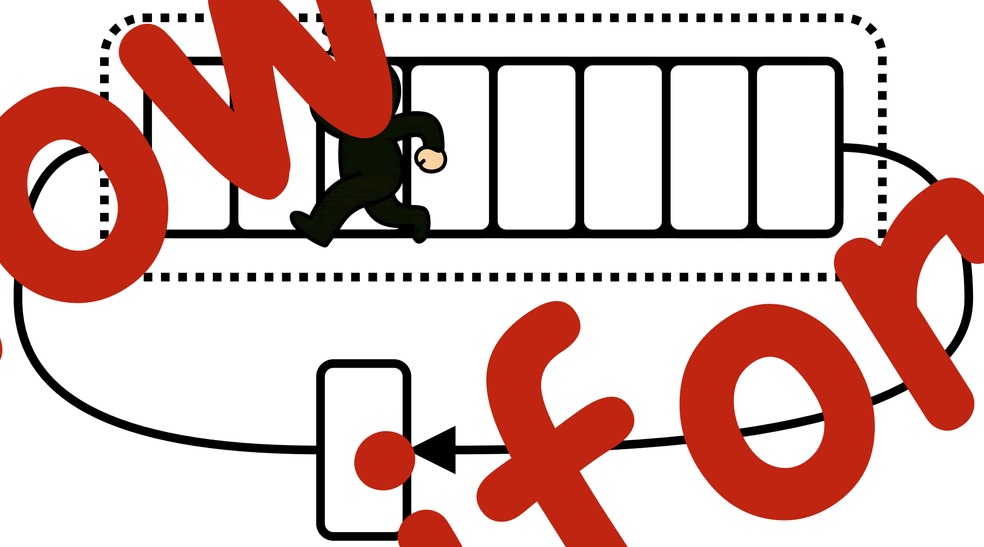
Eviction sets



Minimal eviction sets

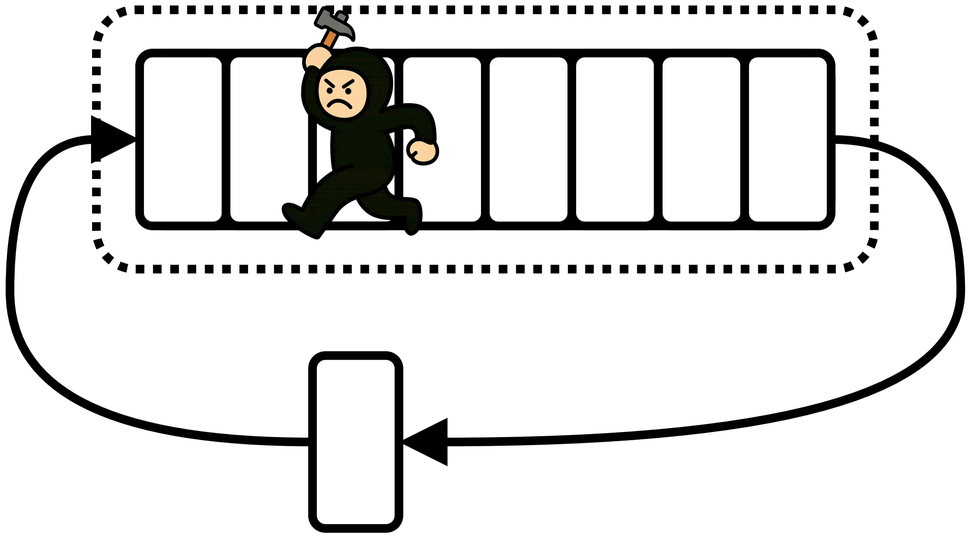


Minimal eviction sets

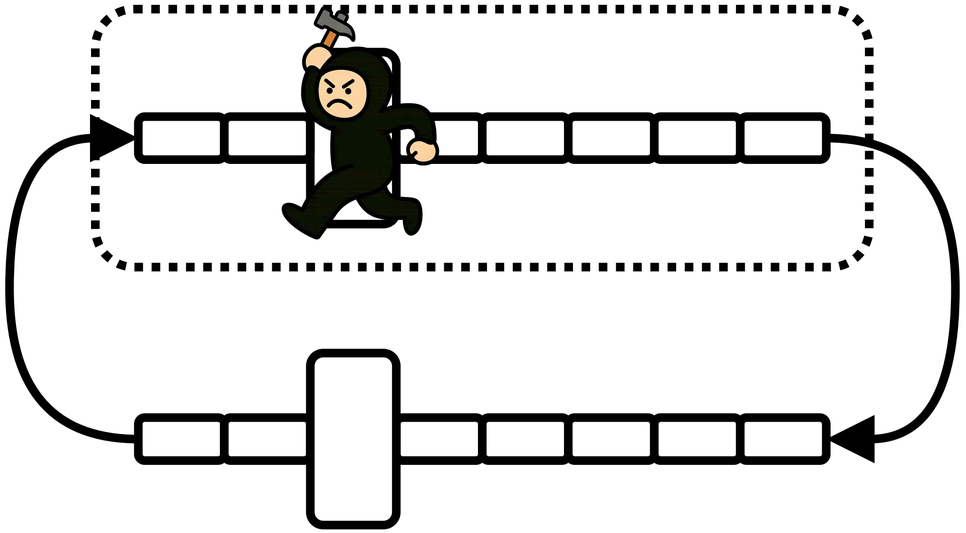


Slow and uniform

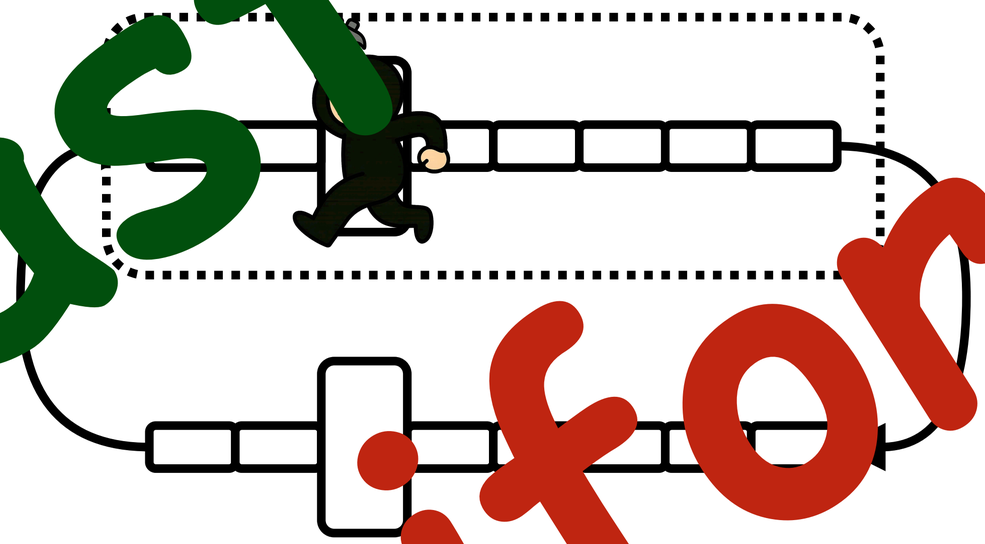
Minimal eviction sets



Introducing cache hits

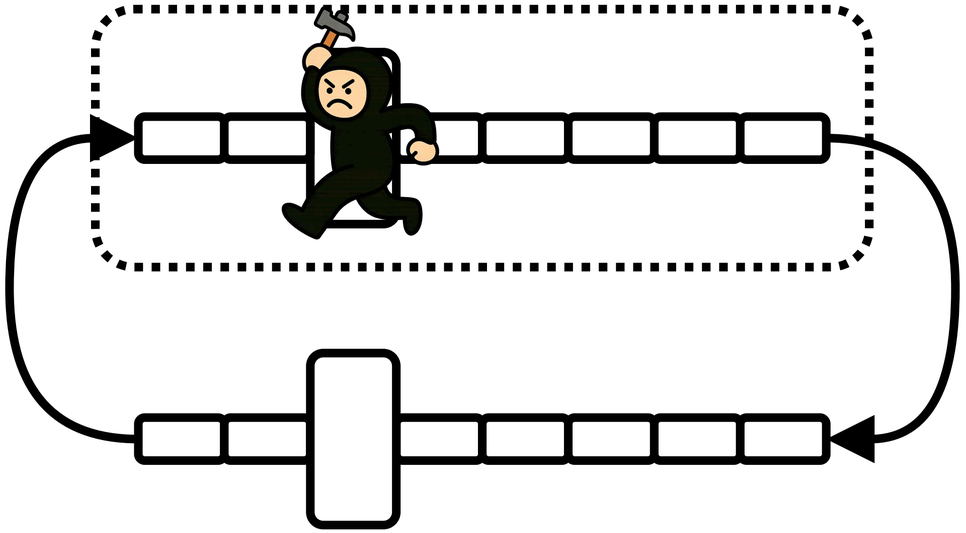


Introducing cache hits

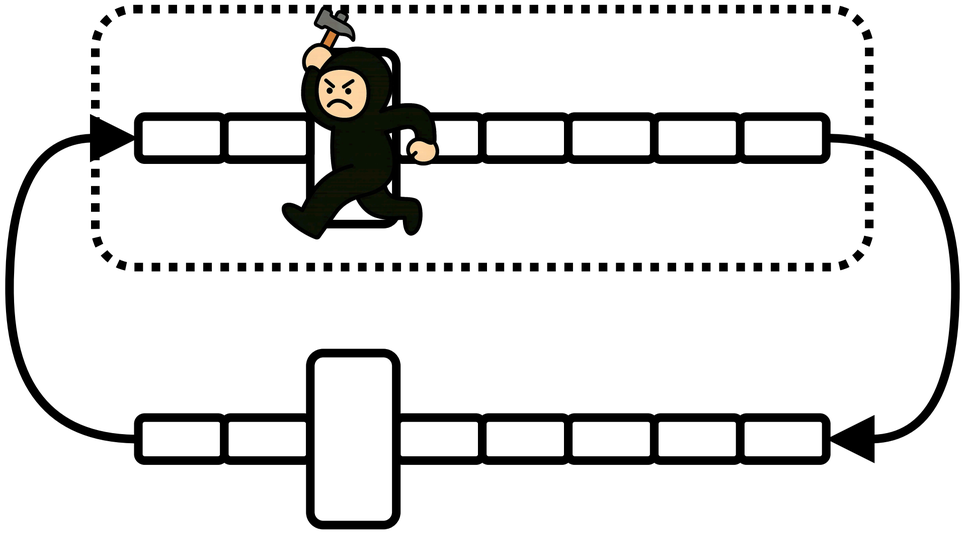


Fast but uniform

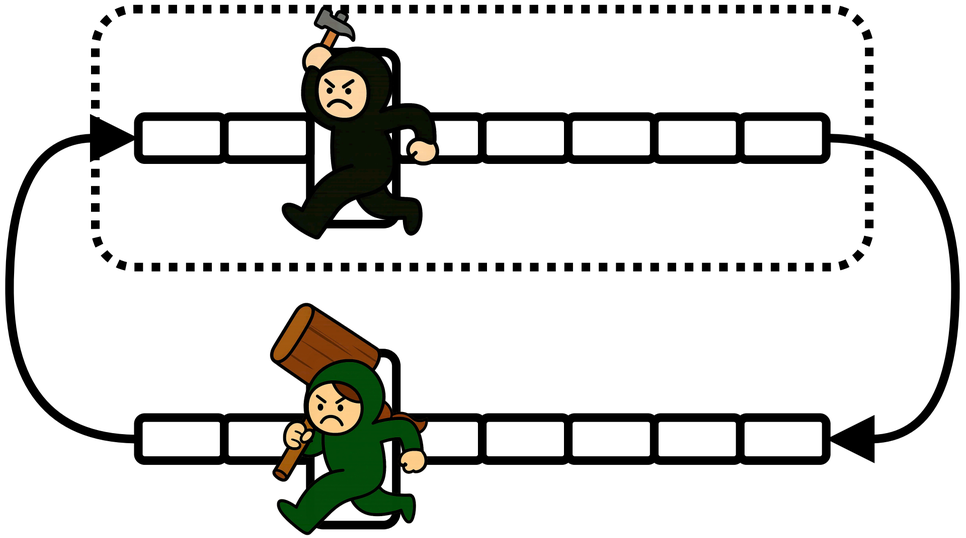
Introducing cache hits



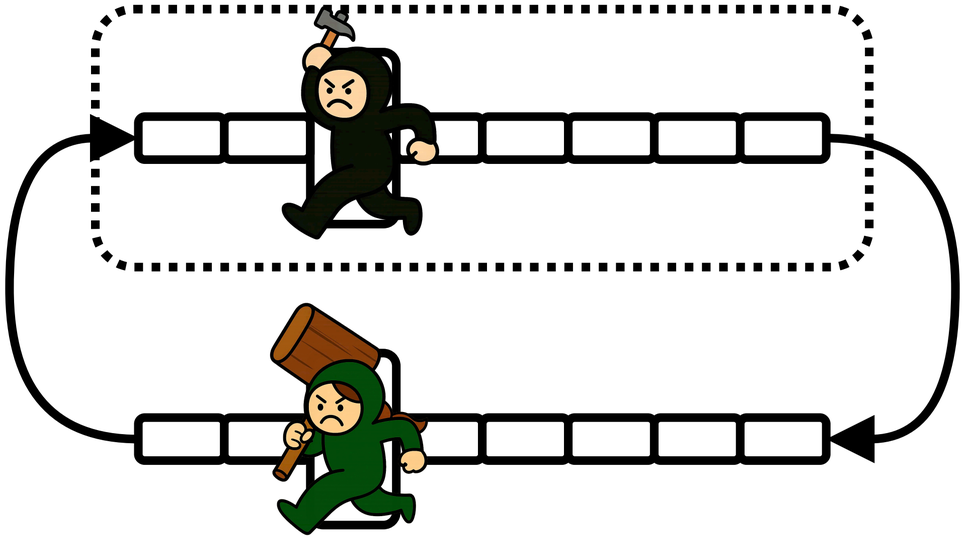
More aggressors



More aggressors



