If it moves, measure it!
or how I drove my wife absolutely crazy with sensors.

Chris Cowan
Kibana Engineer
Measure all the things!

- Presence Sensors
- Motion Detectors
- Power Usage
- Bandwidth Usage
- Contact Switches
- Noise Sensors
- Temperature and Humidity
- Illuminance
- Vibrations
Architecture

SmartThings Hub → SSL Proxy

Logstash → Elasticsearch
SmartThings Hub

- Customizable with Groovy
- Works with Z-Wave Devices
- Works with Zigbee Devices
- Provides Mobile Interface to System
- Their objective is to be open and extensible
Aeon Labs Home Energy Meter

- Clamps to Power Mains
- Exports power usage in real time
- Uses Z-Wave
- Exports Watts and kWh
- Difficult to Install
- You have to risk your life for installation
Behind this panel lives certain death!
According to the guy at hardware store

= SAFETY!
What I thought was going to happen...
What actually happened...
What actually happened...
Watt Usage Chart

Breakfast

Dinner
Energy Usage Chart
Our HVAC Fan Cycles Every Hour
Power By Hour for Last 30 Days
Aeon Labs Multisensor

- Temperature
- Humidity
- Motion
- Illuminance
- Uses Z-Wave
- Runs on 4 AAA Batteries
- Indoor/Outdoors
- Looks like the Eye of Sauron
- Terrifies Children

Most cost effective solution given everything these sensors do along with the SmartThings integration
Motion Sensor Events
Counting Stairs

When the sensor at the top is triggered it looks back 25 seconds for a trigger event at the bottom of the stairs and counts that as 18 stairs.
Humidity Events

It rained last night!
Temperature Events
Temperature Percentiles
Washing Machine Sensor

Wait 50 minutes after a close event to start watching the for the accelerometer events to stop moving for 2 minutes.

- SmartThings Multisensor
- Detects Open and Close
- Detects Temperature
- Detects Movement
Washing Machine Events
Raspberry Pi Garage Door Opener

- Raspberry Pi A+
- Relay Switch
- Magnetic Relay Switch
- Cat 5 Cable
- Integrated with SmartThings Hub
- Door Automatically Opens and Closes

Code available at: https://github.com/simianhacker/rpi-garage-door
Raspberry Pi Garage Door Events
DIY Sound Sensor

- Arduino Uno
- Analog Sound Sensor
- 433mHz Transmitter
- 433mHz Receiver on an Arduino Uno hooked up to a Raspberry Pi
- RPI logs directly to Elasticsearch
Sound Sensor: Carpet Cleaning

Cleaning Carpets Upstairs
Dashboards!
Energy Dashboard

- Avg Power for Last 10 Minutes: 1785
- Cost for Time Period: 3.057
- Total Energy for Time Period: 38.207
Temperature Dashboard
Hodgepodge Dashboard
Technical Stuffs
Data Enrichment with Logstash

• Add a time parts object to each record
  – “timeParts” with attributes for hour, minute, day, weekday, year, month, week year, quarter

• Calculate time difference between “presence” event.
  – When an event with the “valueAsString” attribute set to “present” query Elasticsearch and find the previous “not present” event

• Calculate difference between “energy” events
  – Store the value for the last “energy” event and calculate the “delta”
if [displayName] =~ /^Top|^Bottom/ and [valueAsString] == "active" {

}
if [displayName] =~ /Top|Bottom/ and [valueAsString] == "active" {
    # If the event is Top of Stairway then search for Bottom of Stairway
    if [displayName] =~ /Top/ {
        mutate { add_field => { "tempDisplayName" => "Bottom of Stairway" } }
    } else {
        mutate { add_field => { "tempDisplayName" => "Top of Stairway" } }
    }
}
if [displayName] =~ /^Top|^Bottom/ and [valueAsString] == "active" {
  # If the event is Top of Stairway then search for Bottom of Stairway
  if [displayName] =~ /^Top/ {
    mutate { add_field => { "tempDisplayName" => "Bottom of Stairway" } }
  } else {
    mutate { add_field => { "tempDisplayName" => "Top of Stairway" } }
  }

  # Find the corresponding event in Elasticsearch
  elasticsearch {
    query => "valueAsString.raw:active \n        AND displayName.raw:"%{[tempDisplayName]}" \n        AND isoDate:* TO %{[isoDate]}"
    sort => "isoDate:desc"
    new_field => "previousDate"
  }
}
if [displayName] =~ /'^Top|^Bottom'/ and [valueAsString] == "active" {
    # If the event is Top of Stairway then search for Bottom of Stairway
    if [displayName] =~ /'^Top'/ {
        mutate { add_field => { "tempDisplayName" => "Bottom of Stairway" } }
    } else {
        mutate { add_field => { "tempDisplayName" => "Top of Stairway" } }
    }

    # Find the corresponding event in Elasticsearch
    elasticsearch {
        query => "valueAsString.raw:active \n                AND displayName.raw:"%{[tempDisplayName]}" \n                AND isoDate:[* TO %{[isoDate]}]"
        sort => "isoDate:desc"
        new_field => "previousDate"
    }

    # If the duration between events is less then 25 seconds then add 18 steps to the value
    ruby {
        code => "event['value'] = 18 if (event['@timestamp'] - event['previousDate']) > 25"
    }
}
if [displayName] =~ /^Top|^Bottom/ and [valueAsString] == "active" {
  # If the event is Top of Stairway then search for Bottom of Stairway
  if [displayName] =~ /^Top/ {
    mutate { add_field => { "tempDisplayName" => "Bottom of Stairway" } }
  } else {
    mutate { add_field => { "tempDisplayName" => "Top of Stairway" } }
  }

  # Find the corresponding event in Elasticsearch
  elasticsearch {
    query => "valueAsString.raw:active \n          AND displayName.raw:"%{[tempDisplayName]}" \n          AND isoDate: [* TO %{[isoDate]}]"
    sort => "isoDate:desc"
    new_field => "previousDate"
  }

  # If the duration between events is less then 25 seconds then add 18 steps to the value
  ruby {
    code => "event['value'] = 18 if (event['@timestamp'] - event['previousDate']) > 25"
  }

  # Remove the temporary attributes
  mutate { remove => ["tempDisplayName", "previousDate"] }
}
{  
  "isoDate": "2015-03-08T17:38:46.936Z",  
  "displayName": "Top of Stairway",  
  "descriptionText": "Top of Stairway detected motion",  
  "name": "motion",  
  "valueAsString": "active",  
  "value": 18,  
  "timeParts": {  
    "hour": "17",  
    "minute": "38",  
    "weekday": "Sunday",  
    "week": "10",  
    "day": "8",  
    "month": "March",  
    "year": "2015",  
    "quarter": "1",  
    "weekYear": "2015"  
  },  
  "unit": "stairs"  
}
Use Scripted Fields to Make Your Data Interesting

<table>
<thead>
<tr>
<th>name</th>
<th>script</th>
<th>type</th>
<th>popularity</th>
<th>controls</th>
</tr>
</thead>
<tbody>
<tr>
<td>cost</td>
<td>doc['delta'].value * 0.08</td>
<td>number</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>GB</td>
<td>doc['value'].value / 1048576</td>
<td>number</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>MB</td>
<td>doc['value'].value / 1024</td>
<td>number</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>minutes</td>
<td>doc['value'].value / 60000</td>
<td>number</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>decibels</td>
<td>20.0 * log10 (doc['value'].value +1.)</td>
<td>number</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Questions?