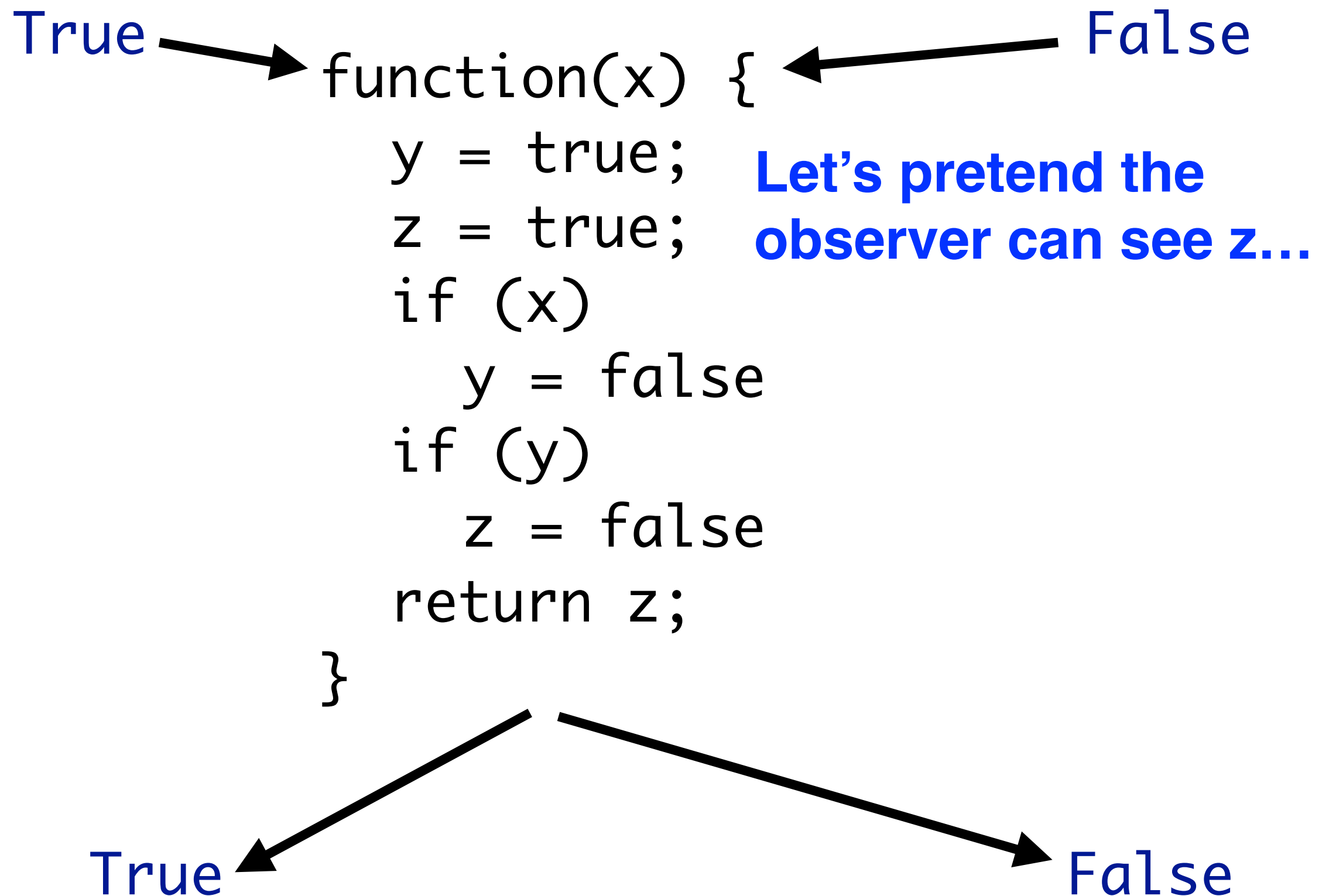


Symbolic Faceted Execution

(possibly with...) Kris Micinski

What does the following function compute?



Should be identity, but apply it to a private variable...

