# Principled Flow Tracking in IoT and Low-Level Applications

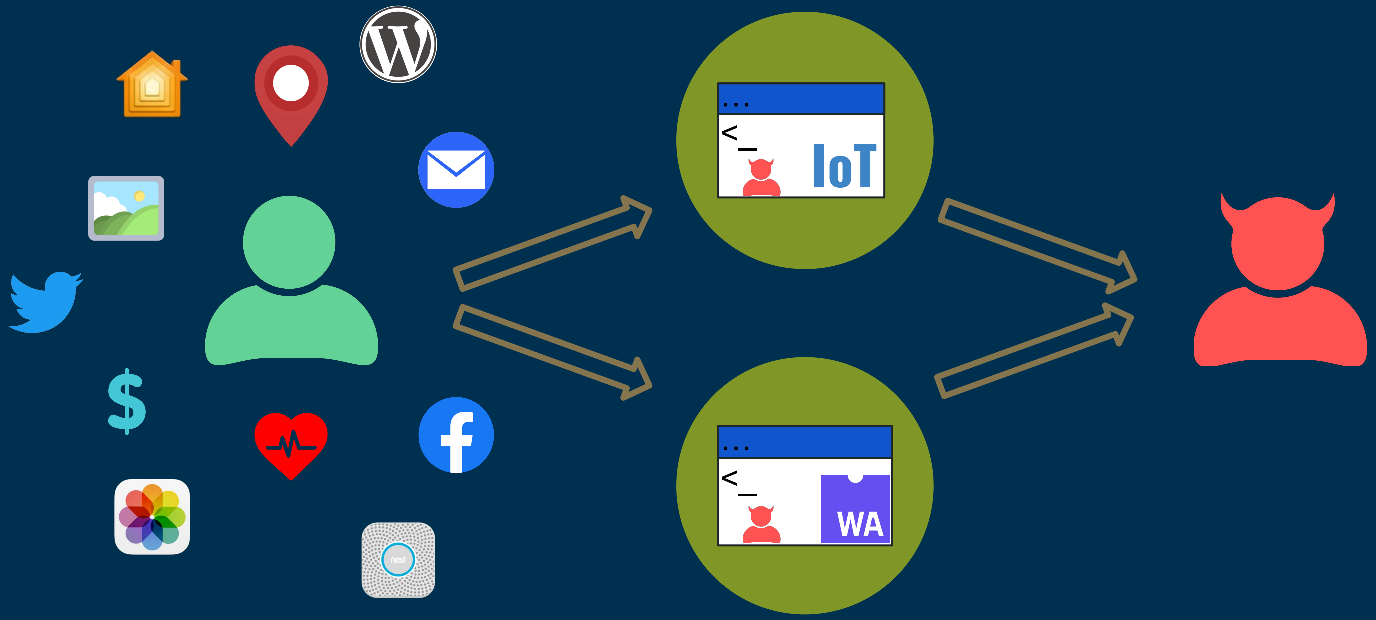
Iulia Bastys

February 8th, 2022

CHALMERS

WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

# Motivation



# IoT apps

## Connecting otherwise unconnected devices and services

event

**if + then +**

event

IFTTT

zapier

