IoT システムを構成するコンポーネントの信頼度に基づいた真正性検証手法の基本設計

濱本 亮[†1]　佐々木 貴之[†1]　森田 佑亮[†1]　三好 一徳[†1]　小林 俊輝[†1]

概要：スマートシティなどの IoT システムでは，不正なコンポーネント(デバイスやゲートウェイ，クラウドなど)がシステムに侵入することでシステム内を流通するデータが盗聴・改ざんされるという脅威がある．これを防ぐためにはコンポーネントの真正性検証が必要となるが，従来は一度真正性が保証されたコンポーネントは永久に信頼されるため，コンポーネントの状態が時間や環境によって変化する一般的な IoT システムへ適用するには不十分であった．そこで，本稿では IoT システムの特徴であるコンポーネントの種類/特性が多種多様かつ時間とともに変化することを踏まえて，真正性検証をコンポーネントの信頼度に応じて実施することを提案する．

キーワード：スマートシティ，IoT，認証，真正性検証，信頼モデル

A Basic Design of A Trust Model for IoT Systems Based on Trustworthiness

Ryo Hamamoto[†1]　Takayuki Sasaki[†1]　Yusuke Morita[†1]　Kazunori Miyoshi[†1]
Toshiki Kobayashi[†1]

Abstract: IoT systems such as smart cities face a threats of incursion of unauthorized components such as devices, gateways and clouds. An eavesdropping or a falsification of data is pursue by unauthorized components. To prevent this issue, an attestation for the component is required. This paper proposes a trust model for IoT systems based on trustworthiness of devices. Moreover, we discuss the effectiveness of our model.

Keywords: Smart city, IoT, Attestation, Trust model

1. はじめに

スマートシティ[1]などの公共系 Internet of Things (IoT) システム[2]では，システム内に IoT デバイス，ゲートウェイ (GW)，クラウド，アプリケーションなど多種多様なコンポーネントがある程度自由に参入できる．しかし，一方では課題として管理者が不明なコンポーネント (不正なコンポーネント) が自由に IoT システムへ侵入する可能性が高く，これを契機として，例えば (1)システム内を流通するデータが盗聴・改ざんされる，(2)システムの正規コンポーネントの資源を使い果たしてシステムを停止させるという攻撃が発生する．この攻撃への対策として，システムに参入しているコンポーネントが正規のコンポーネントであるかを判定する，すなわちコンポーネントの真正性検証 (Attestation) が行われている．Attestation の手法として，Trusted Computing[3]を利用した Remote Attestation (以後，単に Attestation) が盛んに議論されている．一般的な Attestation の手法では，一度 Attestation して結果に問題が無かったコンポーネントはその後永久に信頼される．つまり，1 つのコンポーネントに対して複数回 Attestation は行わず，信頼する/しないのどちらかの判定が出ると以後その判定は変化しない．しかし，コンポーネントの種類や特性が多種多様かつ時間とともに変化する IoT システムでは，前述のような単純な 0/1 の (信頼する/しないが明確に判定でき，時間変化しない) 信頼モデルでは完全にデバイスの真正性検証を実施することは難しい．例えばコンポーネントの内部プロセスがマルウェアに感染すると，システムの運用中に正規コンポーネントが動的に悪意のあるコンポーネントに変化することが考えられる[4]．また，コンポーネントのセキュリティ機能がソフトウェア的に実装される，もしくはハードウェア的に実装されるかに応じて真正性が維持される可能性は変化することが予想できる．従って，IoT システムでは一度コンポーネントに対して信頼関係を構築したとしても，その信頼関係がその後も成立する保証がない．つまり，IoT システムでは，従来の単純な信頼モデルでは完全にコンポーネントの Attestation を実現することは困難であり，時間的・空間的な変化に対応可能な信頼モデルが要求される．しかし，既存の Attestation はこの様な信頼モデルを考慮して設計されていない．

†1 日本電気株式会社 セキュリティ研究所
Security Research Laboratories NEC Corporation

そこで本稿では，コンポーネントに対して時間的・空間的な変化を考慮した信頼度を新たに定義し，Attestation 完了後も定期的にコンポーネントの Attestation を行う方法を提案する．また，提案する方法によって，システム運用中に正規コンポーネント内で不正なプロセスの混入があった場合には，そのコンポーネントを排除可能なことを示す．

以後，第 2 節では関連研究の紹介を行い，第 3 節にて IoT システム向けの信頼モデルとして段階的信頼モデルを提案する．第 4 節にて提案手法である段階的信頼モデルに基づいた Attestation を説明し，第 5 節にて提案手法によって防ぐことができる攻撃を議論する．最後に第 6 節にて本稿のまとめと今後の課題について述べる．

