CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

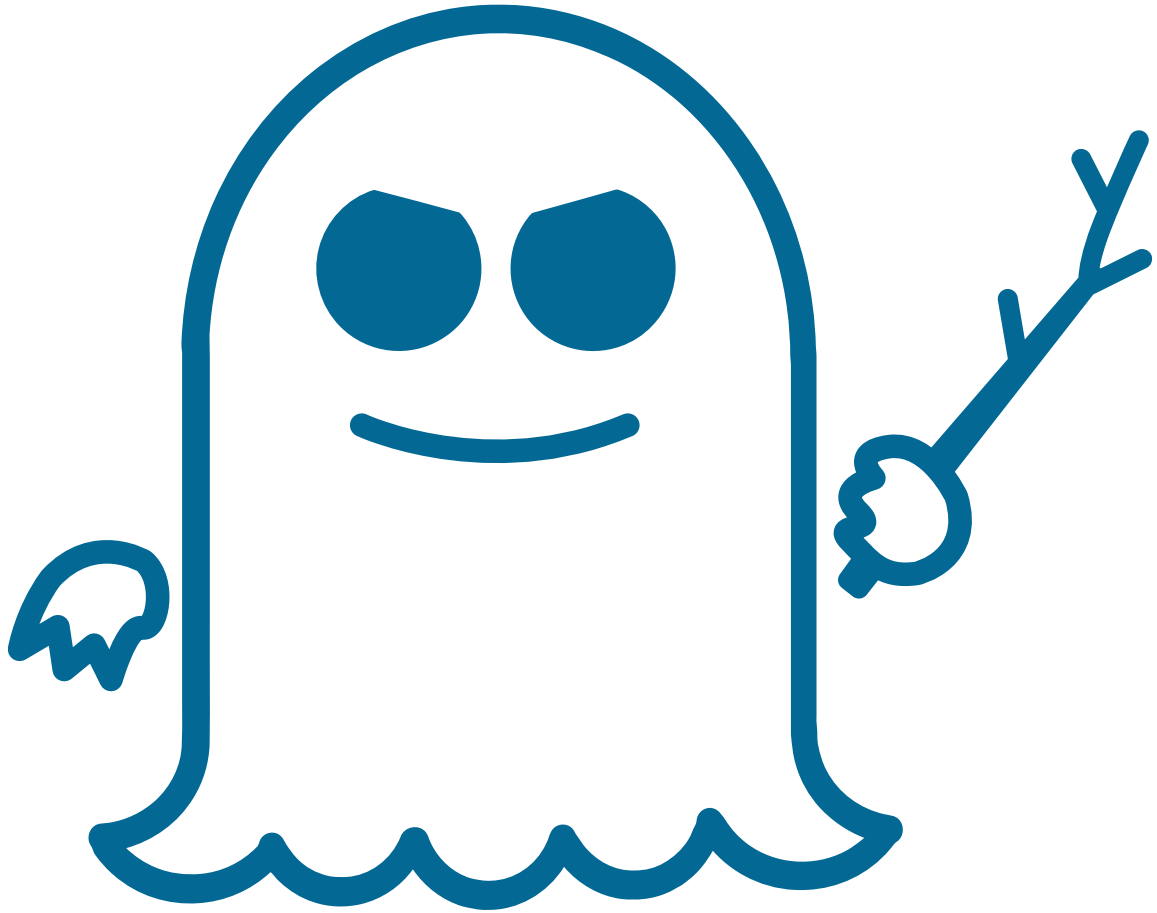
Automatic Detection of Speculative Execution Combinations

Xaver Fabian[1], Marco Guarnieri[2], Marco Patrignani[3]

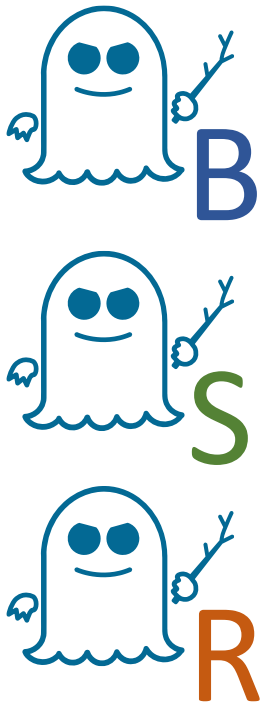
1 CISPA Helmholtz Center for Information Security
2 IMDEA Software Institute
3 University of Trento

Spectre

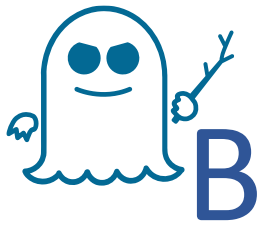
- Vulnerability exploiting Speculative execution in modern processors
- Uses multiple prediction mechanisms
- Mispredictions are rolled back
- Leave footprint in the microarchitectural state (e.g., cache)

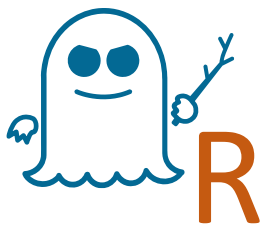


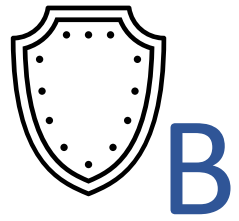
Different Versions of Spectre



Different Versions of Spectre

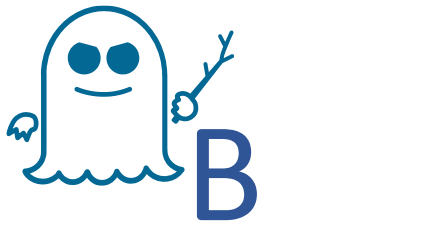


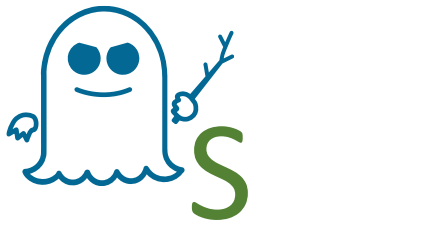


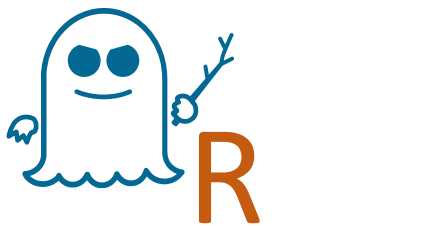


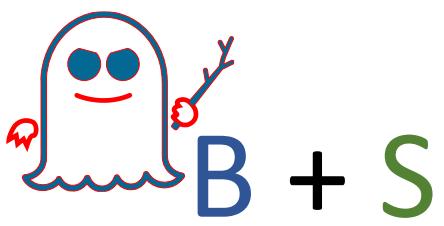
SoK: Practical Foundations for Software Spectre Defenses, Cauligi et al., IEEE S&P, 2022

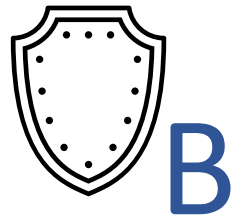
Different Versions of Spectre





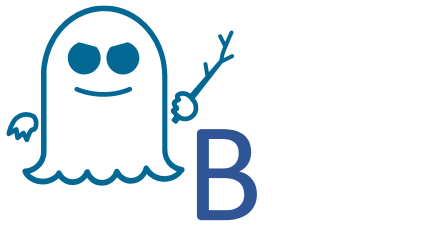


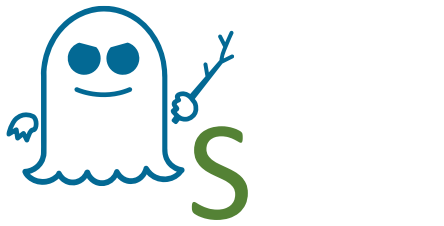


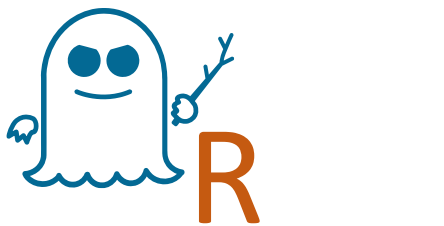


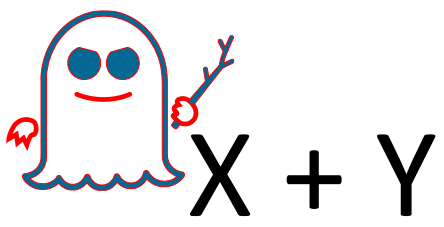
SoK: Practical Foundations for Software Spectre Defenses, Cauligi et al., IEEE S&P, 2022

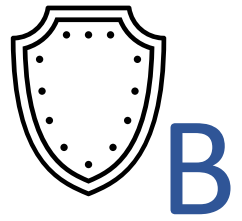
Different Versions of Spectre









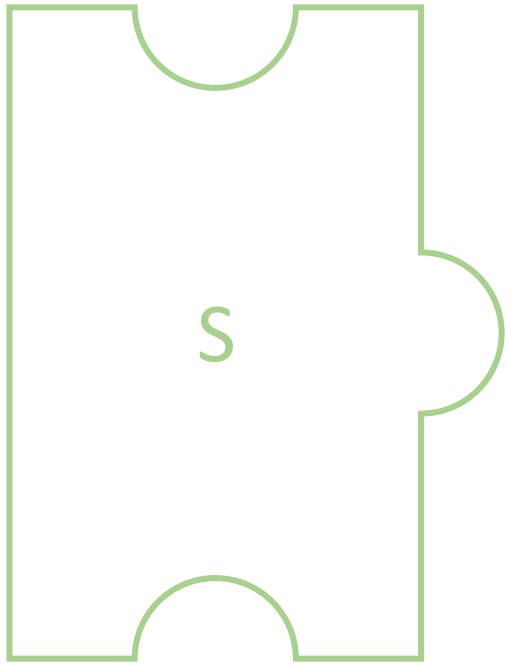


SoK: Practical Foundations for Software Spectre Defenses, Cauligi et al., IEEE S&P, 2022

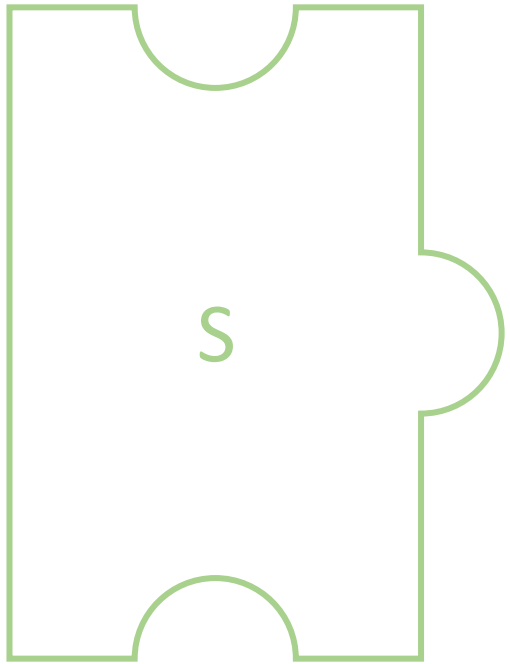
Detecting a Combination

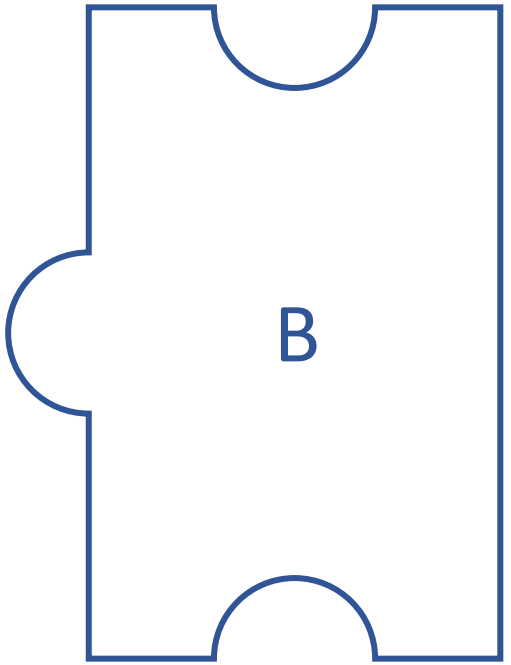
$$\frac{p(\sigma(\mathbf{pc})) \notin [\mathbf{store}\ x, e;\ \boxed{\mathrm{Z}}\ ] \qquad \sigma \xrightarrow{\tau} \sigma'}{\langle p, ctr, \sigma, n+1\rangle \overset{\tau}{\dashrightarrow}_{\mathrm{S}} \langle p, ctr, \sigma', n\rangle}\ (\mathrm{S}\text{:AM-NoSpec})$$

