.

connecting other- wise unconnected *trigger* and *action*  services

the Internet of Things (IoT). TAPs empower users by seamlessly

*Trigger-Action Platforms (TAPs)* play a vital role in fulfilling the promise of

Introduction

1

Module 9: RED SERVICE

● Create a document around it and share the same for assessment.

you thought of in previous phase into transformation.

● Explain in detail the complete steps that will be taken by you to put your design that

● In this phase you need to put your design into innovation to solve the problem.

**Innovation**

**PHASE-4**

**AQM- AIR QUALITY MONITORING**

NM I’D

**au411521106030**

**COLLEGE NAME**

PERI INSTITUTE OF TECHNOLOGY

**YEAR**

III

**DEPARTMENT**

ECE

**REG. NO.**

411521106029

**NAME**

MALIN.S



platform, although it is also available on the IBM Cloud platform [23].

platform, Node-RED is mainly tar- geted for deployment as a single-user

programming for event- driven applications” [36]. As an open-source

devices, APIs and online services”, which provides a way of “low-code

Node-RED is “a programming tool for wiring together hardware

Node-RED Vulnerabilities

its soundness and transparency (Section 3).

framework to express and enforce fine-grained security policies, proving

be they benign, vulnerable, or malicious; and present a mon- itoring

essential model for Node-RED, suitable to reason about nodes and flows,

and app-level vulnerabilities and attacks (Sec- tion 2); propose an

the following, we discuss Node-RED along with overviewing platform-

grained allowlist policies at module-, API-, value-, and context-level. In

transparent moni- toring framework for Node-RED for enforcing fine-

understanding how to monitor Node-RED apps. We present a sound and

Motivated by SandTrap, this work is a step toward formally

Node

ﬂow

Node

context

Node

Node

message

global

Flow

Flow

Node.js

concerns.

run third-party code, TAPs are subject to critical security and privacy

action services. While greatly benefiting from the possibility of apps to

A TAP is effectively a “person-in-the-middle” between trigger and

Malicious app maker

App

Trigger

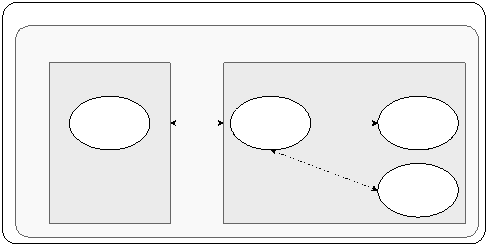
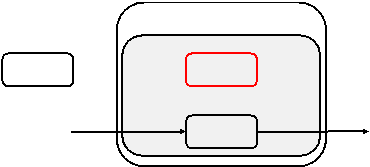
Action

Action

Trigger

App

TAP



},

...

// other

parameters

],

[ NODE2 , NODE3 ] // second output port to nodes 2 and 3

[ NODE1 ],

// first output port to node 1

array of array of strings

function

// type of node , string wires: [

//

id: NODE0 ,

// unique ID of node , string type :

/\* parameters of interest in every node \*/

{

// node 0

[

// list of nodes

third parties.

existing flows pro- vided by the official Node-RED catalog [33] and by

either create their own flows on the platform’s environment or deploy

A flow is a representation of nodes connected together. End users can

nodes.

sharing configuration data, such as login credentials, between multiple

node without sources and sinks, called *configuration* node, is used for

illustrates the code structure of a Node-RED node. A special type of

which can be potentially multiple, unlike the input port. Figure 3

results of (side-effectful) computations on output ports (dubbed *sinks*),

messages on at most one input port (dubbed *source*) and sending the

A node is a reactive Node.js application triggered by receiving

Node-RED platform

}

RED . nodes. registerType (" type - name ", Node Name );

}

});

outputs

node . send ( msg); // or an array of messages

for

multiple

... // functionality of node

function ( msg){

// register a callback when a message is received ... node . on(" input",

= this;

RED . nodes. create Node ( this , config ); var node

Node Name ( config ){

module . exports = function ( RED ){ function

sce- narios for malicious nodes [3]. At the Node.js level, an attacker can

messages within a flow. As shown in Figure 6a, there are various attack

Node.js, are accessible for node developers, as well as the incom- ing

All APIs provided by the underlying runtimes, Node-RED and

Platform-level isolation vulnerabilities

levels. These policies will further be formalized in Section 3.