Power Automate

# IoT apps

CHALMERS

## Connecting otherwise unconnected devices and services

action

if then

trigger

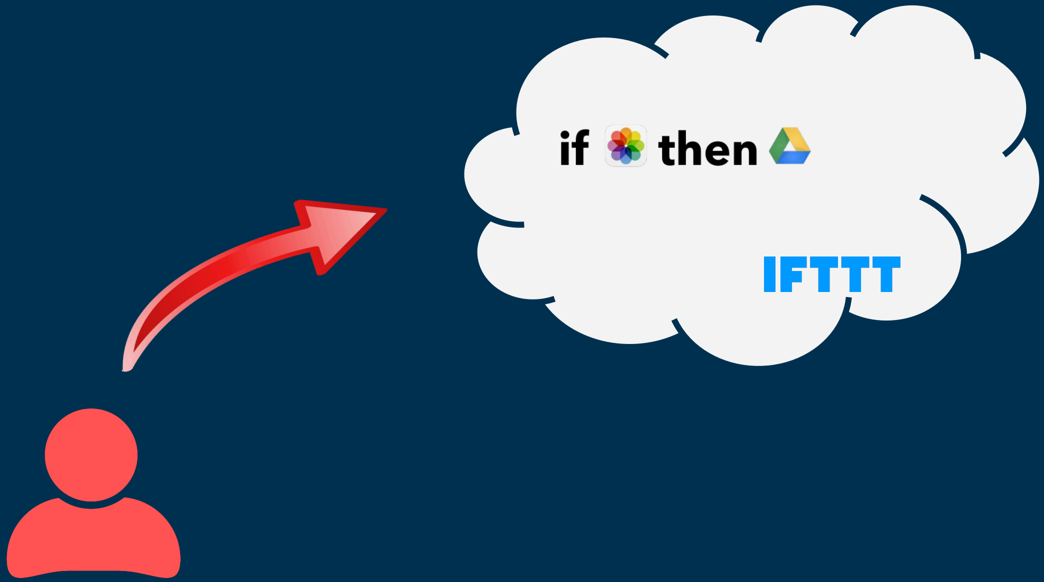
IFTTT

zapier

Power Automate

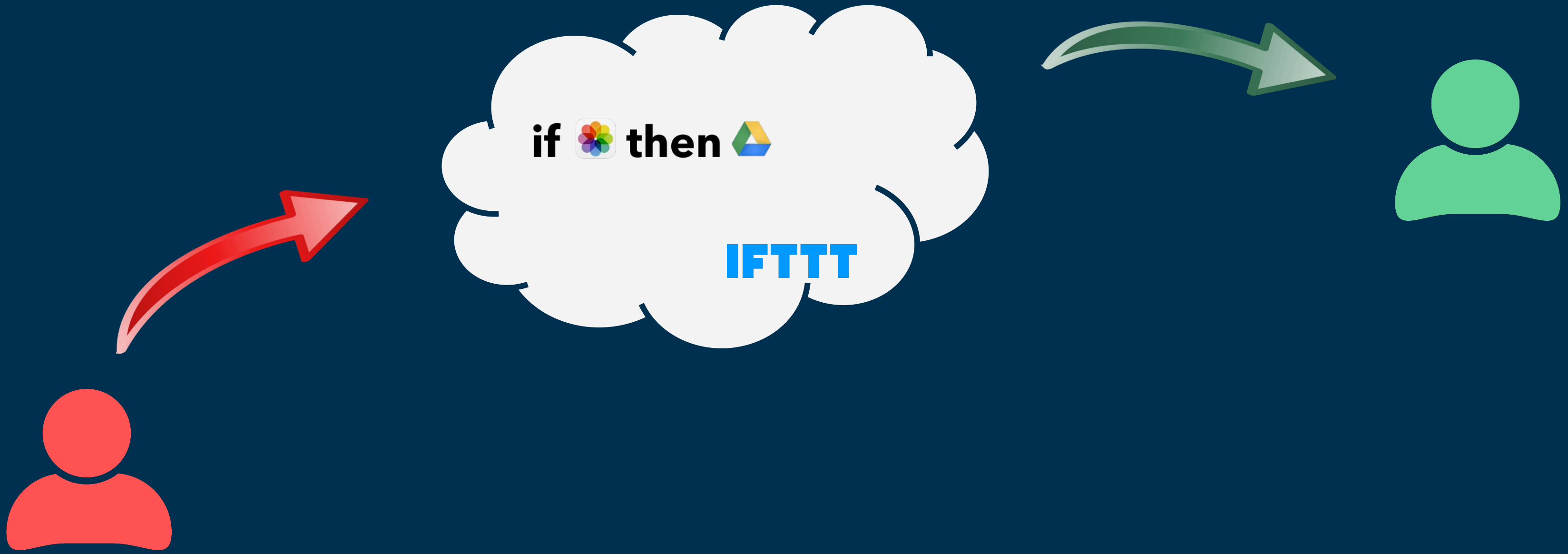
# IoT apps

## 3rd party user publishes an app



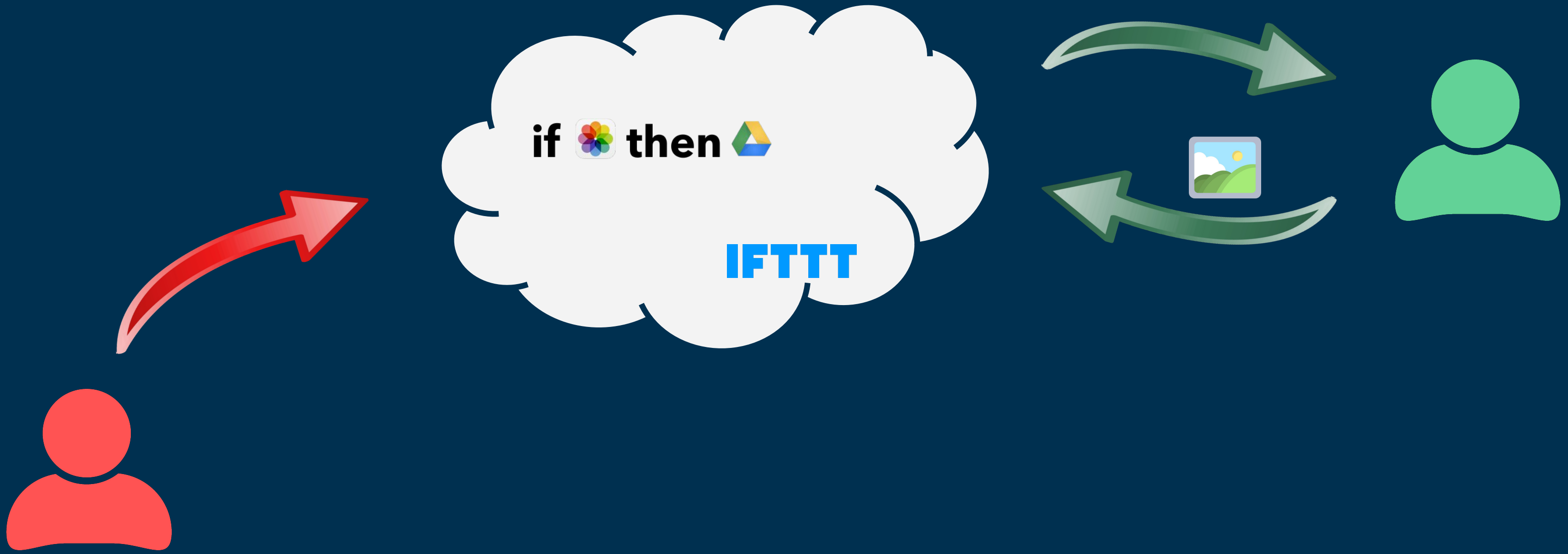
