# P4BID: INFORMATION FLOW CONTROL IN P4

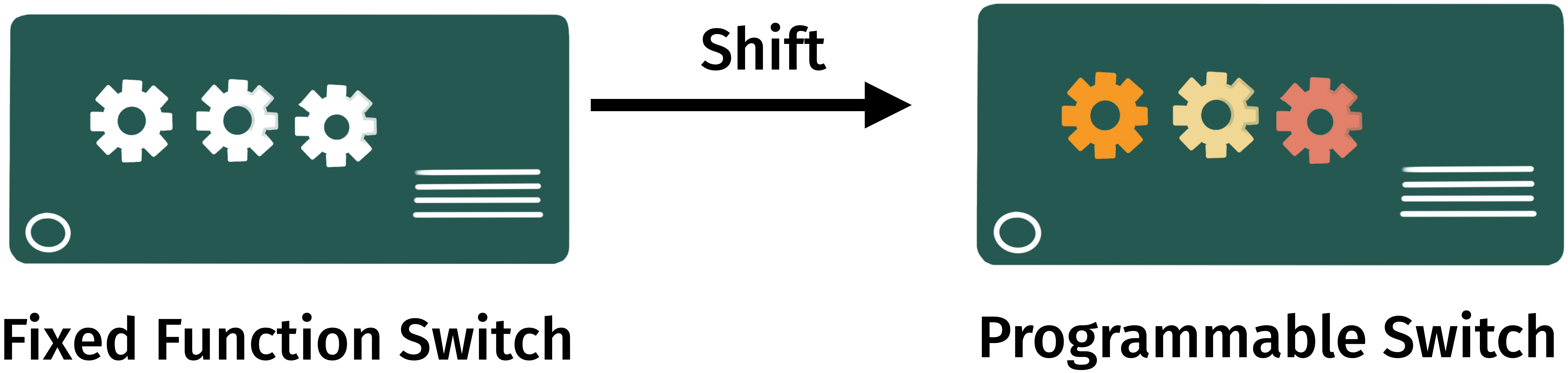
PLDI’22

**Karuna Grewal**, Loris D’Antoni, Justin Hsu

# SHIFTING TRENDS

Fixed Function Switch

# SHIFTING TRENDS

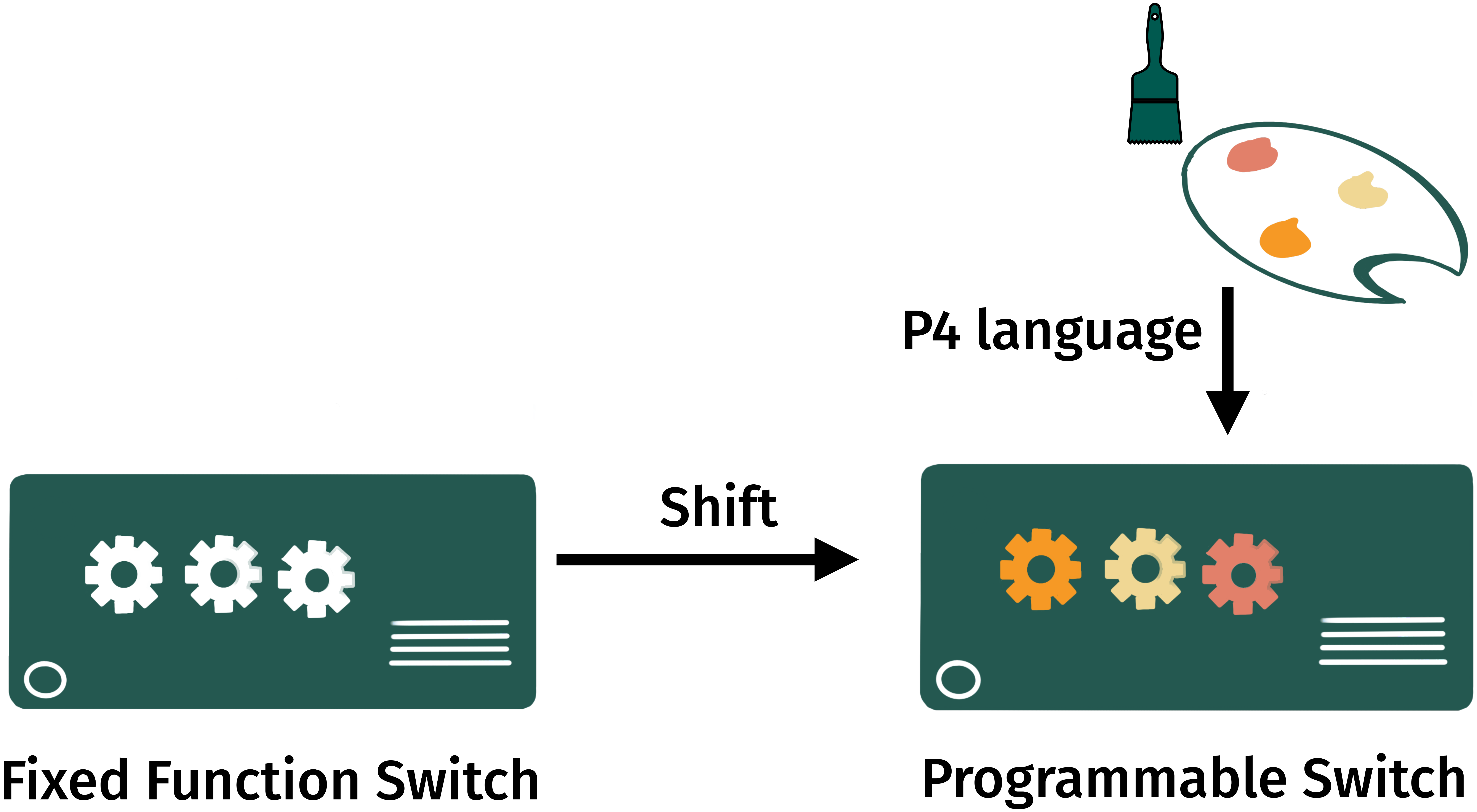


Shift

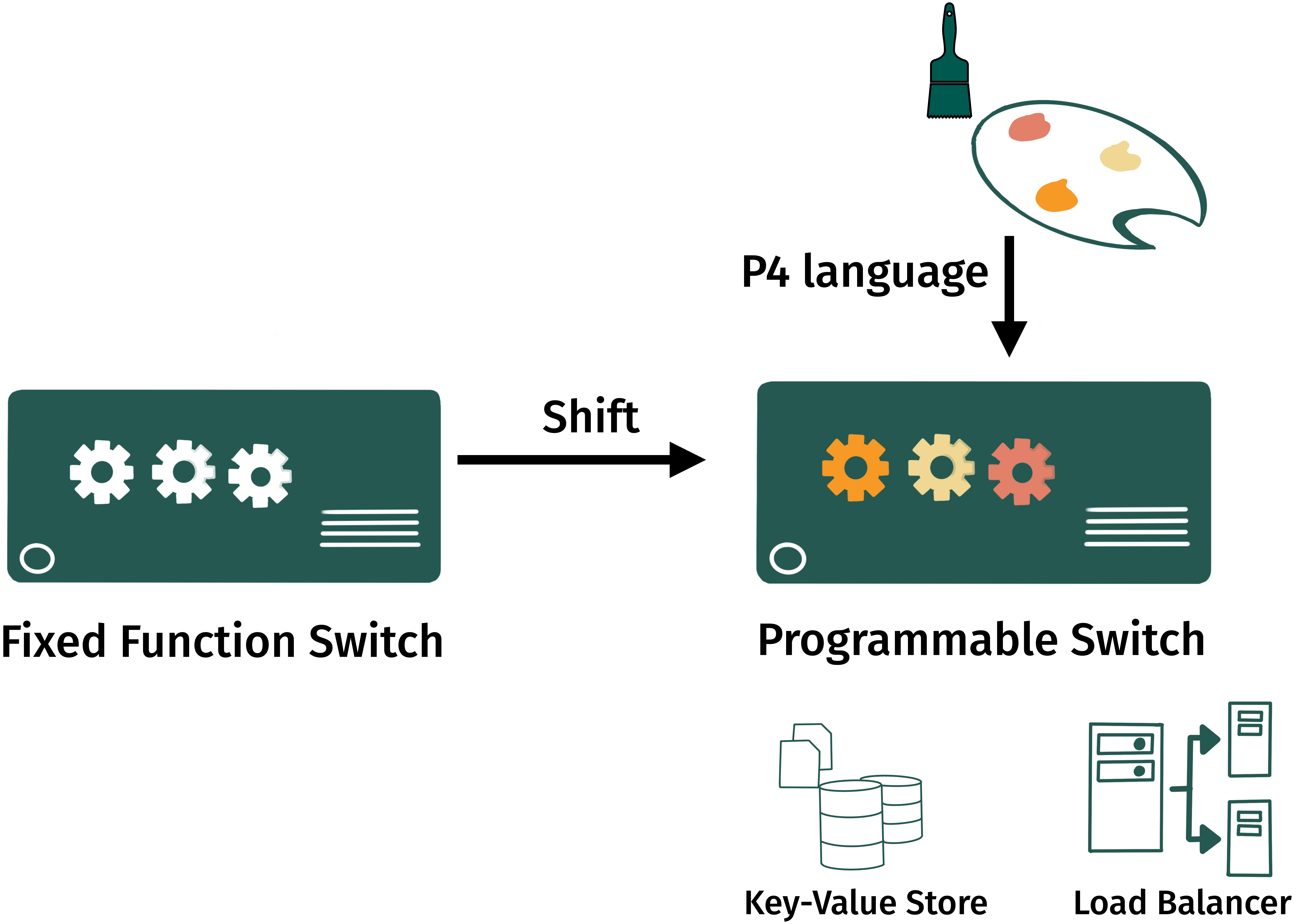
Fixed Function Switch

Programmable Switch

# SHIFTING TRENDS



# SHIFTING TRENDS



P4 language

Shift

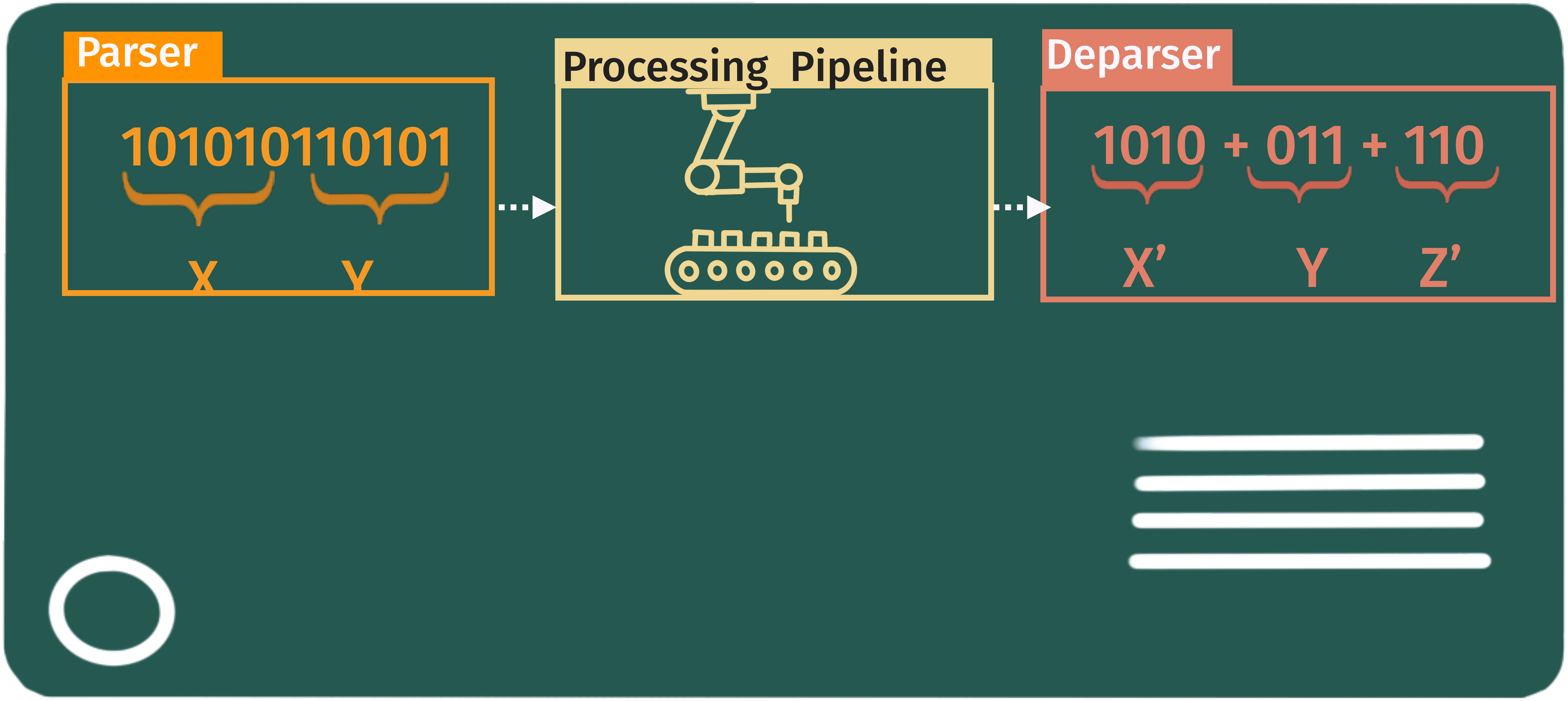
Fixed Function Switch

Programmable Switch

Key-Value Store

Load Balancer

# CUSTOMIZING A SWITCH

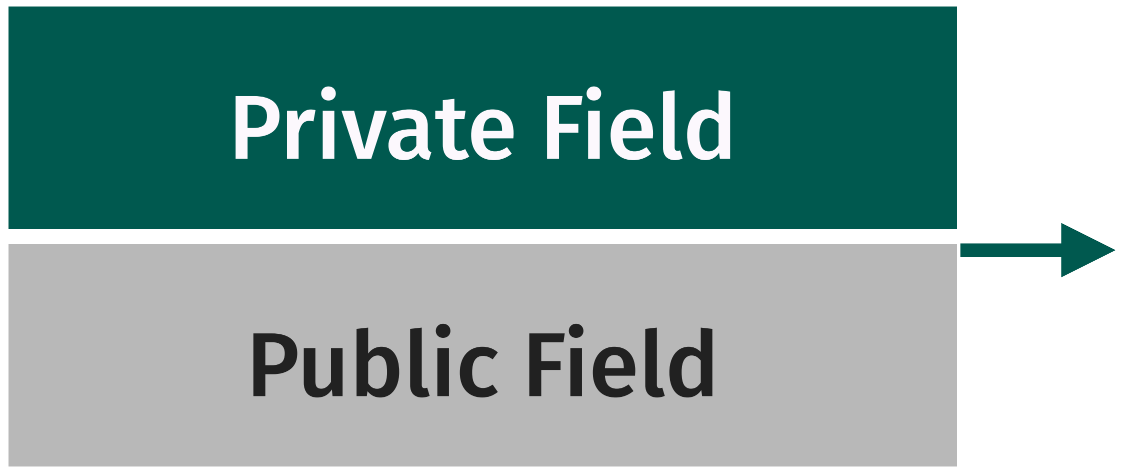


# A NEW CHALLENGE...

Programming Errors → Information Leak

# A NEW CHALLENGE...

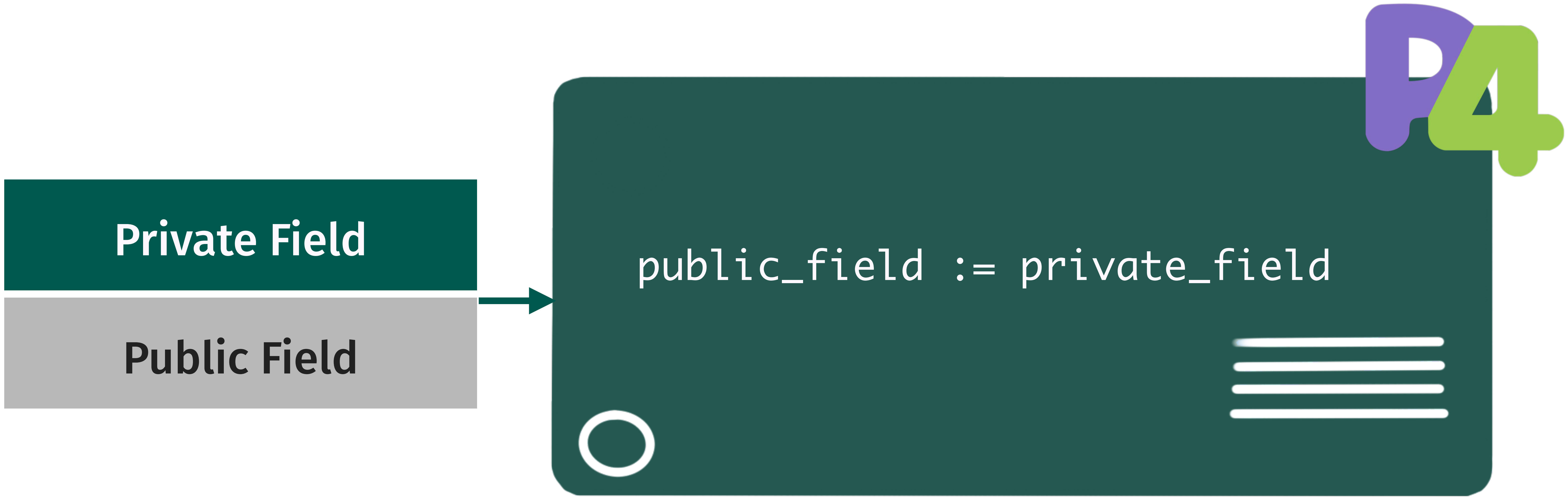
Programming Errors → Information Leak