that mo- tivate enriching the policy mechanism with different granularity

In the following, we discuss Node-RED attacks and vulnerabilities

express fine-grained app-specific security goals.

baseline and advanced policies customized by developers or users to

mechanism to aid developers in designing the policies, enabling both

before deploying the node. SandTrap [3] offers a pol- icy generation

The provided policies can later be vetted by the platform and the user,

Node

Node

Malicious

ﬂow

Node

Node

Node

Malicious

Malicious

message

message

global

global

Flow

Flow

Flow

object

Node-RED

Node.js

module

Node.js

(a)

**(b)**

defines three scope levels for the contexts

is not visible to the user via the graphical interface of a flow. Node-RED

flows that are possible in the flow. It introduces an implicit channel that

interface reflects only a part of the information

through a flow [40]. Therefore the node wiring visible in the user

between different nodes without using the explicit messages that pass

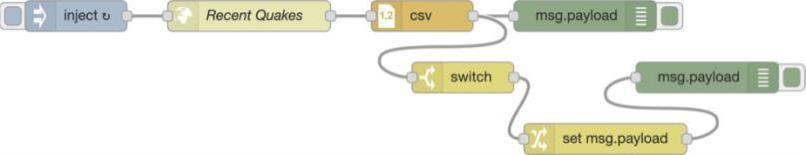
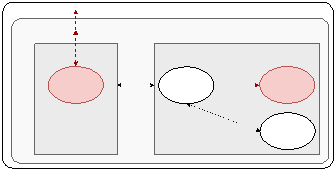
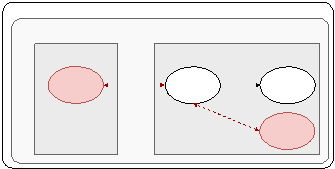
In Node-RED, *contexts* provide a shared communication channel

Figure 4: Node-RED flow structure.

]

...

// other nodes



*k*

*k*

updates the memory  *M*

to  *M*

′

trace of events produced by variable reads and writes. An assignment

accordingly. Assignments operate in a similar manner and record the

channel *I* by consuming the message and updating the memory

handler. The handler executes whenever there is a message in the input

*k*

*M* , *I* , *O*

. Command evaluation models the execution of a node’s

*k*

pro- duced during the evaluation, where  *M*

denotes the memory *M* in *c*,

⟨

Expression evaluation is standard and records the sequence of events

an output message.

messages to execute the code of an event handler and potentially produce

nodes and flows. Nodes are reactive programs triggered by input

**Syntax** We define a core language to capture the reactive nature of

Language syntax and semantics

transparency of the monitor.

to the shared context. Our main formal results are the soundness and

nodes at the level of APIs and their values, along with the access rights

platform. We show how to formalize and enforce security policies for

characterizes the essence of a fine-grained access control monitor for the

This section presents a security model for Node-RED apps and

the JavaScript monitor SandTrap [3].

programming model of Node-RED and was the basis when developing

Our runtime framework formalizes the core of the flow-based

Formalization

create a ma- licious Node-RED node including.

(VR), and Digital Futures.

Foundation for Strategic Research (SSF), the Swedish Research Council

*Acknowledgments* This work was partially supported by the Swedish

API-, value-, and context-level policies.

Node-RED applications by local access checks, supporting module-,

how to soundly and transparently enforce global security properties of

Node-RED applications. Our formalization for a core language shows

cipled framework to enforce fine-grained API control for untrusted

be they benign, vulnerable, or malicious. We have formalized a prin-

essential model for Node-RED, suitable to reason about nodes and flows,

need for a security mechanism for Node-RED, we have proposed an

unsuspecting users to taking over the entire platform. Motivated by the

where the impact ranges from massive exfiltration of data from

the recently-discovered critical exploitable vulnerabilities in Node-RED,

source JavaScript- driven trigger-action platform. We have expanded on

We have investigated the security of Node-RED, an open-

Conclusion

modules, functions, and con- texts. Schreckling et al.

our monitor supports both coarse and fine access control granularity of

including tem- poral patterns over sequences of API calls. By contrast,

API functions in Node-RED. Trace expressions allow for rich policies,

monitoring of parametric trace expressions to check the correct usage of

**Node-RED security and modeling** Ancona et al. [5] investigate runtime

IoT app platforms [7,13] for further details.