# IoT apps

## User installs the app



# IoT apps

## User takes a photo



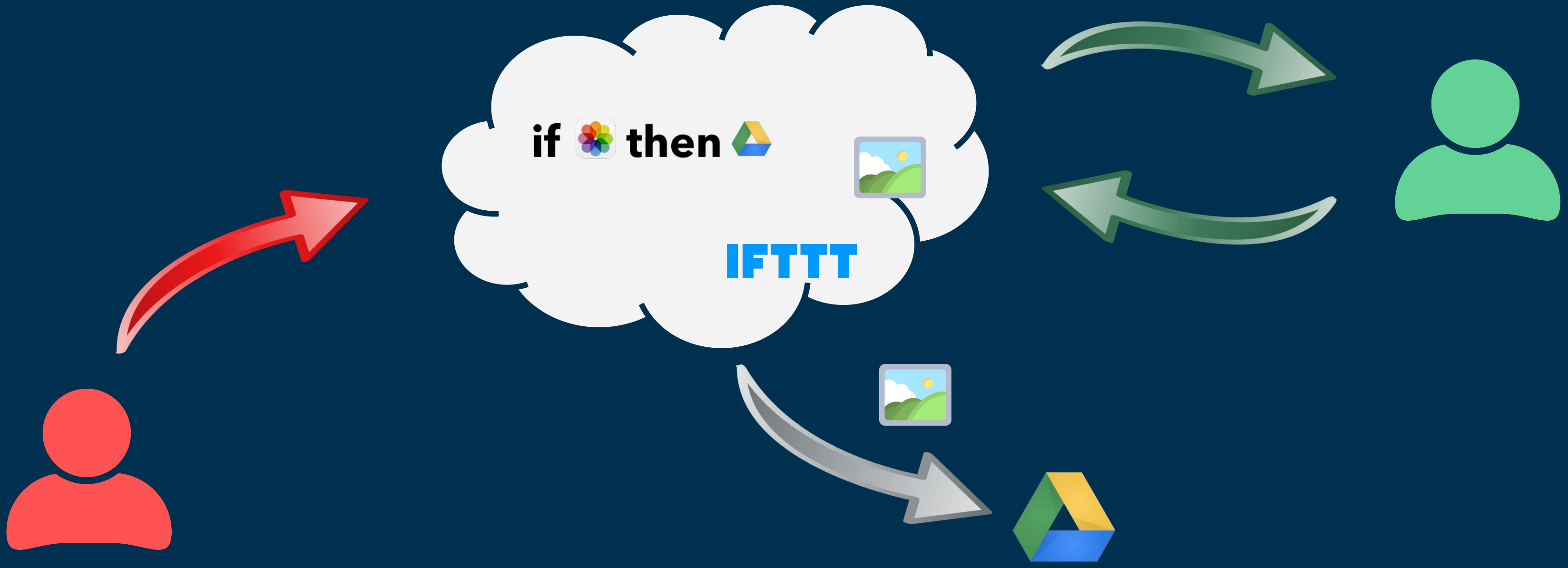
# IoT apps

## Photo is sent to IFTTT

# IoT apps

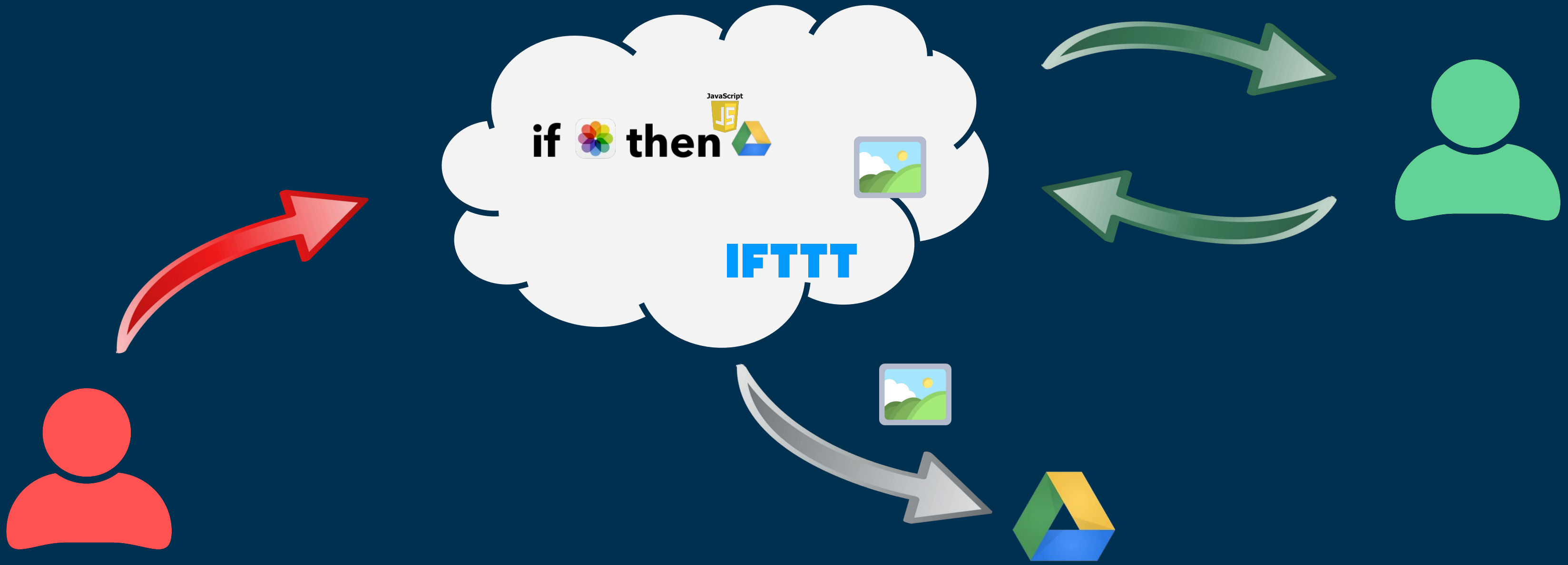
CHALMERS

## Photo is backed up on Google Drive, as expected



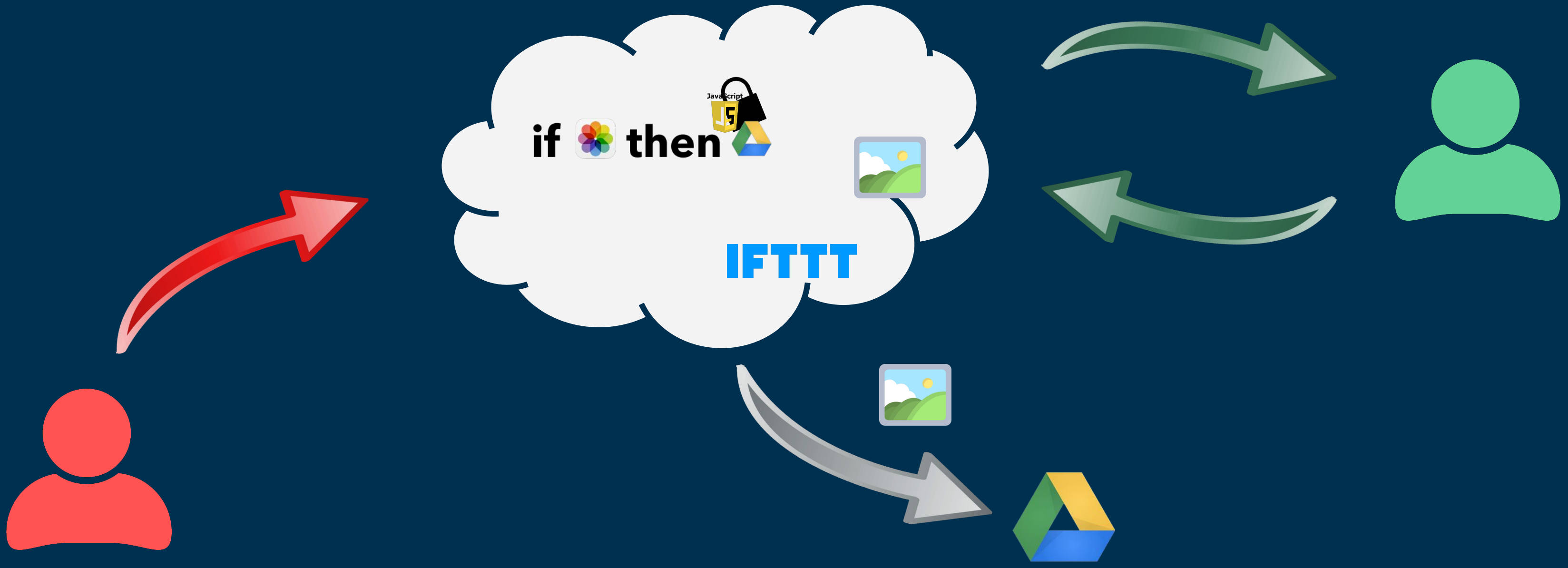
