# ADAPTIVE LEARNING MODEL FOR APPLICATION-BASED TRUST AND RISK SCORING USING CONSISTENT PROFILE CREATION

AUTHORS:

Omar Santos
Jazib Frahim
Yenu Gobena
Hazim Dahir

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 values in large-scale IoT environments by providing an adaptive learning security capability at a fog node that maintains and dynamically learns approved application attributes. The method also allows for ongoing security checks to verify integrity of the data stream at randomized intervals set by the risk tolerance of the application.

## DETAILED DESCRIPTION

The majority of sensors today are built with very lightweight protocols and limited battery life. This trend is likely to continue as sensors become 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. Described herein is the inclusion of adaptive machine learning for the purpose of ongoing risk profiles based on application tolerance. This model focuses on the ability of a fog node to maintain an associated risk profile for all sensors and nodes, and, based on received data, dynamically adjust a risk score. If the risk score falls out of the tolerance level of the application, the data should be "quarantined" until trust increases. This ties together adaptive learning and a first hop security layer while allowing the application to apply important characteristics aside from the traditional authentication and authorization initial handshakes.

Some sensors may go dormant for long periods of time, ranging from a couple of days to months. The reintroduction of these devices can be risky as they could have been moved or 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.

In some cases, if a sensor is not trusted it may be allowed to connect to the network, but read and/or read/write access is not given to any applications. The data may or may not be stored and in some cases that data may be extremely valuable, especially after the sensor has been identified as a valid communicator to the application.

Presented herein is an adaptive learning security capability at a fog node that maintains and learns approved application attributes. Figure 1 illustrates an adaptive learning engine with inputs from services and the application layer. The components of this system are as follows.

**Sensor**: Sensors are the physical IoT devices used for a specific function (e.g., monitoring pressure, temperature, seismic activity, air moisture, wind speed, etc.)

**Fog Node**: The first layer that interacts with these sensors. This layer is responsible for providing authentication, on-boarding, and Machine Learning services to the physical IoT devices.

**Services**: The 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 last known profile.

**Application**: The 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).

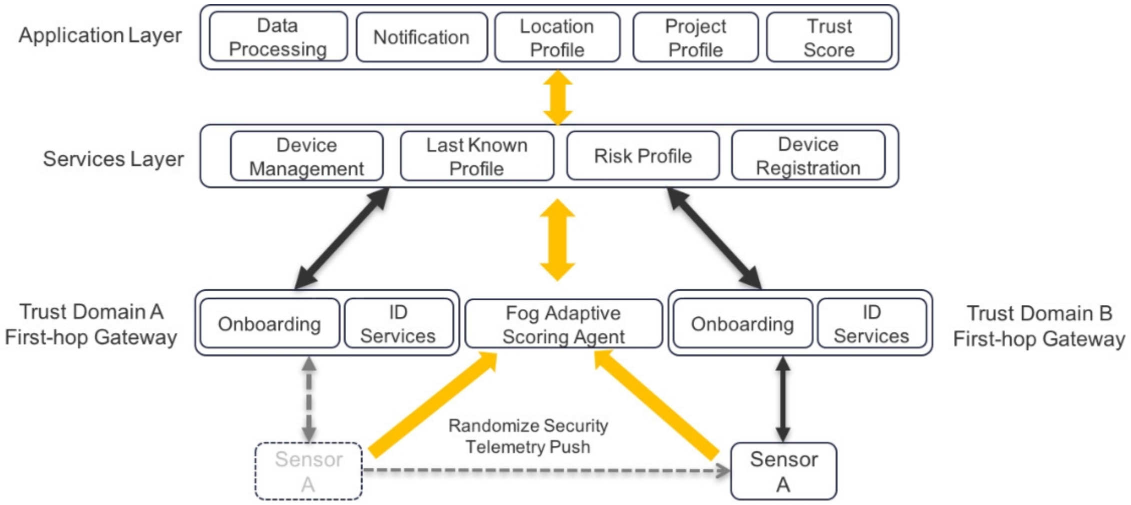

**Figure 1**

The fog node sends out randomized requests when a sensor/node needs to push certain attributes based on the application profile. The fog node maintains a profile of expected parameters that are learned and kept in the last known good profile to detect anomalies during the flow of data. The typical security models (such as authorization, authentication or policy enforcement) occur during the initial handshake prior to data transfer occurring.

An adaptive risk profile method scales a risk profile of a device up or down based on the application profile parameters received. The model maintains both an application profile and the last known profile of the sensor/node on the fog node. On the sensor side, the node may send the application risk parameters (e.g., microcode, device ID, firmware, digital certificate(s), etc.) to enable comparison with expected behavior in order to maintain a running profile of the device information. A lightweight agent may be also be used to enable the comparison.