2. 関連研究

本節では，Remote Attestation の概要と，IoT システムのコンポーネントの 1 つであるデバイスに対する Remote Attestation の効率化に関する先行研究を紹介する．

2.1 Remote Attestation の概要

Trusted Computing[3]における Attestation とは，耐タンパ性を持つチップ (いわゆる Trusted Platform Module: TPM[5]) やメモリ空間 (いわゆる Trusted Execution Environment: TEE[6]) に保存されている Platform Configuration Register (PCR) 値と呼ばれるハッシュ値を検証することを意味する．ハッシュ値の元となる情報には，そのシステムで動作していることが期待されているプロセスなどの構成要素に関する情報が含まれる．また，これらの情報を TPM や TEE に保存することによってハッシュ値の改ざん/漏えいを防ぐことが可能となる．

Remote Attestation は，あるシステムの持つハッシュ値を別の信頼できるシステムから遠隔で検証する手法である．なお，``信頼できる”とは，システムが期待通りに動作している，すなわちシステムが改ざんなどの攻撃を受けていないことを意味する．Remote Attestation を実現するための要素技術として，システム内の構成要素を正しく計測してそれを安全に保存する Trusted Boot，認証要求に応じて，自身のハッシュ値を返す通知，あらかじめ認証側で保存しているハッシュの期待値との比較を行う認証がある．なお，単純に通知をするだけでは Attestation の対象が送ったハッシュ値が本当に Attestation 対象の値なのか判断できない．そこでハッシュ値に署名を施して出力し，その署名に対応する証明書を検証者が保持することでハッシュ値が正規のものであることを第 3 者が検証することが可能となる．次節では，Attestation の高度化・高効率化に関する研究を紹介する．

2.2 Remote Attestation の高度化・高効率化

本節では，Attestation を対象にした既存の研究例を紹介する．まず，文献[7, 8]では，数多くのデバイスをいかに効率よく Attestation するかを議論している．スマートシティなどの公共系 IoT システムの様に端末数が大規模となるシステムではスケーラビリティは避けられない課題となる．スケーラビリティを改善するために，文献[7, 8]ではデバイスの接続関係をツリー状にしてツリーの子の Attestation 結果を親でまとめて認証者へと送るという方式を提案している．これによって，認証に必要となる時間計算量を線形オーダから log オーダに削減することが可能となる．この他にも Attestation を実現するために必要な最低限のアーキテクチャを検討し，組み込み系のデバイスへと実装した報告[9]や，Physical Unclonable Function (PUF) と呼ばれるデバイスに搭載されるチップの個体差 (トランジスタ特性など) を利用してデバイスを Attestation する手法[10]も検討されている．

しかし，文献[7-10]の手法はデバイスの内部プロセスが時間とともに悪意あるプロセスへと変化することは考慮していない．そのため，デバイスの運用中に悪意あるデバイスへ変化した場合はそれを取り除くことができない．これは，既存の Attestation が想定している信頼モデルが 0/1，つまり信頼できる/できないが明確であるため一度コンポーネントに対して信頼関係を構築できれば，以後もその信頼関係が成立するためである．この課題を解決するためには，まずシステムやシステムのコンポーネントに対する信頼モデルそのものを変える必要がある．次節にてその詳細を議論する．

3. 段階的信頼モデル: IoT システム向けの信頼モデルの提案

本節では，従来から検討されている IT システムにおける信頼モデルを IoT 向けに拡張することを考える．一般的な IT システムでは，起動時に一度 Attestation を行い，その結果が正しければ永続的に対象のコンポーネントを信頼するという 0/1 でコンポーネントの信頼を定義できる．つまり，