platforms in general. We refer the reader to surveys on the security of

and modeling, monitor implementation, and securing trigger-action

We discuss the most closely related work on Node-RED security

Related work

respect to the security condition.

node-level monitor and prove its soundness and transparency with

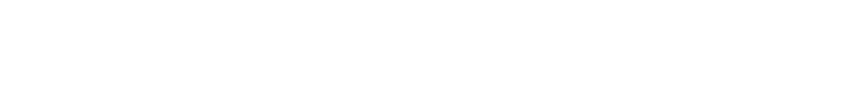
the shared context. Then, we present the semantics of a fine-grained

security policies, represented as allowlists of API calls and accesses to

based security con- dition. The condition is parametric on node-level

We leverage our trace-based semantics to define a semantics-

Security condition and enforcement



**iOS SDK**

Object Pascal

Debugger integrated in Xcode IDE

**iOS SDK**

Objective-C, Swift

Debugger integrated in Xcode IDE

**BlackBerry**

Java

Debugger integrated in IDE

Kotlin

monitor available

**Android**

in C, C++,

Debugger integrated in Eclipse, standalone debugging

Java but portions of code can be

**Platform**

**Programming language**

**Debuggers available**

hardware.

party software that is officially supported for the purpose of developing mobile apps for that

for the design of a particular hardware platform (e.g. Apple, Google, etc) as well as any third-

First party tools include official SDKs published by, or on behalf of, the company responsible

**First-Party**

Notable tools are listed below.

● Cross-platform accommodations/support

● SDKs to access device features

● UI design tools

and provide the following abilities:

Front-end development tools are focused on the user interface and user experience (UI-UX)

**Front-end development tools**

applications for one or more target platforms.

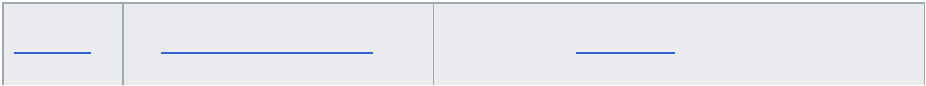
made from many components and tools which allow a developer to write, test, and deploy

The software development packages needed to develop, deploy, and manage mobile apps are

