DYNAMIC TRUST AND RISK SCORING USING LAST-KNOWN-PROFILE LEARNING

Authors et. al.: Hazim Dahir
Omar Santos
Jazib Fahim
Yenu Gobena

IP.com Number: IPCOM000247388D
IP.com Electronic Publication Date: August 31, 2016

DYNAMIC TRUST AND RISK SCORING USING LAST-KNOWN-PROFILE LEARNING

AUTHORS:

Hazim Dahir
Omar Santos
Jazib Fahim
Yenu Gobena

CISCO SYSTEMS, INC.

ABSTRACT

Presented herein is a distributed and dynamic security threat and risk calculation method for Internet of Things (IoT) environments. Dynamic changes of IoT infrastructure are detected, and a "Risk Score" profile is derived from multiple "current" or "previously known" factors about the sensor or previous communication patterns. The Risk Score is updated and maintained over time. This method allows for enumerating and classifying IoT asset value in large-scale IoT environments.

DETAILED DESCRIPTION

The majority of sensors today are built with very light weight protocols with limited battery life. This trend is here to stay as sensors continue to get smaller in their form-factors in order to accommodate a wider set of applications and use-cases. Consequently, sensors are only able to share limited information about their identity with the upper layers of the stack when communicating with their first-hop gateway.

Another challenge is that some of these sensors may go dormant for long periods of times ranging from a couple of days to months. The re-introduction of these devices can be risky as they could have been moved, compromised, or a rogue new sensor could have been added either maliciously or inadvertently. As a large number of sensors are placed in a variety of large-scale environments it may become difficult to authenticate and trust individual sensors based on information carried in the communication exchange. Typically, if a sensor is not trusted it may be allowed to connect to the network but not access or write to any applications. The data may or may not be stored and in some cases that data may be extremely valuable especially once the sensor has been identified as a valid communicator into the application.

In some cases, sensors from one project may end up being on a different project (and perhaps on a different authentication domain). Consider a scenario where sensors are used to monitor seismic activity in California for a project sponsored by a first entity. At the end of the assignment, the same sensors might be used to measure seismic activity in Japan for a project sponsored by a second entity. The authentication gateways and perhaps the authentication domains could be completely different based on the project type and the deployment scenario. Ignoring all other challenges around a large shared secret or public-key infrastructure deployment, a sensor can certainly be authenticated using a shared secret or public-key exchange. However, a successful authentication only validates the identity of the sensor (I am talking to the sensor that I should be talking to). It does not validate the business context (Is it registering/onboarding for the project that it should be a part of?) as well as the technical context (is it tampered in any way since it was last seen online?) of the sensor.

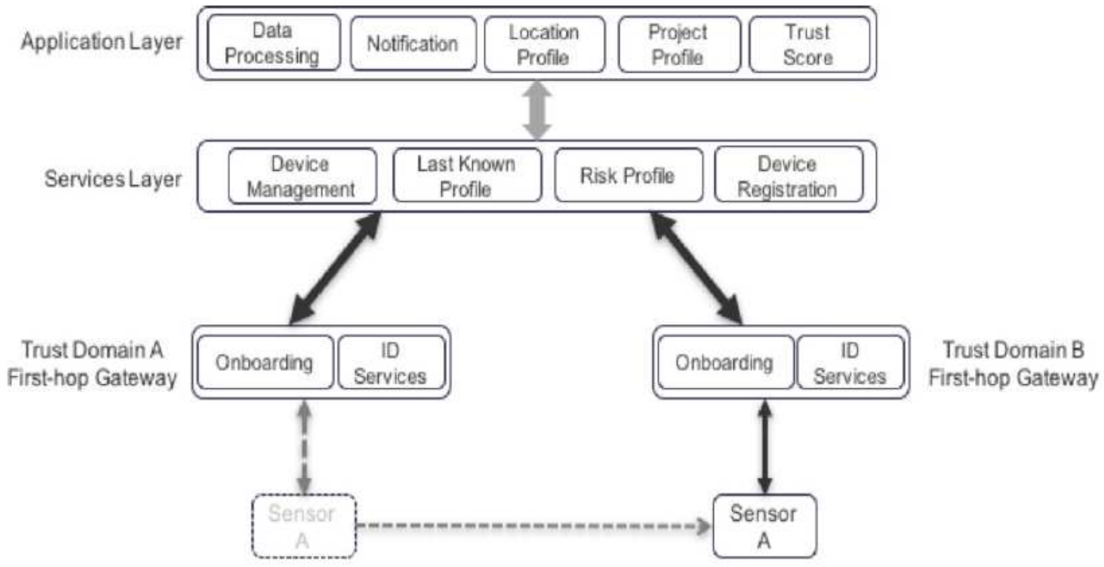
A method is needed for building hierarchical (or tiered) trust model that can authenticate a sensor at the first-hop gateway to validate its assigned identity and place a "limited trust" on it, validate the sensor's profile to increase its trust, and keep increasing its trust by leveraging the business intelligence and information obtained from the upper layers of the communication stack