Detecting a Combination

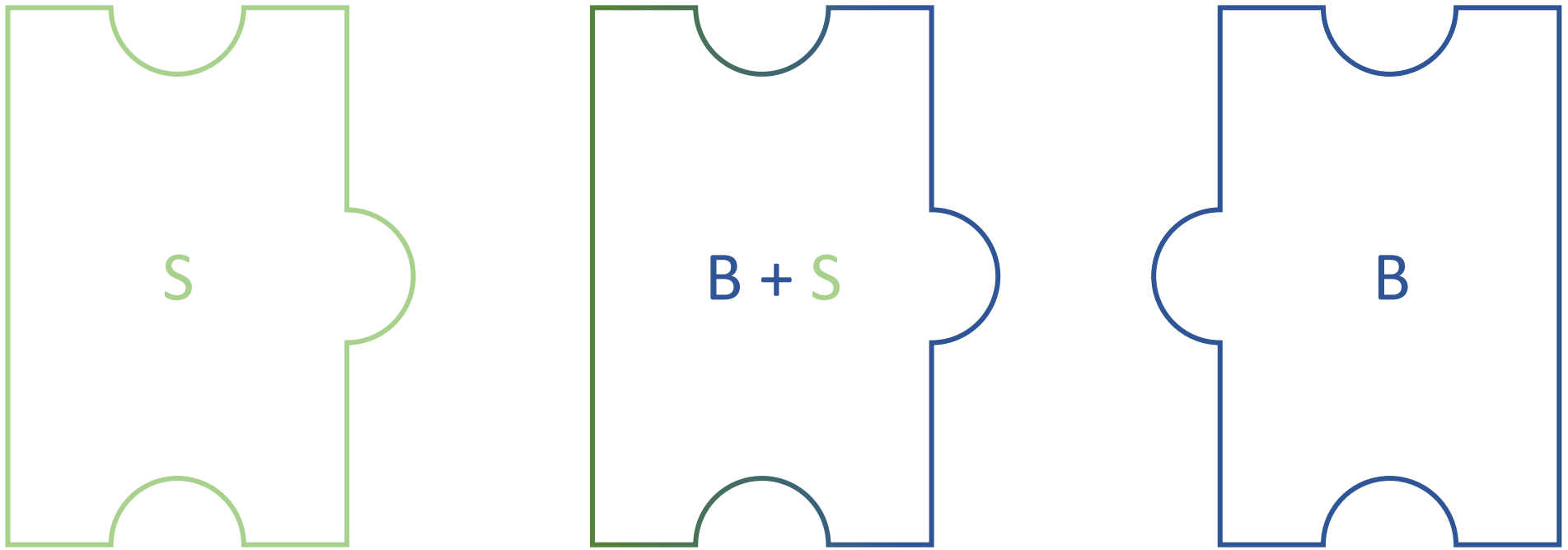


Detecting a Combination

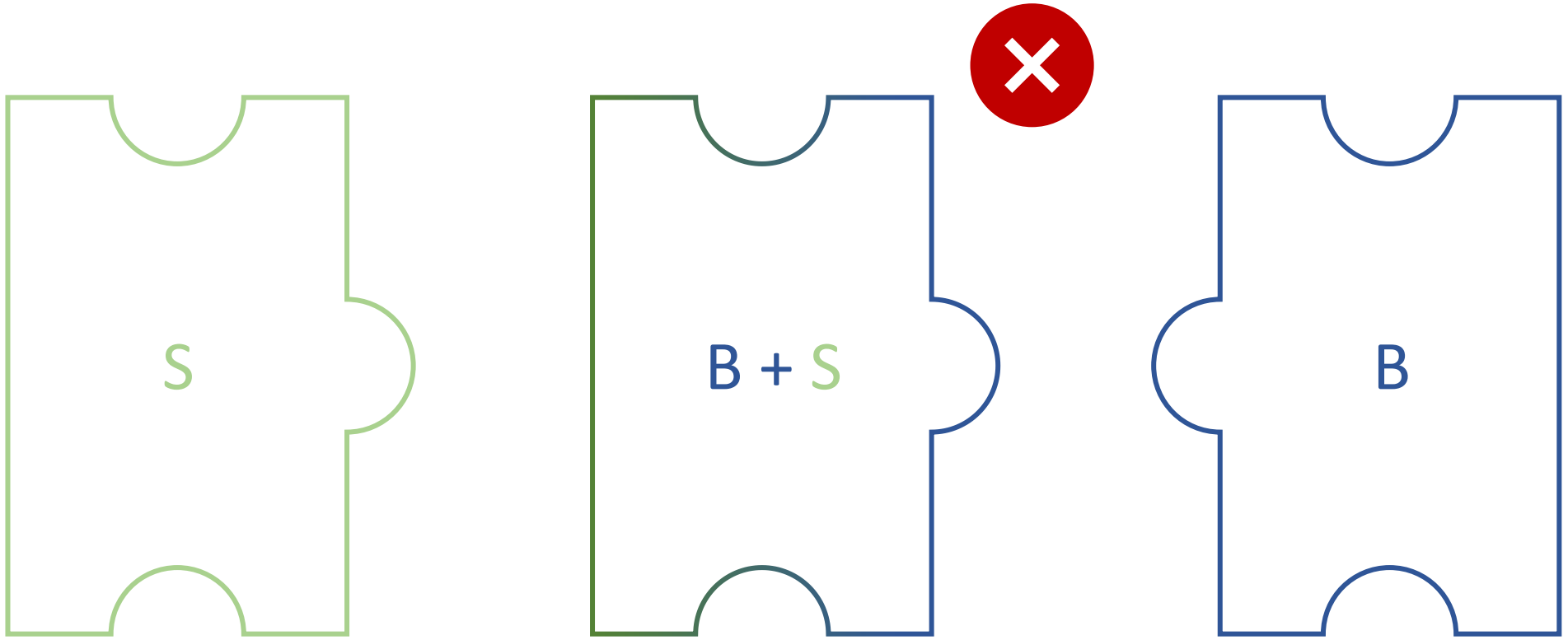




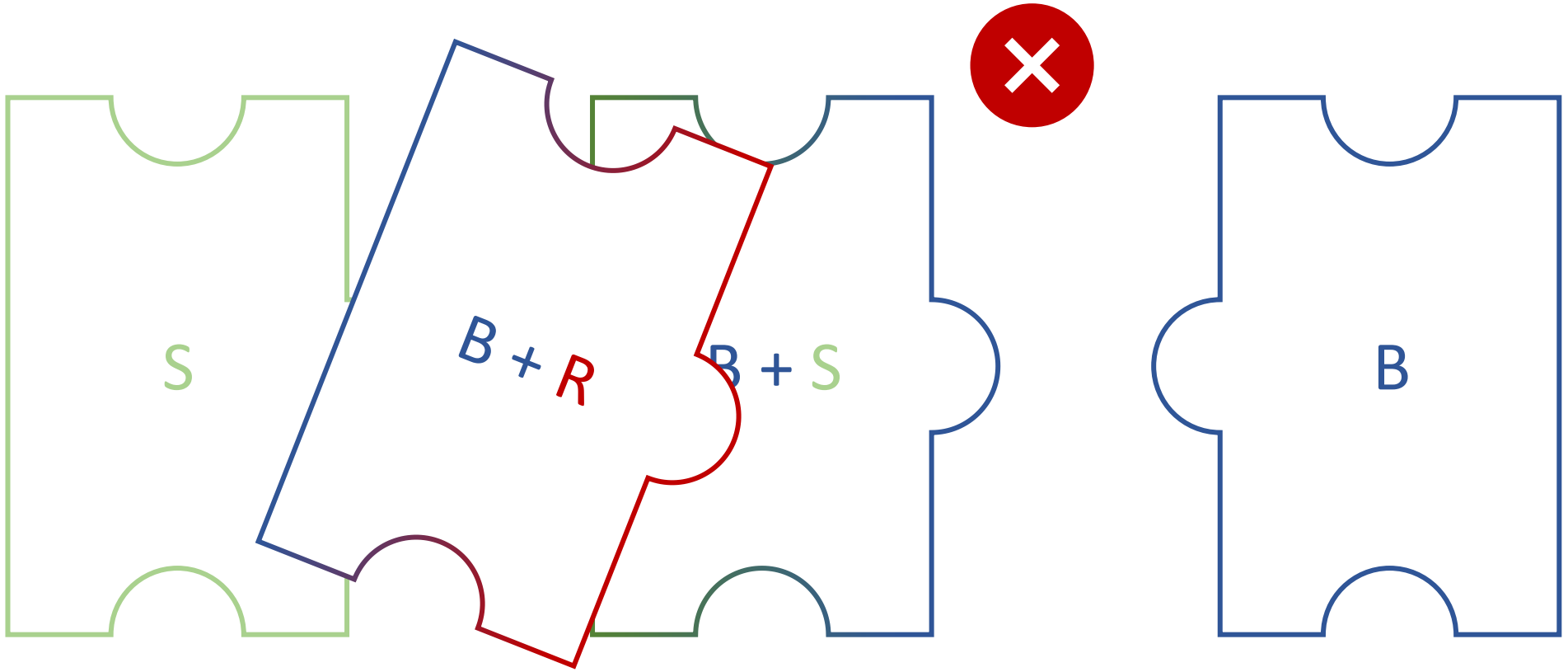
Detecting a Combination



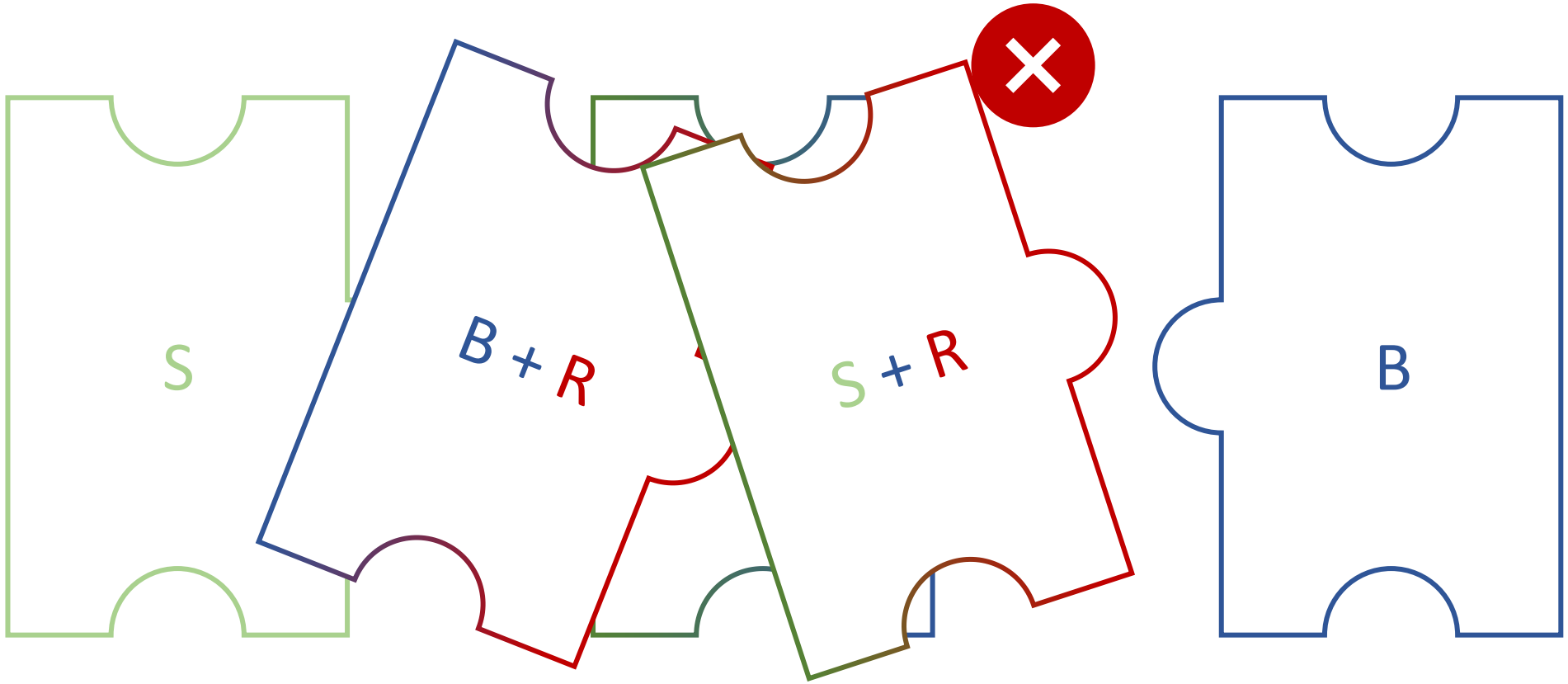
Detecting a Combination



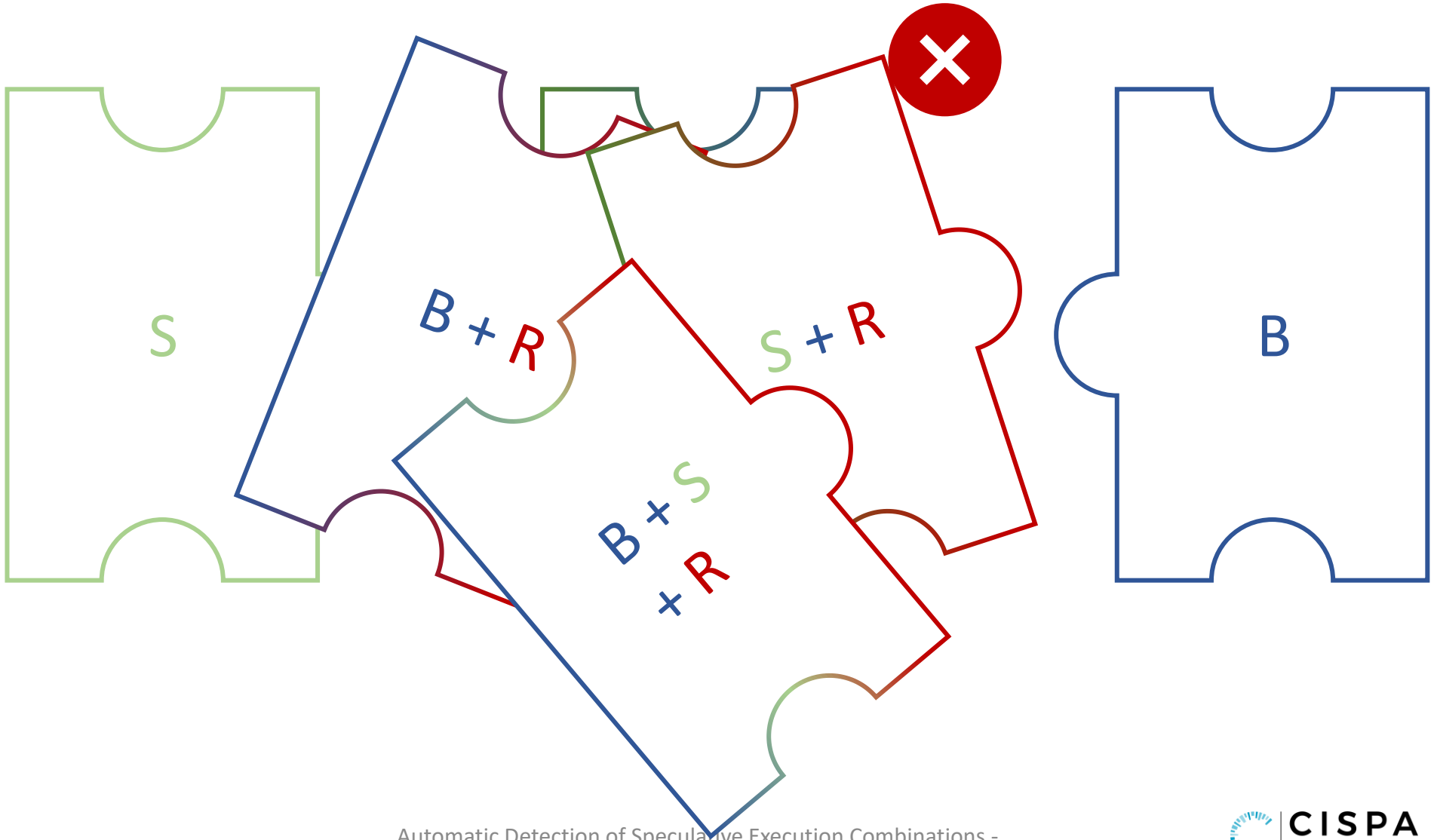
Detecting a Combination



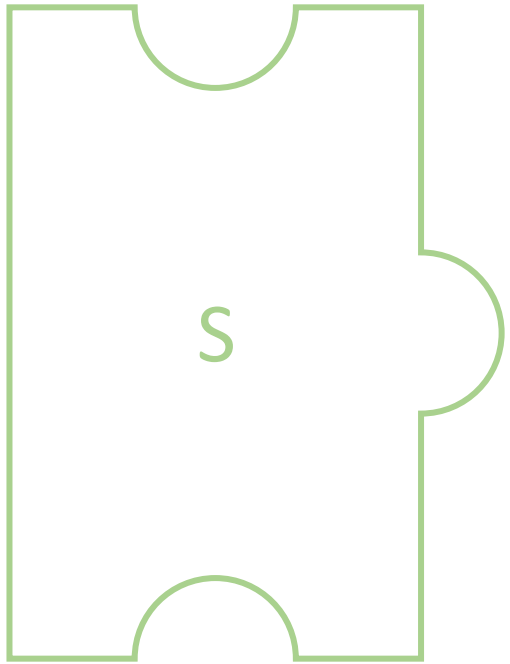
Detecting a Combination

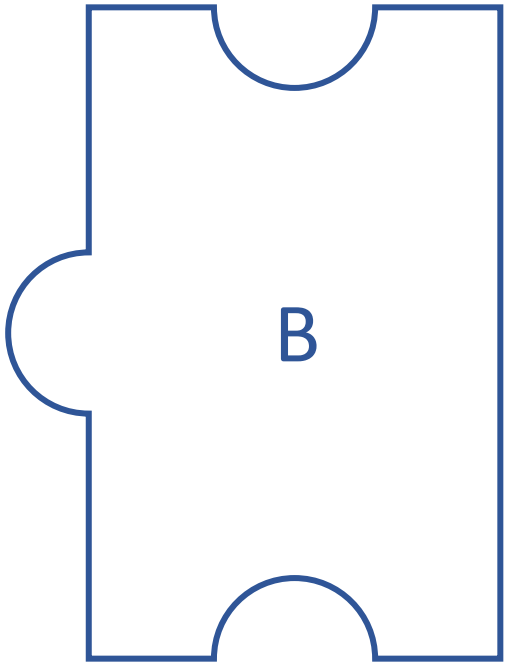


Detecting a Combination

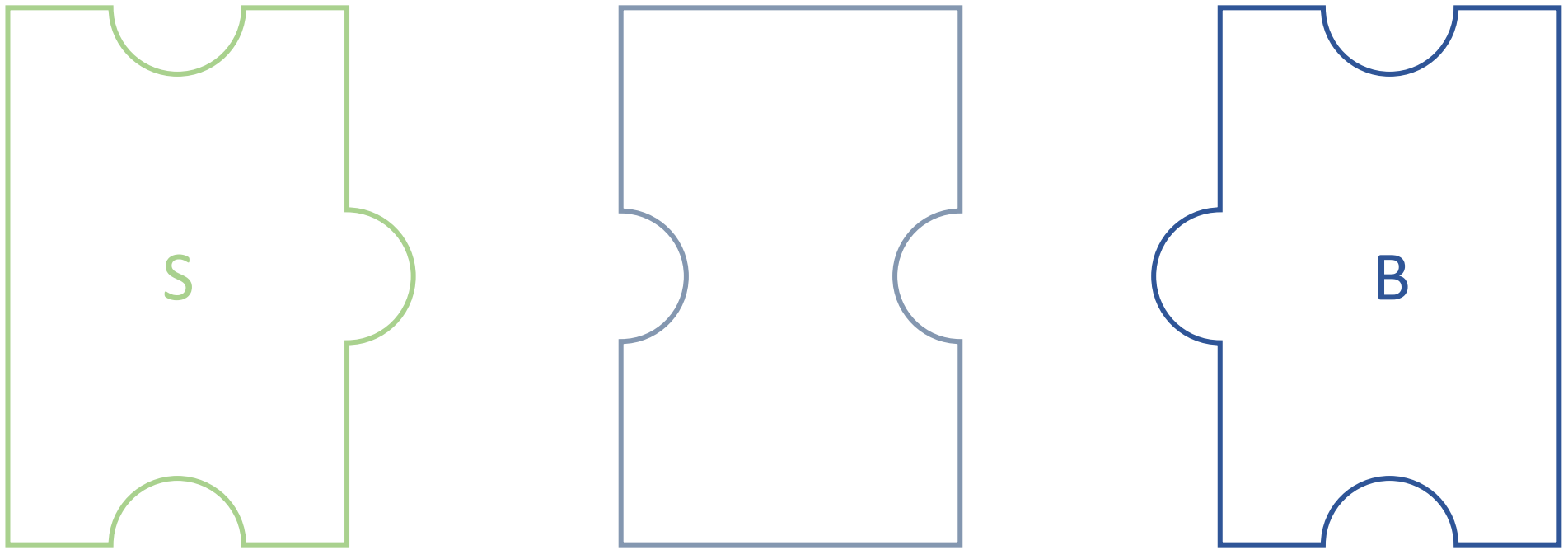