# IoT apps

## The app may execute JavaScript, invisible to the user



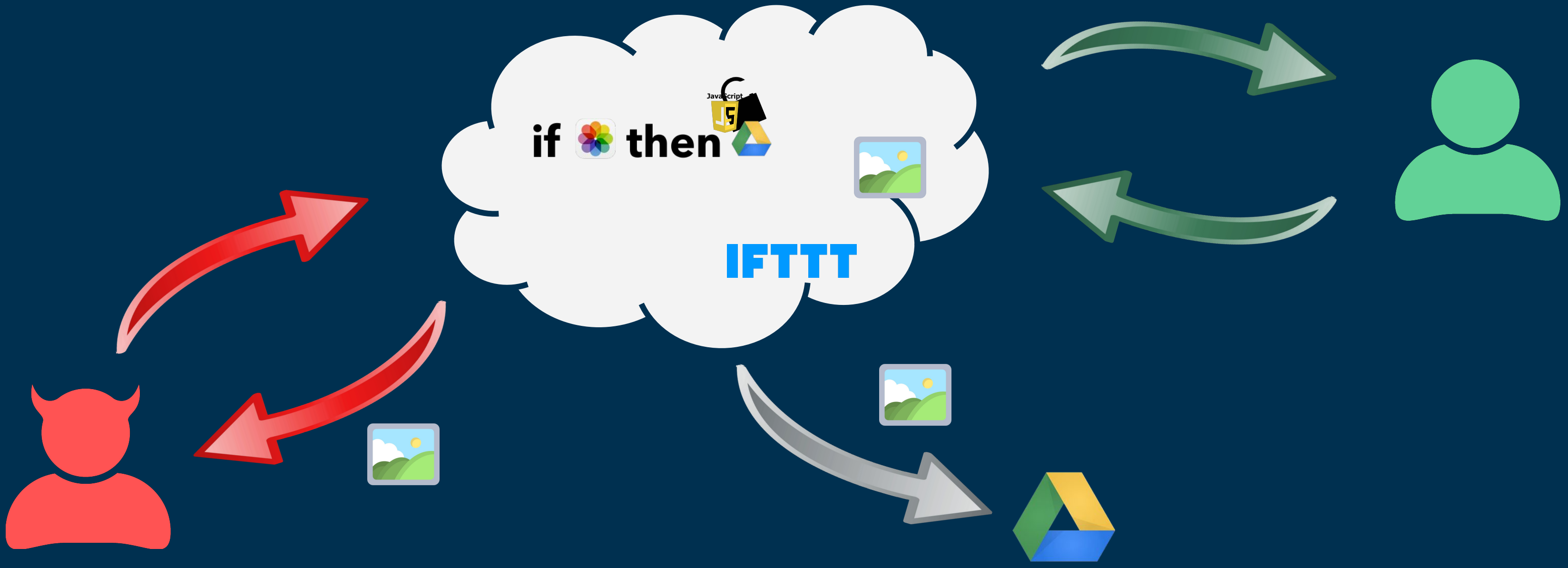
# IoT apps

## JavaScript sandboxed



# IoT apps

## Sandboxing mechanism evaded

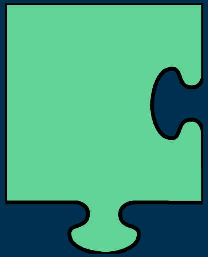


# WebAssembly (Wasm)

- low-level programming language
- portable and fast
- high-performance web-applications