This tiered trust model can keep increasing the trust level of the sensor dynamically until it is allowed to have unrestricted communication as a part of a project.
This provides a mechanism for security as well as data integrity by listening and collecting, but not allowing compromised or rouge sensor data as part of the analytics or data lake of a particular application.

Presented herein is a solution that addresses the challenges related to device or configuration compliance via the use of last known profile servers. This allows for the ability to apply dynamic profile policy checks with set and expected parameters such as microcode, identifier (ID), software to ensure sensors' compliance. For example, if the microcode or firmware of a sensor has been altered (maliciously or otherwise), this method will be able to determine the changes based on changes in the data or communication patterns relayed by the sensor. The profile attributes may be weighted based on characters of application expected behavior of the sensors. For example, there may be less weight if the sensor has not communicated back for a period of time versus a microcode or location change.

The methods described herein rely on building and maintaining a "Risk Score" profile derived from multiple "current" or "previously known" factors about the sensor and/or its previous communication patterns. This knowledge base, referred herein as the "Last Known Profile" (LKP), maintains an encrypted profile for each sensor containing information, such as the sensor embedded ID, model, firmware, last known location, last known address, last gateway address, and last reading count. The list can be expanded to include any sensor attribute and is not limited to the attributes mentioned here. Weights can be assigned to these parameters to determine how risky a particular sensor is to the application itself. A per application profile can also be used since the expected behaviors may differ from application to application.

FIG. 1



With reference to FIG. 1 above, an IoT system is described that comprises 4 layers, including:

1. **Sensor**: Sensors are the physical IoT devices used for a specific function (monitoring pressure, temperature, seismic activity, air moisture, wind speed to name a few)

2. **Trust Domain**: The first layer that interacts with these sensors. This layer is responsible for providing authentication and onboarding services to the physical IoT devices.
3. **Services**: Service layer acts as the intermediary layer between the application and the trust domain, responsible for the management and maintenance of the IoT sensors. This layer is also responsible for collecting sensor attributes and keeping them in the LKP.
4. **Application**: Application layer is ultimately responsible for the management and assignment of sensors to a project. This layer collects useable data from the sensors that are eventually used for analytics and/or predicting events (such as earthquakes).

The diagram in FIG. 1 above shows two trust domains that could be across multiple companies, providers, or locations. The same concept can be used in a single trust domain as shown in FIG. 2 below and described in the following section.

In order for a sensor to be a part of a project, the following needs to happen:

1. Sensors present their identity information to the first-hop gateway to be authenticated into the system.
2. The services layer collects information about the sensor (its current profile).
3. Before allowing access to the application layer, the current profile is compared with previously known "good or acceptable" profile. A new sensor can be categorized by the "Device Registration" process in the Services Layer or by a "controller" with a default value of "neutral". This default "risk score/profile" is a template-based profile that can be created based on the type of application. The different variables can be weighted based on specific predetermined criteria. For instance, a higher weight can be assigned to things like microcode, the type of data that is being carried, image configuration, firmware, location, etc. These variables are not limited to these types of categories and states. Any information that can be collected from the sensors can be used to define an accepted or base risk profile.

4. Once the sensor starts sending data and it is provisioned by the registration process or controller, the score is adjusted based on different criteria including the state, the posture, and types of data and applications used. Again, this is not limited to just these categories or states.

5. Based on the results of this comparison process, a new risk score is determined. Only sensors or devices that meet the minimum risk score for a proposed application will be allowed to partake in the system. The risk score can be weighted to give things such as ID or firmware change a less favorable output versus time of last communication (communication of sensors are meant to be infrequent so this maybe an expected behavior). Factors that determine the new risk score may include, but not limited to:
    1. Location (previous or new)
    2. Time of Last Connectivity and Communication (thresholds can be set)
    3. Device Information
        1. Firmware
        2. Image
        3. Configuration Parameters
    4. Change in Identification
        1. Name
        2. Certificate (Chain)
        3. Unique identifier

Note: The risk score only helps to determine if there were significant changes in a sensor profile. Should I ever trust this sensor knowing that its firmware was tempered? Will the data produced by such sensor be trustworthy?