Detecting a Combination

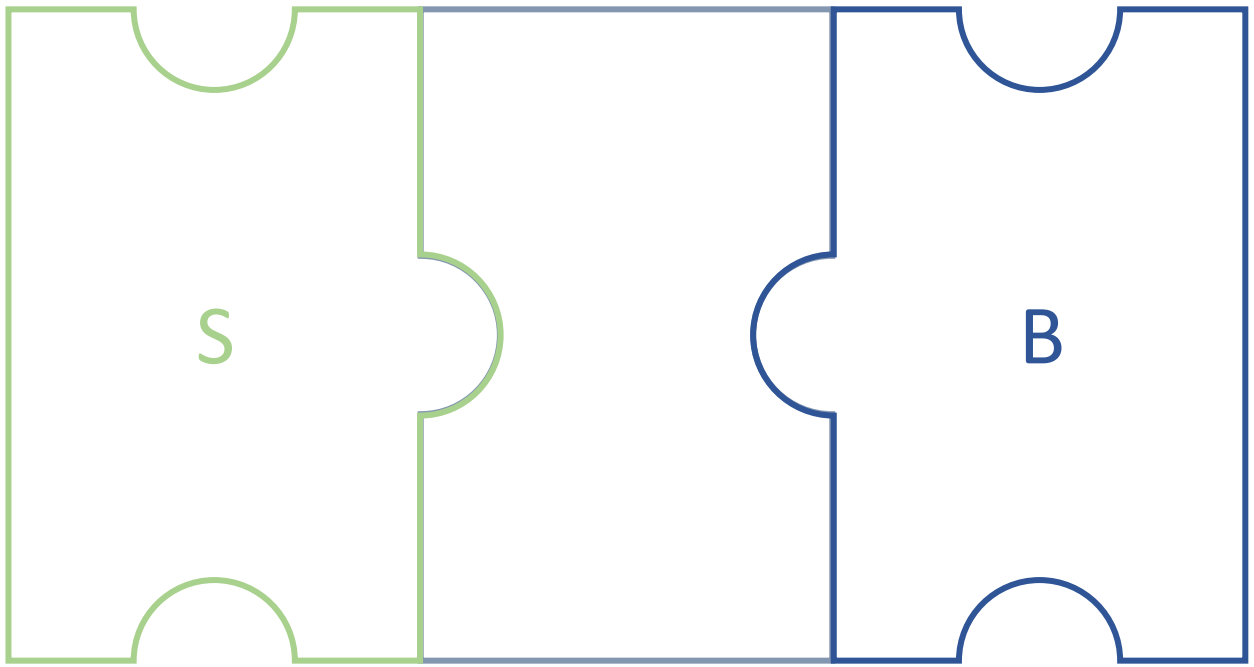




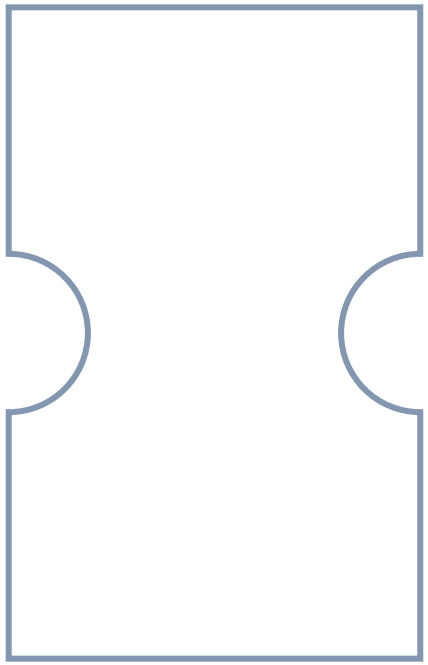
Detecting a Combination



Detecting a Combination



Detecting a Combination

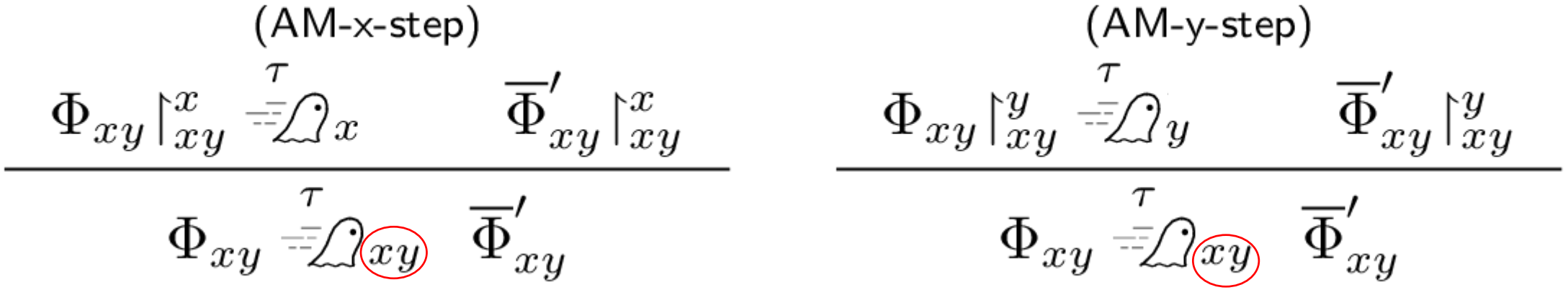


Detecting a Combination

$$\frac{\Phi_{xy}\restriction^{x}_{xy} \overset{\tau}{\dashrightarrow} x \qquad \overline{\Phi}'_{xy}\restriction^{x}_{xy}}{\Phi_{xy} \overset{\tau}{\dashrightarrow} xy \quad \overline{\Phi}'_{xy}}\ \text{(AM-x-step)}$$