Platform

DEVELOPMENT

Module 10: MOBILE APPLICATION



**Lazarus**

Object Pascal

API

**IBM MobileFirst Studio**

HTML5, CSS3, JavaScript, and native SDK languages w/ Native Worklight

**Smart Devices**

development, then code is automatically generated for each platform

**GeneXus for Mobile and**

Knowledge representation and declarative programming-modeling for easy

**DragonRAD**

Visual drag & drop tiles

**Solar2D**

Lua

**Codename One**

Java

**Basic4android**

Visual Basic similar syntax

**Appcelerator**

JavaScript

**App Inventor for Android** Visual blocks-based programming language, with Interface designer

**MobileTogether**

XPath/XQuery, Action Trees visual programming language

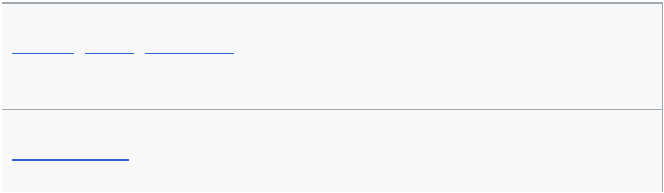
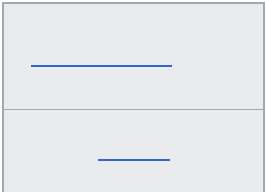
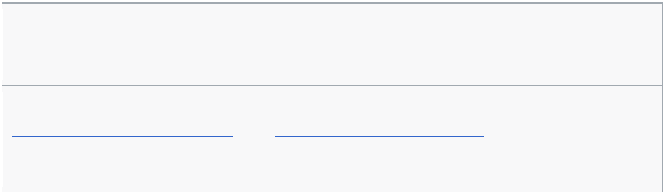
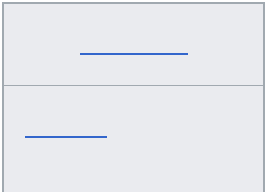
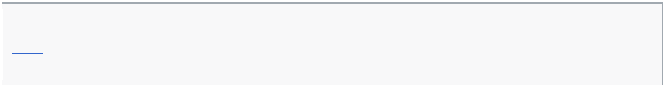
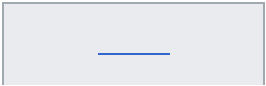
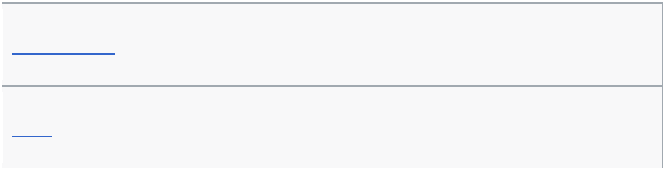
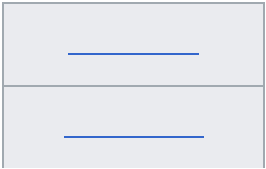
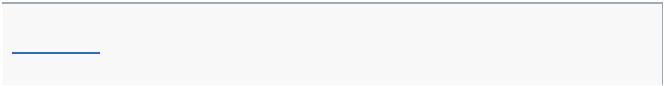
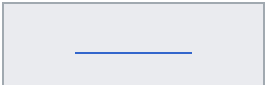
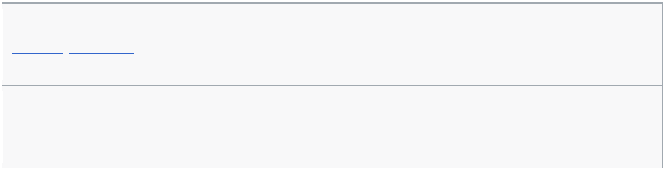
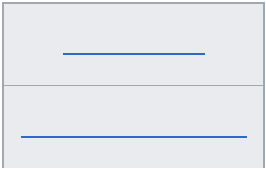
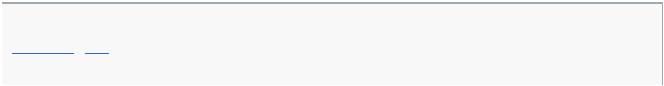
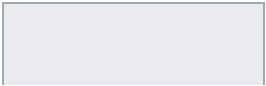
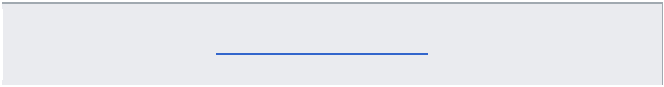
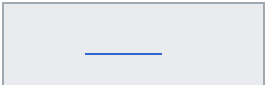
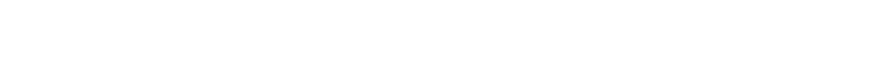
**Accelerator**

HTML5, C#

**Platform**

**Programming language**

**Third Party**



**Ubuntu Touch**

*Web-based*: HTML5, CSS, JavaScript *Native*: QML, C, C++

**Tizen**

*Web-based*: HTML5, CSS, JavaScript *Native*: C, C++

**Symbian**

C++

**Python**

Python

**Palm OS**

C, C++, Pascal

**OpenFL**

Haxe (similar to Actionscript and Java)

**.NET Compact Framework** C#, VB.NET, Basic4ppc

**Firefox OS**

HTML5, CSS, JavaScript

**BREW**

C; the APIs are provided in C with a C++ style interface

**Adobe AIR**

ActionScript, HTML, CSS, JavaScript

**Platform**

**Programming language**

mobile apps.

Many system-level components are needed to have a functioning platform for developing