Incoming Packet Header

# A NEW CHALLENGE...

## Programming Errors → Information Leak



Private Field

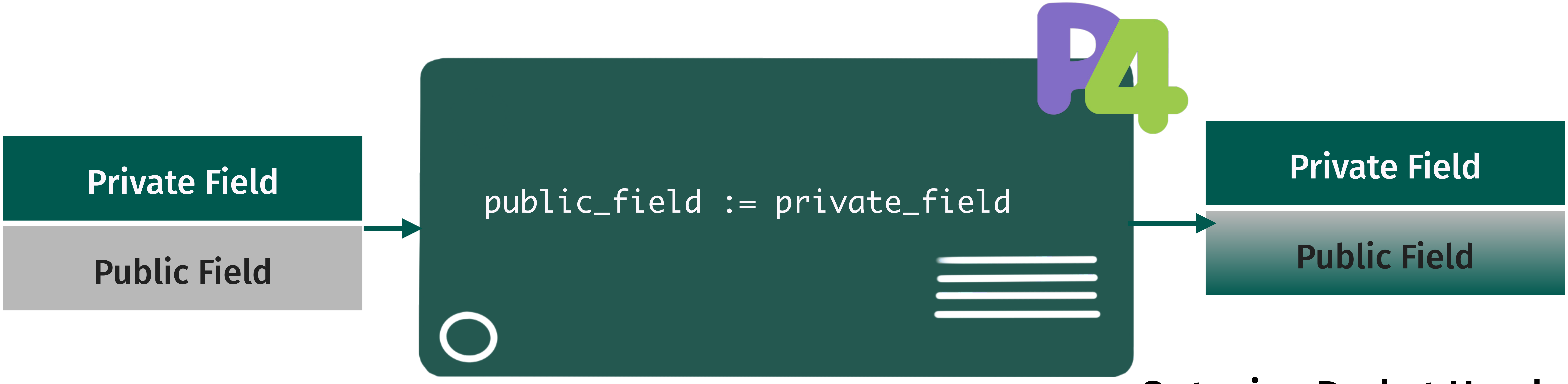
Public Field

```
public_field := private_field
```

Incoming Packet Header

# A NEW CHALLENGE...

## Programming Errors → Information Leak



Private Field

Public Field

```
public_field := private_field
```

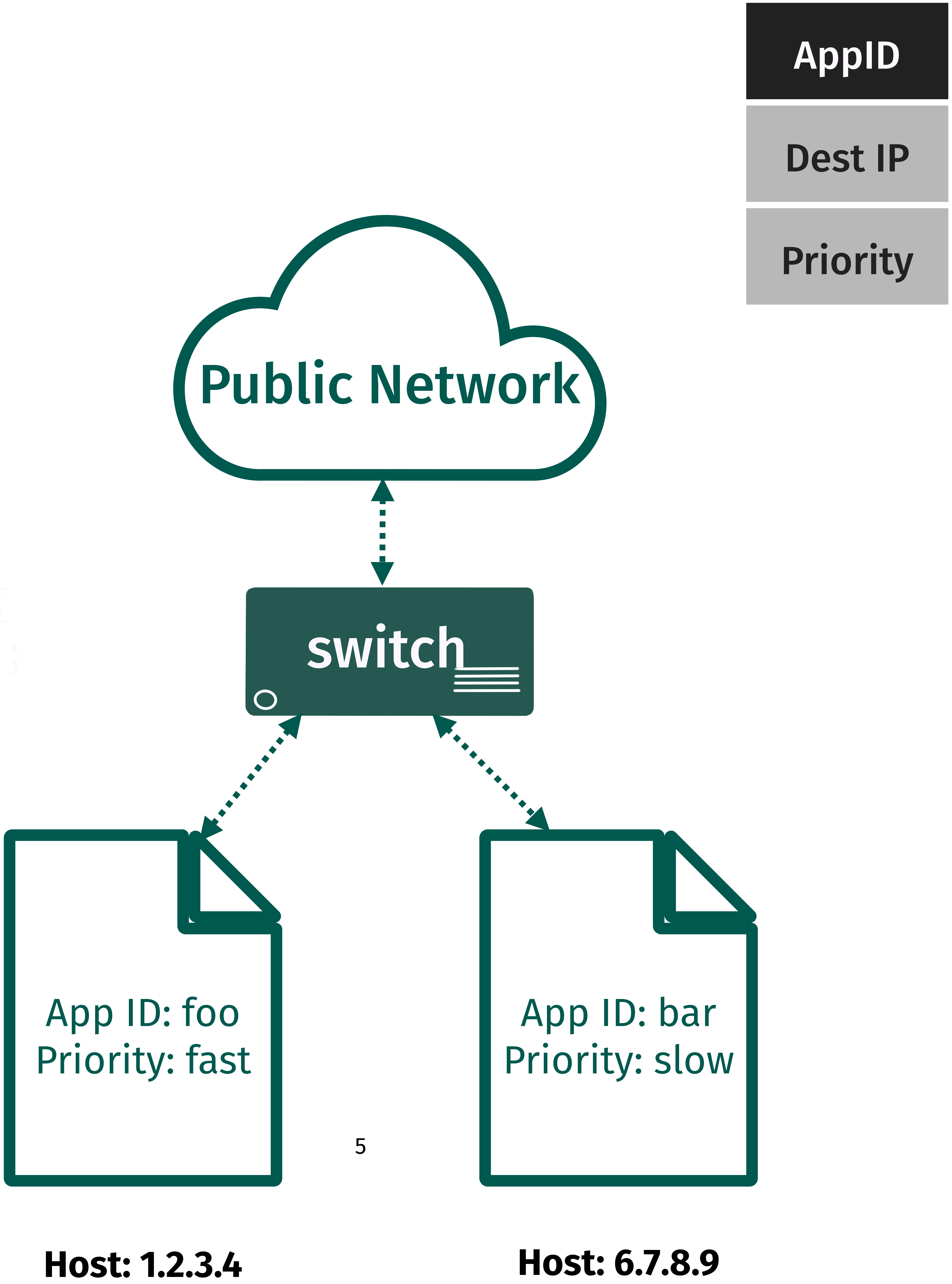
Private Field

Public Field

Incoming Packet Header

Outgoing Packet Header

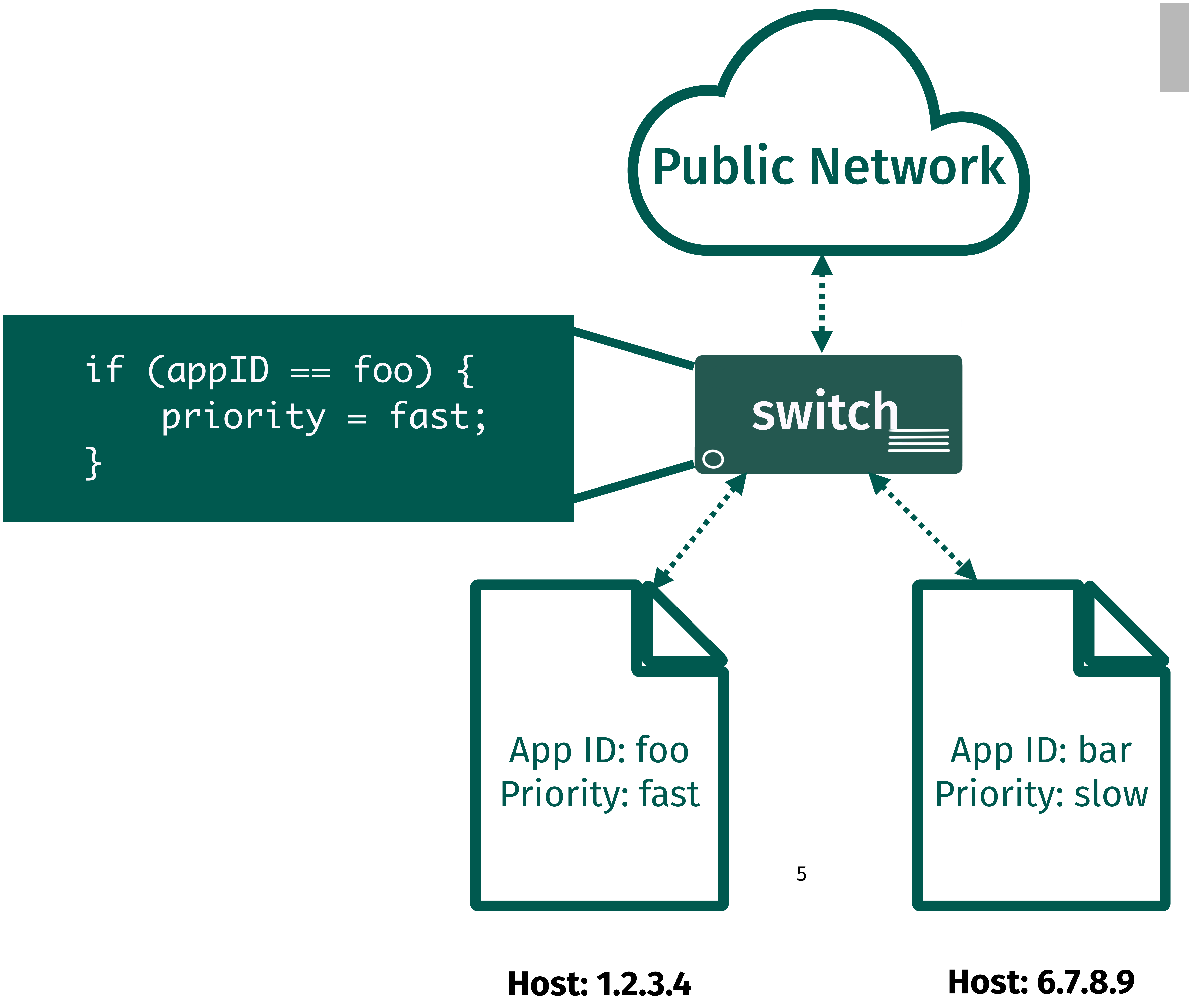
```
if (appID == foo) {
    priority = fast;
}
```



AppID

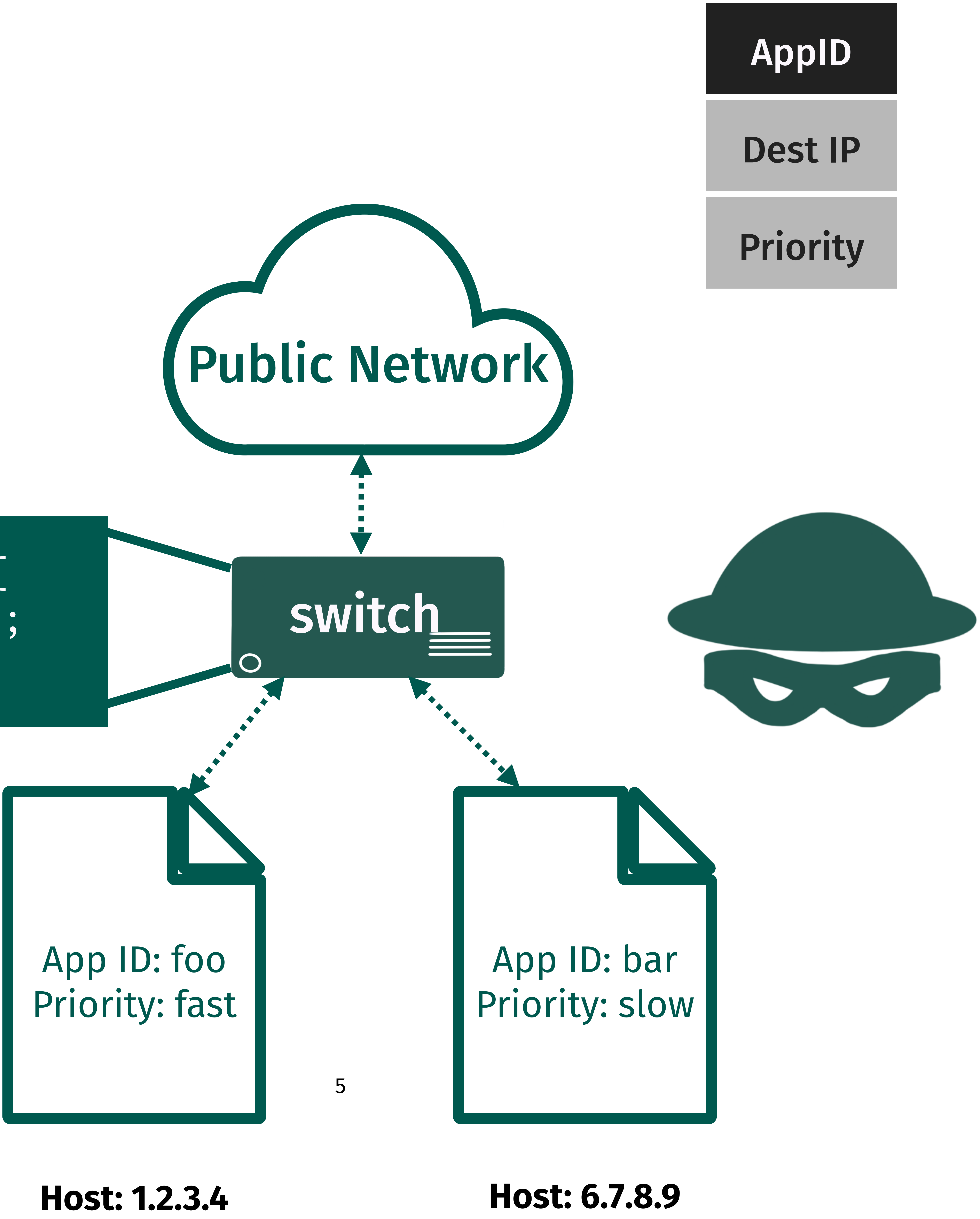
Dest IP

Priority



```
if (appID == foo) {
    priority = fast;
}
```

```
if (appID == foo) {
    priority = fast;
}
```