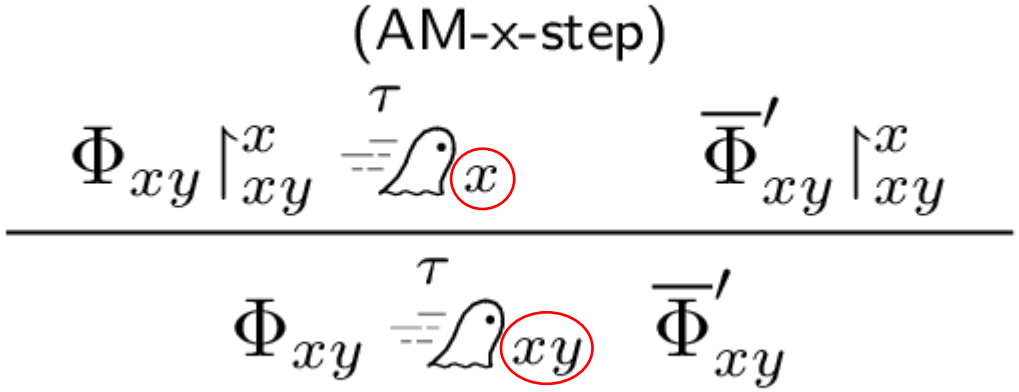
$$\frac{\Phi_{xy}\restriction^{y}_{xy} \overset{\tau}{\dashrightarrow} y \qquad \overline{\Phi}'_{xy}\restriction^{y}_{xy}}{\Phi_{xy} \overset{\tau}{\dashrightarrow} xy \quad \overline{\Phi}'_{xy}}\ \text{(AM-y-step)}$$

Detecting a Combination

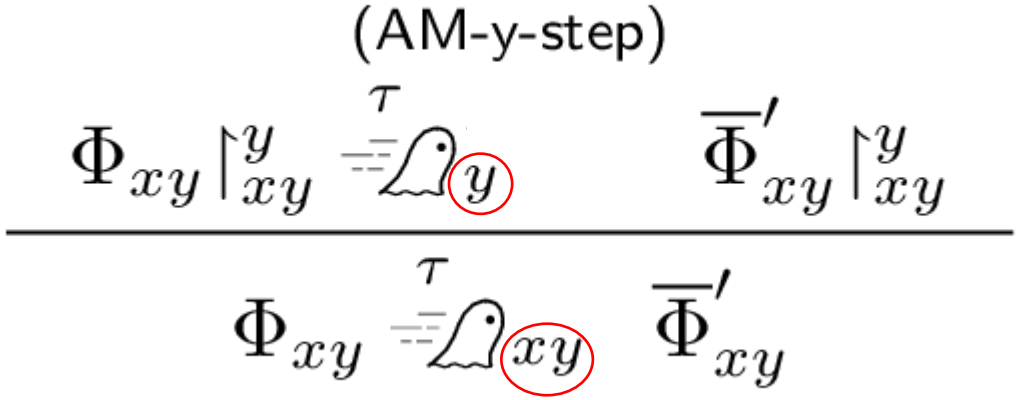


Detecting a Combination

(AM-x-step)



(AM-y-step)

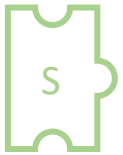


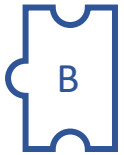
Well Formed Composition (WFC)

- Confluence
- Recovering the behaviour of the source semantics

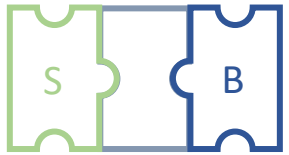
Well Formed Composition (WFC)

- Confluence
- Recovering the behaviour of the source semantics

- Secure (

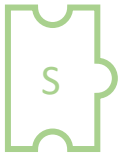
and 

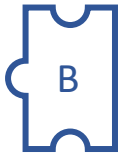
) and WFC (

) -> Secure (

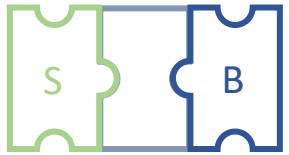
)

Well Formed Composition (WFC)

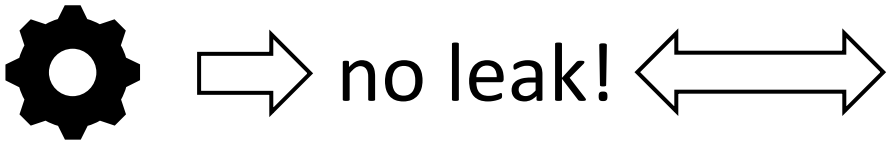
- Confluence
- Recovering the behaviour of the source semantics
- Secure (

and 

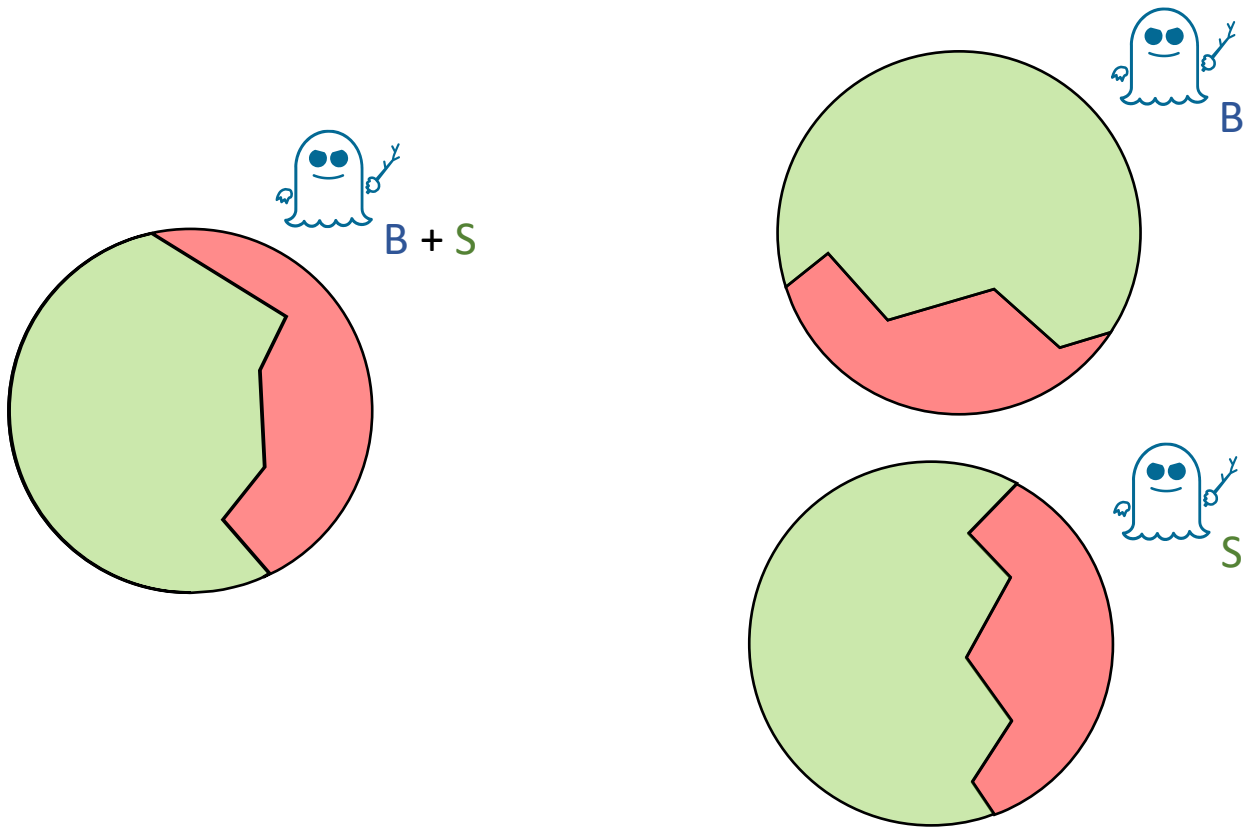
) and WFC (

) -> Secure (

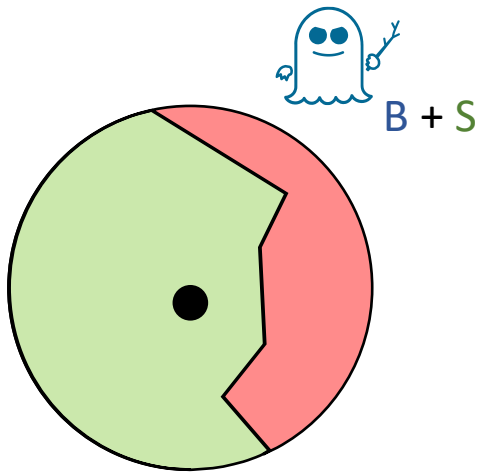
)
- Correctness of the Spectector algorithm