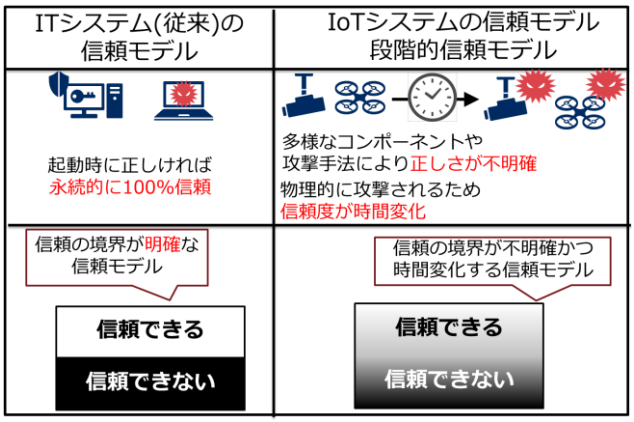


図 1　IT システムと IoT システムにおける
信頼モデルの違い

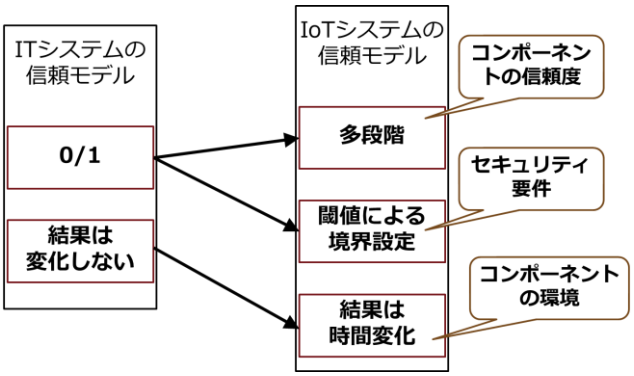


図 2 信頼モデルの変化

IT システムの信頼モデルは，信頼の境界が明確な信頼モデルであるといえる．しかし，IoT システムは IT システムとは異なり，(a) 物理的に攻撃されやすい，(b) システムを構成するコンポーネントやコンポーネントへの攻撃手法が多種多様という特徴がある．そのため，一度コンポーネントを Attestation して信頼を構築しても，永続的に対象のコンポーネントを信頼できるわけではない. そのため単純な 0/1 の信頼モデルではなく，コンポーネントの信頼を段階的に(時間・空間的変化を考慮して)表現するモデルが必要である．

そこで本研究では，信頼の境界が不明確かつ時間変化する信頼モデル(段階的信頼モデル)を提案する(図 1)．段階的信頼モデルでは，コンポーネントの正しさを段階的に表現するために, コンポーネントの信頼度という値を導入する．コンポーネントの信頼度はコンポーネントの環境や状態によって定義される．そのため，信頼度は時間の関数となる(時間変化とともに増減する). また, 不明確となった信頼の境界を決定づけるために，閾値と呼ばれるパラメータを導入する．閾値の決定には，システムの管理者やユーザの望むセキュリティ要件を組み込む．つまり，システムを構成するコンポーネントに対して，どれだけの信頼度を望むのかを閾値によって決定づける．以上，IoT システムの信頼モデルの IT システムの信頼モデルからの変化を図 2 にまとめる．

4. 段階的信頼モデルに基づいた真正性検証手法

本節では，第 3 節にて提案した段階的信頼モデルに基づいた Attestation として，コンポーネントの信頼度に応じて Attestation 実施が必要か不要かを判定する方式を提案する．また，提案手法のパラメータについてその設計方針を述べる．

4.1 概要

本節では，提案手法の概要を説明する．提案手法では，時刻 t におけるコンポーネントの信頼度 $T(t)$を定義し，そ

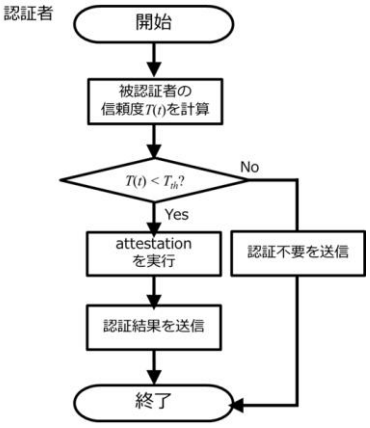


図 3 提案手法のフローチャート

の値を元に Attestation を行うか否かを判定する．具体的には，被認証者のコンポーネントに設定されている信頼度が自身の想定する信頼度の閾値 T_{th} より低ければ，対象のコンポーネントを Attestation する．この手順によって，内部情報が動的に書き換えられた不正なコンポーネントをシステムから排除することが可能となる．認証者が実施する動作のフローチャートを図 3 に示す．認証者は被認証者の信頼度 $T(t)$を計算し，管理者の設定した閾値 T_{th} と比較する．$T(t) \geqq T_{th}$ であれば認証不要を被認証者に送信し，そうでなければ認証装置に実装されている Attestation プロトコルを実行する．なお，信頼度や認証判定機能を改ざんや漏えいなどの脅威から守るためにも，コンポーネントの持つ TPM や TEE などのセキュア領域にそれらを実装することを推奨する．

4.2 提案手法におけるパラメータ設計

本節では提案手法のパラメータであるコンポーネントの信頼度 $T(t)$とその閾値 T_{th} について設計方針を検討する．

4.2.1 信頼度の設計

まず，$T(t)$の前提について述べる．本稿では，大きく 4 つの前提を置く: (1) 信頼度は[0, T_{up}]の値をとる (T_{up}:信頼度の上限. 例えば 100[%]など), (2) 初期値 $T(0)$は T_{init} とする (例えば 50[%]など), (3) Attestation が実施されなければ $T(t)$は時間経過とともに減少する, (4) Attestation が実施されると $T(t)$は増加 (回復) する．なお，パラメータ設計を簡略化するために $T_{up} = T_{init}$ としてもよい. T_{up} と T_{init} の設計については，管理者の運用ポリシに従うものとする．