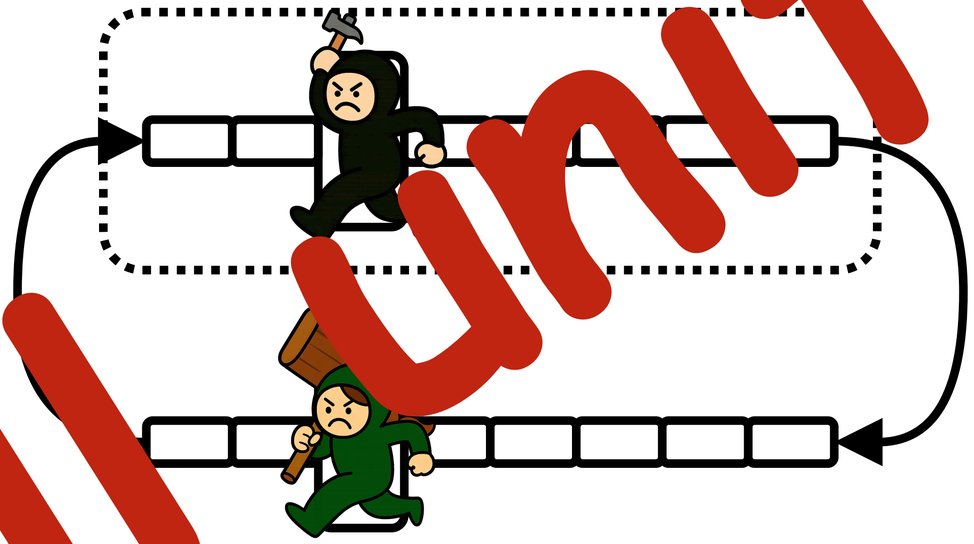
More aggressors



1x 

1x 

More aggressors

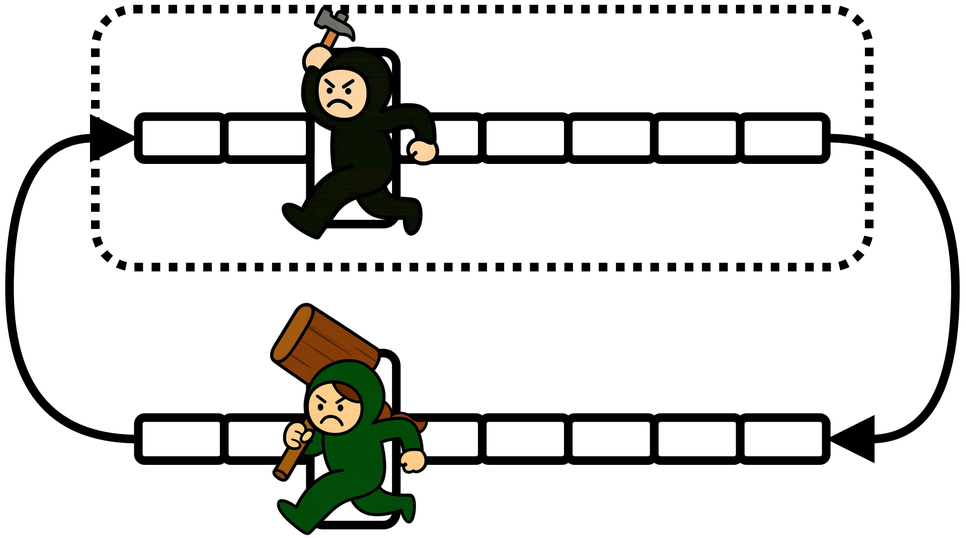


1×

1×

Still uniform

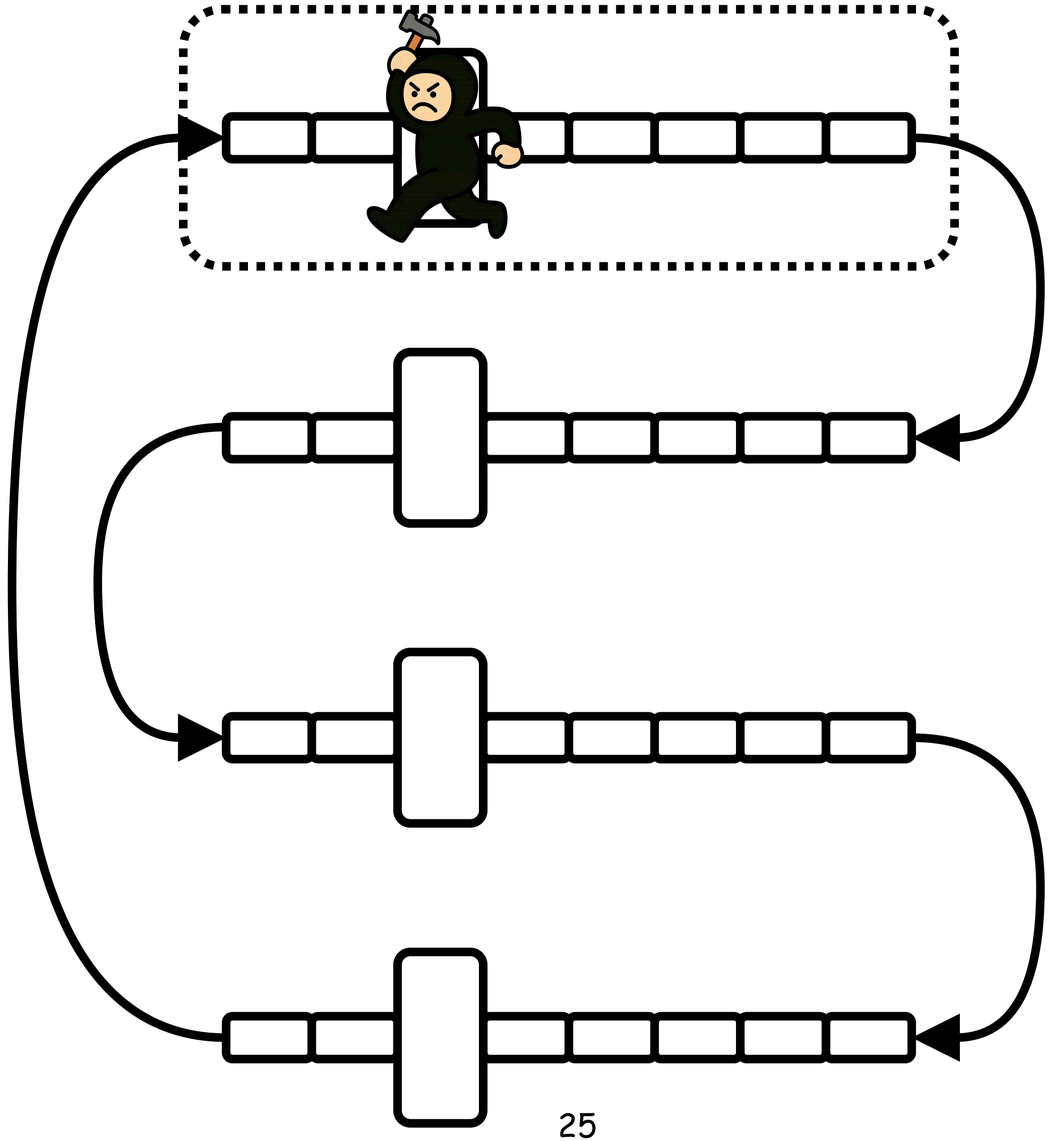
More aggressors



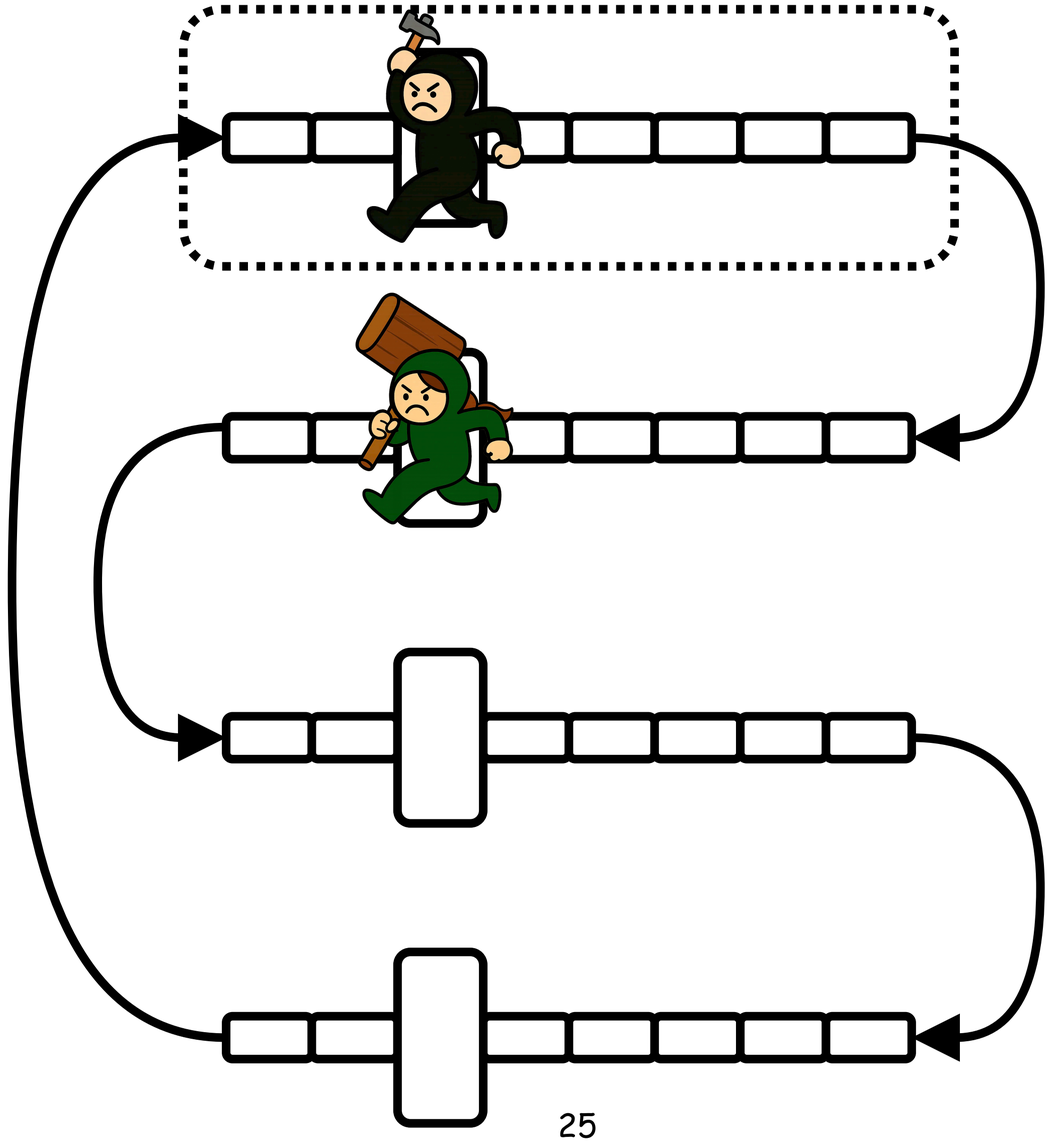
1x 

1x 

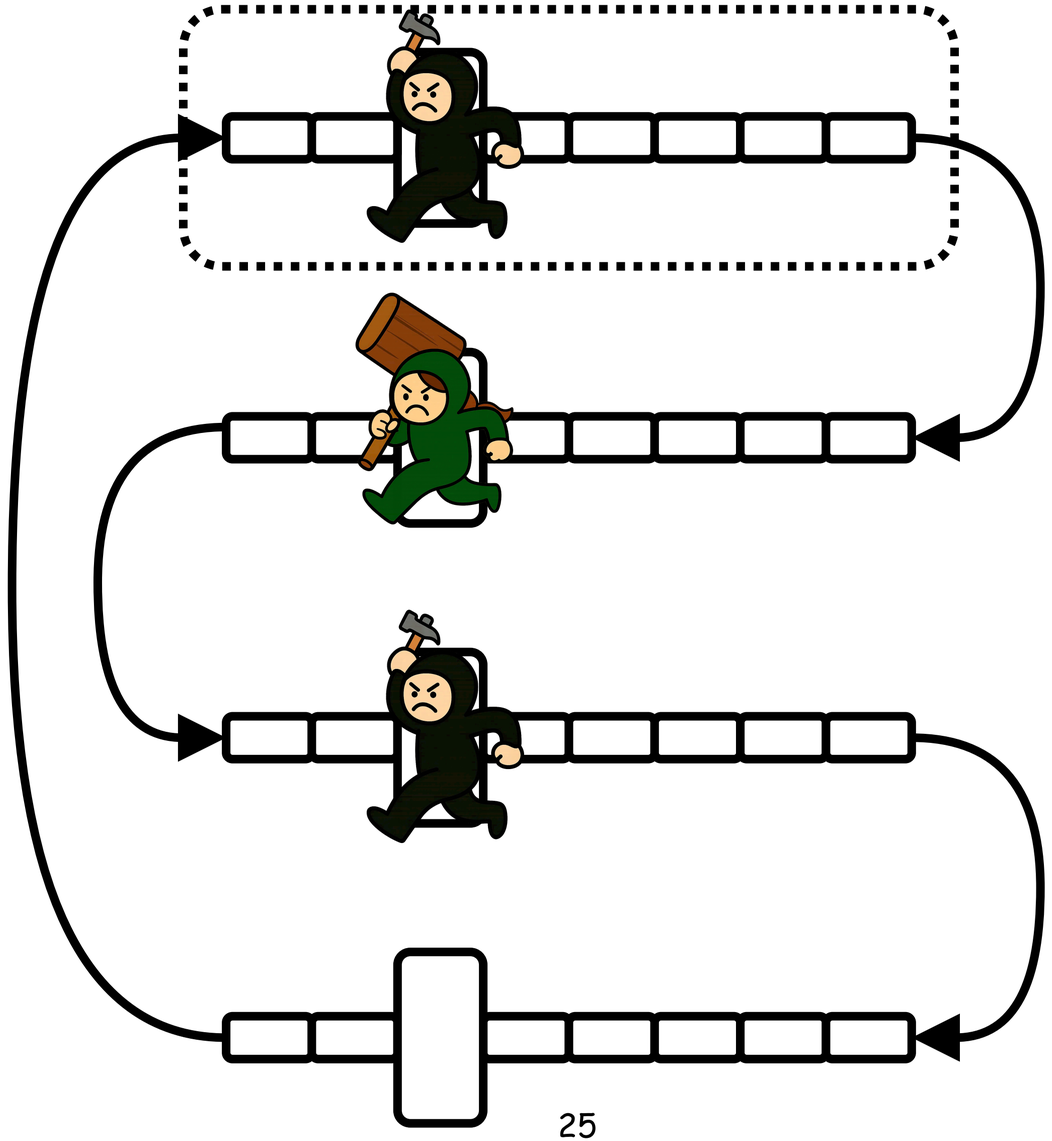
Introducing lanes



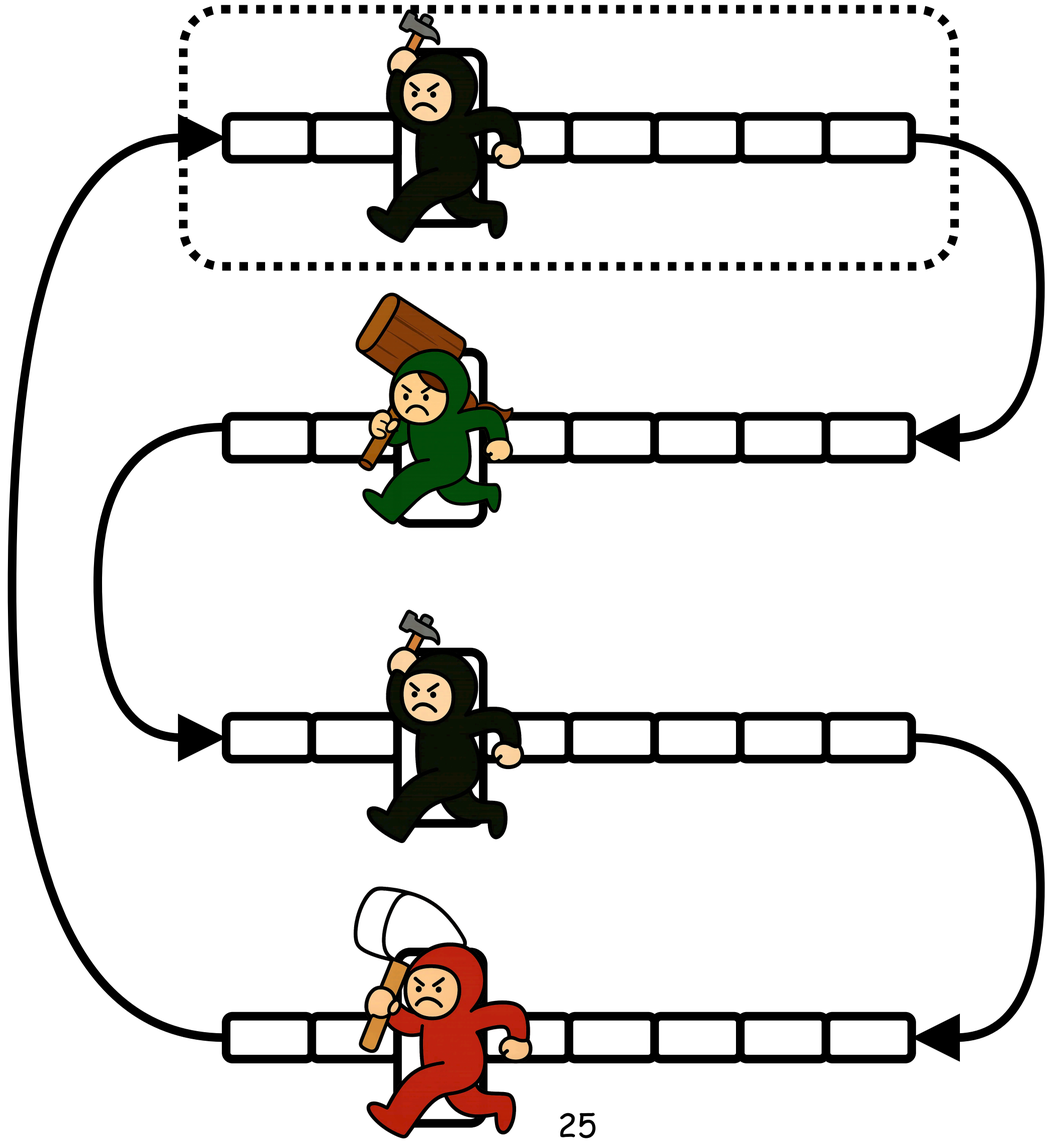
Introducing lanes



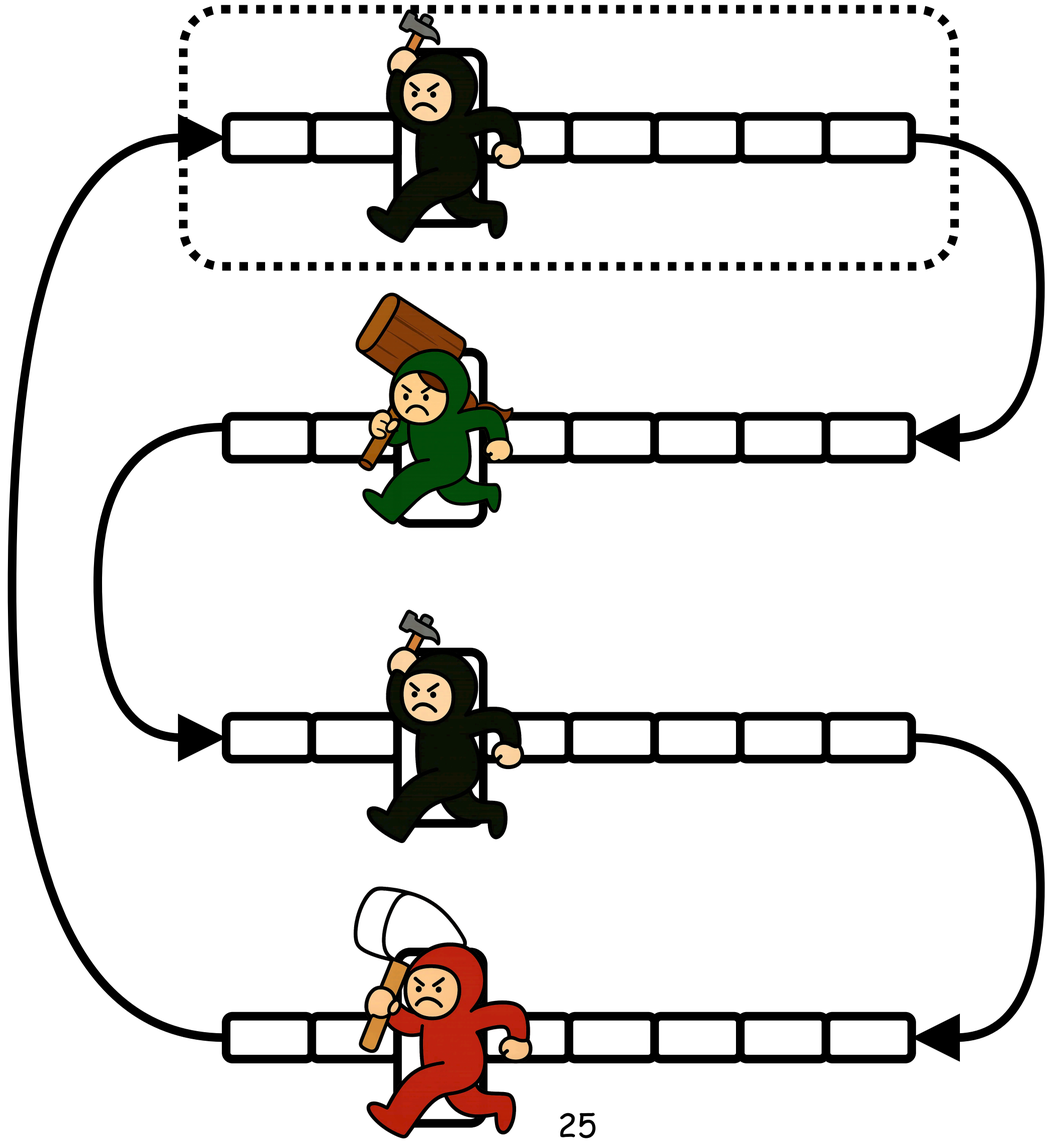
Introducing lanes



Introducing lanes



Introducing lanes



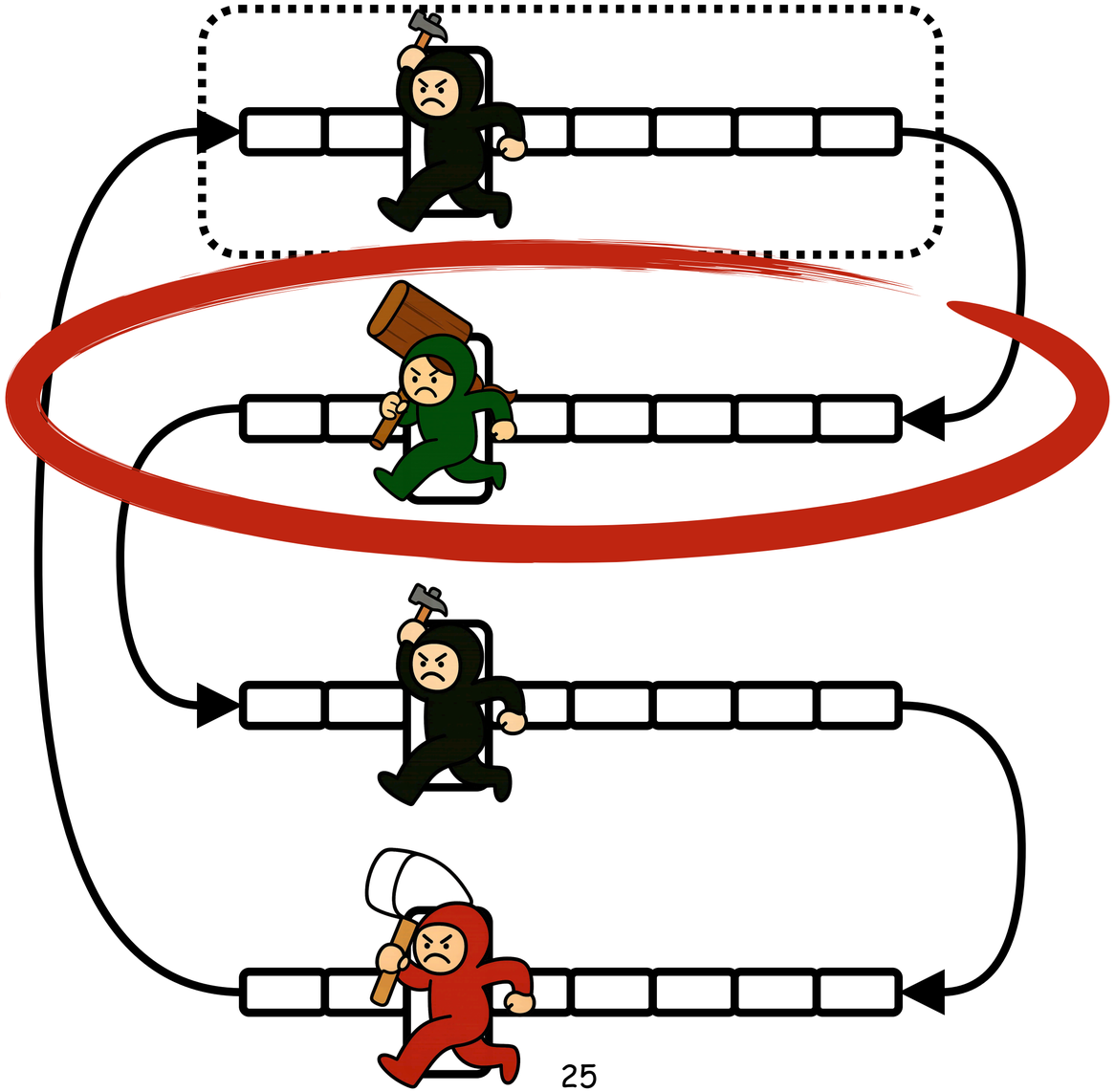
2x 

1x 

1x 

Introducing lanes

Lane