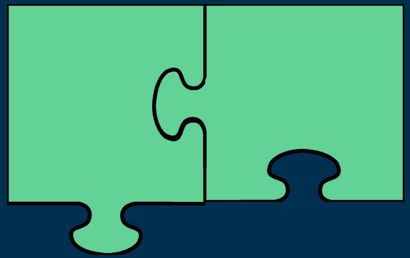
# WebAssembly apps

Memory safe and sandboxed execution environment



# WebAssembly apps

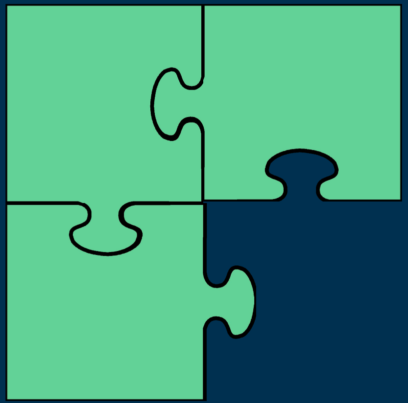
Memory safe and sandboxed execution environment



Separate memory and code space

# WebAssembly apps

Memory safe and sandboxed execution environment



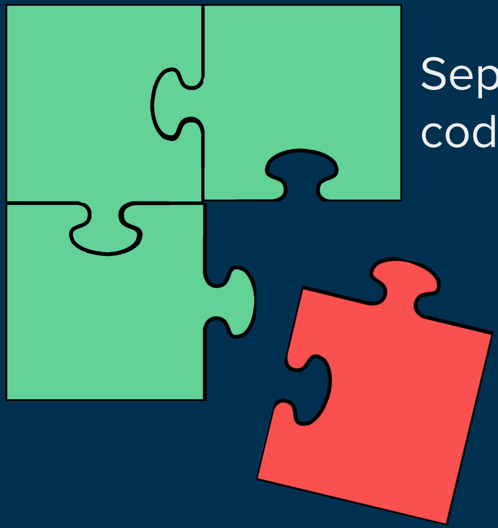
Separate memory and code space

Structured control flow

# WebAssembly apps

Memory safe and sandboxed execution environment

Separate memory and code space



Structured control flow

No guarantees for information flows

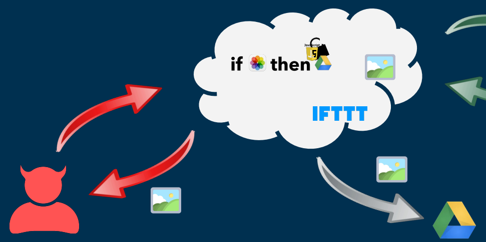
# Current security guarantees



## IoT apps

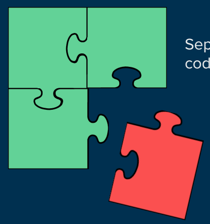
**inefficient**

Sandboxing mechanism **evaded**




## WebAssembly apps

**insufficient**

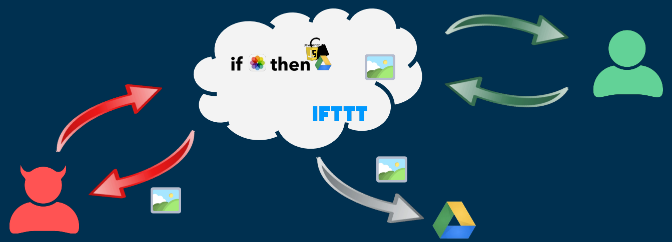


# Current security guarantees

IoT apps

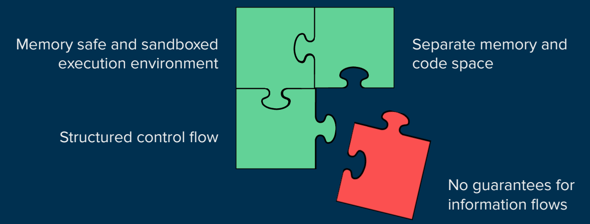
inefficient

Sandboxing mechanism evaded



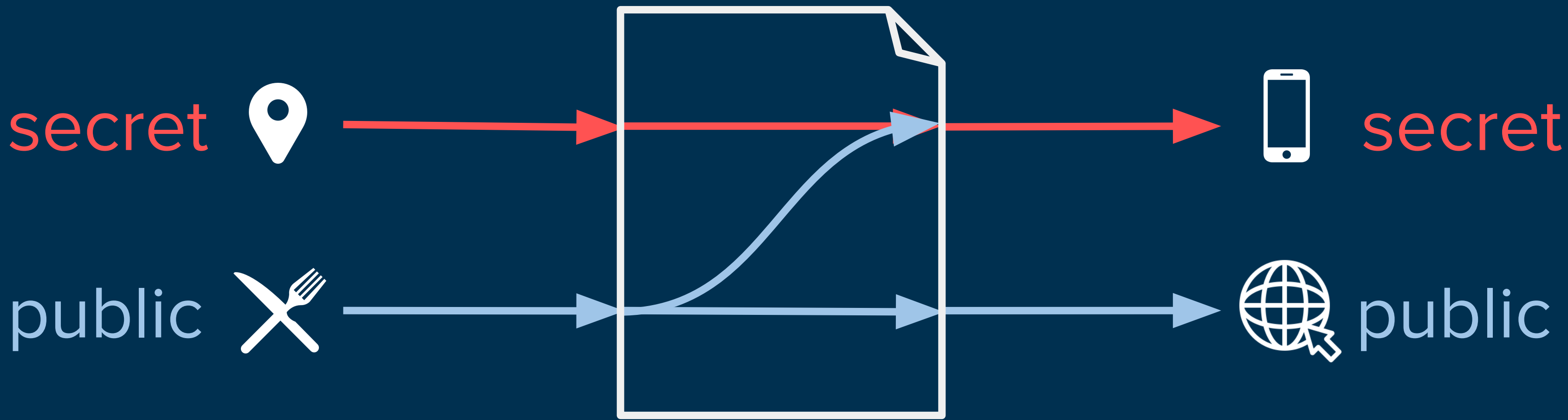

WebAssembly apps

insufficient




- Information flow control (IFC)
  - formal security guarantees

# Noninterference



# Tracking flows

$x^{public} := y^{secret}$

```
if (y^secret) then
    x^public := true
else
    x^public := false
```