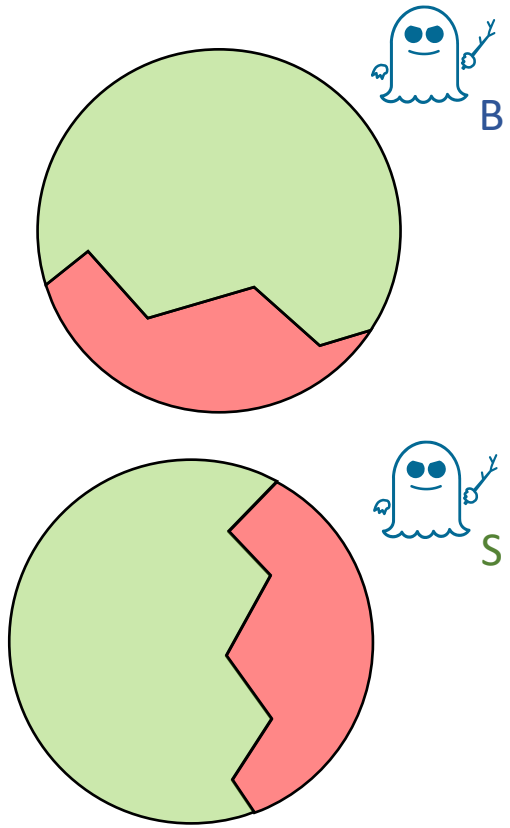


Relationship source and Composition

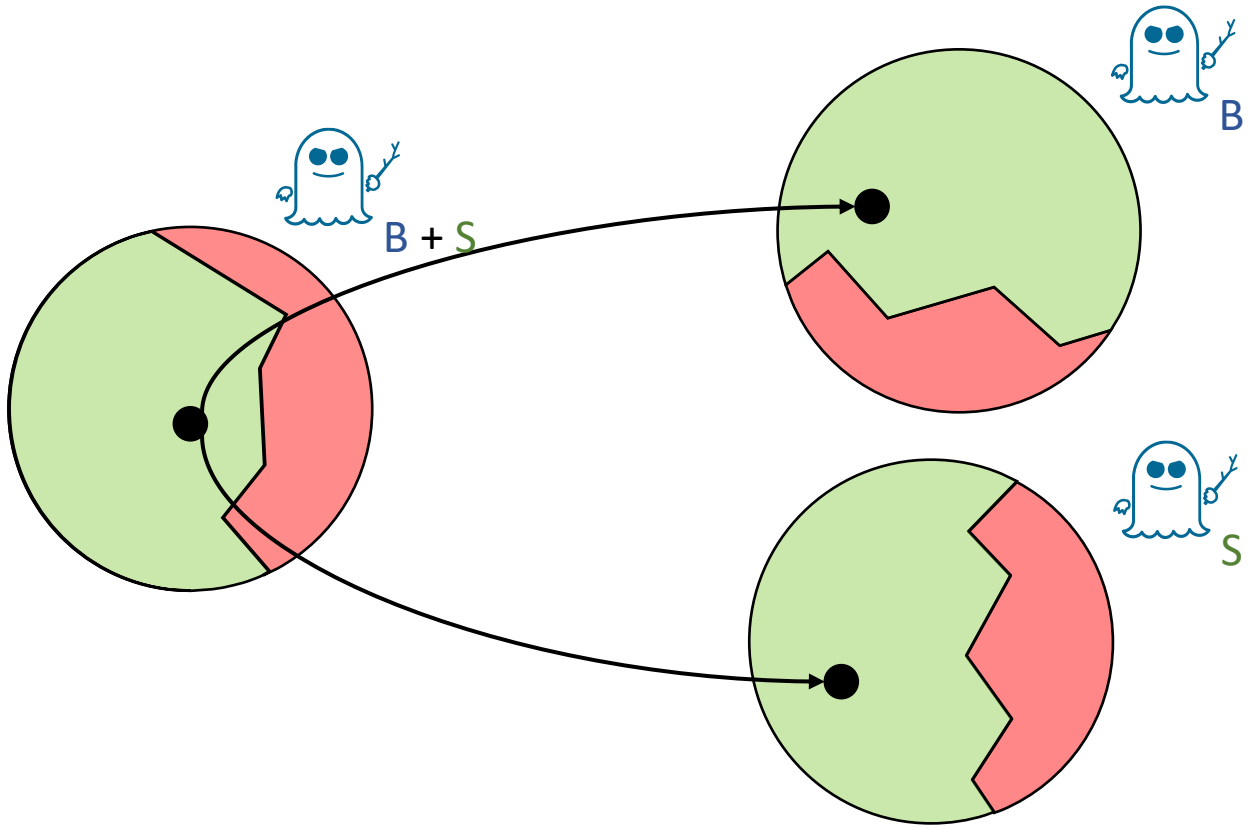


Relationship source and Composition

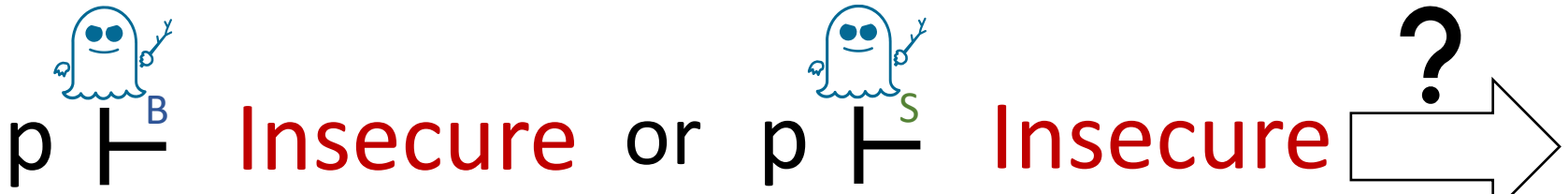


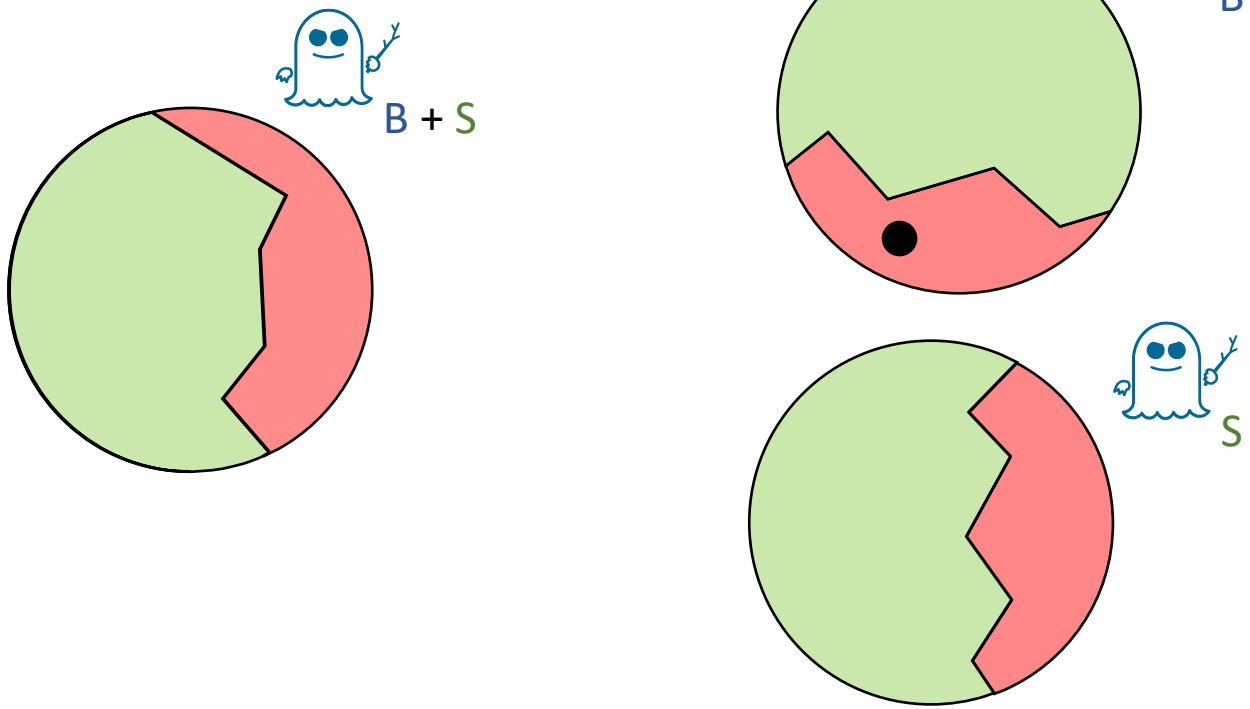


Relationship source and Composition

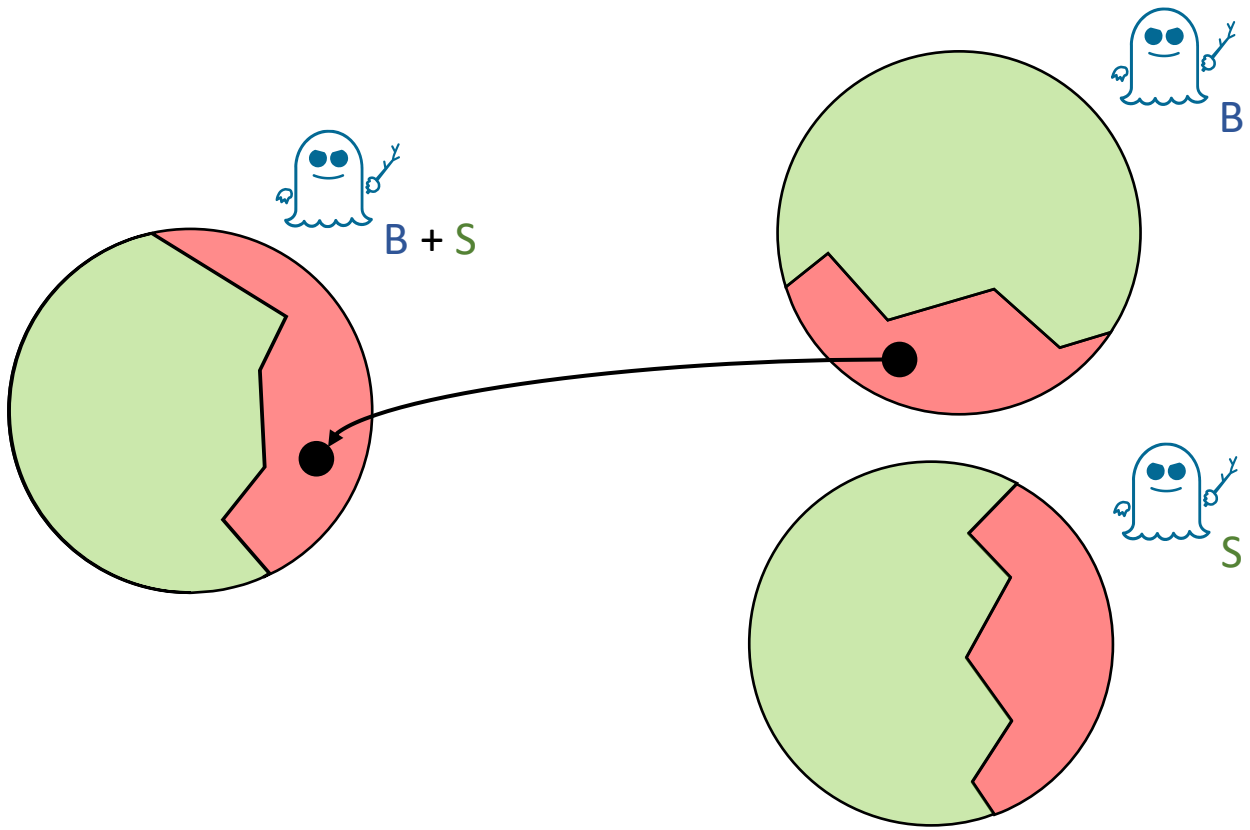


Relationship source and Composition

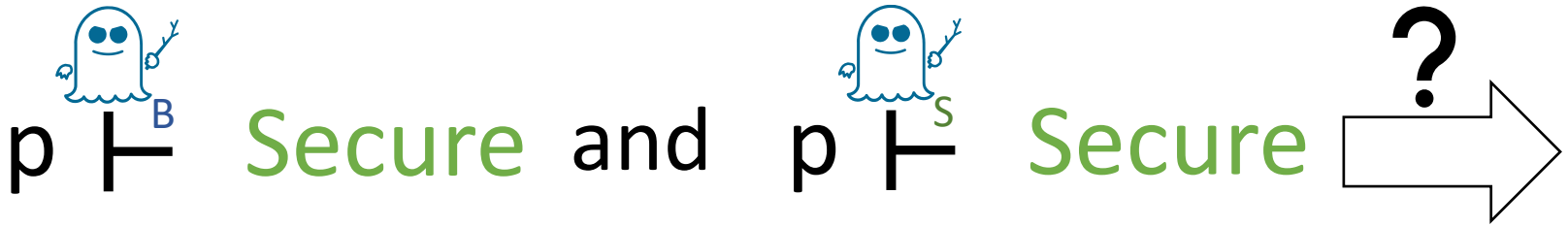


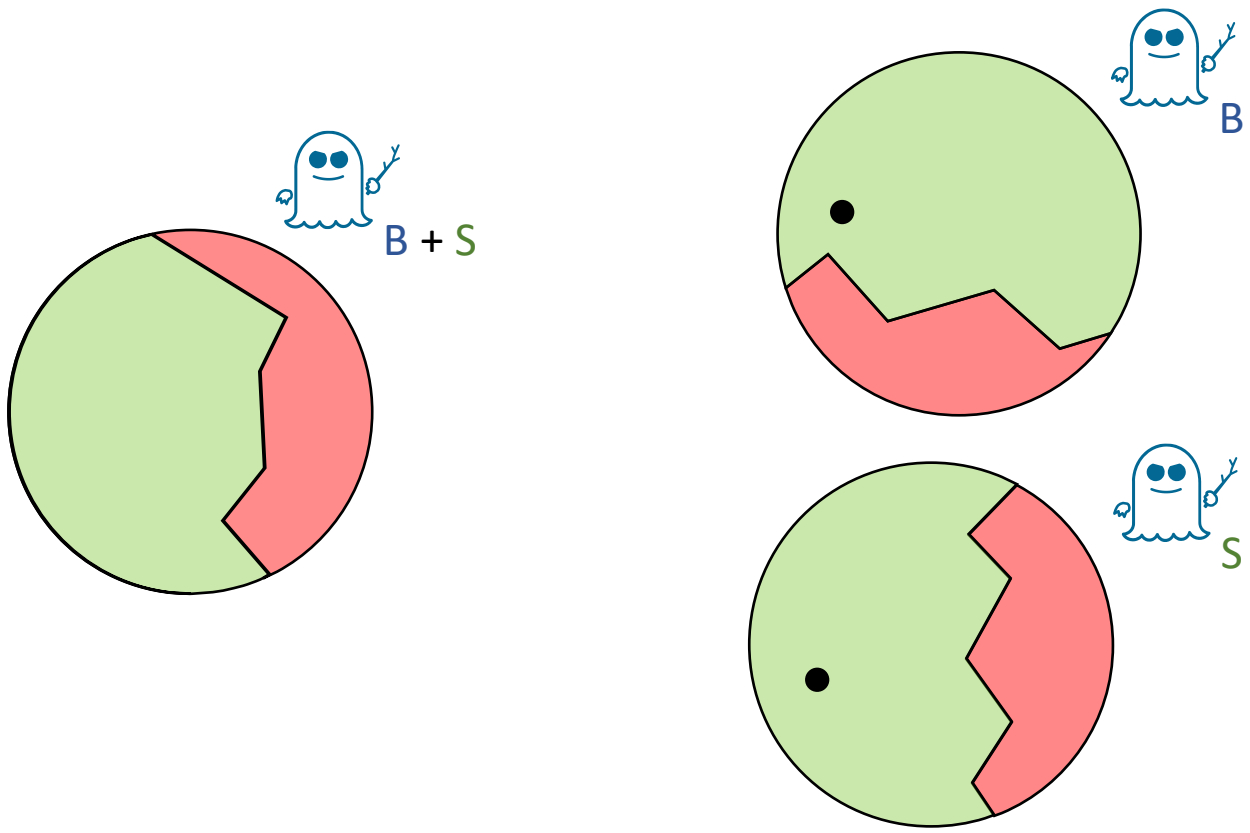


Relationship source and Composition



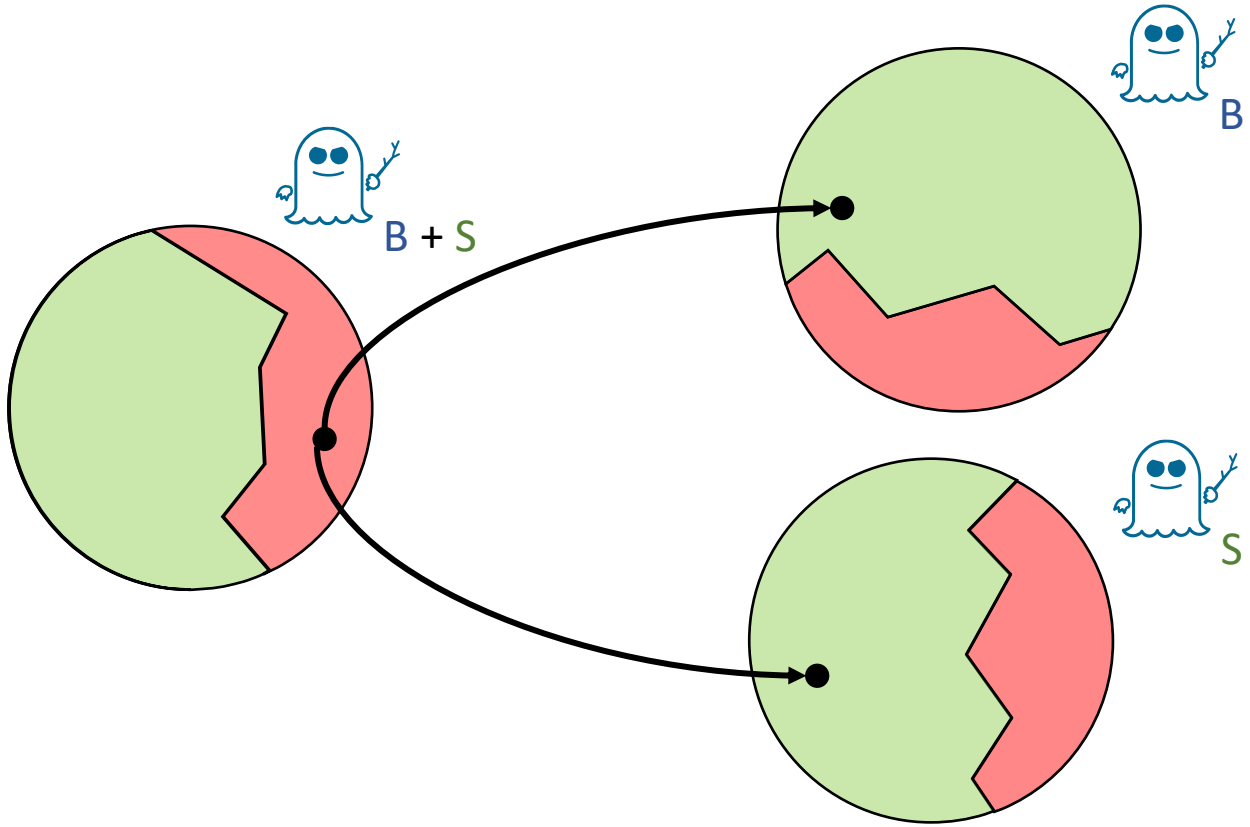
Relationship source and Composition





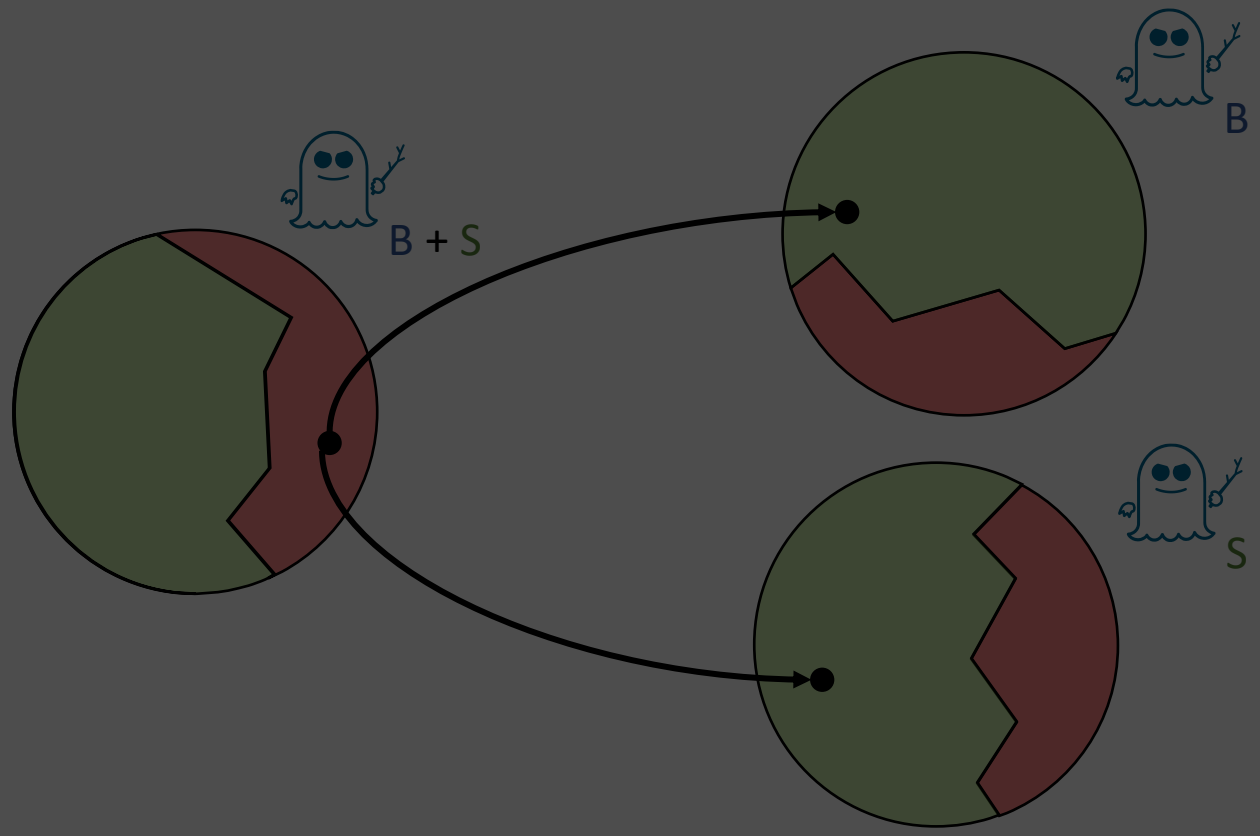
Relationship source and Composition