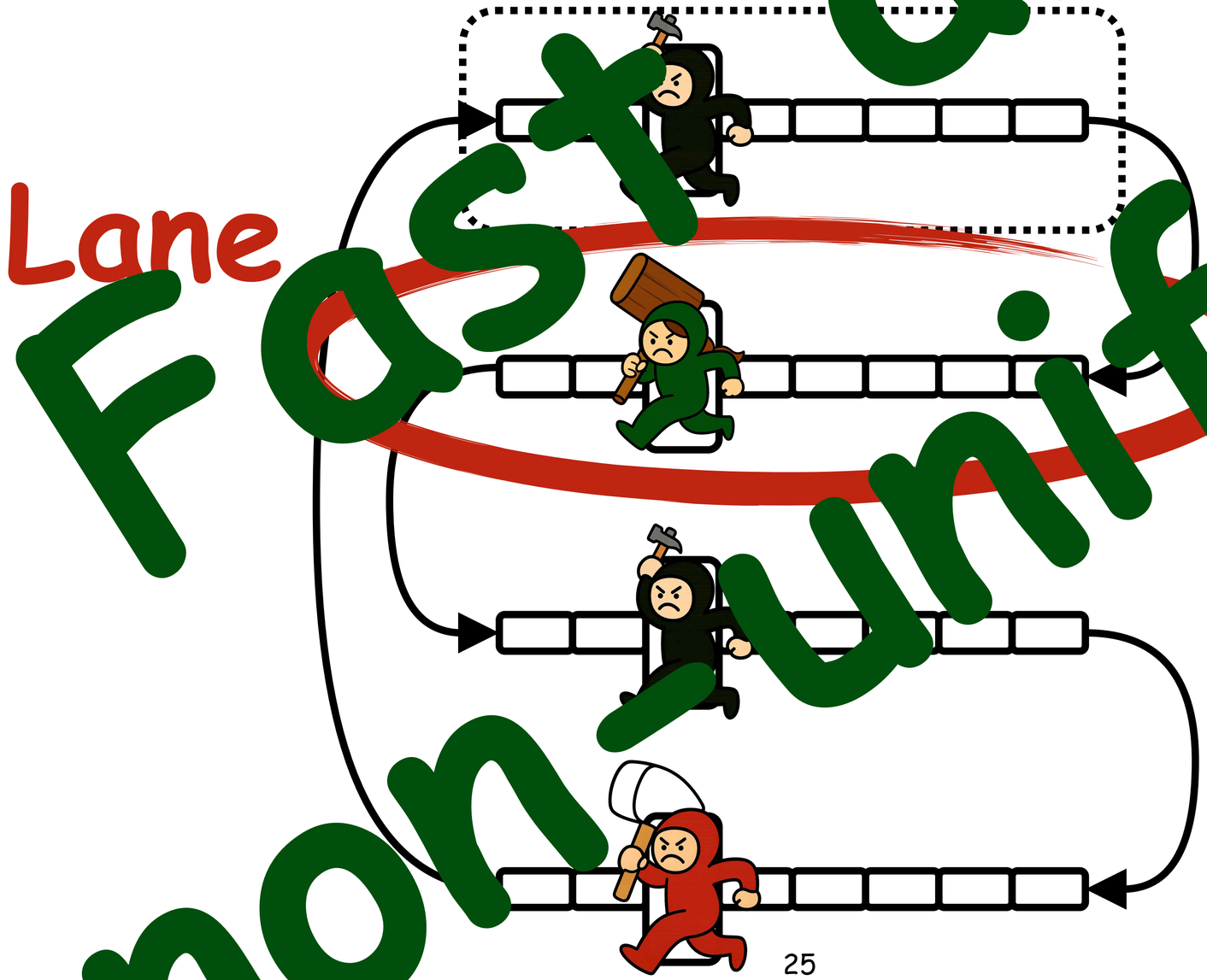


2x

1x

1x

Introducing lanes



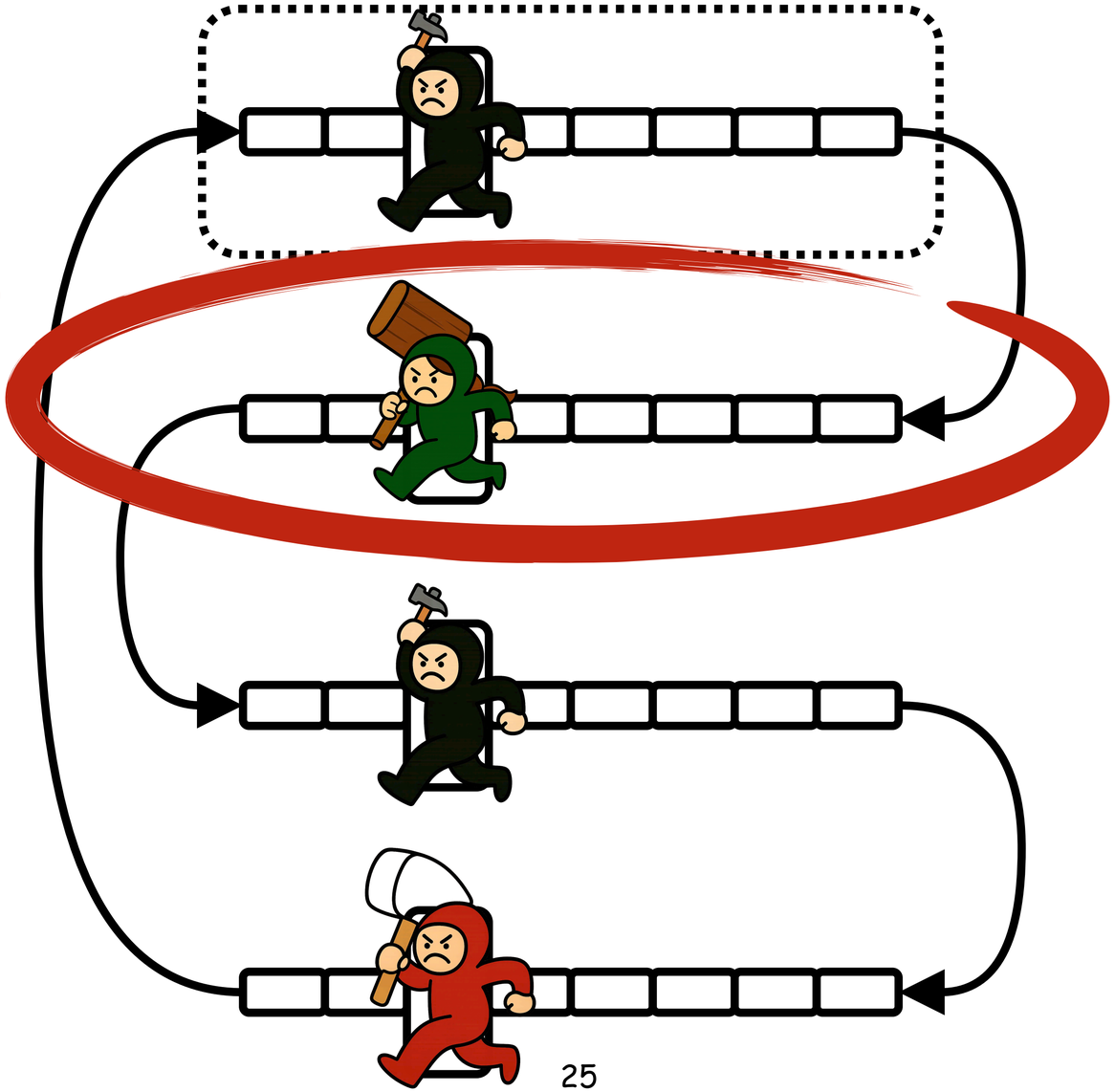
2×

1×

1×

Introducing lanes

Lane



2x

1x

1x

Some examples

Some examples

50% · 25% · 25%

Some examples

50% · 25% · 25%

50% · 33% · 17%

Some examples

50% · 25% · 25%

50% · 33% · 17%

50% · 38% · 12%

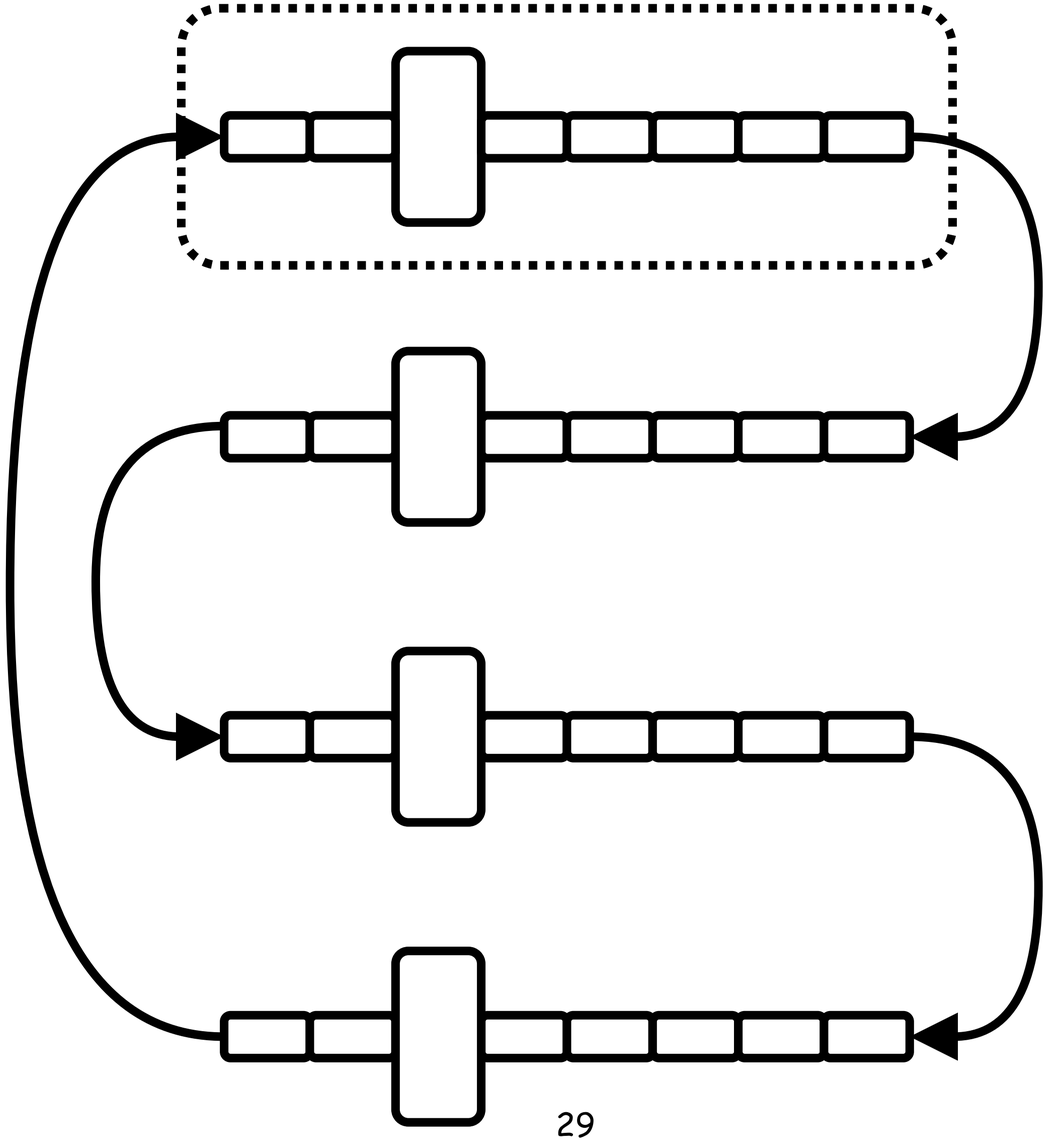
Insight 3

Few patterns, big impact

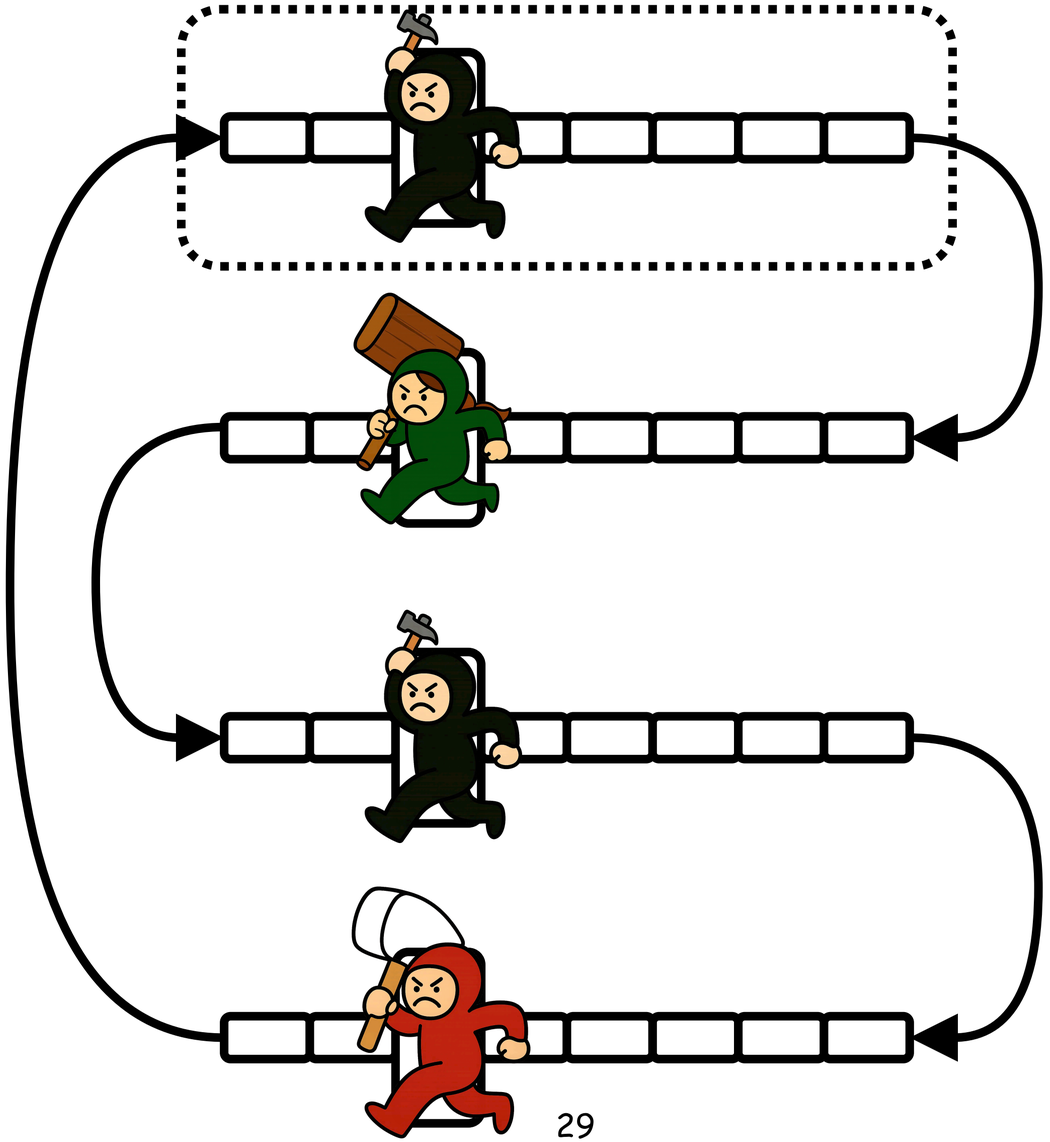
Few patterns, big impact

Found two types of patterns that bypass all mitigations

Recall

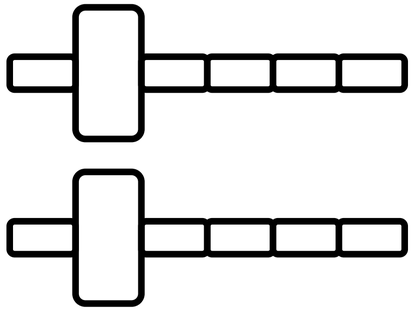


Recall