**System software**

● Reporting and statistics

● Client actions

● Data encryption

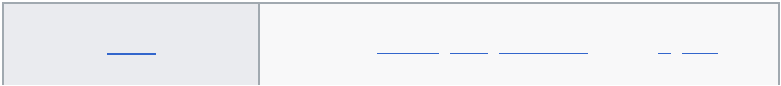
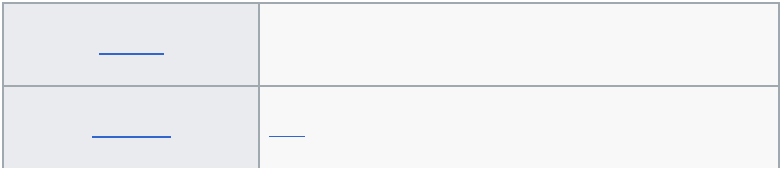
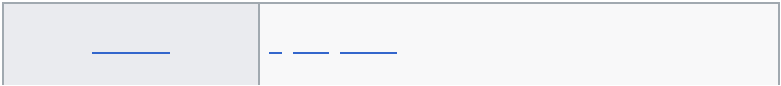
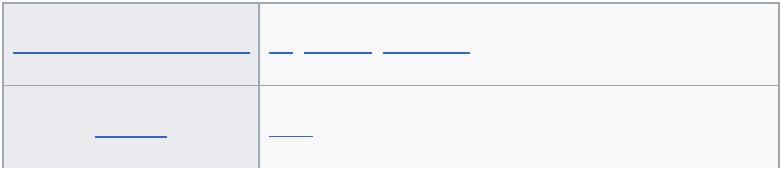
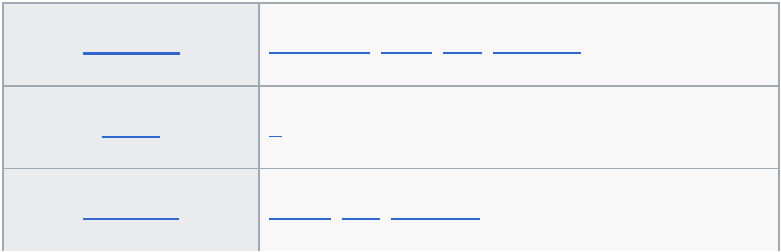
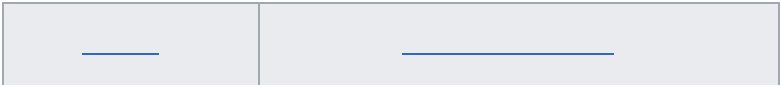
● App wrapping for security

platform component. Features include

departments often need stop-gap, tactical solutions that layer atop existing apps, phones, and

With bring your own device (BYOD) becoming the norm within more enterprises, IT

**Security add-on layers**



● Ranorex: Test automation tools for mobile, web and desktop apps.

and devices.

● eggPlant: A GUI-based automated test tool for mobile app across all operating systems

Developer was officially declared End of Life by the end of 2014.

[8]

applications to devices such as iPhone, BlackBerry, Android, and the Palm Pre. MobiOne

for Windows that helps developers to code, test, debug, package and deploy mobile Web

● **MobiOne** Developer - a mobile Web integrated development environment (IDE)

Windows Server 2008 R2 and later versions of Windows.

technology, it requires Windows Automation API 3.0. It is pre-installed on Windows 7,

● **Windows UI Automation** - To test applications that use the Microsoft UI Automation

screen, keyboard and trackwheel will work with application.

emulate the functionality of actual BlackBerry products and test how the device software,

● **BlackBerry Simulator** - There are a variety of official BlackBerry simulators available to

websites for use with iPhone. iPhoney will only run on OS X 10.4.7 or later.

instead is designed for web developers who want to create 320 by 480 (or 480 by 320)

can be used while developing web sites for the iPhone. It is not an iPhone simulator but

● **iPhoney** - gives a pixel-accurate web browsing environment and it is powered by Safari. It

This tool has been tested and works using Internet Explorer 7, Firefox 2 and Safari 3.

● **TestiPhone** - a web browser-based simulator for quickly testing iPhone web applications.

hardware and software features of a typical mobile device (without the calls).

● **The official Android SDK Emulator** - a mobile device emulator which mimics all of the

it.

complex Android SDK. It can be installed and Android compatible apps can be tested on

as a standalone app, without having to download and install the complete and

● **Google Android Emulator** - an Android emulator that is patched to run on a Windows PC

systems.

examples of tools used for testing application across the most popular mobile operating

mobile phones to which developers may not have physical access. The following are

later subjected to field testing. Emulators provide an inexpensive way to test applications on

Mobile applications are first tested within the development environment using emulators and

**Mobile app testing**

strong correlation between application performance and user satisfaction.

experience. Performance is another important criteria, as research on mobile apps indicates a

cross-platform development it is also important to consider the impact of the tool on the user

existing infrastructure and development skills. When targeting more than one platform with

Criteria for selecting a development platform usually contains the target mobile platforms,

**Windows Phone**

C#, Visual Basic, C, C++

**Windows Mobile**

C, C++

**webOS**

JavaScript, CSS, HTML, C and C++ through the PDK

**Platform**

**Programming language**