AppID

Dest IP

Priority

foo

1.2.3.4

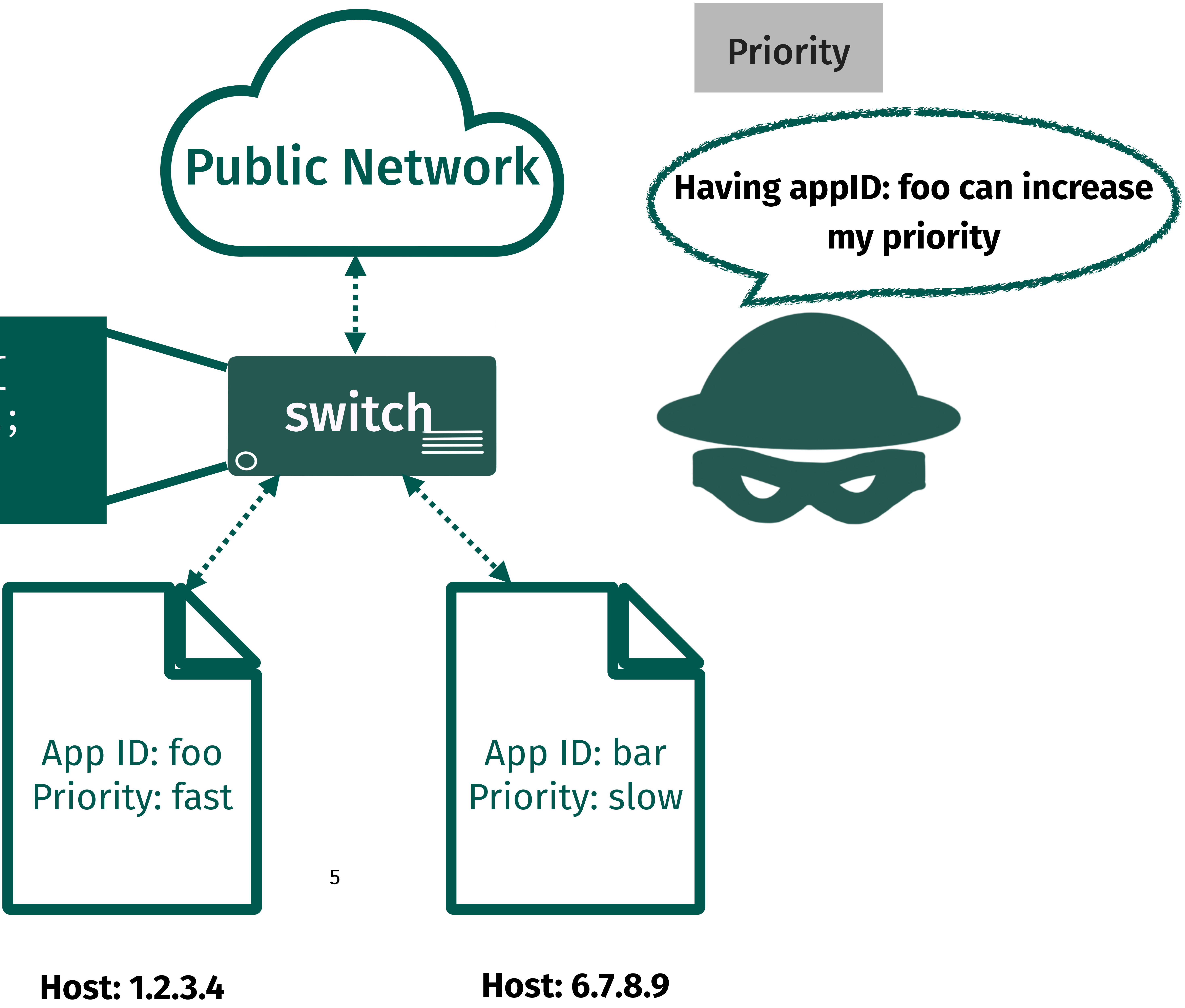
?

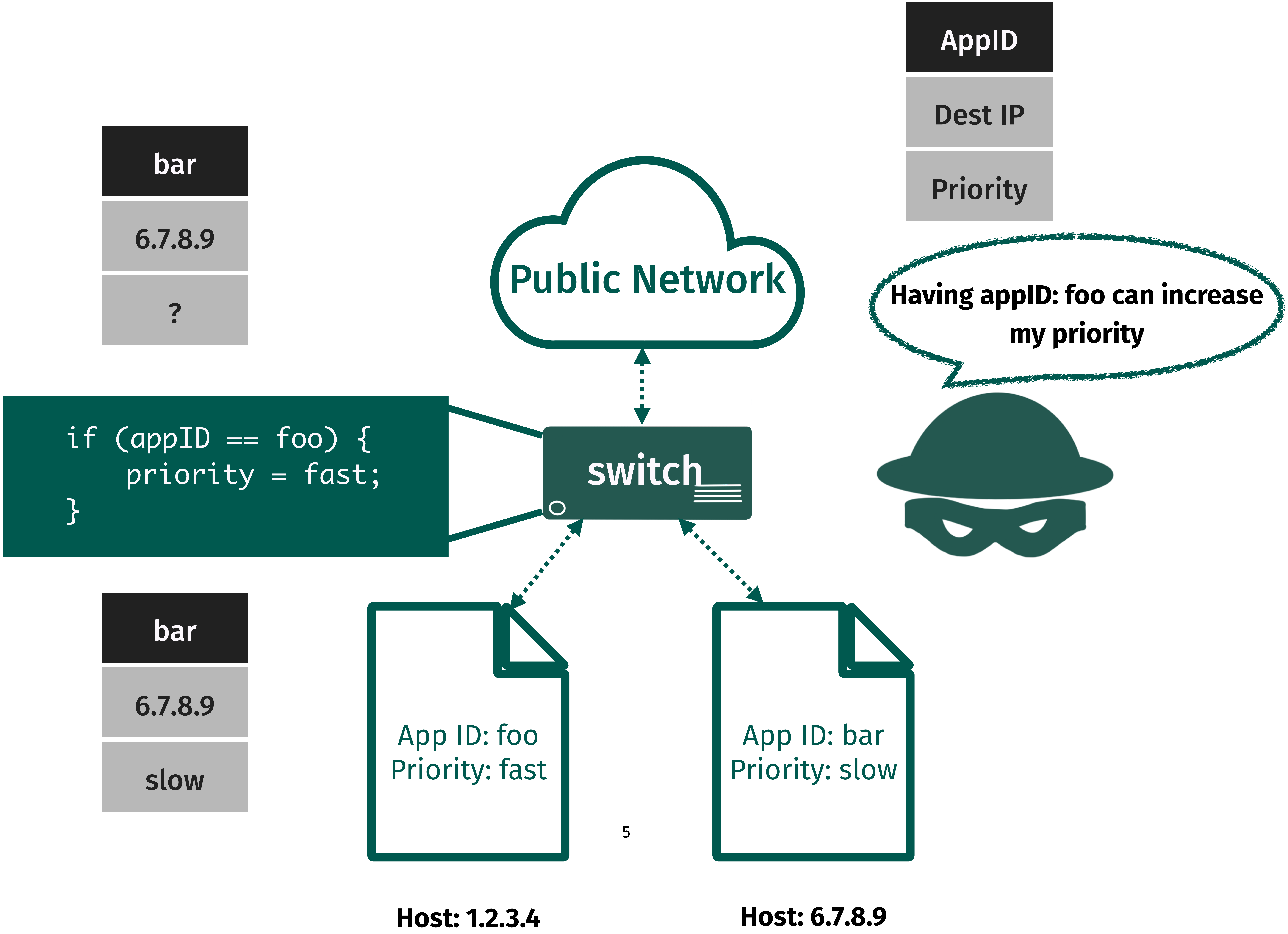
```
if (appID == foo) {
    priority = fast;
}
```

foo

1.2.3.4

fast





AppID
Dest IP
Priority

bar
6.7.8.9
?