**Explicit flows**

**Implicit flows**

# Enforcement mechanisms

## Static

$$\Gamma \vdash c : \tau$$

security type

program

security context

# Enforcement mechanisms

CHALMERS

## Static

$\Gamma \vdash c : \tau$

## Dynamic

$(c, st, S)$

program state

security state

# Enforcement mechanisms

CHALMERS

## Static

$\Gamma \vdash c : \tau$

## Dynamic

$(c, st, S) \longrightarrow (c', st', S')$

# Enforcement mechanisms

CHALMERS

## Static

$\Gamma \vdash c : \tau$

## Dynamic

$(c, st, S) \xrightarrow{e} (c', st', S')$

attacker observation

# Enforcement mechanisms

CHALMERS

## Static

$\Gamma \vdash c : \tau$

## Dynamic

$(c, st, S) \xrightarrow{\not\epsilon} \lightning$

# Enforcement mechanisms

CHALMERS

## Static

$\Gamma \vdash c : \tau$

## Dynamic

$(c, st, S) \xrightarrow{\cancel{e}} \lightning$

## Hybrid

$\Gamma \vdash c : \tau$

$(c, st, S) \xrightarrow{e}_{\tau} (c', st', S')$

# Challenges

CHALMERS

Securing Wasm apps

New security characterization and enforcement mechanism

Securing IoT apps

New security characterization and enforcement mechanism

?

# Challenges



Securing Wasm apps

New security characterization and enforcement mechanism

Securing IoT apps

New security characterization and enforcement mechanism

?

Design principles

Security characterization and enforcement for new applications

# Challenges

CHALMERS

Securing Wasm apps

New security characterization and enforcement mechanism

Securing IoT apps

New security characterization and enforcement mechanism

?

Design principles

Security characterization and enforcement for new applications

Enforcement granularity

Expressiveness relation between enforcement mechanisms

# Challenges


New security characterization and enforcement mechanism

New security characterization and enforcement mechanism

Security characterization and enforcement for new applications

Towards widespread adoption of IFC techniques

Expressiveness relation between enforcement mechanisms

# Thesis structure


- Securing IoT apps
- Securing Wasm apps
- Design principles
- Enforcement granularity
- Automatic labeling

# Thesis structure


- Securing IoT apps
- Securing Wasm apps
- Design principles
- Enforcement granularity
- Automatic labeling

# If This Then What? Controlling Flows in IoT Apps

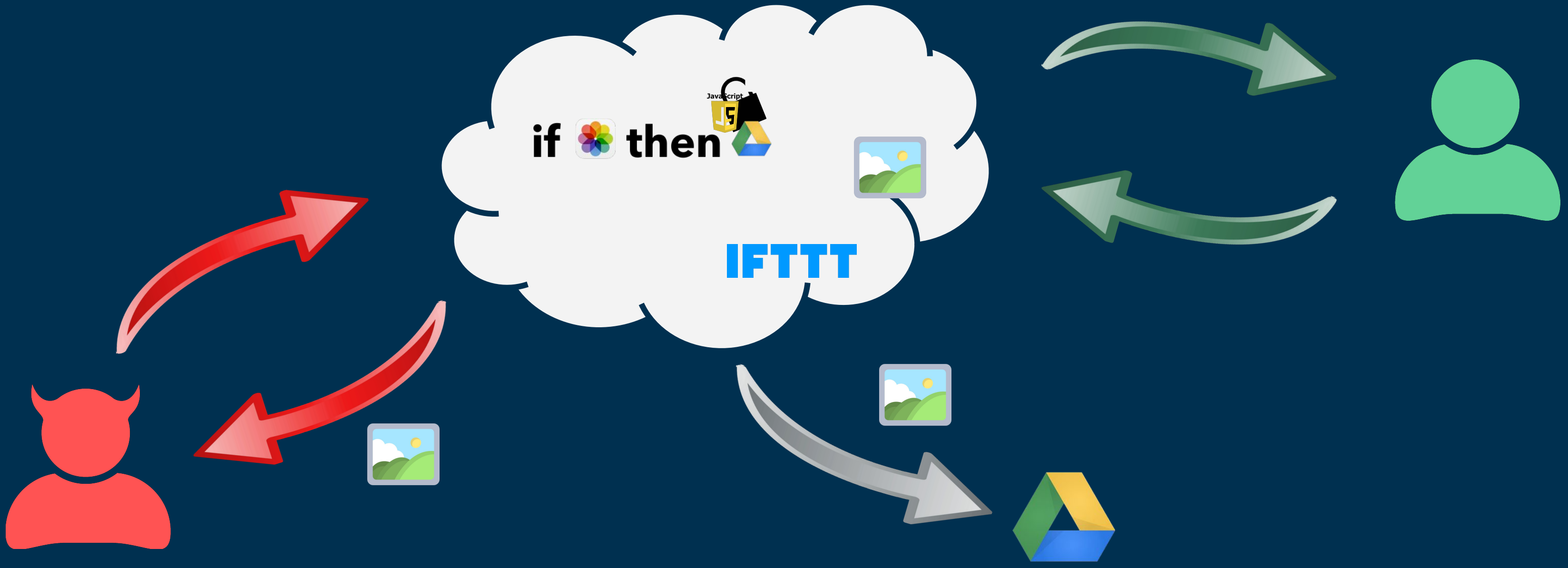
*CCS 2018*

Iulia Bastys, Musard Balliu, Andrei Sabelfeld

- IoT apps recap
- URL-based attacks
- Projected security
- FlowIT

# IoT apps recap

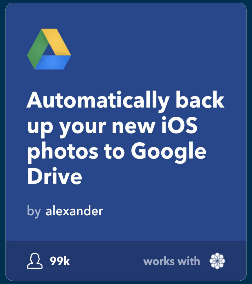
## Sandboxing mechanism evaded



# URL-based attacks

## URL upload attack

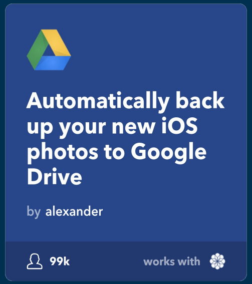
```
GoogleDrive.uploadFileFromUrlGoogleDrive.setURL(...)
```



