Poster: Attack Surface Modelling in Trigger Action Platforms

Pubali Datta, Adam Bates
Department of Computer Science
University of Illinois at Urbana- Champaign
{pdatta2, batesa}@illinois.edu

I. Problem Description

The use of end-user programming in form of trigger- action rules is becoming increasingly popular with the advancement of Internet of Things (IoT) and smart technologies. Home automation platforms like IFTTT, Zapier, Microsoft Flow [1] are popular among users for their ease of use. These platforms let users create new functions in their homes by stitching different IoT devices and online services in the form of simple rules in natural language, e.g., *"If humidity goes over 55% send notification to open window"*. While these platforms can be used easily, they also pose great security risks through interplay of several rules installed in a smart-home.

Predominantly mentioned platforms are closed where rule behaviors are described in only natural language making it difficult to observe rule interactions leading to security issues. Previous works [2] have focussed on detecting integrity and secrecy violation at rule level and user level with a significant amount of manual work to process natural language rule descriptions. However, the attack surface of rules installed in a smart home using such platforms were not discussed previously. This work aims to process natural language rule descriptions automatically, without any manual effort, into an information flow (IF) graph that describes the attack surface of the smart home. Moreover, our technique enables an user to perform reachability queries over the IF graph to find sources of flows to a specific device or service installed in the smart-home.

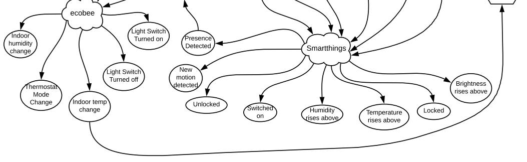


Fig. 1. Initial attempts at building Trigger-Action information flow graphs suffered from state explosion and false dependencies.

II. Preliminary Experiments

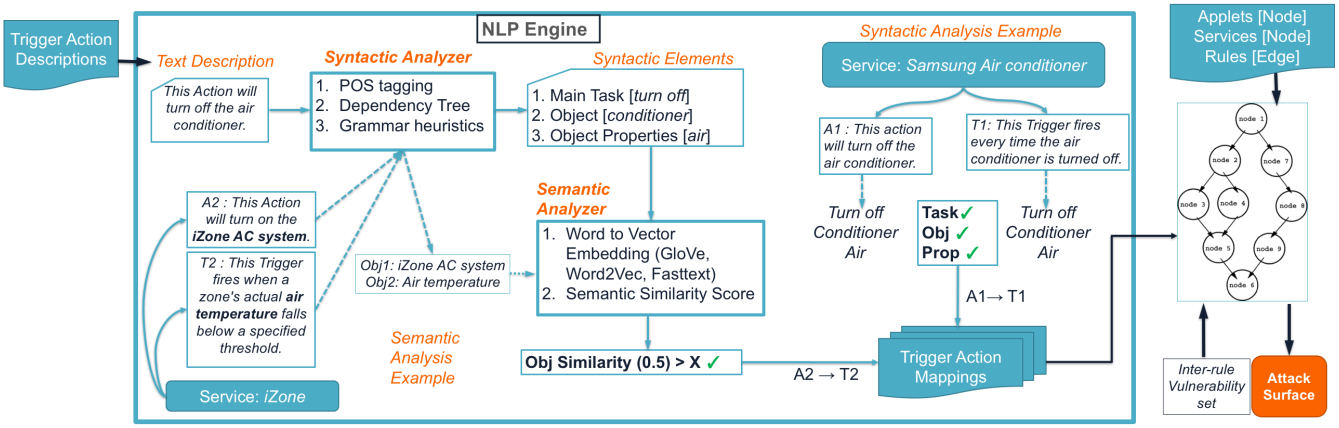
To get an idea of how trigger- action rules interact with each other, we obtained a dataset (2016) of trigger- action rules from the popular platform IFTTT [3]. Individual rules in this dataset are in form of *"If Trigger, Then Action"*, where each *trigger* and *action* is linked to some channel. As an example, in *"If humidity goes over 55% send notification to open window"*, trigger *'humidity goes over 55%'* is linked to channel *Netatmo Weather Station* and action *'send notification to open window'* is linked to channel *IFTTT notifications*. In IFTTT platform, each rule is called a *recipe* (from 2017, each rule is called an *applet* and *channel* is renamed as *service*).

We parsed individual recipes in an information flow (IF) graph as following: $Channel \rightarrow Trigger \rightarrow Recipe \rightarrow Action \rightarrow Channel$. Then we combined IF graphs for all recipes in a single IF graph. As evident from this single recipe graph, tracking flow from trigger to action is trivial, because the flow lies in the rule itself. But tracking flow from an action to other possible triggers it could invoke is non- trivial since the descriptions of actions and triggers are in natural language. So, the simplest way to model inter-rule flows is to link a service's inbound actions to all outbound triggers, which leads to state explosion and spurious flows in the graph (Figure 1). Unfortunately, many of the flows in this graph shows false flows that does not describe actual rule interaction. For example, *Smartthings*'s "Activate Device" action would not trigger "Presence Detected"; device activation and presence detection are actually two independent features that can be manipulated through this service. *It is thus apparent that in order to obtain an accurate and precise attack surface graph, we must decompose channels into their underlying components in order to identify true inter-rule flows.*

III. System Design

We present the system architecture of NLP- aided information flow analysis system for trigger- action platforms in Figure 2. We strive to automatically find precise information flow from action to trigger using existing natural language techniques [5]. IFTTT platform website was scraped to collect recent dataset of applets and service descriptions, 512 Services and 18892 applets were obtained. These text descriptions act as an input to our NLP engine.