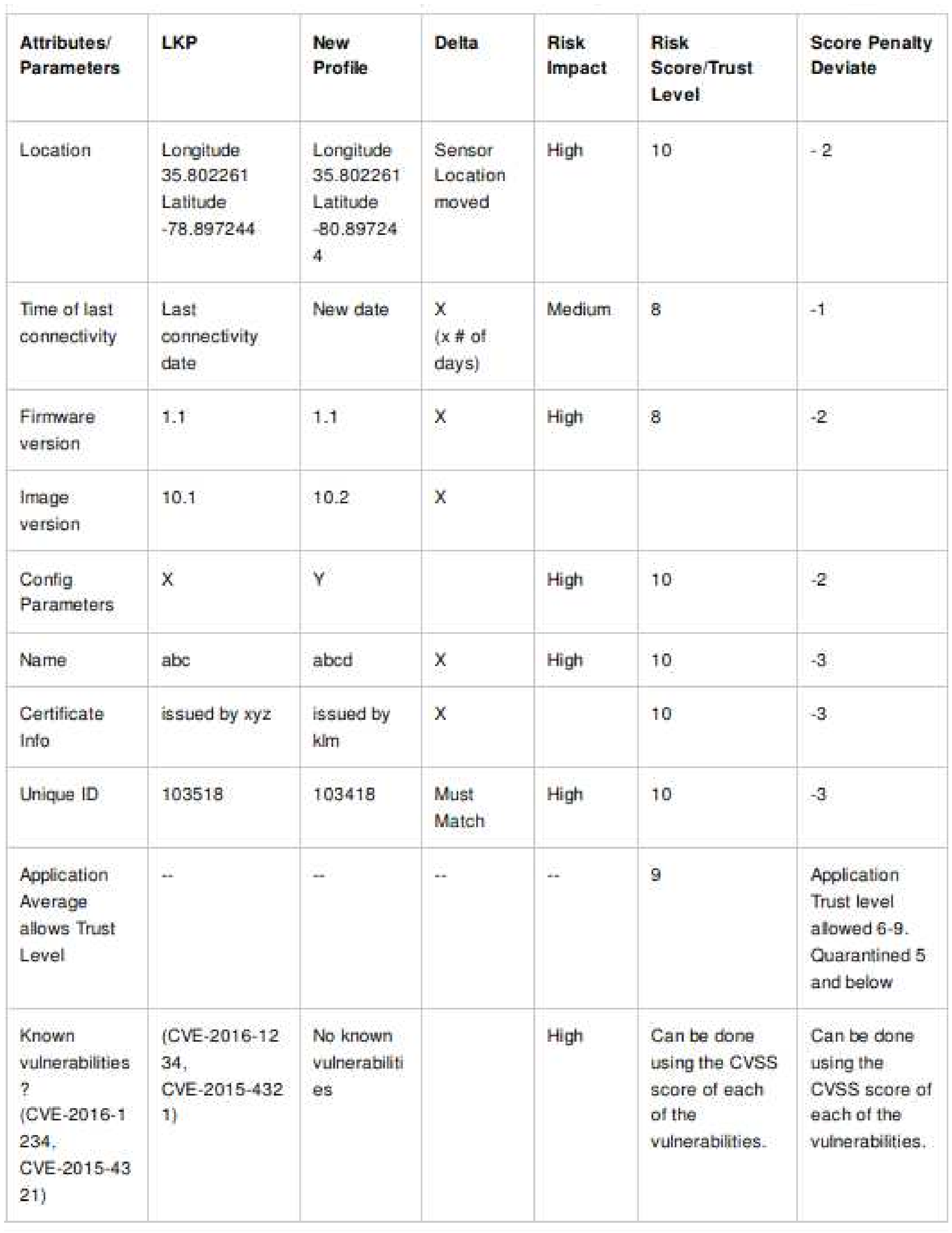
5. The application layer, after receiving the risk score and the most recent profile of a sensor, compares sensor's information with "Project" information to determine the sensor's "Trustworthiness" which is

represented by a "Risk Score. " For example, the sensor should be in Japan based on its project assignment, but it is showing up in Thailand.