# URL-based attacks

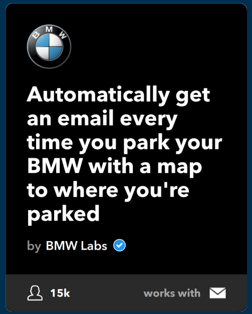
## URL upload attack

```
GoogleDrive.uploadFileFromUrlGoogleDrive.setURL(...)
```




## URL markup attack

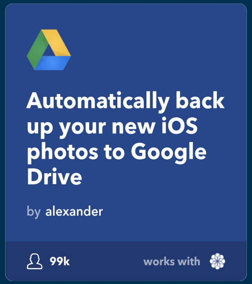
```
Email.sendMeEmail.setBody(...)
```



# URL-based attacks

## URL upload attack

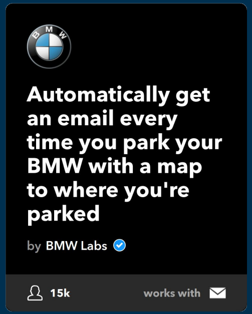
```
GoogleDrive.uploadFileFromUrlGoogleDrive.setURL(...)
```




## URL markup attack

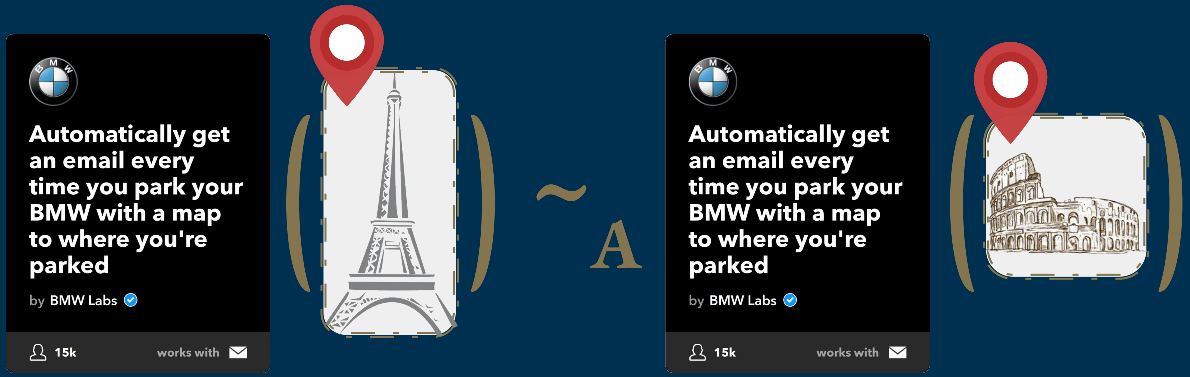
```
https://attacker.com?secret
```

```
Email.sendMeEmail.setBody(...)
```



# Projected security (PS)

## Attacker's observations on the sink are the same




```
www.attacker.com?loc|A = [www.attacker.com?loc]
www.ifttt.com/logo.png|A = []
```

# FlowIT

- Dynamic monitor for PS

  $\langle c,m,S,\Gamma\rangle \xrightarrow[pc\ n]{} \langle c',m',S',\Gamma'\rangle$

- JSFlow-based implementation

- Evaluation on 60 apps (30 secure and 30 insecure)
  - No false negatives
  - Single false positive (on "artificial" code)

# Clockwork: Tracking Remote Timing Attacks

*CSF 2020*

Iulia Bastys, Musard Balliu, Tamara Rezk, Andrei Sabelfeld

- Remote timing attacks
- Remote secure programs
- Clockwork

# Remote timing attacks

## clock, branch, I/O

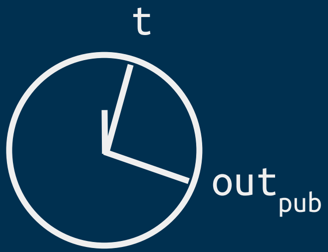
$t =$ 

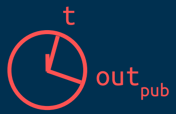
`if secret then { ... }`

$out_{pub}(t)$

`secret = false`


`secret = true`




Remote attacker observation: `secret = true` if 

# Remote timing attacks

Clockwork

## I/O, branch, I/O

```
outpub(
if sec
outpub(
```

## cache

```
if secre
outpub(1)
h1 = h2
outpub(2)
```

## high delay

```
t =
if h % 2 = seconds(t) % 2 then h = h
else h = h; … ; h = h
outpub(1)
```

# Constant-time security

Clockwork

- popular in cryptographic implementations (e.g. AES, DES, SHA256, RSA)
- no branching on secret data
- useful for **local** attacker models
- too restrictive for **remote** attacker models