$p \vdash_{B}$ Secure and $p \vdash_{S}$ Secure $\not\Rightarrow$ $p \vdash_{B+S}$ Secure



Relationship source and Composition

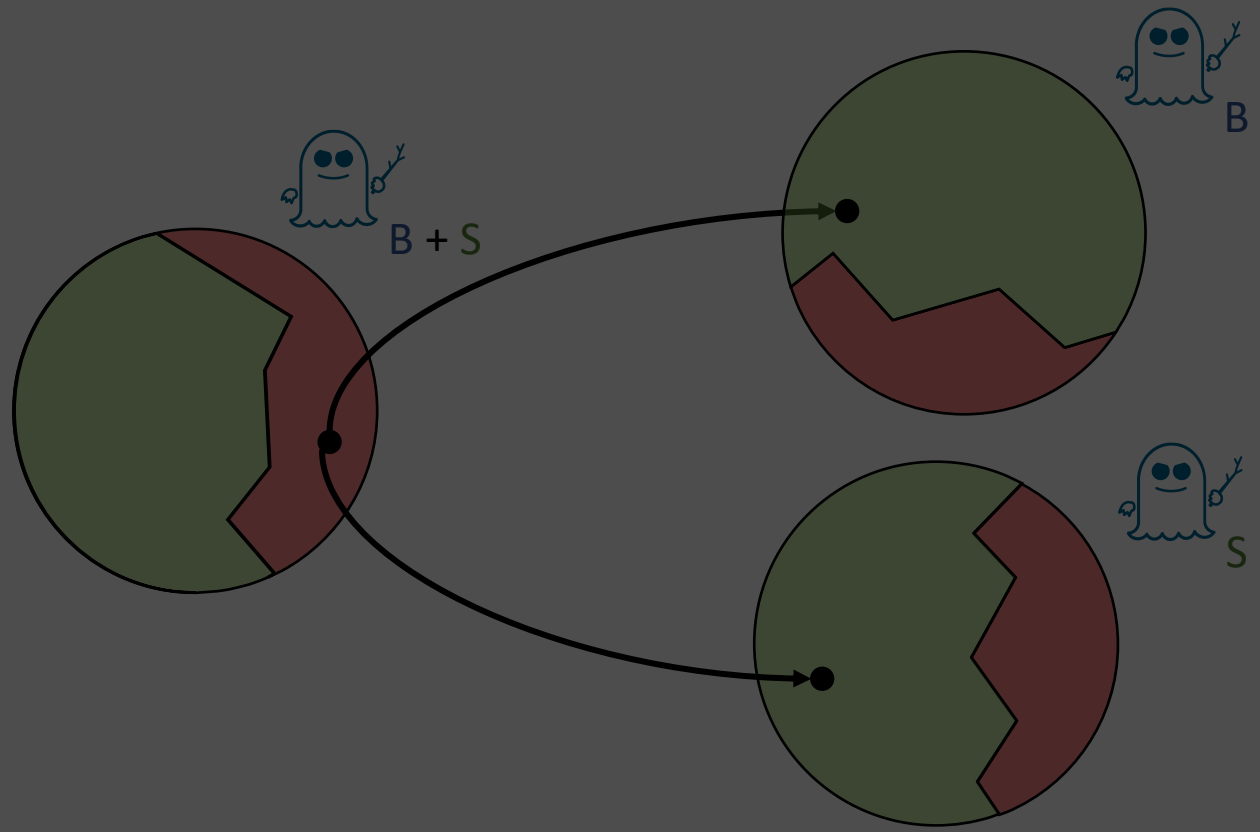
$p \vdash_{B}$ Secure and $p \vdash_{S}$ Secure $\not\Rightarrow$ $p \vdash_{B+S}$ Secure




```
1. x = 0
2. p = &secret
3. p = &public
4. if x != 0
5.     y = A[*p]
6.     z = B[y]
```

Relationship source and Composition

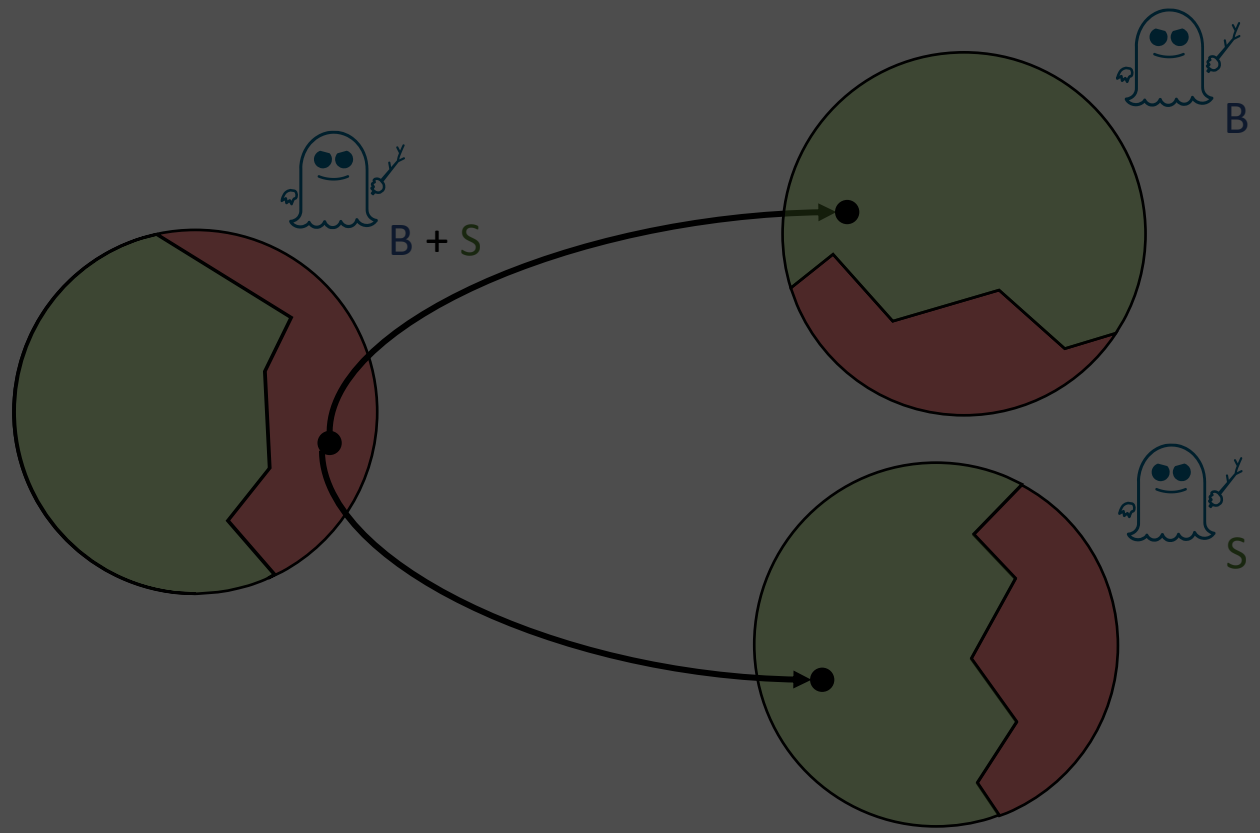
$p \vdash_{B}$ Secure and $p \vdash_{S}$ Secure $\not\Rightarrow$ $p \vdash_{B+S}$ Secure




```
1.  x = 0
2.  p = &secret
3.  p = &public
4.  if x != 0
5.      y = A[*p]
6.      z = B[y]
```