6. The "Risk Score" will determine the trust level and privileges. For example, the trust score below 4 (e.g., change in certificate chain) could instruct the application to simply ignore all data produced by a sensor regardless of its accuracy. A score between 4 and 7 could place a sensor in a gray-area where certain data elements are accepted. A trust score above 8 could allow the application to trust all data elements produced by the sensor.

7. To reduce the risk score (and increase trust levels), the following steps could be taken:
    1. Manual Approval based on an Alert sent and acted upon by a Security of project administrator. This would applicable to one sensor or a group of sensors participating in the same "process' or "project".
    2. Validation of profile based on Data being relayed over a specific period of time (e.g. expected behavior of the experiment being conducted).
    3. Dynamic training models (pre-existing machine learning).

The profile attributes may be weighted based on characteristics of application expected behavior of the sensors. For example, there maybe less weight if the sensor has not communicated back for a period of time versus a microcode or location change. Below is an example chart that shows a default template and a score deviation if a parameter does not match.



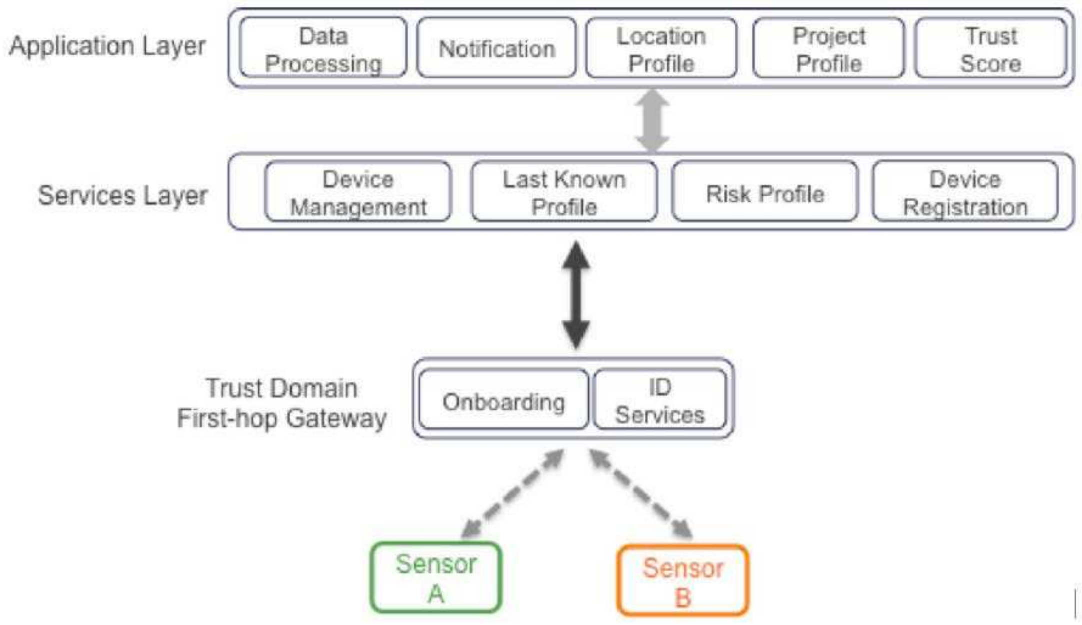
Attributes/ Parameters	LKP	New Profile	Delta	Risk Impact	Risk Score/Trust Level	Score Penalty Deviate
Location	Longitude 35.802261 Latitude -78.897244	Longitude 35.802261 Latitude -80.89724 4	Sensor Location moved	High	10	- 2
Time of last connectivity	Last connectivity date	New date	X (x # of days)	Medium	8	-1
Firmware version	1.1	1.1	X	High	8	-2
Image version	10.1	10.2	X			
Config Parameters	X	Y		High	10	-2
Name	abc	abcd	X	High	10	-3
Certificate Info	issued by xyz	issued by klm	X		10	-3
Unique ID	103518	103418	Must Match	High	10	-3
Application Average allows Trust Level	--	--	--	--	9	Application Trust level allowed 6-9. Quarantined 5 and below
Known vulnerabilities ? (CVE-2016-1 234, CVE-2015-43 21)	(CVE-2016-12 34, CVE-2015-432 1)	No known vulnerabiliti es		High	Can be done using the CVSS score of each of the vulnerabilities.	Can be done using the CVSS score of each of the vulnerabilities.