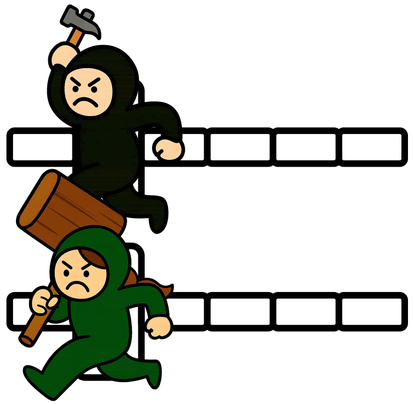


Example

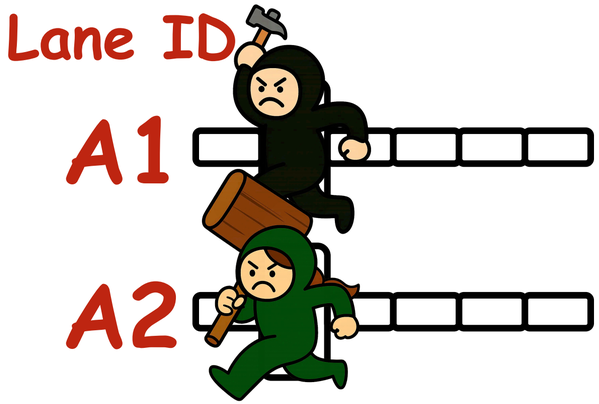
Example



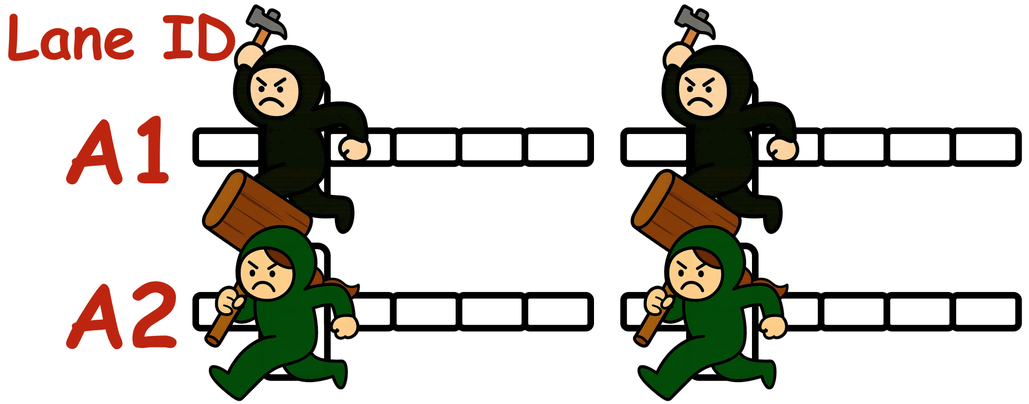
Example



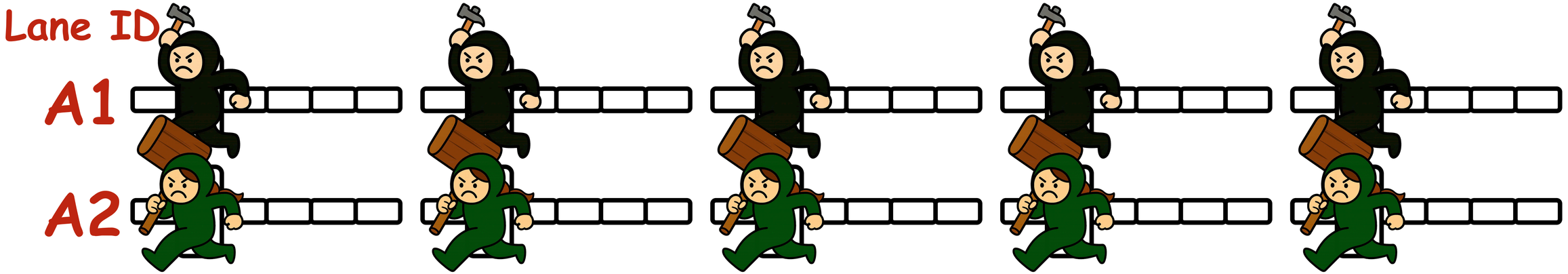
Example



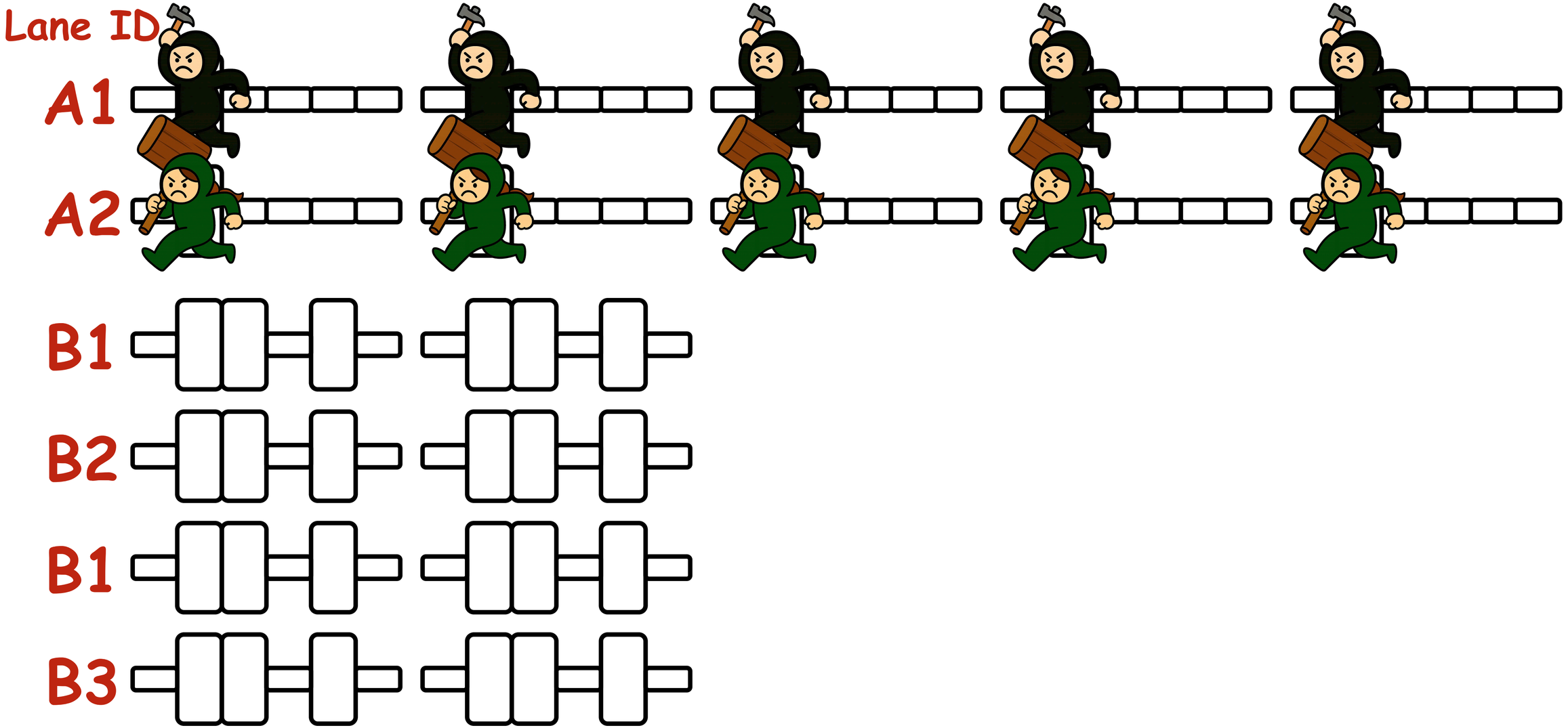
Example



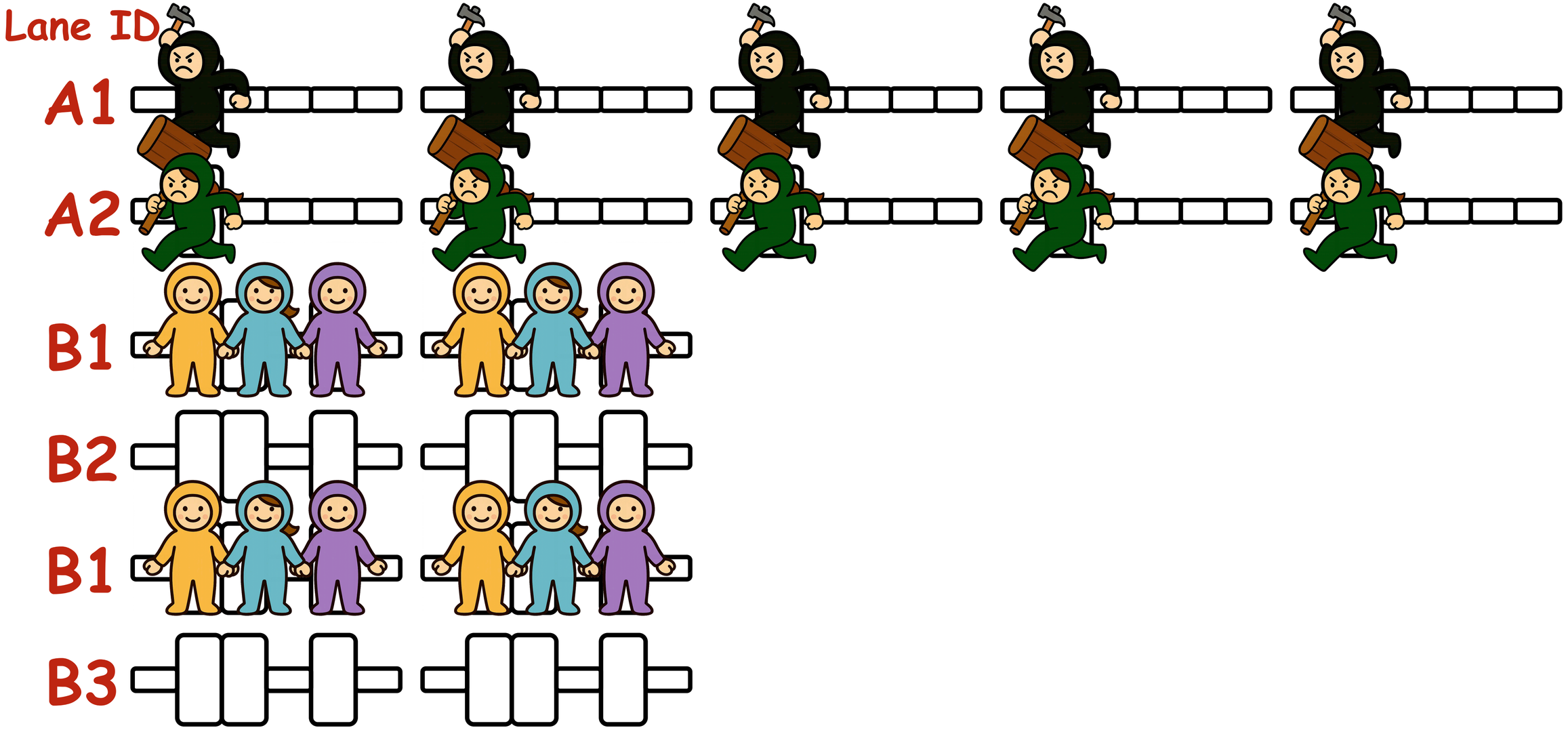
Example



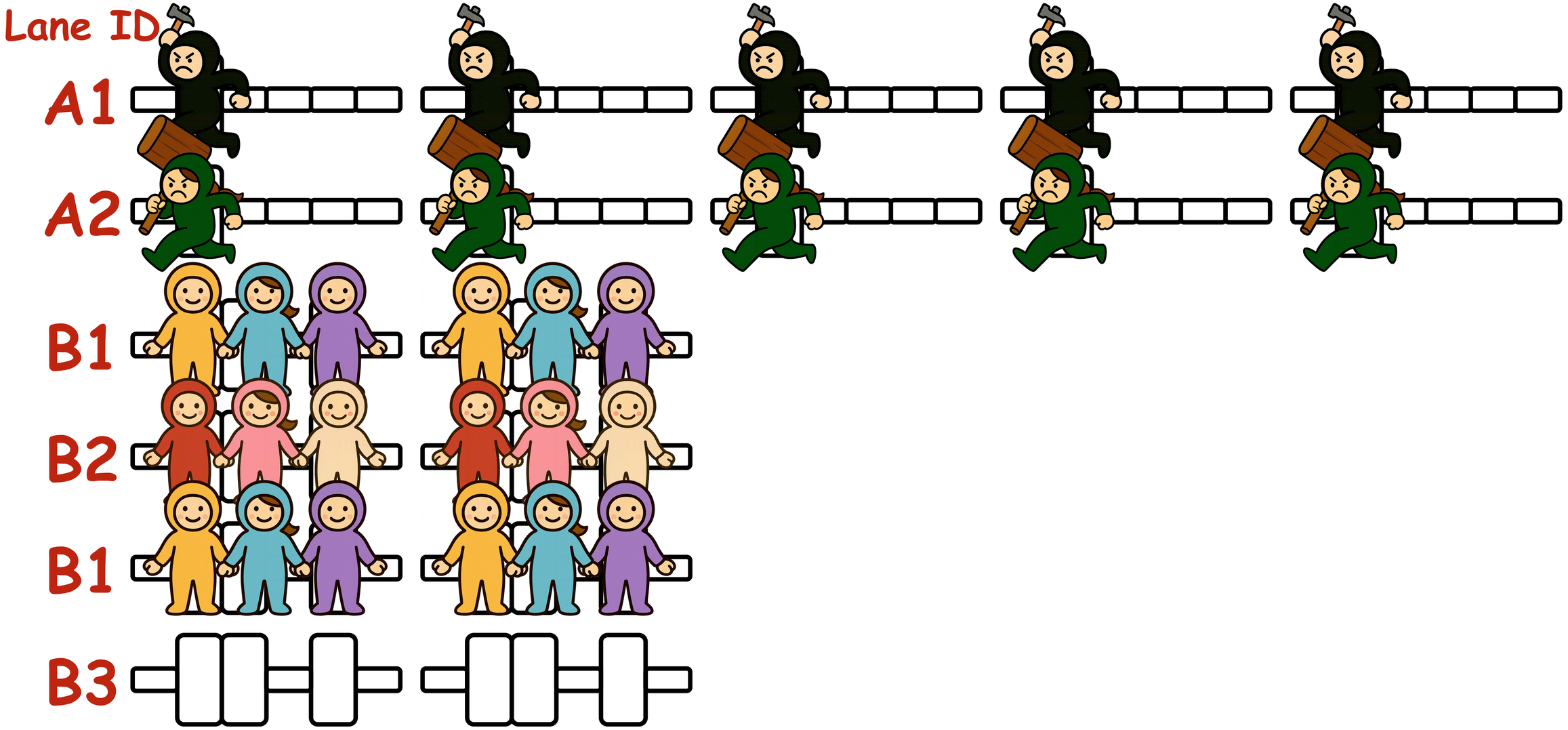
Example



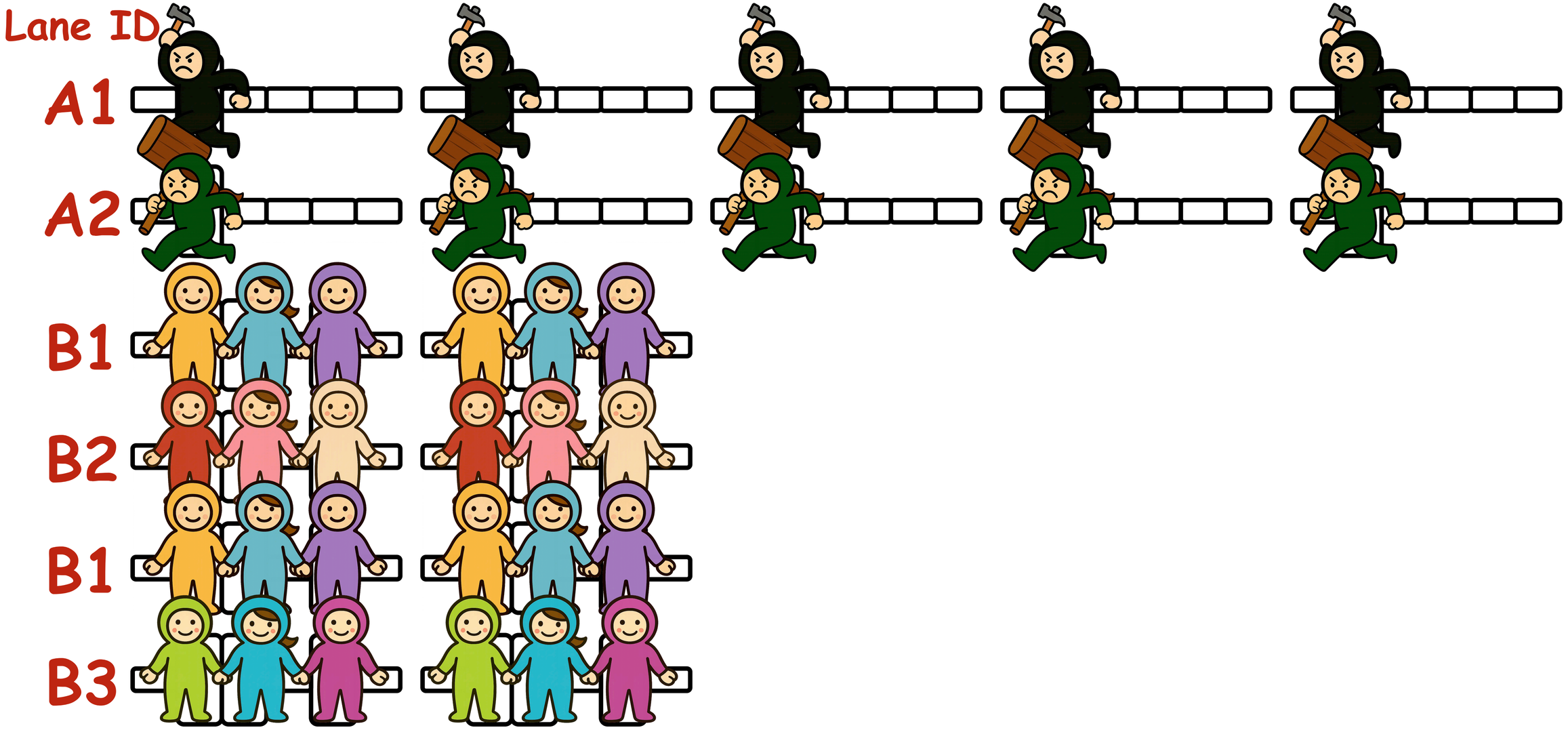
Example



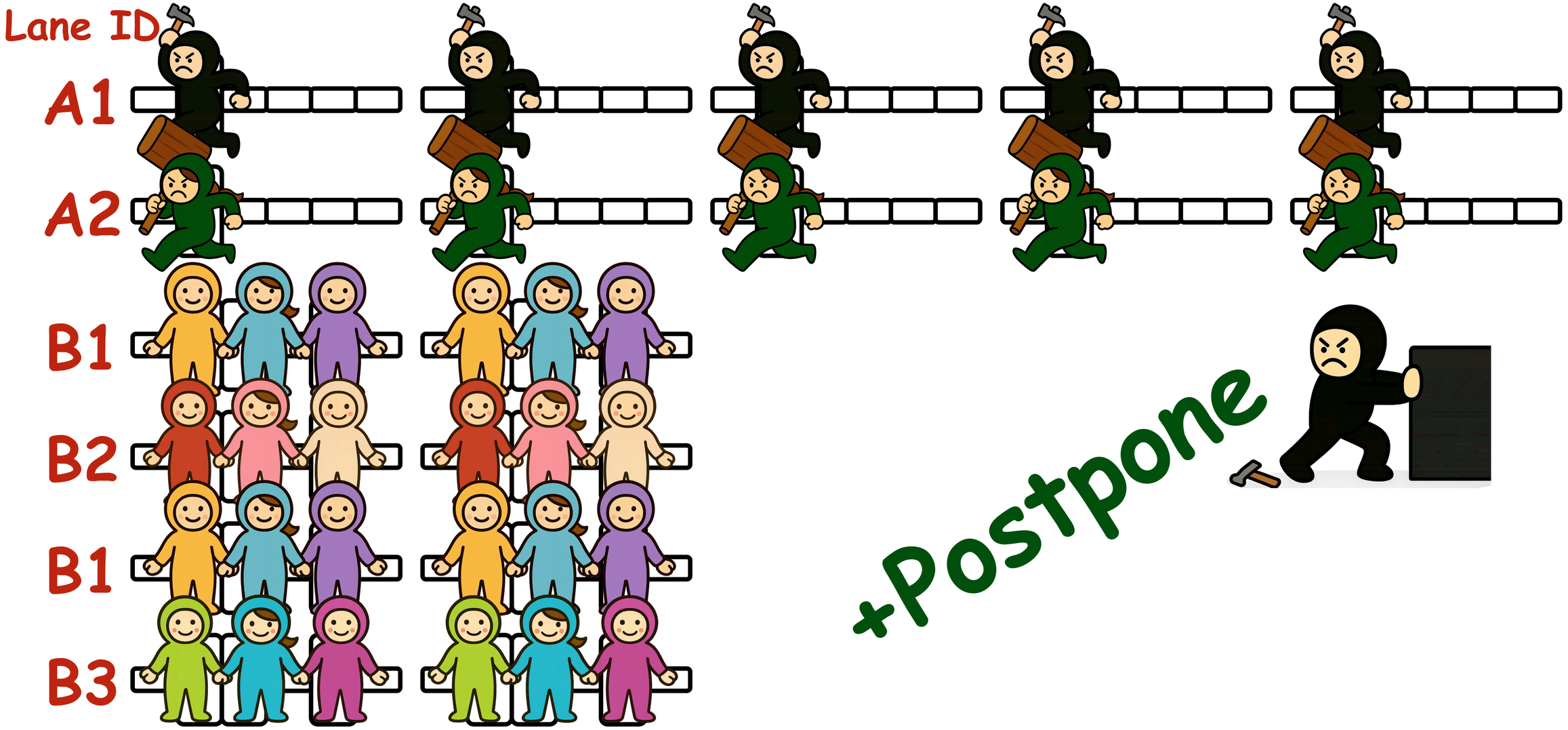
Example



Example



Example



Exploit

Type-flipping

Type-flipping

Type-flipping

1

Allocator exhaustion.

Type-flipping

1