Syntactic Analysis. Using NLP techniques, text descriptions of actions and triggers are tagged using *parts-of-speech*



Trigger Action Descriptions
NLP Engine
Text Description
This Action will turn off the air conditioner.
Syntactic Analyzer
1. POS tagging
2. Dependency Tree
3. Grammar heuristics
Syntactic Elements
1. Main Task [turn off]
2. Object [conditioner]
3. Object Properties [air]
Syntactic Analysis Example
Service: Samsung Air conditioner
A1 : This action will turn off the air conditioner.
T1: This Trigger fires every time the air conditioner is turned off.
Turn off Conditioner Air
Task ✓
Obj ✓
Prop ✓
Turn off Conditioner Air
A1→ T1
Applets [Node]
Services [Node]
Rules [Edge]
node 1
node 2
node 7
node 3
node 4
node 8
node 5
node 9
node 6
A2 : This Action will turn on the iZone AC system.
T2 : This Trigger fires when a zone's actual air temperature falls below a specified threshold.
Service: iZone
Obj1: iZone AC system
Obj2: Air temperature
Semantic Analyzer
1. Word to Vector Embedding (GloVe, Word2Vec, Fasttext)
2. Semantic Similarity Score
Semantic Analysis Example
Obj Similarity (0.5) > X ✓
A2 → T2
Trigger Action Mappings
Inter-rule Vulnerability set
Attack Surface

Fig. 2. NLP-aided Information Flow Analysis of Trigger-Action Platforms.

(POS) tags and subsequently dependency parsing is employed to create a dependency tree for the sentence representing grammatical relationships. Using these dependency relationships and predefined heuristics, syntactic elements can be extracted from the description, which are - root verb (*task*), the main object and the properties of the object. Due to similarity in descriptions of actions and triggers inside a single service, syntactic element matching results in detection of flow from action to trigger.

Semantic Analysis. Unfortunately, syntactic analysis alone is insufficient to detect all inter-rule flows. This is because trigger and action text descriptions often refer to semantically-related objects (e.g., *temperature* and *thermostat*) without referencing the exact same object. This calls for semantic analysis technique like word to vector embedding [6] to compute semantic similarity between words. If the similarity surpasses a predefined threshold between main objects of an action and a trigger, then we determine a flow from said action to trigger.

Using the detected flows in NLP engine, the spurious flows in IF graph in Figure 1 can be pruned leading to a highly precise graph capturing correct flows among the set of automation rules. This IF graph can be leveraged by an user to perform queries e.g., *"Which services can affect my door lock?"* and retrieve the attack surface of the specified door lock.

IV. Results

We manually inspected action and trigger descriptions of services scraped from the IFTTT platform website to obtain ground truth. Then we employed our tool and compared the *number of flows detected* to the manually computed results, as shown in Table I. The semantic analyzer is still under development causing our system to miss several semantic flows leading to low number of true positives. The false positive occurs when object properties are specified using grammar rules our system do not monitor, leading to a false match.

TABLE I
Summary of Error Rates.

Attribute	Observed Value
#True Positive	422
#False Negative	152
#False Positive	1164
#True Negative	5251
FP rate α (type I error)	0.18
FN rate β (type II error)	0.26
#Total flows (naïve strategy)	6989
#Total flows (NLP-aided strategy)	1586 (23%)

V. Future Work

Refining the semantic analyzer to improve the accuracy for flow detection is ongoing. Additionally, to test the effectiveness of our system to find security vulnerabilities in real environment, representative smart home configurations are required. Since this information is proprietary to the platforms, generating practical models of rule configurations in user homes is another important thread of work.

Acknowledgment

The authors would like to thank Qi Wang and Dr. Carl Gunter from University of Illinois, Urbana- Champaign for their helpful suggestions and resources.

References

[1] Fernandes et al., *Decentralized Action Integrity for Trigger-Action IoT Platforms*, appeared in The Network and Distributed System Security Symposium (NDSS), 2018.

[2] Surbatovich et al., *Some Recipes Can Do More Than Spoil Your Appetite: Analyzing the Security and Privacy Risks of IFTTT Recipes*, in Proceedings of the 26th International Conference on World Wide Web (WWW), Pages 1501-1510 , 2017.

[3] Ur et al., *Trigger-Action Programming in the Wild: An Analysis of 200,000 IFTTT Recipes*, in Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI), Pages 3227-3231, 2016.

[4] Nest Thermostat service in IFTTT, *https://ifttt.com/nest_thermostat*.

[5] Stanford CoreNLP, *https://stanfordnlp.github.io/CoreNLP*.

[6] GloVe: Global Vectors for Word Representation, *https://nlp.stanford.edu/projects/glove*.