The business impact stems from the technical impact, but requires a deep understanding of what is important to the company running the application. Technical impact can be broken down into factors aligned with the traditional security areas of concern: confidentiality, integrity, availability. The goal is to estimate the magnitude of the impact on the system if the vulnerability were to be exploited.

- Loss of confidentiality - How much data could be disclosed and how sensitive is it? Minimal non-sensitive data disclosed, minimal critical data disclosed, extensive non-sensitive data disclosed, extensive critical data disclosed, all data disclosed.
- Loss of integrity - How much data could be corrupted and how damaged is it? Minimal slightly corrupt data, minimal seriously corrupt data, extensive slightly corrupt data, extensive seriously corrupt data, and all data totally corrupt.
- Loss of availability - How much service could be lost and how vital is it? Minimal secondary services interrupted, minimal primary services interrupted, extensive secondary services interrupted, extensive primary services interrupted, all services completely lost.

FIG. 2 below is a diagram depicting a single domain with both a good and a tampered sensor.

FIG. 2



This solution proposes a distributed architecture leveraging first-hop gateways that provide authentication and on-boarding services to potentially millions of globally deployed sensors. If a new sensor needs to be on-boarded, a template-based default "risk score/profile" is applied based on the application/project. However, if a previously registered sensor is seen by a gateway (after x # of days, different gateway etc.), it is re-profiled to determine its current posture. Based on the comparison of these two profiles, a dynamic threat and risk profile is determined.

Leveraging the diagram of FIG. 2 covers a two stage process of security: a first at the service layer, and a second at the application layer. The application level security provides the ultimate check and compliance associated with a particular application. The critically of this function, aside from ensuring the sensor has passed the minimum risk profile, is around data integrity and compliance. A good example is in agriculture and the concept of script farming. A farmer following a script on when to water, when to fertilize is guaranteed a better yield. That yield comes at a premium but is also a big competitive advantage for the crop and equipment manufactures. The data integrity of "compliance" whether or not the farmer or crop grower followed instructions is based on valid data from the sensors. The application layer risk profile provides a mechanism to prevent someone placing a rouge sensor, tampering with a sensor, and manipulating data for the script farming compliance application. This rouge sensor could be placed by someone that did not follow the script, a competitor, a malicious attacker looking to throw off data. The proposed two stage dynamic risk profile allows for this kind of protection aligning to the characteristics and behavior of sensors (dormant, low power, small compute (inhibits complex security schemas)).

As previously mentioned, the risk score will be determined by the application owner. The application owner also has the flexibility to weight one attribute higher than another. For example, if the application characteristics calls for sensors to go dormant for long periods of time, then a sensor should not be penalized for that. However, if the sensor is talking constantly, then that falls outside of expected behavior and it would be penalized for that from a score perceptive. Another example is if is expected that there will be a lot of micro-code upgrades, then this can be weighted less that one of the other attributes.

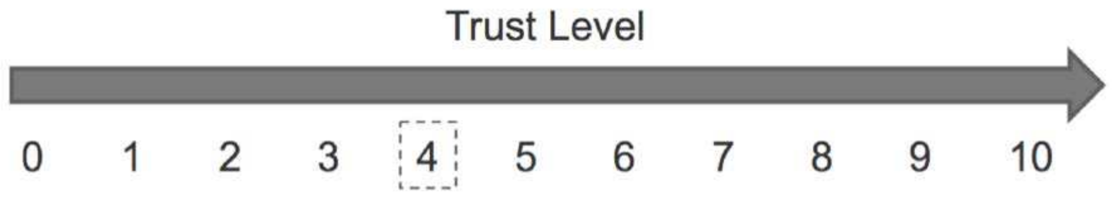
"Scripted Farming" or Integrated Farming is an IoT opportunity being evaluated by seed manufacturers, industrial equipment manufacturers, farmers, etc. Again, the goal of scripted farming is to provide a prescription of how to increase (and guarantee) yield. As a seed manufacturer, the farmer and the associated equipment would be instructed to water on Monday, Fertilize on Tuesday, Do nothing on Wednesday, etc. By doing, the farmer is guaranteed a better yield, i.e., 20% more crops. That comes at a cost either upfront or as a supplier who may ask for 10% of the profit from the yield. The way the script is determined is by sensors in the ground measuring moisture, PH Levels, fertilizer levels, etc. A manufacturer of seeds would want to ensure that he is getting good data back to ensure the script is accurate. If a sensor is tampered with or malfunctioning based on characteristics of the application, then the whole formula is at risk. This could be done by a competitor, hackers, foreign nations to impact our food supply, etc. The second value in risk score in this use case would be around making sure the script was followed, i.e., compliance. A farmer could say he did not get any better yield because he did not follow the script, or other reasons. The use case shows why both on the supply and consumer side the risk score helps with data integrity of the application.