Public Network

```
if (appID == foo) {
    priority = fast;
}
```

switch

Having appID: foo can increase my priority

bar
6.7.8.9
slow

App ID: foo
Priority: fast

Host: 1.2.3.4

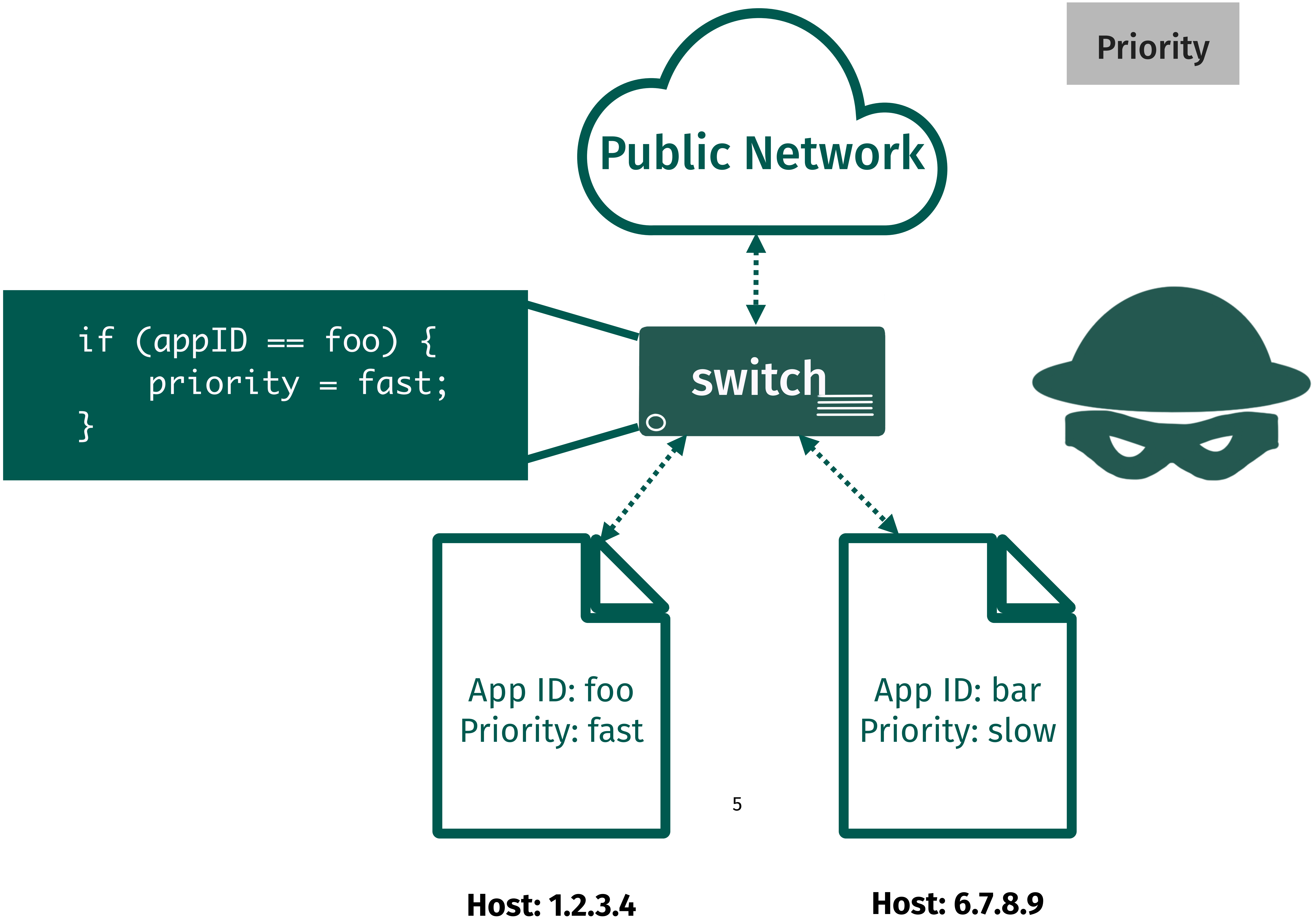
App ID: bar
Priority: slow

Host: 6.7.8.9

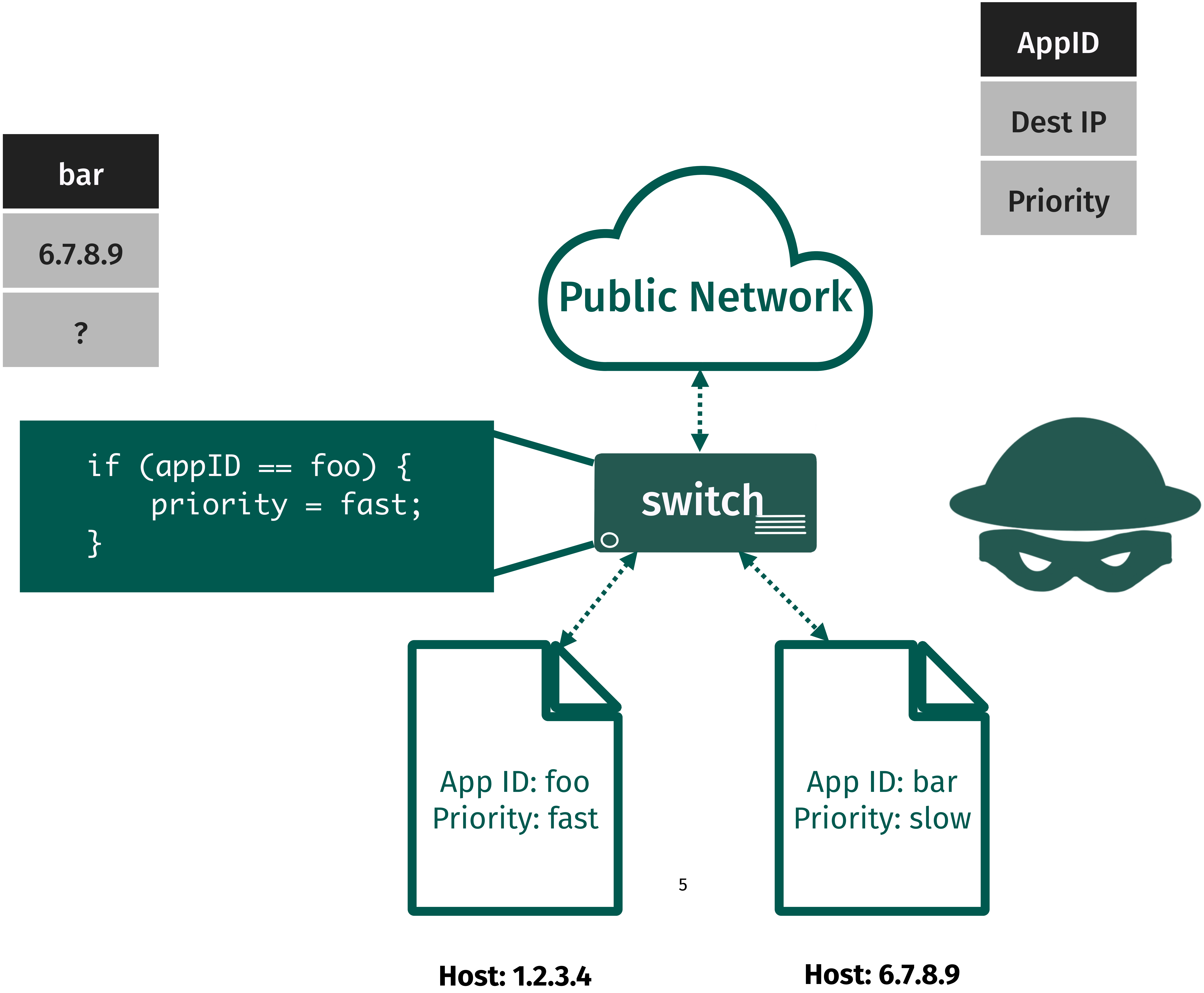
AppID

Dest IP

Priority



```
if (appID == foo) {
    priority = fast;
}
```



| AppID |
|---|
| Dest IP |
| Priority |

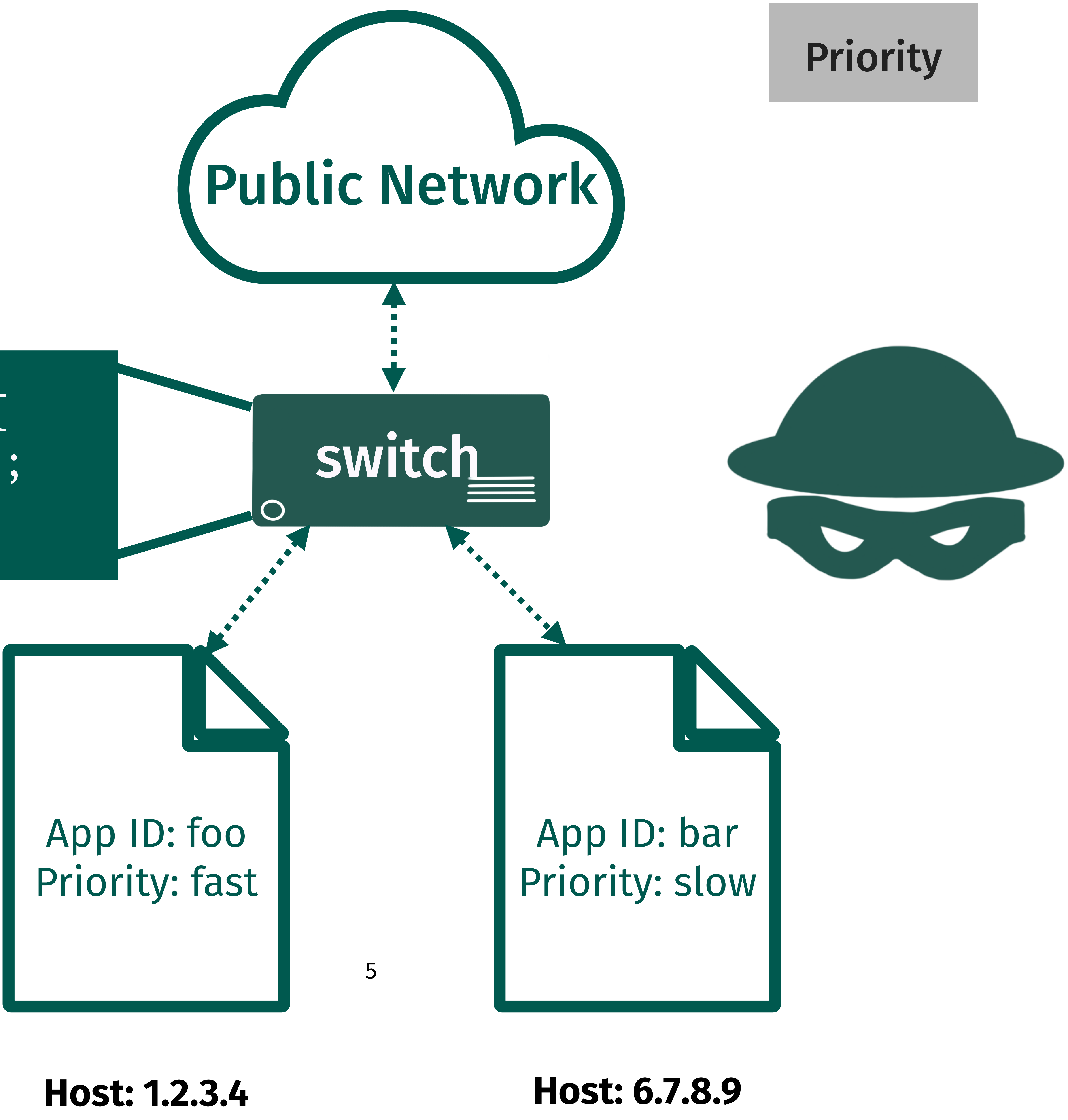
| bar |
|---|
| 6.7.8.9 |
| ? |

```
if (appID == foo) {
    priority = fast;
}
```

| |
|---|
| AppID |
| Dest IP |
| Priority |

| |
|---|
| ~~bar~~ |
| 6.7.8.9 |
| ? |

```
if (appID == foo) {
    priority = fast;
}
```



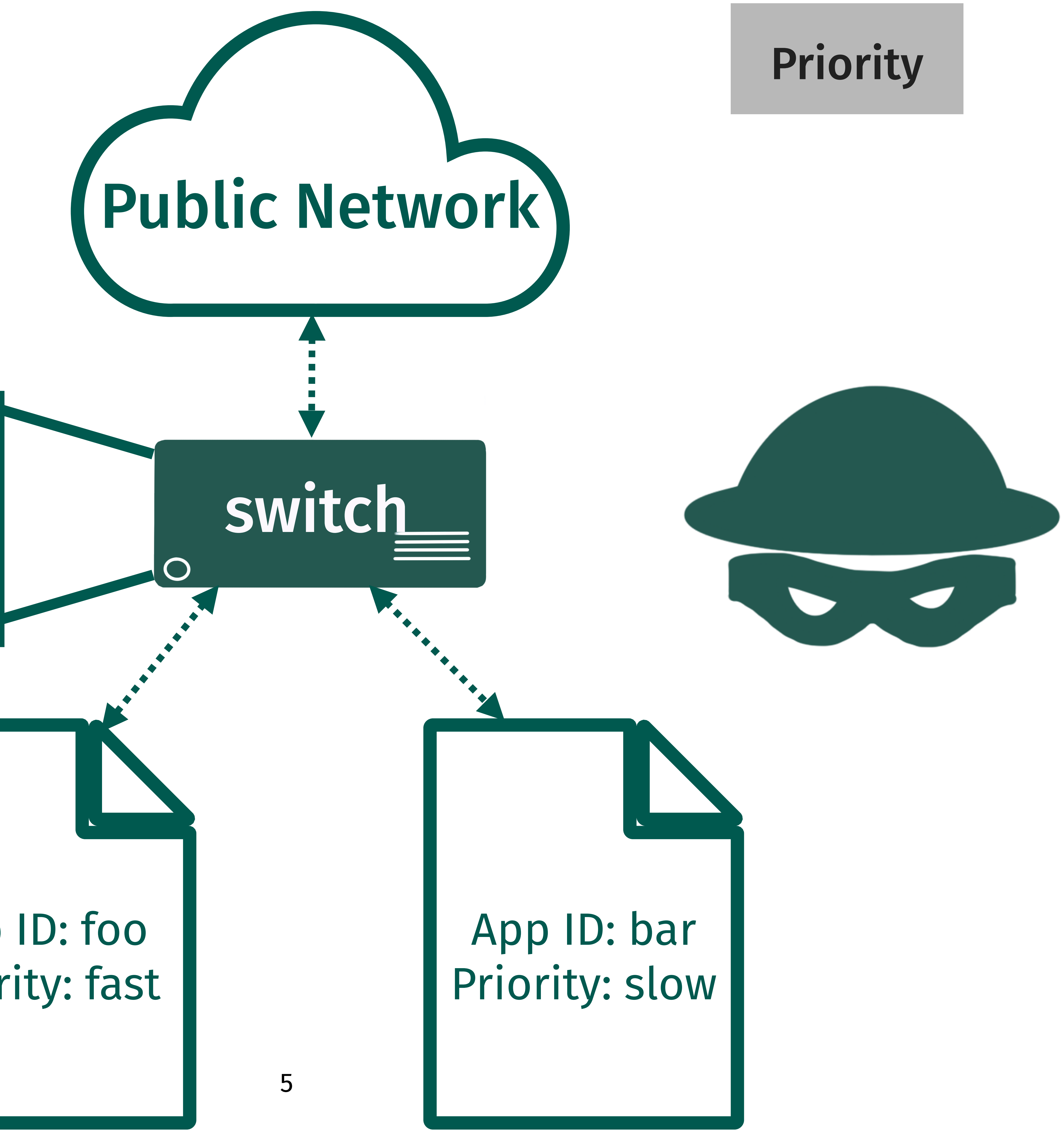
| ~~bar~~ | foo |
|---|---|
| 6.7.8.9 | 6.7.8.9 |
| ? | ? |

AppID

Dest IP

Priority

```
if (appID == foo) {
    priority = fast;
}
```



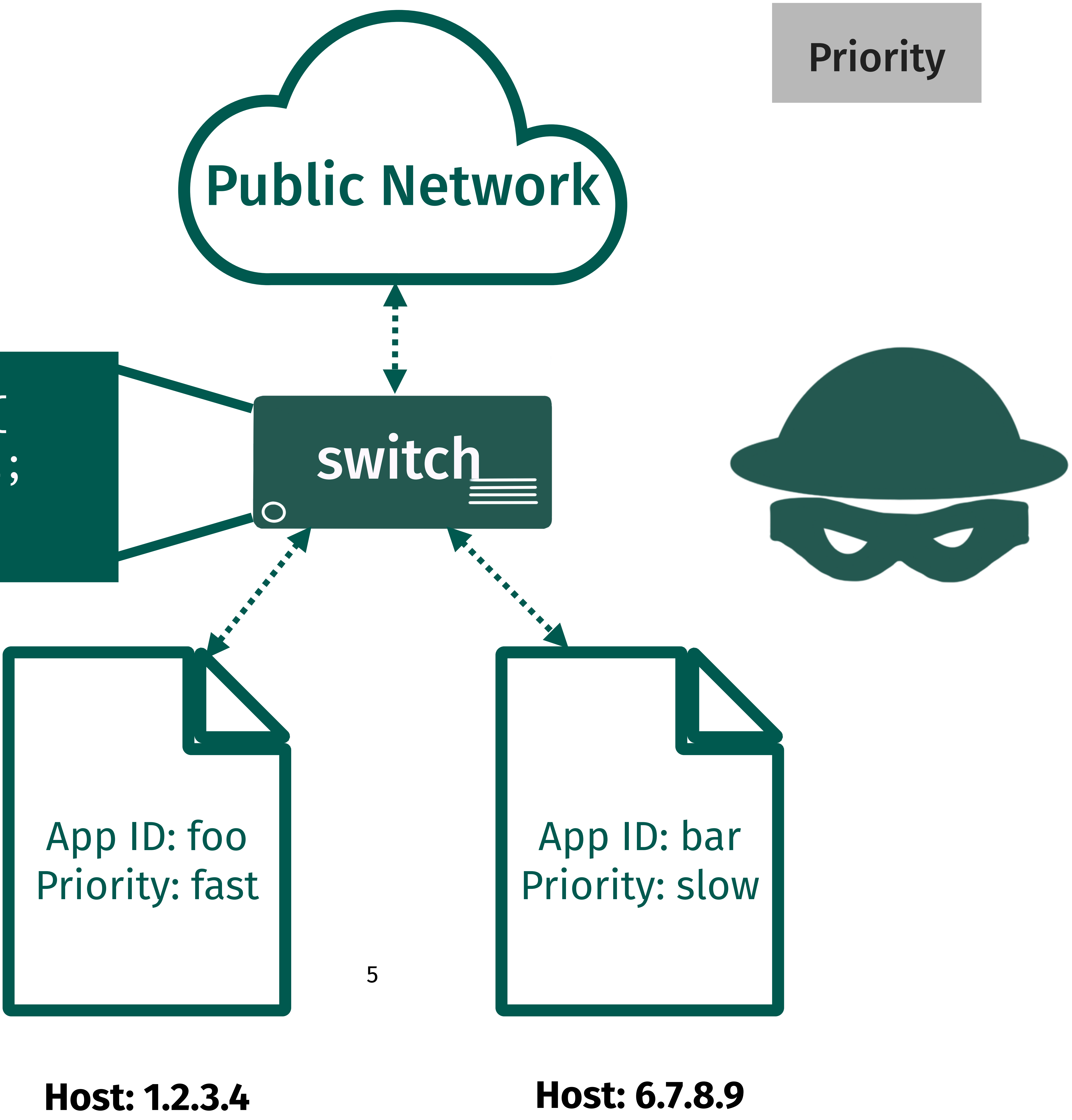
**Host: 1.2.3.4**

**Host: 6.7.8.9**

| AppID |
|---|
| Dest IP |
| Priority |

| ~~bar~~ | foo |
|---|---|
| 6.7.8.9 | 6.7.8.9 |
| ? | ? |

```
if (appID == foo) {
    priority = fast;
}
```



| foo |
|---|
| 6.7.8.9 |
| fast |

# CONTRIBUTIONS

# P4BID

Information flow control type system for P4

Implement P4BID on top of P4’s reference compiler

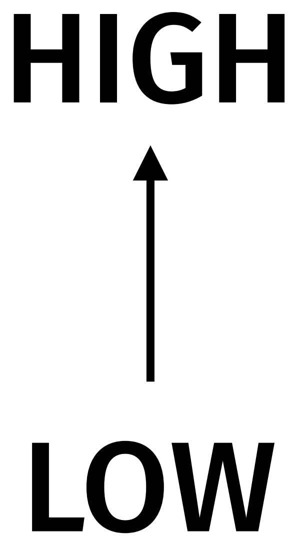
Encode networking properties as IFC properties