Do you have prior medical conditions???



```
Thread 1:  
function(x) {  
    y = true;  
    z = true;  
    if (x)  
        y = false  
    if (y)  
        z = false  
    return z;  
}
```

```
Thread 2:  
send(insurer, z);
```



Ways to tame this

- For a “public” observer:
 - Pretend that the input x was NULL (\perp)
 - Run the program and propagate \perp
- For a “private” observer, run program in separate addr space
- Run the program twice
 - Give public view to public (when sending network data)
 - Keep the private view inside

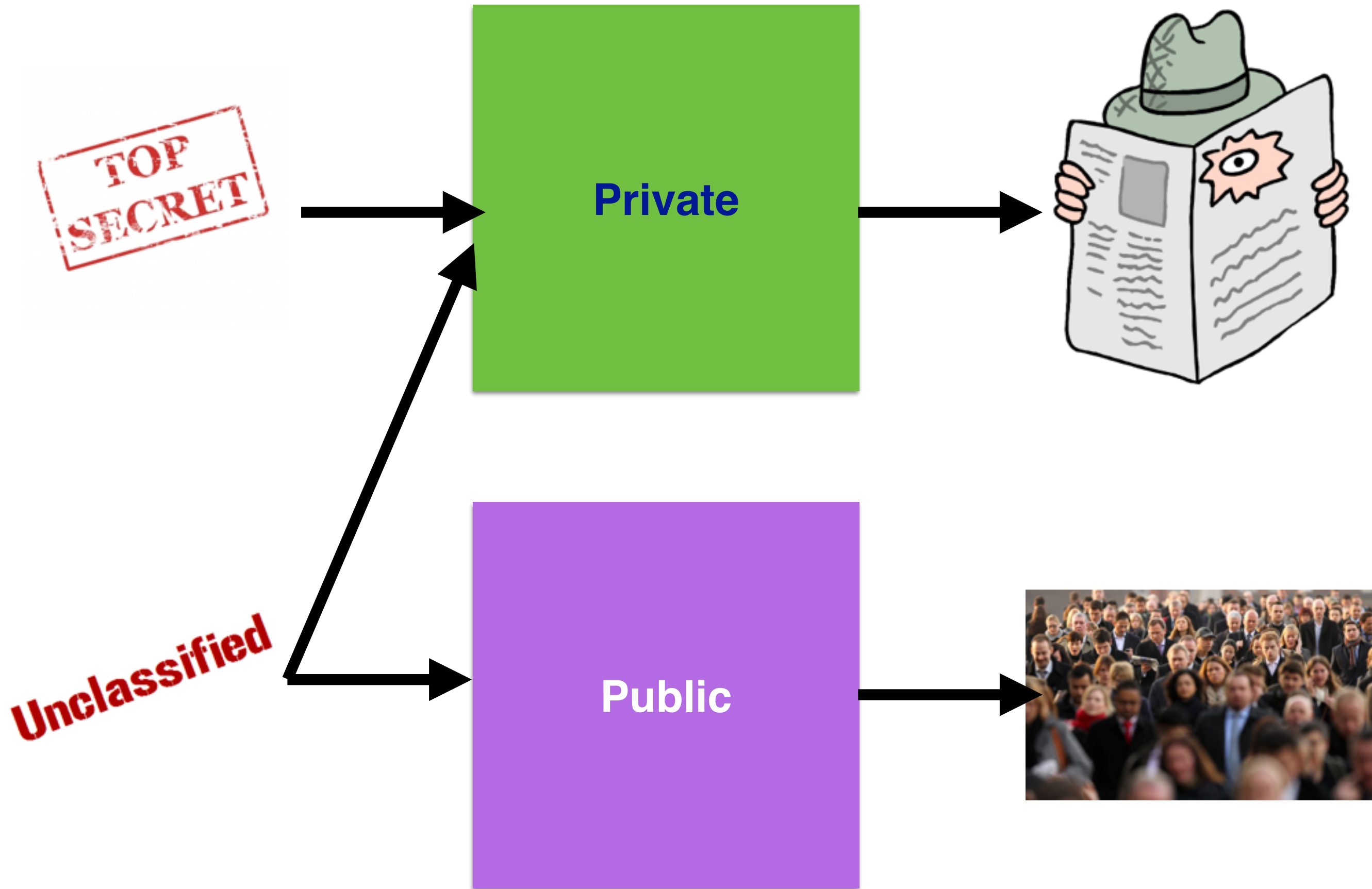
Running with \perp

Treat \perp as false

```
function( $\perp$ ) {  
    y = true;  
    z = true;  
    if ( $\perp$ )  
        y = false  
    if (y)  
        z = false ☒  
    return z;  
}
```

I get **False** no matter what...