The machine-learning fog agent on the fog node can be programmed to use different training methods to calculate and deduce the application tolerance. For instance, an implementer can use either a nearly zero-knowledge train model or a supervised method. In the zero-knowledge train model, the algorithm is incrementally trained starting with a little knowledge on the sensor "state," "behavior," or related information. In the supervised method, sets of known trajectories, sensors, and application state and attributes can be fed to the algorithm. In addition, the zero-knowledge train model and supervised method may be combined with additional enrichment methods by using non-reinforcement learning (nRL) and reinforcement learning (RL). In the nRL method, a misclassified risk or trust level is not further used in the model-training phase. Hence, the algorithm is no longer aware of unsuccessful predictions. In the RL method, a misclassified risk or trust level is introduced into the knowledge base, thus appropriately updating the model, as illustrated in the application layer tolerance sliding scale of Figure 2.

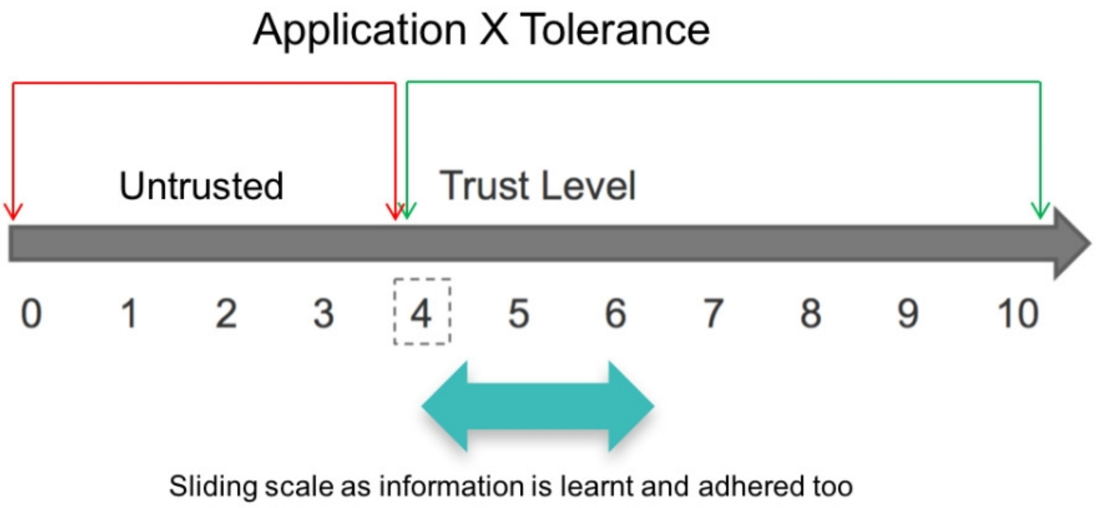

**Figure 2**

The trust level can be pre-classified in a predetermined or arbitrary scale. The following example shows a scale of zero (0) through ten (10). Zero is the least trusted state and ten is the highest trust state.

To preserve data integrity, if a sensor or host falls below the minimum risk tolerance, the application can be set to either discard data or store the data in a quarantine mode until further action. This can be accomplished either manually or via a machine learning algorithm decision. The application setting may allow data to be completely discarded if the score falls below a threshold (e.g., below 4) that indicates that it is completely untrustworthy.

A sensor/node can also be placed in a gray area between zero and ten if data is allowed but classified as "less trusted". When the sensor or host falls into the grey area, the application owner can choose to quarantine that data until further action. The machine learning algorithm can check the expected attributes per the application settings and continually check and compare expected behavior frequency of attribute or data messages (e.g., whether it falls within the expected times, the number of times the sensor or host violated the policy previously, etc.) before allowing free flowing data and appropriate storage.

Using this machine learning approach, each application-identified attribute is compared with the learned behavior of the sensor. When the sensor/node deviates sufficiently (e.g., frequency of communication, location change etc.), the fog node flags and quarantines the sensor and the data. The application owner may decide, based on a template, the characterization of the application and associated risk score that must be met for each attribute, which contributed to the determination of the average overall application acceptable trust level.

The fog node pushes a "randomization" alert to the agent for upstream reporting. This provides additional security to ensure that sensors or nodes are not tampered with. The fog node may only report back when told to do so through these randomized alerts. Sensor push requires application security attributes when the timer expires. If the parameters are not satisfied and the sensor is unable to send an update for a certain time period, the fog node downgrades the corresponding trust score. When a new update is received by the fog node, it is compared with, and set equal to, the last known profile.

The methods described herein achieve a distributed and dynamic IoT security threat and risk calculation for dynamic changes of IoT infrastructure. Additionally, these methods provide the capability of enumerating and classifying IoT asset value in large-scale IoT environments. They can also allow an implementer to automatically, quickly, and accurately classify data based on the risk calculation and trust level of distributed and dynamic IoT environments. The methods 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. The methods use the visibility of data to apply the correct access control, improve monitoring, and educate users as to which data should be considered sensitive.

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 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. These methods allow for enumerating and classifying IoT asset value in large-scale IoT environments, by providing an adaptive learning security capability at a fog node that maintains and dynamically learns approved application attributes. The methods also allow for ongoing security checks to verify integrity of the data stream at randomized intervals set by the risk tolerance of the application.