# A QUICK REVIEW OF INFORMATION FLOW CONTROL

# INFORMATION FLOW CONTROL REFRESHER

# INFORMATION FLOW CONTROL REFRESHER

**Security lattice**

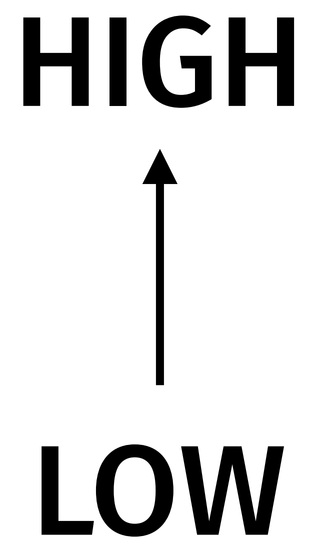


HIGH

LOW

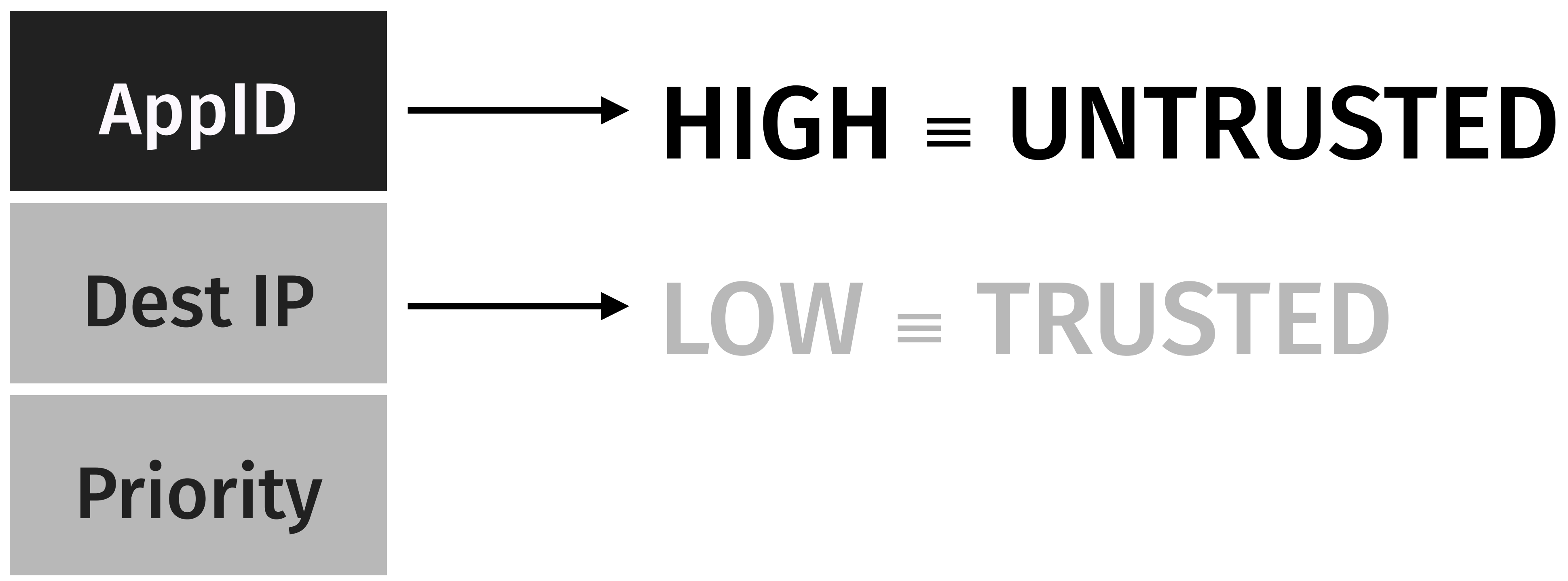
# INFORMATION FLOW CONTROL REFRESHER

## Security lattice


HIGH

LOW

## Label the variables

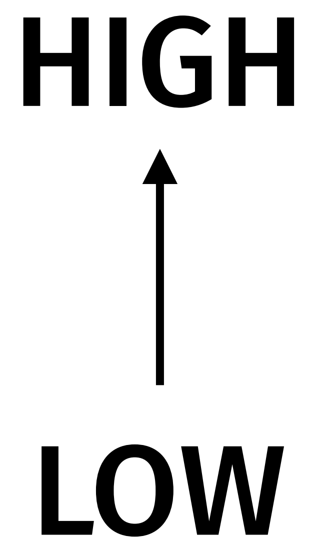

AppID → HIGH ≡ UNTRUSTED

Dest IP → LOW ≡ TRUSTED

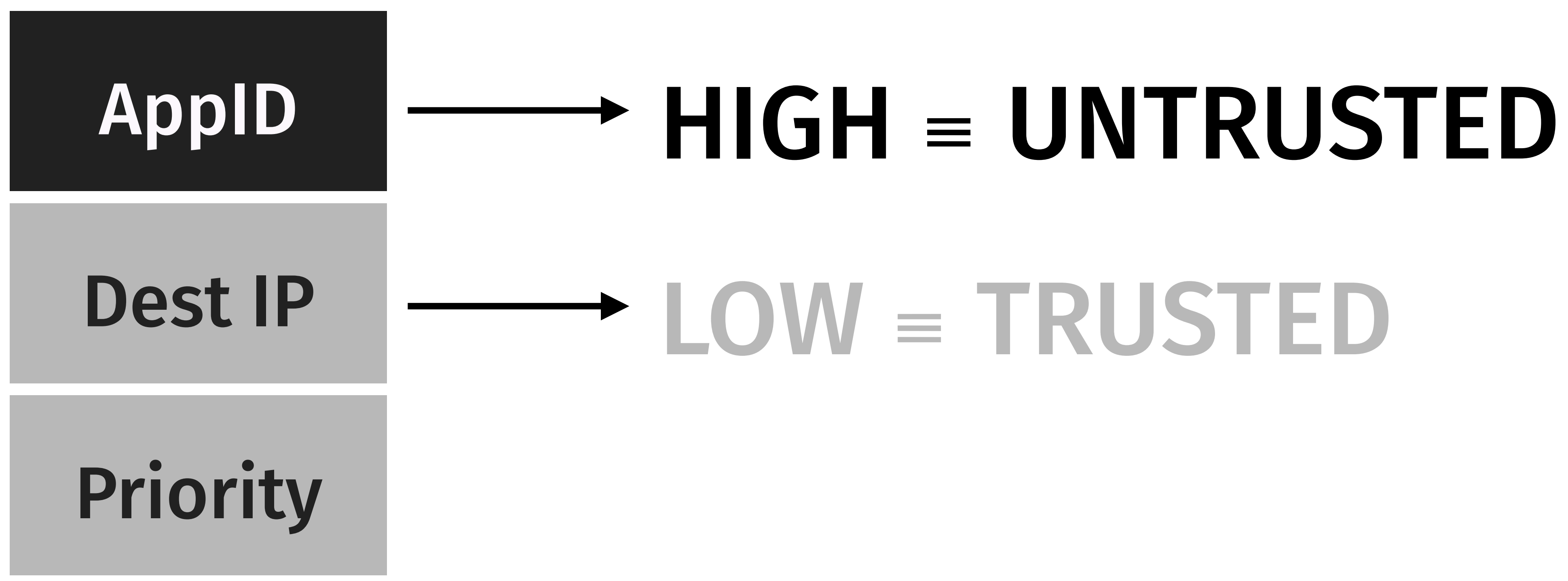
Priority

# INFORMATION FLOW CONTROL REFRESHER

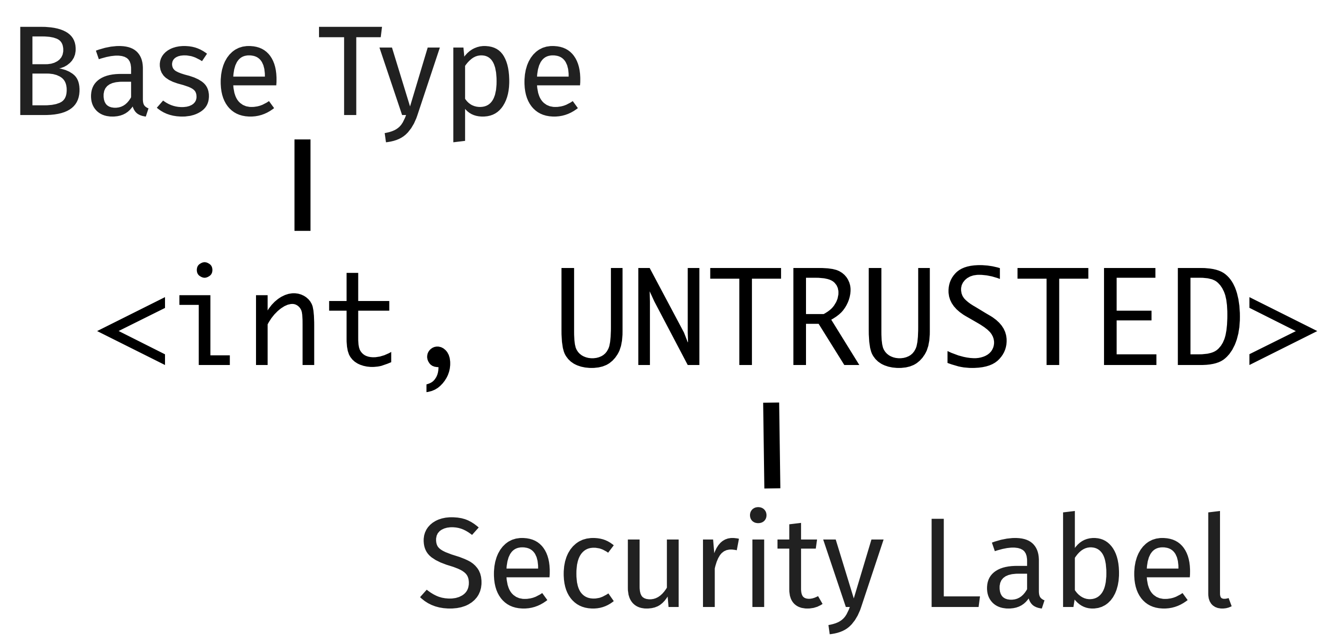
**Security lattice**



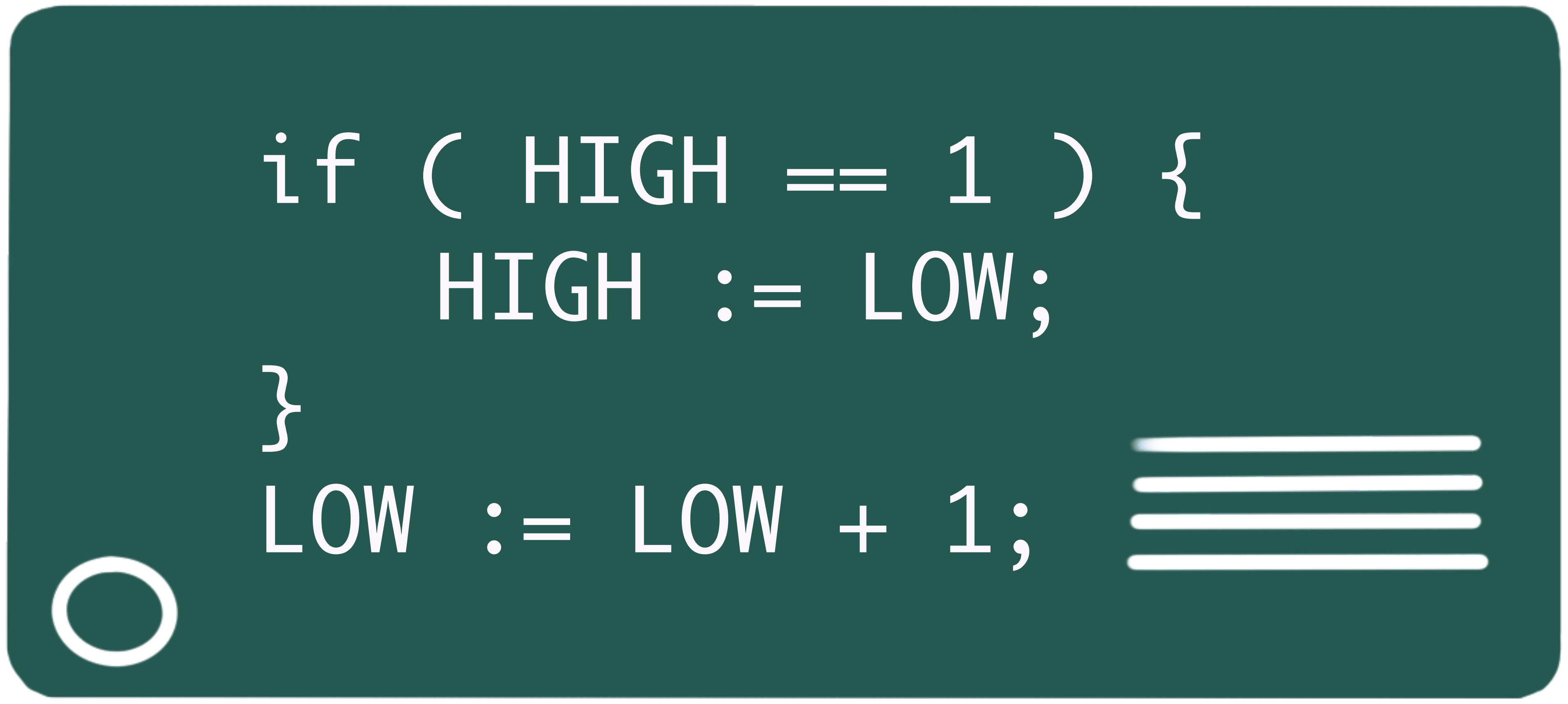
**Label the variables**



**Security types**



# TYPE SYSTEM GUARANTEES

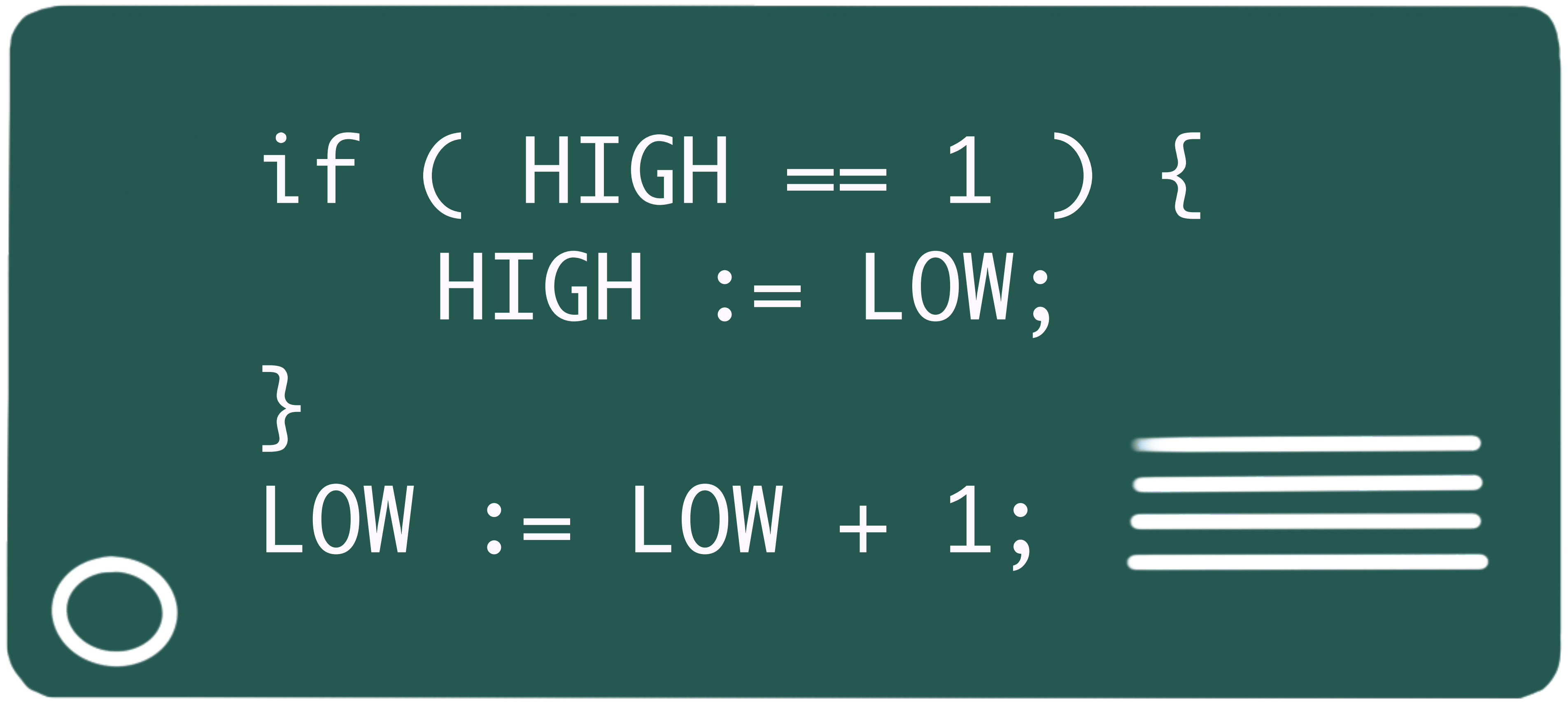


```
if ( HIGH == 1 ) {
    HIGH := LOW;
}
LOW := LOW + 1;
```