# Constant-time insecure programs

Clockwork

### branch, I/O

```
if secret then { ... }
```

$\texttt{out}_{\texttt{pub}}\texttt{(1)}$

### I/O, I/O, branch

$\texttt{out}_{\texttt{pub}}\texttt{(1)}$

$\texttt{out}_{\texttt{pub}}\texttt{(2)}$

```
if secret then { ... }
```

# Remote secure programs

D Clockwork

**branch, I/O**

```
if secret then { ... }
```

$\texttt{out}_{\texttt{pub}}\texttt{(1)}$

**I/O, I/O, branch**

$\texttt{out}_{\texttt{pub}}\texttt{(1)}$

$\texttt{out}_{\texttt{pub}}\texttt{(2)}$

```
if secret then { ... }
```

Remote attacker observation: `secret` $\in$ `{true, false}`

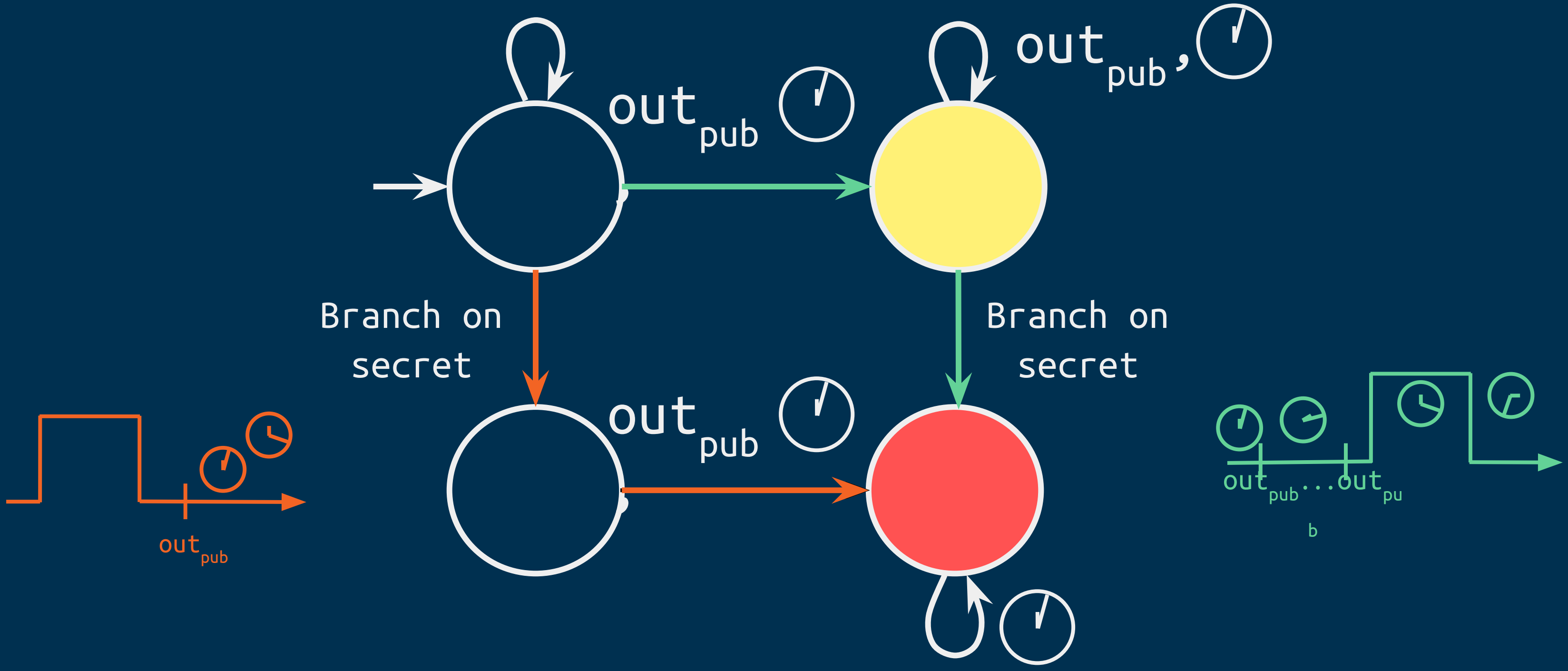
# Patterns of remote secure programs



**one public output** after
branching on secret
if no prior clock read OR
public outputs

**any public outputs** before
branching on secret;
unrestricted clock reads

# Clockwork


out pub
out pub ,
Branch on secret
Branch on secret
out pub
out pub
out pub ... out pu b

# Clockwork

Clockwork

- Dynamic monitor for RS
- JSFlow-based implementation

JSFlow

- Case studies
  - IFTTT
  - Open Verificatum

# A Principled Approach to Securing WebAssembly

*Manuscript*

Iulia Bastys, Maximilian Algehed, Alexander Sjösten, Andrei Sabelfeld

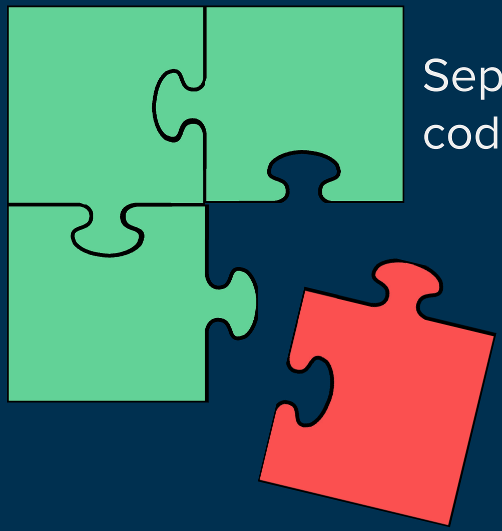
- WebAssembly apps recap
- SecWasm

# WebAssembly apps recap

Memory safe and sandboxed execution environment

Separate memory and code space

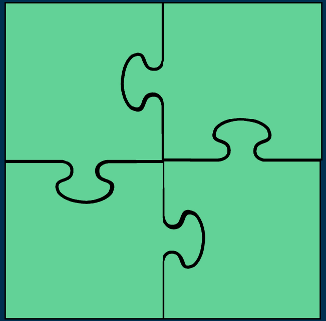
Structured control flow



No guarantees for information flows

# SecWasm

- Hybrid monitor
  - $\gamma, C \vdash c \dashv \gamma'$
  - $(st, S, c) \Downarrow (st', S', \theta)$



Secure flow of information

# Conclusion