Data Classification Based on Trust Level

Trusting a sensor largely affects how to trust data from that sensor. The sensor's integrity and its validation play a crucial role. It helps to evaluate how much data could be corrupted and how damaged it is: minimal slightly corrupt data, minimal seriously corrupt data, extensive slightly corrupt data, extensive seriously corrupt data, all data totally corrupt. It is important to be able to protect/trust a sensor based on its potential risk and the potential of it being compromised and reporting something other than what it should be reporting.

The trust level can be pre-classified in a predetermined or arbitrary scale. The following example shows a scale of 0 through 10. Zero (0) is the least trusted state and ten (10) is the highest trust level. As previously mentioned, the trust level or score can place the sensor or a compute node in a gray-area; where data could be allowed, but classified as "less trusted". See FIG. 3 below as an example.

FIG. 3

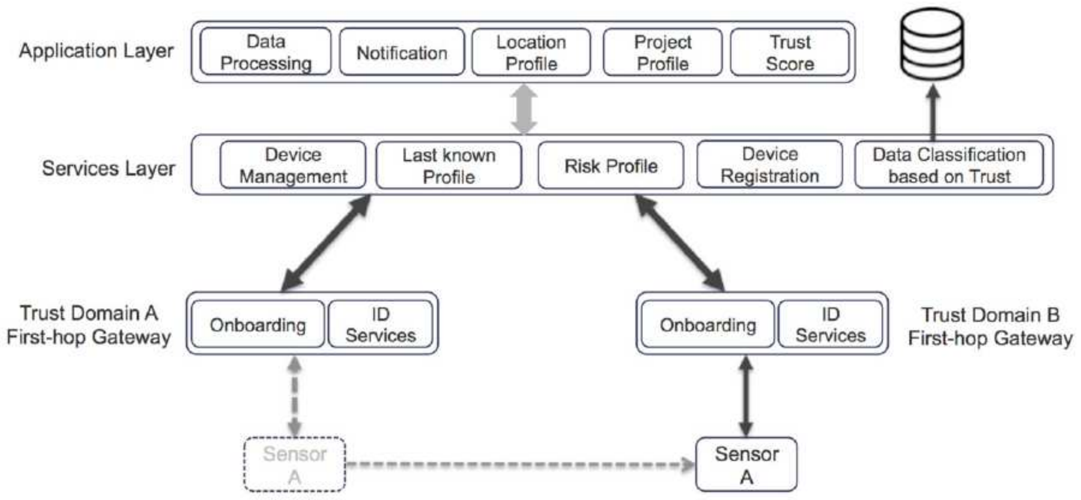


The trust level can place the sensor or a compute node in a gray-area; where data could be allowed, but classified as "less trusted". For example, data from a device with a score of 4 can be accepted, but used in a different way than data coming from a sensor or compute node that has a trust level of 10. The data classification entity can be within the services layer and such entity can "mark or classifies" the data to be stored or used differently. For instance, data from a sensor with trust level score is 4 can be used for informational purposes only, but not to automate an action based on such data.

Just as the weights are adjustable to meet the characteristics and behavior of the application, the application risk can be static or a range. As an application owner it can choose to allow only sensors and data with a trust score of 10. One could also choice a range I will accept data from anything that has a risk of 7 and higher. This allows for data consumption as the sensor is still being validated.

The data classification entity can be within the services layer and such an entity can "mark or classify" the data to be stored or used differently, as shown in the example in FIG. 4 below.

FIG. 4



In summary, presented herein is a distributed and dynamic security threat and risk calculation method for Internet of Things (IoT) environments. Dynamic changes of IoT infrastructure are detected, and the "Risk Score" profile is derived from multiple "current" or "previously known" factors about the sensor or previous communication patterns. The Risk Score is updated and maintained over time. This method allows the capability of enumerating and classifying IoT asset value in large-scale IoT environments. This method can also allow an implementer to automatically, fast and accurately classify data based on the risk calculation and trust level of distributed and dynamic IoT environments. The method of performing automated risk calculations, trust scoring, and identifying and classifying data based on those calculations can greatly reduce the overall security risk and the cost to classify data and use that for visibility in order to apply the correct access control, improving monitoring, and even educate users on what is sensitive data.