False



Public sees computation as if it had no secret input

Multiple principles

Code from lots of different people
running in same browser (mashups)

The image is a screenshot of a Facebook news feed from 2013. It illustrates the concept of multiple principles running in the same browser through several annotations:

- Ad:** A large red "Ad" label with an arrow pointing to a sponsored post for cloudERP, which features a tablet displaying a PowerPoint slide titled "POWERPOINT".
- News Feed:** A red box labeled "News Feed" encompasses the right side of the interface, including several advertisements: "Your New Career" from adexchanger.com, "RACKSPACE" (Build Scalable Websites and Applications In The Cloud), "Adblade Ads" (Be a marketing hero and use Adblade ads to reach new customers), and "EXPRESS DEAL" (Priceline Express Deals).
- Actual content:** A green oval and arrow labeled "Actual content" point to a post by Caroline McCarthy, which reads: "I wrote something for Date Report (thanks for the opp Melissa!) about why the real problem with PrincetonMomGate is that she's not-so-implicitly rushing..."

The interface also shows a search bar at the top, a left sidebar with navigation links (Socialcam, Gifts, Photos, App Center), and a list of friends.

Execute multiple copies

- Need to execute multiple copies of program
 - So z stays distinct for the rest of execution
- Now consider we have more than just one observer
 - Facebook, Yahoo Ads, and random.com ads
 - Now need to execute many copies of program
- Doesn't scale very well! (In general 2^n)

Faceted execution

- Key concept: proxy data with tags
 - (principal, private view, low view)
 - **If I'm principal k , I see the result V_h otherwise I see V_l**

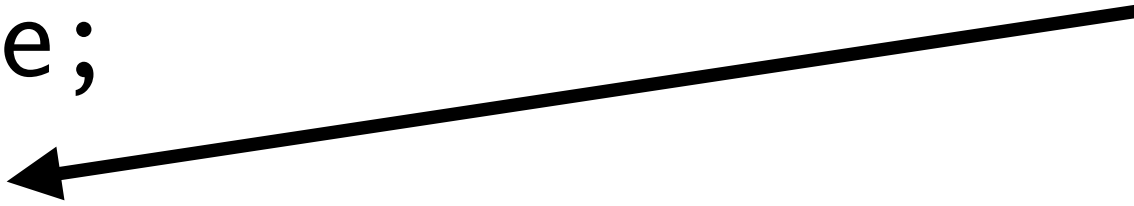
$$\{ k \mid V_h, V_l \}$$

Principal (e.g., **FB**, foo.com)

The way **k** sees the result of the computation

The way **the rest of the world** sees the computation

```
function(x = <FB | true ,  $\perp$ >) {  
  y = true;  
  z = true;  
  if (x)  
    y = false  
  if (y)  
    z = false  
  return z;  
}
```

```
function(x = <FB | true ,  $\perp$ >) {  
  y = true;  
  z = true;  
  if (x)   
    y = false  
  if (y)  
    z = false  
  return z;  
}
```

What happens **here**

Execute **twice**:

- Once for private facet
- Once for public facet
- join back together

Private

- x is true
- y is false

Public

- x is false (\perp)
- y is true

Join them! y becomes <FB | false , true>

```
function(x = <FB | true ,  $\perp$ >) {  
  y = true;  
  z = true;  
  if (x)  
    y = false  
  if (<FB | false, true>)  
    z = false  
  return z;  
}
```

Apply trick again...

after if z = <FB | true, false>

Using faceted values

- Imagine some malicious code tries to send out z
- Should see computation as if private inputs were \perp
- Use **projection**:

`send(foo.com, z = <FB | true, false>)`

Check to see if foo.com is FB

NO!
false

- This is a **simple trick** that solves a **big problem**
- Some technical details I can help you figure out
 - But works pretty well!
- Also supports **declassification**
- Potential research:
 - might propagate facet to place where it **isn't needed**
 - Can eliminate facet that is of form $\langle k \mid v_1, v_1 \rangle$
 - Also places where facet is never projected?

Silly example of when facets are unnecessary...

```
function(x = <FB | v ,  $\perp$ >) {  
  y = true;  
  if (x+x-2*x = 0)  
    y = true  
  else  
    y = false  
  return <FB | false , false>;  
}
```

*Probably gets more interesting with more
interesting domains / higher order control flow...*

Your project

- Read and figure out the faceted execution paper
- Implement faceted execution in Redex
- Research: eliminate unnecessary facets
 - **Use symbolic execution to prove agreement**
 - **Show on various examples**
 - Prove implementation is correct
 - Extend actual implementation

Thanks!

- I look forward to talking to you about this!
- Please talk to me if you're interested