Relationship source and Composition

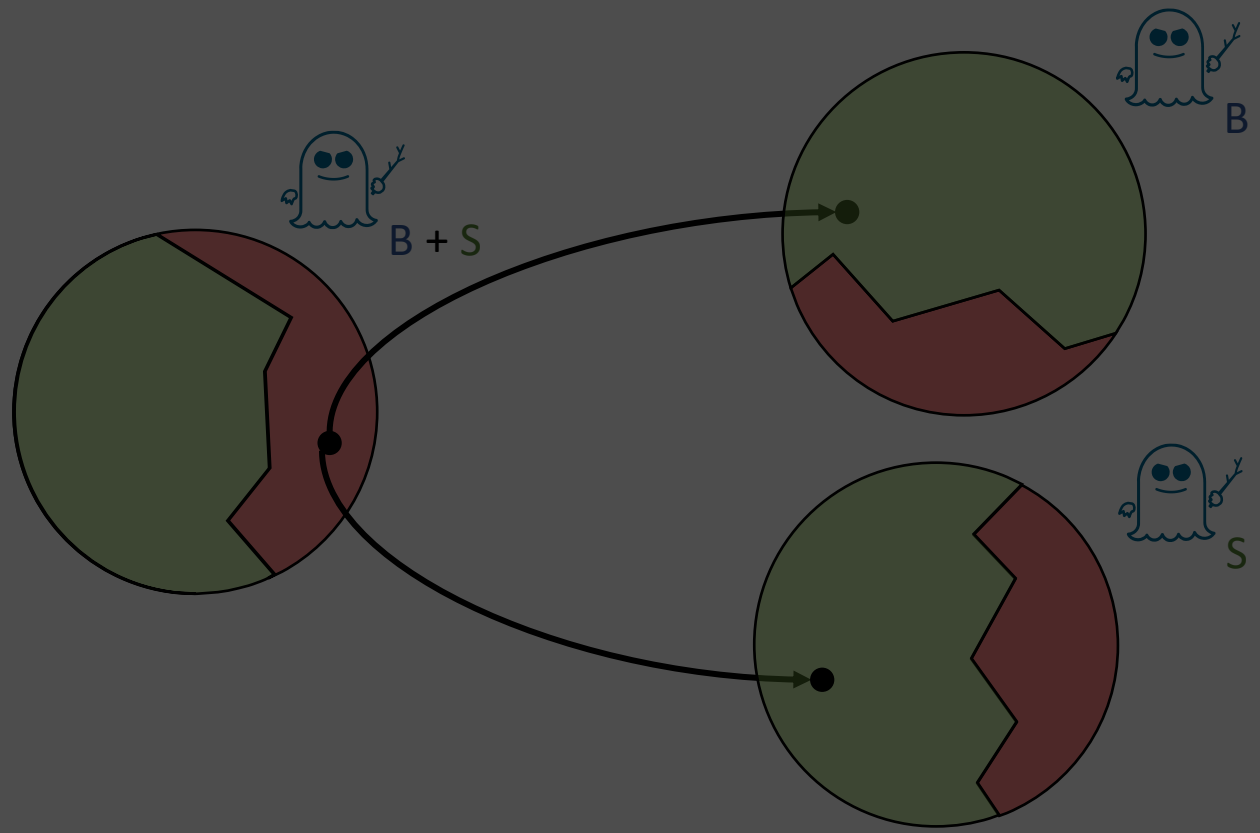
$p \vdash_{B}$ Secure and $p \vdash_{S}$ Secure $\not\Rightarrow$ $p \vdash_{B+S}$ Secure




```
1. x = 0
2. p = &secret
3. p = &public
4. if x != 0
5.     y = A[*p]
6.     z = B[y]
```

Relationship source and Composition

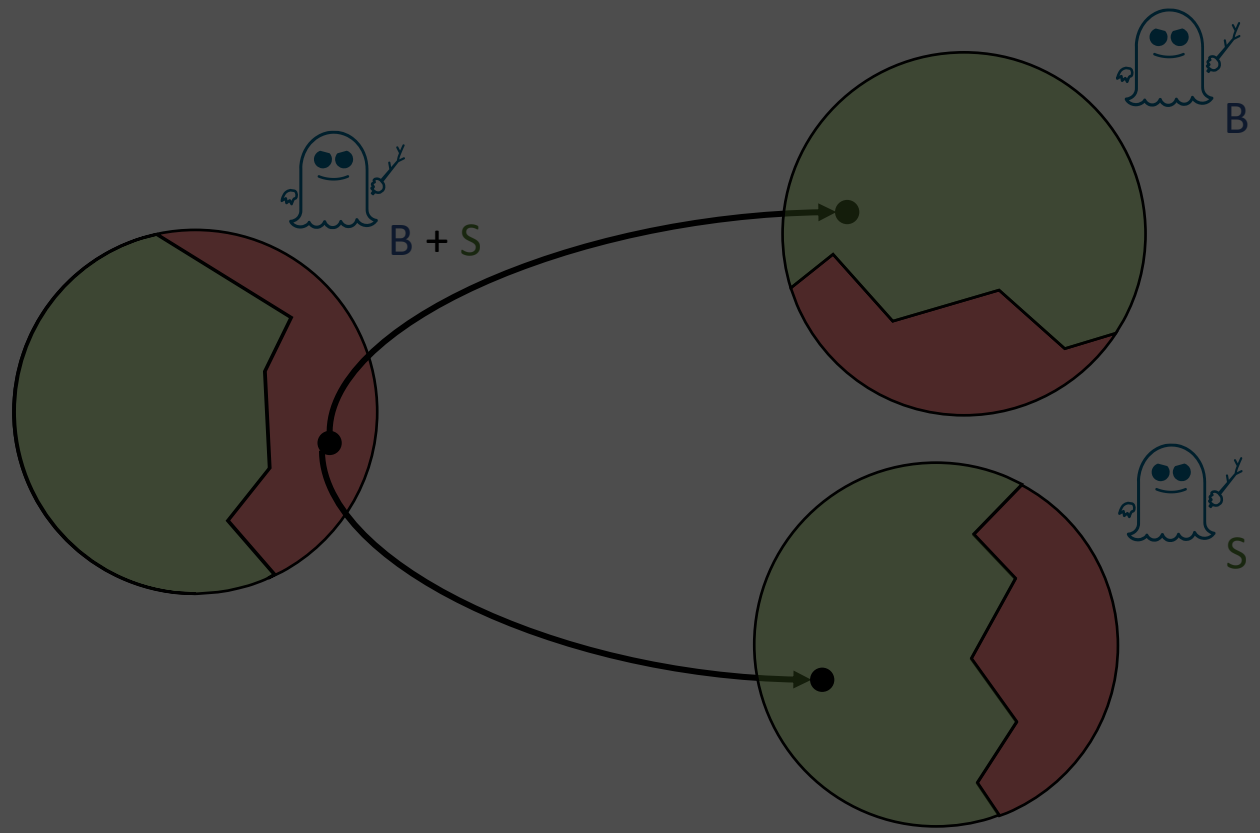
$p \vdash_{B}$ Secure and $p \vdash_{S}$ Secure $\not\Rightarrow$ $p \vdash_{B+S}$ Secure




```
1. x = 0
2. p = &secret
3. p = &public
4. if x != 0
5.     y = A[*p]
6.     z = B[y]
```

Relationship source and Composition

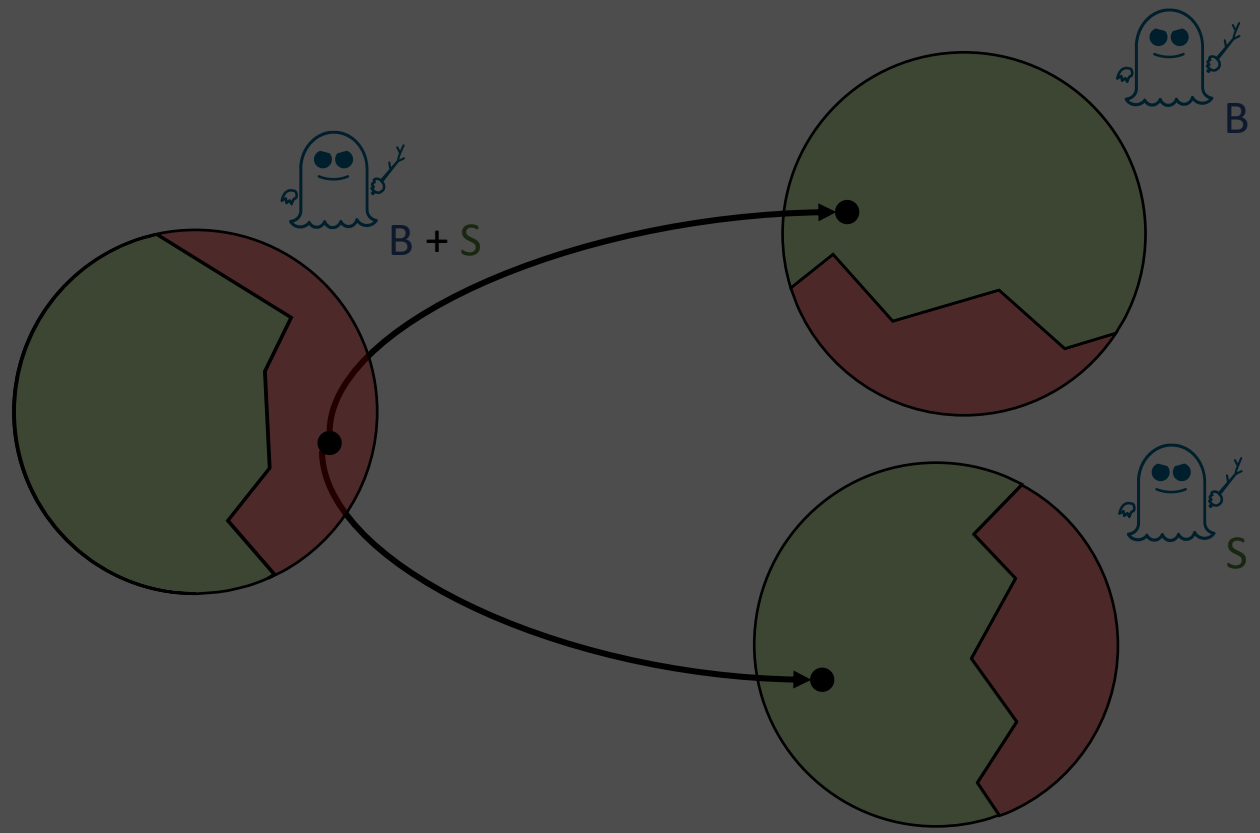
$p \vdash_{B}$ Secure and $p \vdash_{S}$ Secure $\not\Rightarrow$ $p \vdash_{B+S}$ Secure




```
1. x = 0
2. p = &secret
3. p = &public
4. if x != 0
5.     y = A[*p]
6.     z = B[y]
```

Relationship source and Composition

$p \vdash_{B}$ Secure and $p \vdash_{S}$ Secure $\not\Rightarrow$ $p \vdash_{B+S}$ Secure