Allocator exhaustion.

Detect 10 MB of contiguous memory using a cache side channel.

Type-flipping

2

Build one eviction set.

Type-flipping

3

Use it to build many more.

Type-flipping

4

Hammer: use slight variations of the Posthammer patterns.

Type-flipping

4

Hammer: use slight variations of the Posthammer patterns.

Search for exploitable bit flips.

Type-flipping

5

Hammer softly to prevent undesired bit flips.

Type-flipping

6

Launch exploit.

Type-flipping

6

Launch exploit.

Free the arrays that contain exploitable bit flips.

Type-flipping

6

Launch exploit.

Free the arrays that contain exploitable bit flips.

Allocate target arrays.

Type-flipping

6

Launch exploit.

Free the arrays that contain exploitable bit flips.

Allocate target arrays.

Safely rediscover the exploitable bit flips.

Type-flipping

7

Retrigger first bit flip: 0 to 1.

Type-flipping

7

Retrigger first bit flip: 0 to 1.

Turns an array into a floating point number.

Type-flipping

8

Retrigger the second bit flip: 1 to 0.

Type-flipping

8

Retrigger the second bit flip: 1 to 0.

Turns a floating point number into a string reference.

Type-flipping

9

Retrigger the first bit flip: 0 to 1.

Type-flipping

9

Retrigger the first bit flip: 0 to 1.

Creates a valid but fake array.

Type-flipping

10

We now control the array's data pointer.

Type-flipping

10

We now control the array's data pointer.

Read/write anywhere in the address space of the JavaScript process.

Type-flipping

Free of huge pages

10

We now control the array's data pointer.

Read/write anywhere in the address space of the JavaScript process.

Type-flipping

Free of huge pages

10

Pretty reliable (40% success rate)

We now control the array's data pointer.

Read/write anywhere in the address space of the JavaScript process.

Type-flipping

Free of huge pages

10

Pretty reliable (40% success rate)

We now control the array's data pointer.

Read/write anywhere in the address space of the JavaScript process.

Conclusion

Conclusion

Conclusion

Refresh postponement

Conclusion

1

Refresh postponement

2

Conclusion

1

Refresh postponement

2 Lanes for non-uniformity

Conclusion

1

Refresh postponement

2 Lanes for non-uniformity

3

Conclusion

1

Refresh postponement

2 Lanes for non-uniformity

3 Pattern generalization