# TYPE SYSTEM GUARANTEES

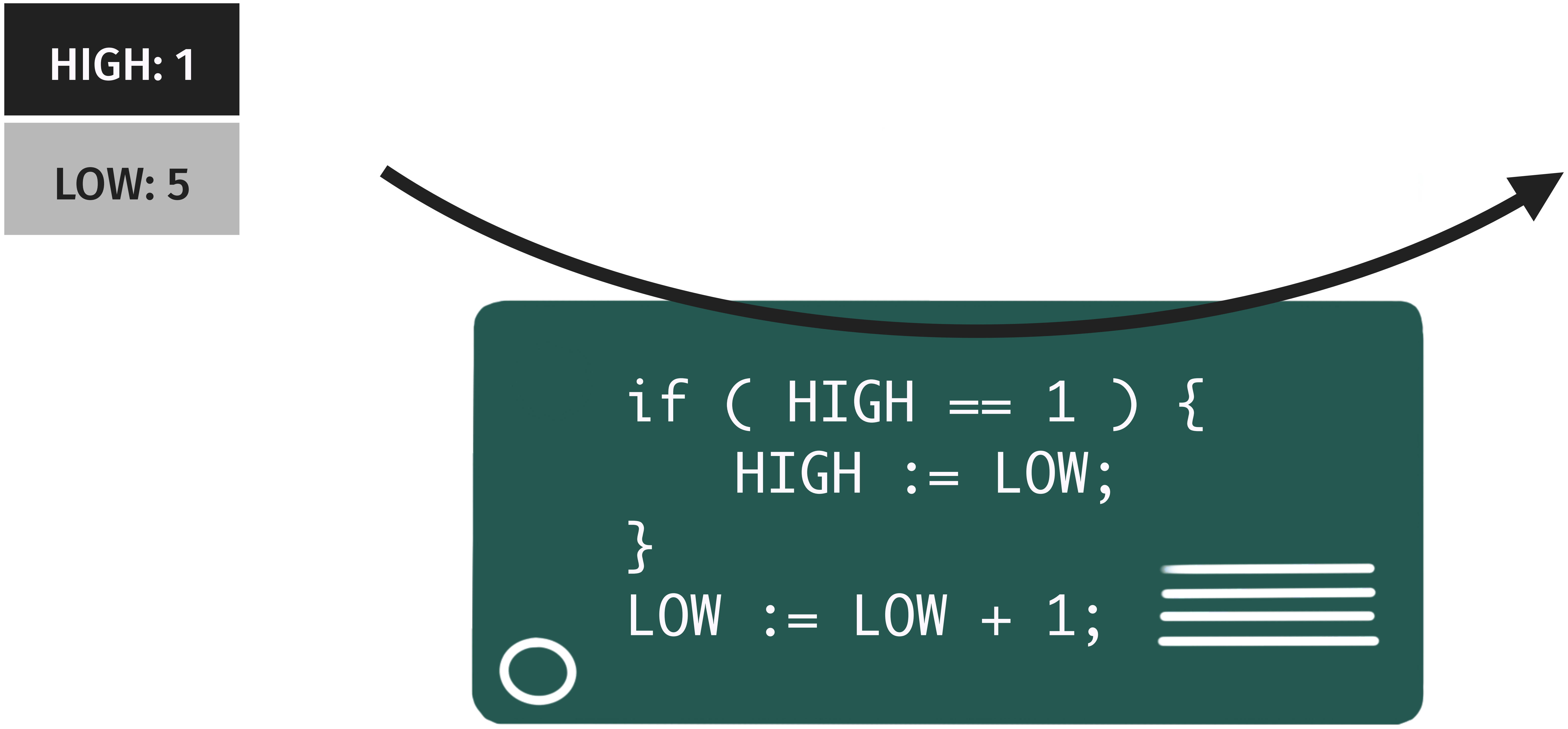
HIGH: 1

LOW: 5



```
if ( HIGH == 1 ) {
    HIGH := LOW;
}
LOW := LOW + 1;
```

# TYPE SYSTEM GUARANTEES

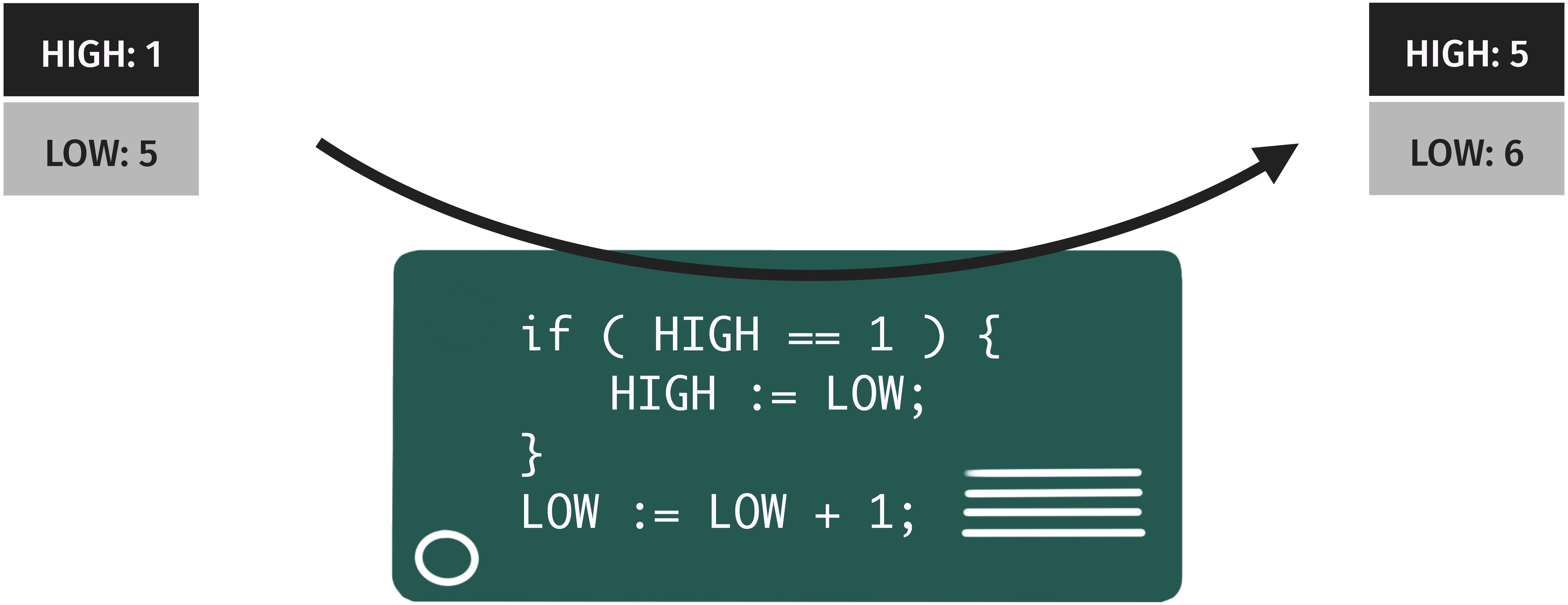


HIGH: 1

LOW: 5

```
if ( HIGH == 1 ) {
    HIGH := LOW;
}
LOW := LOW + 1;
```

# TYPE SYSTEM GUARANTEES



HIGH: 1

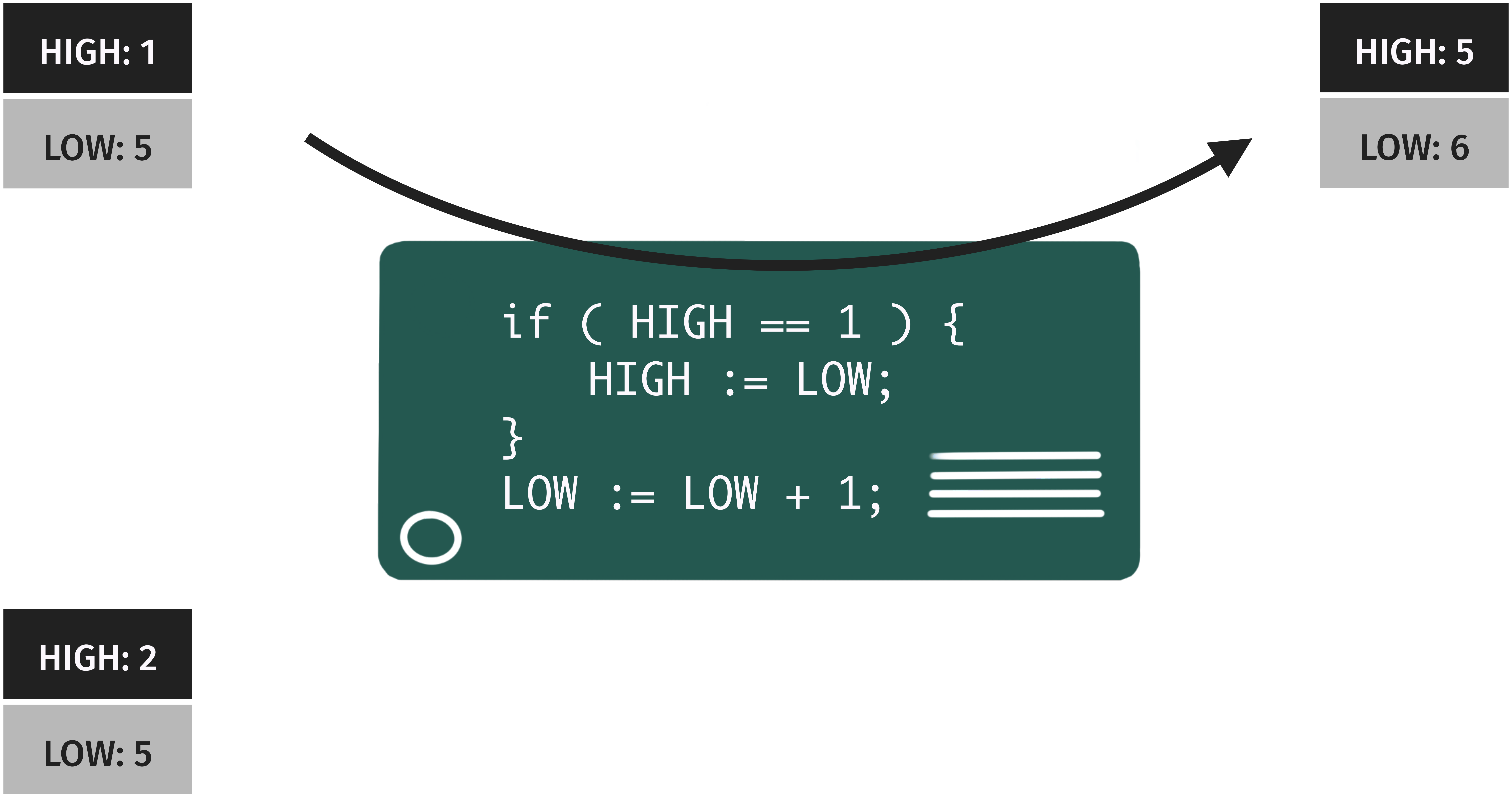
LOW: 5

```
if ( HIGH == 1 ) {
    HIGH := LOW;
}
LOW := LOW + 1;
```