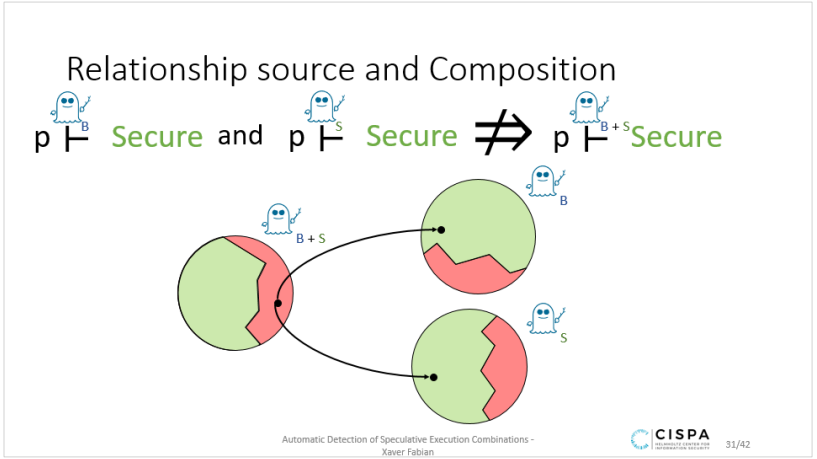



```
1. x = 0
2. p = &secret
3. p = &public
4. if x != 0
5.     y = A[*p]
6.     z = B[y]
```

Evaluation : Combinations

Test case		$\mathcal{B}$ [2]	$\mathcal{S}$	$\mathcal{R}$	$\mathcal{B}$+$\mathcal{S}$	$\mathcal{S}$+$\mathcal{R}$	$\mathcal{B}$+$\mathcal{R}$	$\mathcal{B}$+$\mathcal{S}$+$\mathcal{R}$
listing 1	(-)	●	●	●	○	●	●	○
listing 5	(-)	●	●	●	●	○	●	○
listing 4	(-)	●	●	●	●	●	○	○
listing 6	(-)	●	●	●	●	●	●	○
listing 1 Fence	(+)	●	●	●	●	●	●	●
listing 5 Fence	(+)	●	●	●	●	●	●	●
listing 4 Fence	(+)	●	●	●	●	●	●	●
listing 6 Fence	(+)	●	●	●	●	●	●	●

(c) Results for the Spectre-Comb programs where "listing x Fence" denotes the patched version (using `lfence`) of "listing x"



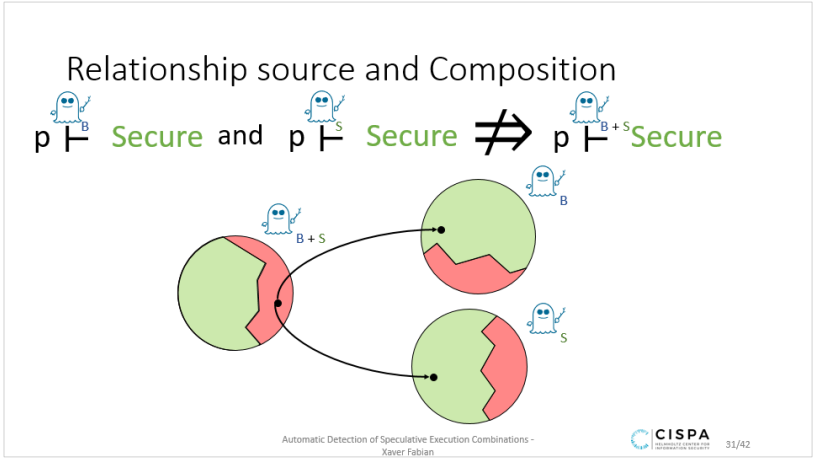
- ● : **Secure** under the resp. Semantics
- ○ : **Insecure** under the resp. Semantics

[2] Spectector, Guarnieri et al., IEEE S&P, 2020

Evaluation : Combinations

Test case		[2] B	S	R	$B+S$	$S+R$	$B+R$	$B+S+R$
listing 1	(-)	●	●	●	○	●	●	○
listing 5	(-)	●	●	●	●	○	●	○
listing 4	(-)	●	●	●	●	●	○	○
listing 6	(-)	●	●	●	●	●	●	○
listing 1 Fence	(+)	●	●	●	●	●	●	●
listing 5 Fence	(+)	●	●	●	●	●	●	●
listing 4 Fence	(+)	●	●	●	●	●	●	●
listing 6 Fence	(+)	●	●	●	●	●	●	●

(c) Results for the Spectre-Comb programs where "listing x Fence" denotes the patched version (using `lfence`) of "listing x"



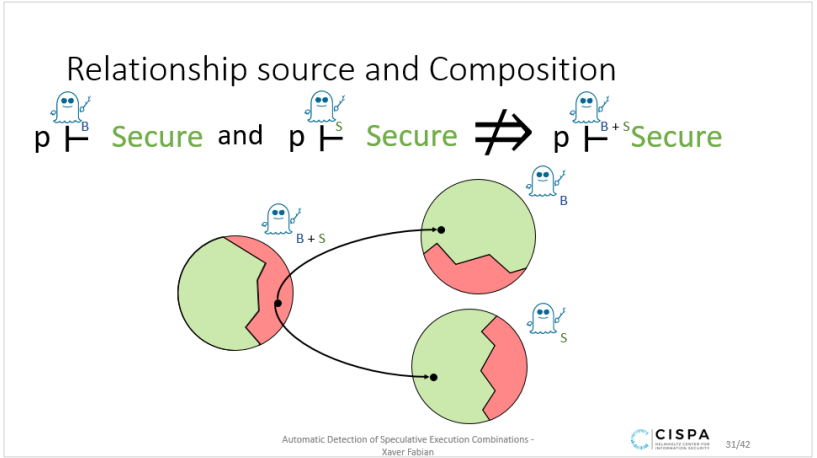
- ● : **Secure** under the resp. Semantics
- ○ : **Insecure** under the resp. Semantics

[2] Spectector, Guarnieri et al., IEEE S&P, 2020

Evaluation : Combinations

Test case		B [2]	S	R	B+S	S+R	B+R	B+S+R
listing 1	(-)	●	●	●	○	●	●	○
listing 5	(-)	●	●	●	●	○	●	○
listing 4	(-)	●	●	●	●	●	○	○
listing 6	(-)	●	●	●	●	●	●	○
listing 1 Fence	(+)	●	●	●	●	●	●	●
listing 5 Fence	(+)	●	●	●	●	●	●	●
listing 4 Fence	(+)	●	●	●	●	●	●	●
listing 6 Fence	(+)	●	●	●	●	●	●	●

(c) Results for the Spectre-Comb programs where "listing x Fence" denotes the patched version (using `lfence`) of "listing x"



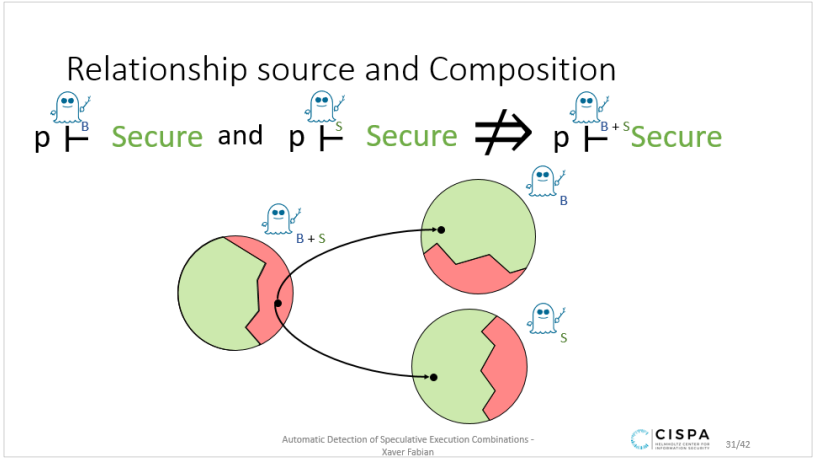
- ● : **Secure** under the resp. Semantics
- ○ : **Insecure** under the resp. Semantics

[2] Spectector, Guarnieri et al., IEEE S&P, 2020

Evaluation : Combinations

Test case		B [2]	S	R	$B+S$	$S+R$	$B+R$	$B+S+R$
listing 1	(-)	●	●	●	○	●	●	○
listing 5	(-)	●	●	●	●	○	●	○
listing 4	(-)	●	●	●	●	●	○	○
listing 6	(-)	●	●	●	●	●	●	○
listing 1 Fence	(+)	●	●	●	●	●	●	●
listing 5 Fence	(+)	●	●	●	●	●	●	●
listing 4 Fence	(+)	●	●	●	●	●	●	●
listing 6 Fence	(+)	●	●	●	●	●	●	●

(c) Results for the Spectre-Comb programs where "listing x Fence" denotes the patched version (using `lfence`) of "listing x"

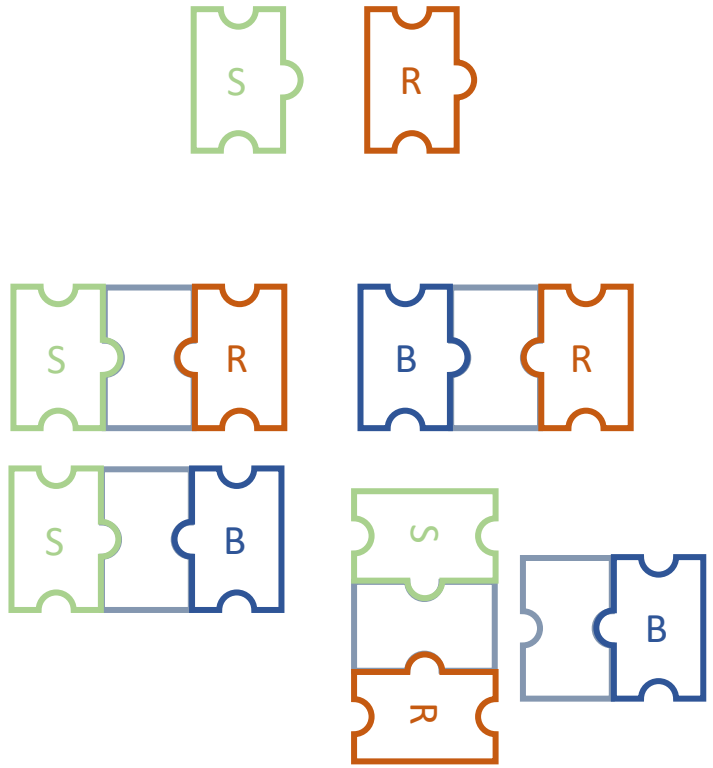


- ● : **Secure** under the resp. Semantics
- ○ : **Insecure** under the resp. Semantics

[2] Spectector, Guarnieri et al., IEEE S&P, 2020

In the paper you will find…

- Source Semantics
- A general framework for combinations
- => Derivable Lemmas from framework
- Created all combinations
- Implementation as extension of Spectector
- Evaluation



Spectre Examples


```
1. if idx < A_size
2.     y = A[idx]
3.     z = B[y]
4. end
```


```
1. p = &secret
2. p = &public
3. temp = B[*p]
```

What about this one?

```
1. x = 0
2. p = &secret
3. p = &public
4. if x != 0
5.     y = A[*p]
6.     z = B[y]
7. end
```

Evaluation

Test case		λ_S None	λ_S Fence
case01	(-)	○	●
case02	(-)	○	●
case03	(+)	●	●
case04	(-)	○	●
case05	(-)	○	●
case06	(-)	○	●
case07	(-)	○	●
case08	(-)	○	●
case09	(+)	●	●
case10	(-)	○	●
case11	(-)	○	●
case12	(+)	●	●
case13	(-)	○	●

(a) Results for the Spectre-STL programs under the λ_S semantics against unpatched programs (column "None") and programs patched with `lfence` (column "Fence")

Test case		λ_R None	λ_R Fence	λ_R Retpoline
ret2spec_c_d	(-)	○	●	●
ca_ip	(-)	○	●	●
ca_oop	(-)	○	●	●
sa_ip	(-)	○	●	●
sa_oop	(-)	○	●	●

(b) Results for the Spectre-RSB programs under the λ_R semantics against unpatched programs (column "None"), programs patched with `lfence` (column "Fence"), and programs patched with retpoline (column "Retpoline")

● : **Secure** under the resp. Semantics

○ : **Insecure** under the resp. Semantics

Evaluation

Test case		λ_S None	λ_S Fence
case01	(-)	○	●
case02	(-)	○	●
case03	(+)	●	●
case04	(-)	○	●
case05	(-)	○	●
case06	(-)	○	●
case07	(-)	○	●
case08	(-)	○	●
case09	(+)	●	●
case10	(-)	○	●
case11	(-)	○	●
case12	(+)	●	●
case13	(-)	○	●

(a) Results for the Spectre-STL programs under the λ_S semantics against unpatched programs (column "None") and programs patched with `lfence` (column "Fence")

Test case		λ_R None	λ_R Fence	λ_R Retpoline
ret2spec_c_d	(-)	○	●	●
ca_ip	(-)	○	●	●
ca_oop	(-)	○	●	●
sa_ip	(-)	○	●	●
sa_oop	(-)	○	●	●

(b) Results for the Spectre-RSB programs under the λ_R semantics against unpatched programs (column "None"), programs patched with `lfence` (column "Fence"), and programs patched with retpoline (column "Retpoline")

● : **Secure** under the resp. Semantics

○ : **Insecure** under the resp. Semantics

Stronger than sum of its parts