HIGH: 5

LOW: 6

# TYPE SYSTEM GUARANTEES



HIGH: 1

LOW: 5

```
if ( HIGH == 1 ) {
    HIGH := LOW;
}
LOW := LOW + 1;
```

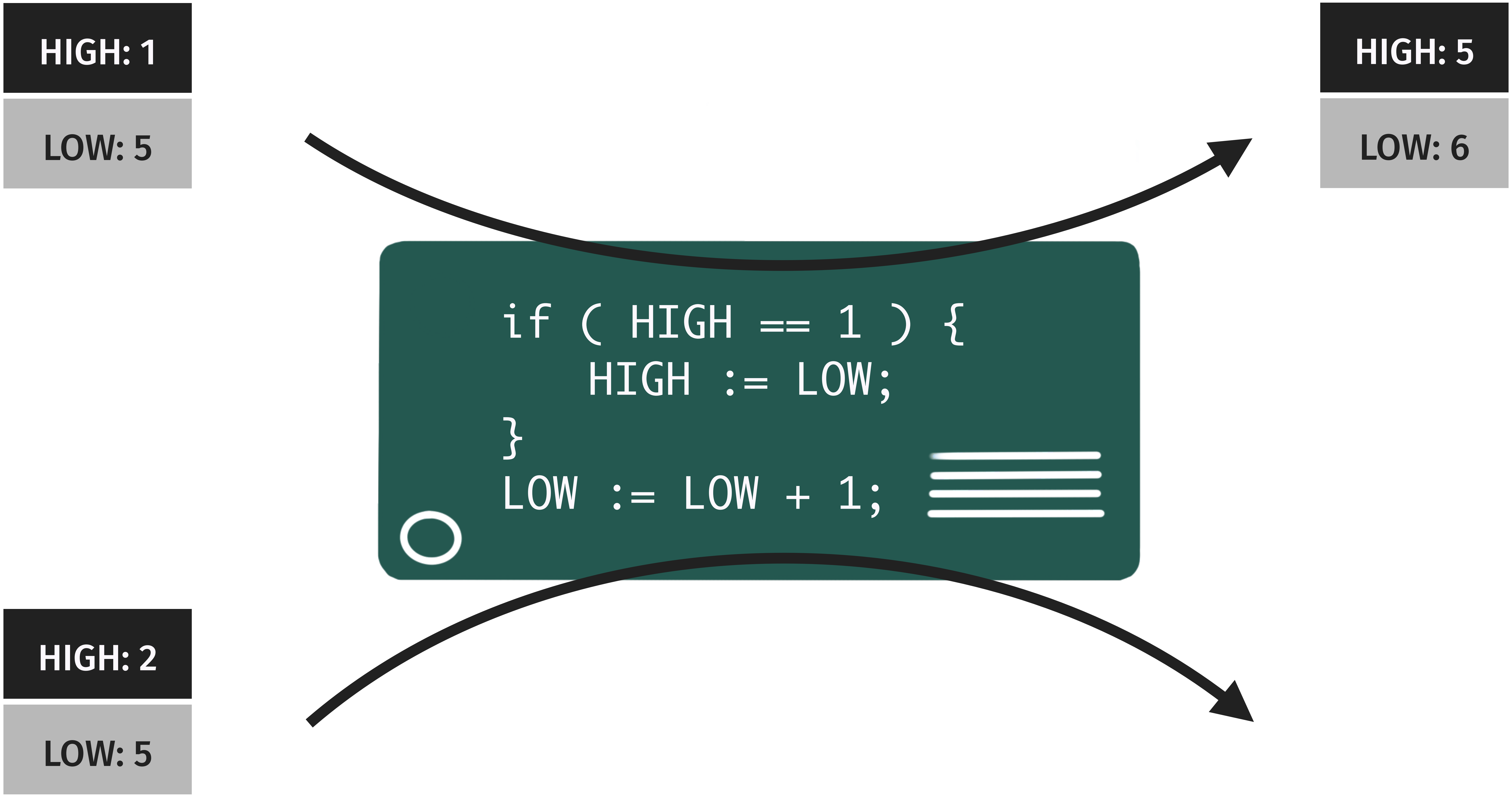
HIGH: 5

LOW: 6

HIGH: 2

LOW: 5

# TYPE SYSTEM GUARANTEES



HIGH: 1

LOW: 5

```
if ( HIGH == 1 ) {
    HIGH := LOW;
}
LOW := LOW + 1;
```

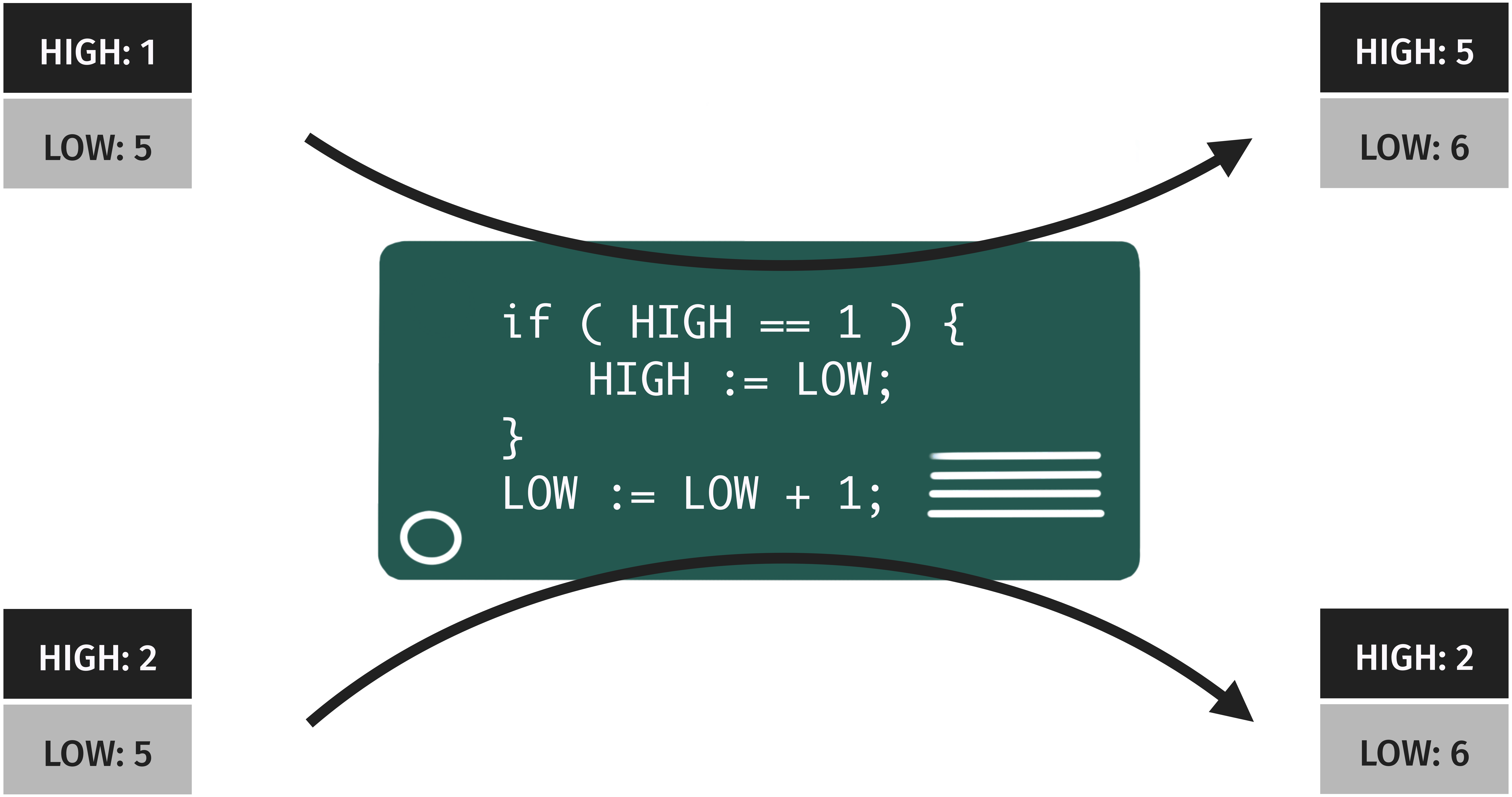
HIGH: 5

LOW: 6

HIGH: 2

LOW: 5

# TYPE SYSTEM GUARANTEES



HIGH: 1

LOW: 5

HIGH: 2

LOW: 5

```
if ( HIGH == 1 ) {
    HIGH := LOW;
}
LOW := LOW + 1;
```

HIGH: 5

LOW: 6

HIGH: 2

LOW: 6

# INFORMATION FLOW CONTROL CHALLENGES IN P4

# P4 LANGUAGE: EXAMPLE *APP2PRIORITY*

# P4 LANGUAGE: EXAMPLE *APP2PRIORITY*

**Action Declaration**

```
action set_priority(int new_priority) {
    hdr.priority = new_priority;
}
```

# P4 LANGUAGE: EXAMPLE *APP2PRIORITY*

**Action Declaration**

```
action set_priority(int new_priority) {
    hdr.priority = new_priority;
}
```

**Table Declaration**

**(Installed at Runtime)**

```
table app2priority {
    key = {
        hdr.appID;
    }
    actions = {
       set_priority;
    }
}
```

# P4 LANGUAGE: EXAMPLE *APP2PRIORITY*

**Action Declaration**

```
action set_priority(int new_priority) {
    hdr.priority = new_priority;
}
```

**Table Declaration**

**(Installed at Runtime)**

```
table app2priority {
    key = {
        hdr.appID;
    }
    actions = {
        set_priority;
    }
}
```

**Statement**

```
apply {
    app2priority.apply();
}
```

# LEAKS IN TABLES

# LEAKS IN TABLES

```
table match_action {
    key = { high_key; }
    actions  = { modify_low_field; }
}
```

# LEAKS IN TABLES

```
table match_action {
    key = { high_key; }
    actions  = { modify_low_field; }
}
```

⦀

```
if (high_key == foo) {
    modify_low_field();
}
else if (high_key == bar) {
    skip;
}
```

# LEAKS IN TABLES

```
table match_action {
    key = { high_key; }
    actions  = { modify_low_field; }
}
```

⦀

```
if (high_key == foo) {
    modify_low_field();
}
else if (high_key == bar) {
    skip;
}
```

Branch on a HIGH variable

# LEAKS IN TABLES

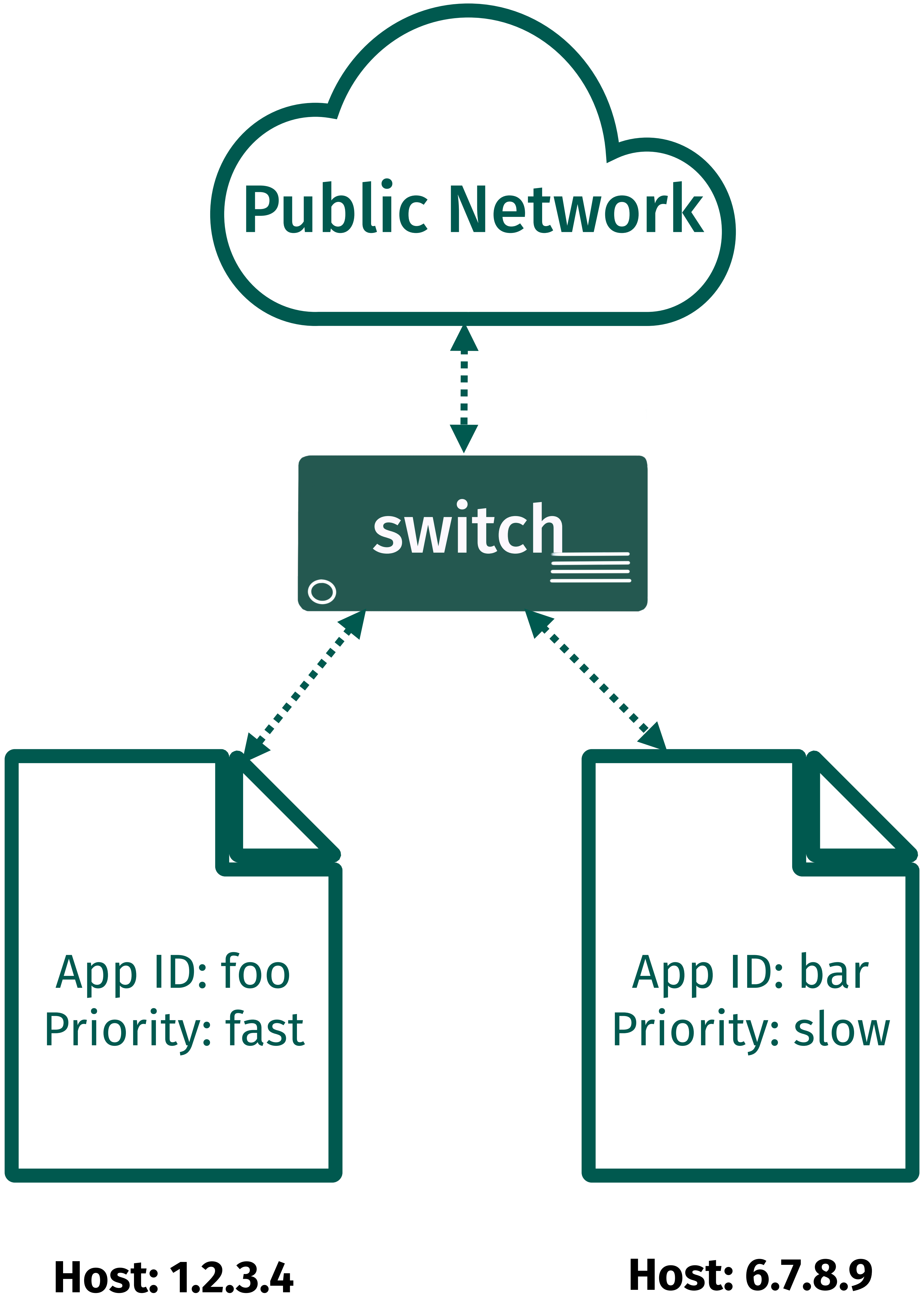
```
table match_action {
    key = { high_key; }
    actions  = { modify_low_field; }
}
```

|||

```
if (high_key == foo) {
    modify_low_field();
}
else if (high_key == bar) {
    skip;
}
```

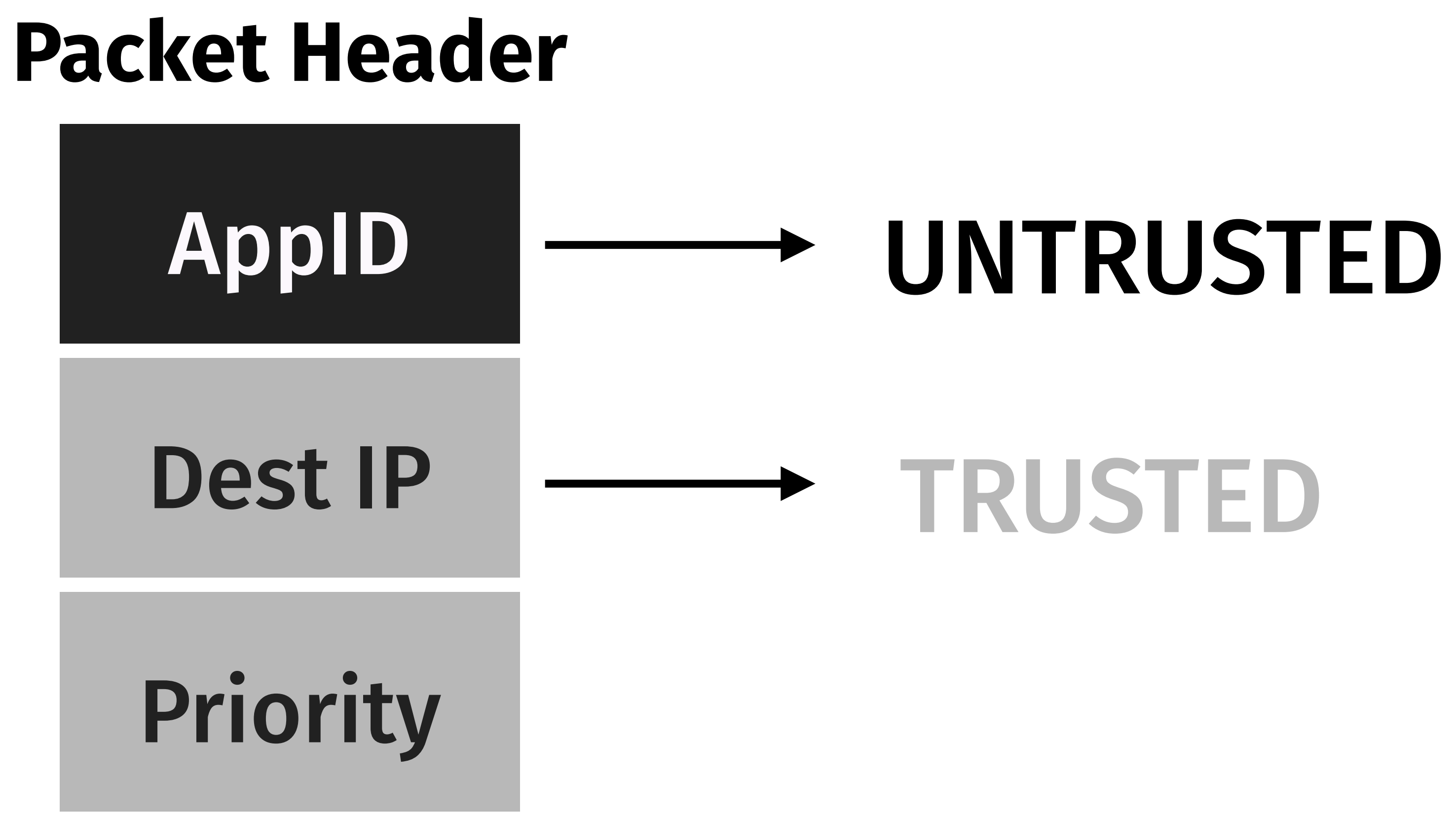
Branch on a HIGH variable

Action writes to a LOW variable



# REVISITING RUNNING EXAMPLE

# SECURITY TYPES



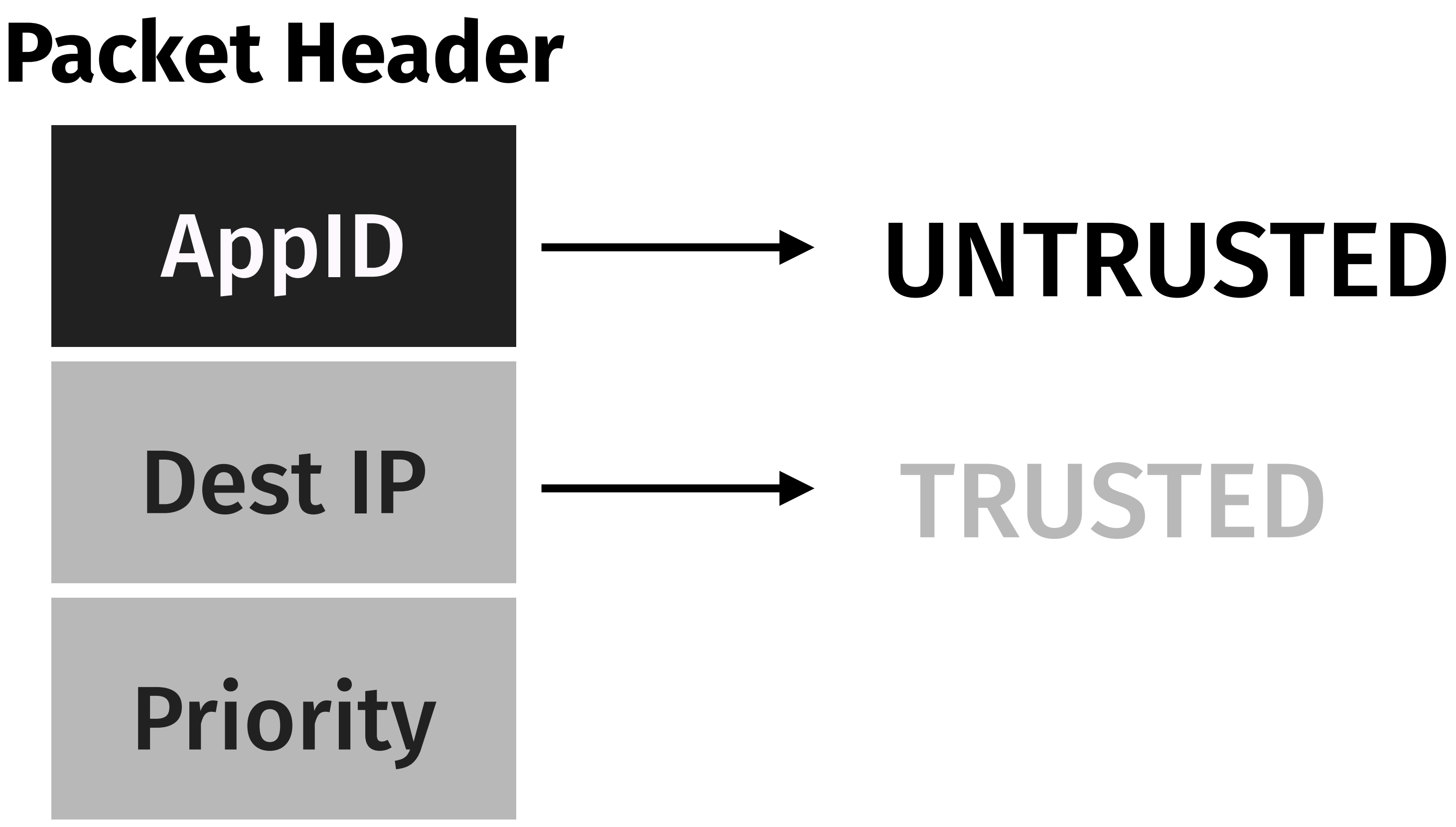
Packet Header

AppID → UNTRUSTED

Dest IP → TRUSTED

Priority

# SECURITY TYPES


Packet Header

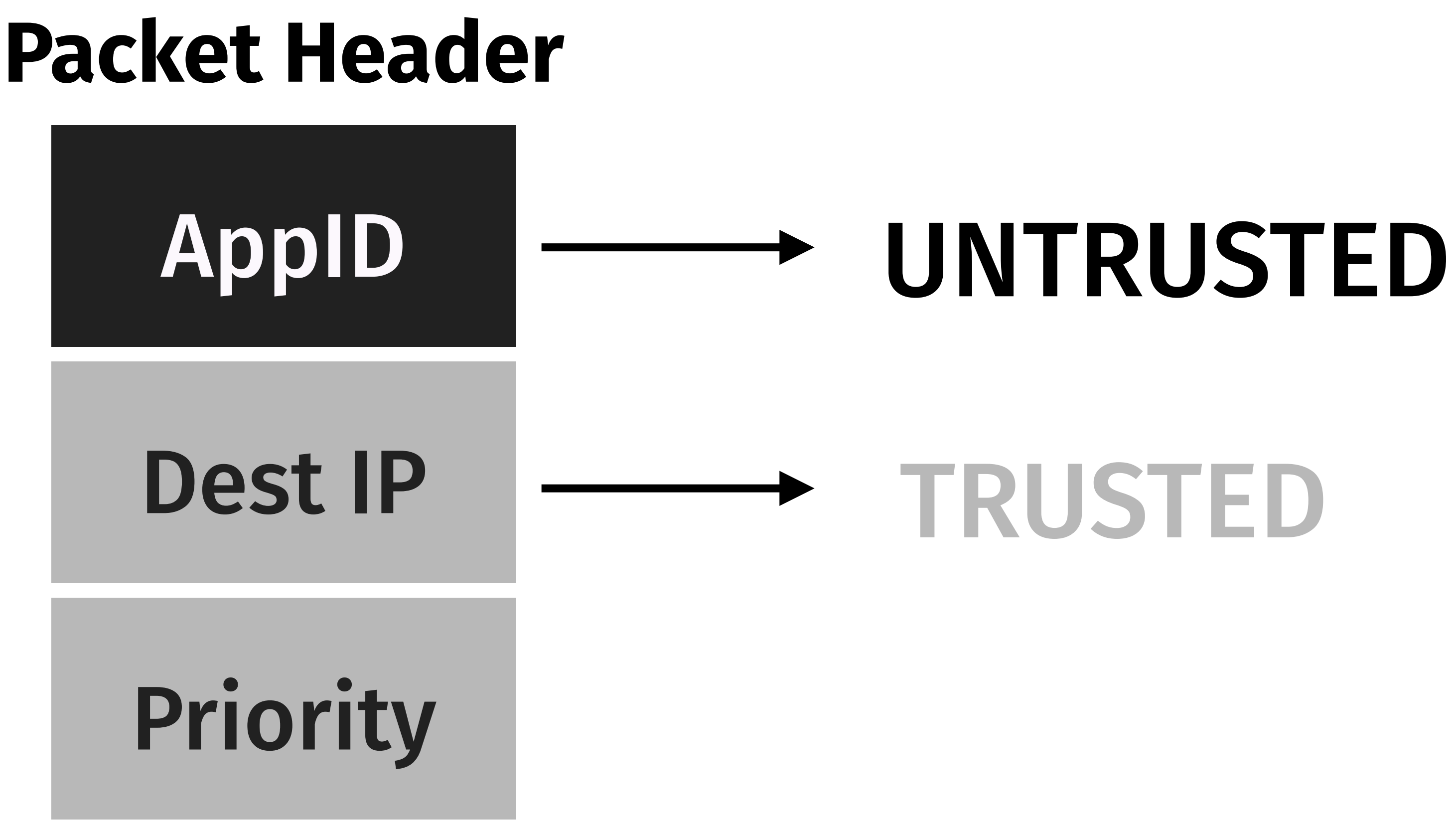
AppID → UNTRUSTED

Dest IP → TRUSTED

Priority

```
AppID:    <AppID_t, UNTRUSTED>,
DestIP:   <DestIP_t, TRUSTED>,
Priority: <Priority_t, TRUSTED>
```

# SECURITY TYPES



```
hdr {
        AppID:    <AppID_t, UNTRUSTED>,
        DestIP:   <DestIP_t, TRUSTED>,
        Priority: <Priority_t, TRUSTED>
    }
```

# BUGGY TABLE

**Table Declaration**

```
table app2priority {
    key = { hdr.appID; }
    actions  = { set_priority; }
}
```

# BUGGY TABLE

## Table Declaration

```
table app2priority {
    key = { hdr.appID; }
    actions  = { set_priority; }
}
```

|||

```
if (hdr.appID == foo) {
    set_priority(prio1);
}
else if (hdr.appID == bar) {
    set_priority(prio2);
}
...
```

# BUGGY TABLE

**Table Declaration**

```
table app2priority {
    key = { hdr.appID; }
    actions  = { set_priority; }
}
```

⦀

```
if (hdr.appID == foo) {
    set_priority(prio1);
}
else if (hdr.appID == bar) {
    set_priority(prio2);
}
...
```

Branch on an UNTRUSTED variable

# BUGGY TABLE

**Table Declaration**

```
table app2priority {
    key = { hdr.appID; }
    actions  = { set_priority; }
}
```

⦀

```
if (hdr.appID == foo) {
    set_priority(prio1);
}
else if (hdr.appID == bar) {
    set_priority(prio2);
}
...
```

Branch on an UNTRUSTED variable

Action writes to a TRUSTED variable

# DETECTING LEAKS IN P4BID

# TYPING JUDGEMENT

$$\Gamma \vdash_{pc} stmt \dashv \Gamma'$$

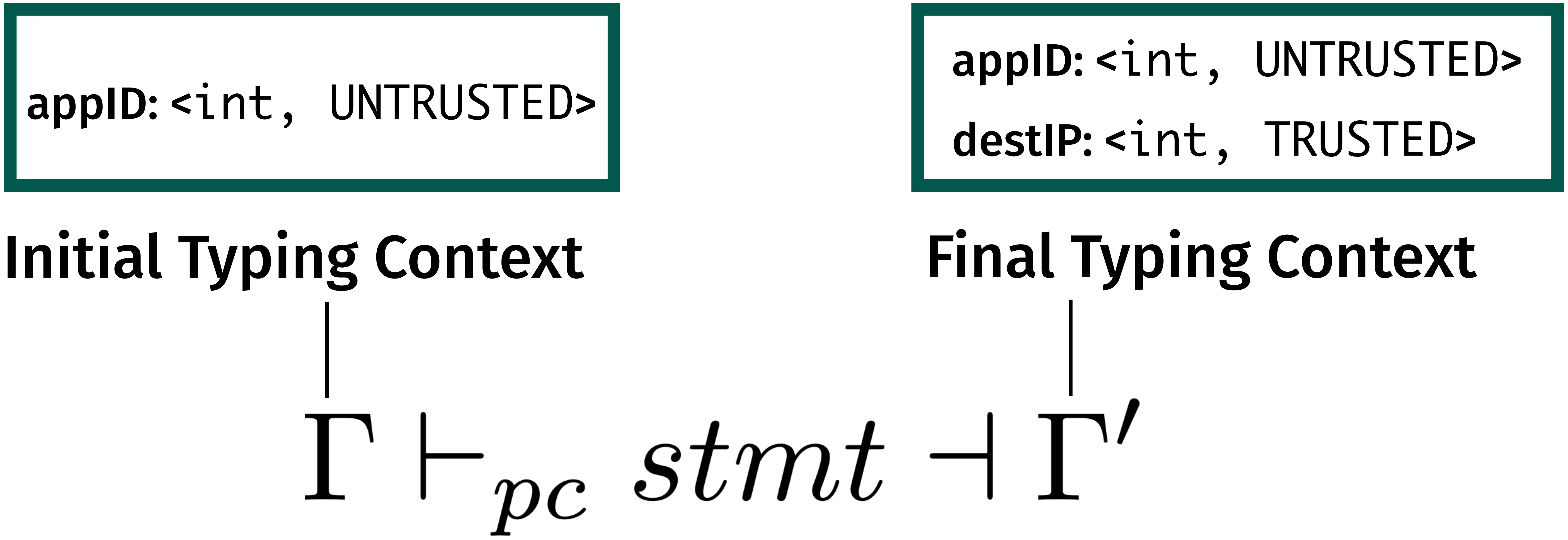
# TYPING JUDGEMENT

**appID:** `<int, UNTRUSTED>`

**Initial Typing Context**

$$\Gamma \vdash_{pc} stmt \dashv \Gamma'$$

# TYPING JUDGEMENT



appID: `<int, UNTRUSTED>`

Initial Typing Context

appID: `<int, UNTRUSTED>`

destIP: `<int, TRUSTED>`

Final Typing Context

$$\Gamma \vdash_{pc} stmt \dashv \Gamma'$$

# TYPING JUDGEMENT

**appID:** `<int, UNTRUSTED>`

**Initial Typing Context**

**appID:** `<int, UNTRUSTED>`

**destIP:** `<int, TRUSTED>`

**Final Typing Context**

$$\Gamma \vdash_{pc} stmt \dashv \Gamma'$$

**no writes to variables below PC**

# NON-INTERFERENCE THEOREM

Suppose

$$\Gamma \vdash_{pc} stmt \dashv \Gamma'$$

# NON-INTERFERENCE THEOREM

Suppose

$$\Gamma \vdash_{pc} stmt \dashv \Gamma'$$

then

stmt is **non-interfering**, i.e, no **High** to **Low** information flow

# LEAKY TABLE

```
table app2priority {
   key = { hdr.appID; }
   actions  = { set_priority; }
}
```

$$\Gamma \vdash_{LOW} \text{app2priority.apply()} \dashv \Gamma'$$

**NOT provable!!**

# LEAKY TABLE

```
table app2priority {
   key = ~~{ hdr.appID; }~~
   actions   = { set_priority; }
}
```

$$\Gamma \vdash_{LOW} \text{app2priority.apply()} \dashv \Gamma'$$

**NOT provable!!**

# LEAKY TABLE

```
table app2priority {
   key = ~~{ hdr.appID; }~~   hdr.destIP
   actions   = { set_priority; }
}
```

$$\Gamma \vdash_{LOW} \text{app2priority.apply()} \dashv \Gamma'$$

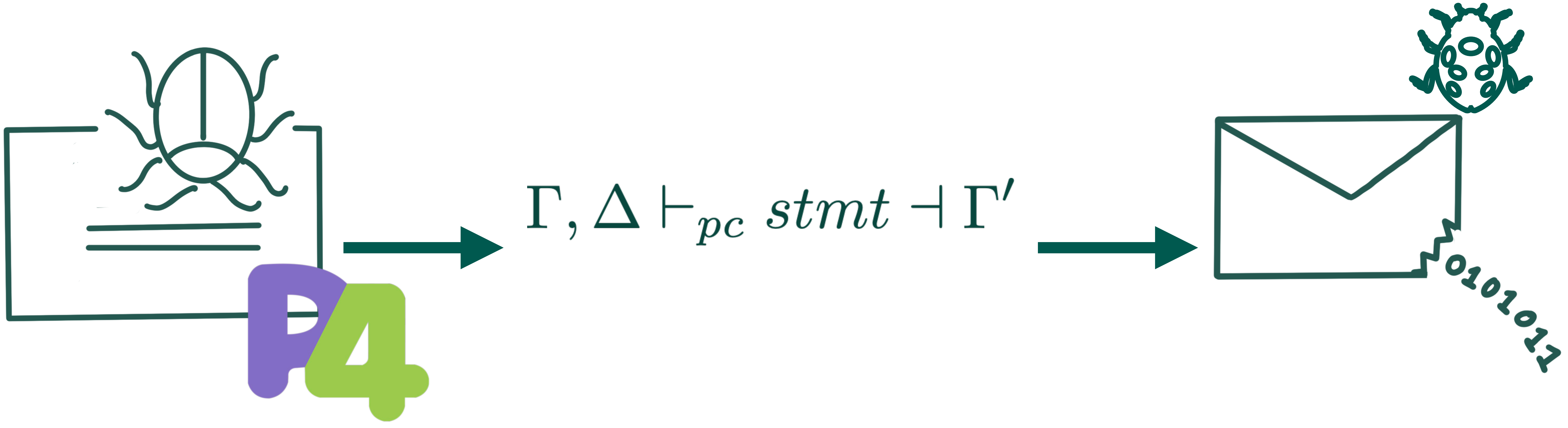
**NOT provable!!**

# LEAKY TABLE

```
table app2priority {
    key = ~~{ hdr.appID; }~~   hdr.destIP
    actions   = { set_priority; }
}
```

$$\Gamma \vdash_{LOW} \text{app2priority.apply()} \dashv \Gamma'$$

**Provable**



$$\Gamma, \Delta \vdash_{pc} stmt \dashv \Gamma'$$

**P4 Program** **IFC Type System** **Catches IFC bugs**

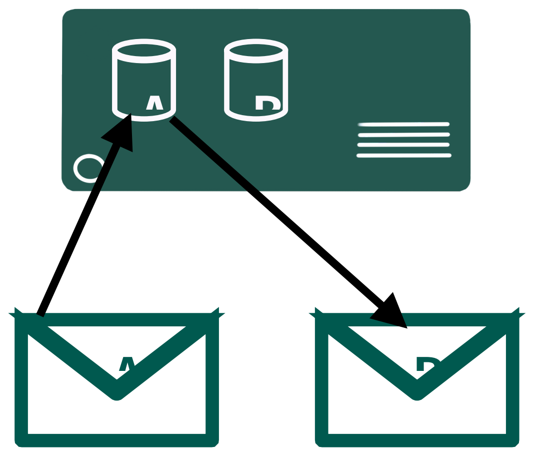
## SEE THE PAPER FOR MORE ON...

Full type system
Non-interference theorem and proof
Several network scenarios using IFC, including isolation

[2204.03113] P4BID: Information Flow Control in P4
arxiv.org

## OPEN QUESTIONS ...

Recirculation



Inter-Packet Leaks