次に，信頼度 $T(t)$をどのようなルールで更新するかについて，対象とする IoT システムの重要度の観点から定性的に議論する．例えば，発電所などの重要インフラに関連する施設に構築される IoT システムの場合は，不正なコンポーネントの攻撃によって発生するインシデントがインフラ停止につながることから，頻繁な Attestation が必要であると考えられる．一方，農地の温度センサや湿度センサなどのシステムの場合は, センサの持つ計算資源が貧弱であり，

表 1 システムの重要度に応じた信頼度増減の定性評価

	重要度 高	重要度 低
Tの増加	小	大
Tの減少	大	小

かつ攻撃者も攻撃対象として選びにくい(攻撃するメリットが少ない)ことを考えるとそれほど頻繁な Attestation は不要と考えられる．以上の検討から，システムの重要度に応じた信頼度の更新方法は，表 1 のようにまとめることができる．

次に，信頼度 $T(t)$を具体的に計算する場合の変数について議論する．本稿では，時間に依存する変数と，時間に依存しない変数とに分類する．まず，時間依存する変数の例として，以下がある:

(1) 時間に依存する変数
- (1-1) 最終 Attestation からの経過時間
- (1-2) (モバイルコンポーネント) システムの重要な場所からの距離
- (1-3) (モバイルコンポーネント) コンポーネントの隣接コンポーネント数

ここで，(1-1)は時間の経過に伴い単調増加するが，その他は単調には変化しないと考えられる．そこで本稿では，議論の簡単化のため $T(t)$の計算のベースとして最終 Attestation からの経過時間を利用する．次に，時間依存しない値として，以下が考えられる:

(2) 時間に依存しない変数
- (2-1) (固定コンポーネント)システムの重要な場所からの距離
- (2-2) (固定コンポーネント)コンポーネントの隣接コンポーネント数
- (2-3) コンポーネントに実装された CPU の種類
- (2-4) コンポーネントが利用できる機能の種類

時間に依存しない変数を導入することで，コンポーネントの状態を Attestation に反映することが出来る．次に，信頼度の時間変化を数式によって具体的に記述することを考える．時間変化による減少を表す数式の代表例として，例えば以下が考えられる．

$$T(t) = T(0) - \alpha t \ (\alpha > 0) \qquad (1)$$

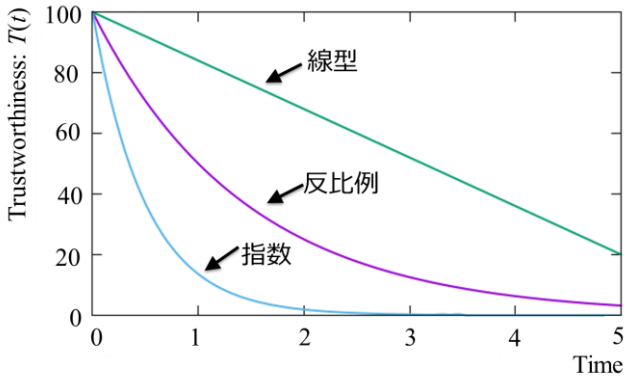


図 4 減少の数値計算例

$$T(t) = T(0) / (\alpha t + 1) \ (\alpha > 1) \qquad (2)$$

$$T(t) = T(0) \exp(-\alpha t) \ (\alpha > 0) \qquad (3)$$

それぞれ, 式(1)が線型的な減少，式(2)が反比例的な減少，式(3)が指数的な減少を意味する．ここで，t は前回の attestation からの経過時間，αは減少の程度を表す時間依存しないパラメータである．一方, Attestation による回復した後の信頼度の例としては以下が考えられる．

$$T(t) + \beta \ (\beta > 0) \qquad (4)$$

$$\beta T(t) \quad (\beta > 1) \qquad (5)$$

ただし，上限 T_{up} を超える場合は T_{up}

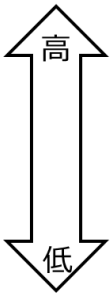
それぞれ，式(4)が線型的な増加， 式(5)が定数倍的な増加を意味する．また，増加後の信頼度が上限を超えた場合には信頼度を上限値としている．βは増加の程度を表す時間に依存しないパラメータである．ここで，図 4 に減少例を示す．図 4 において，横軸が Attestation からの経過時間，縦軸は信頼度の値を意味している．図 4 より，関数形によって時間とともに減少する信頼度の値は異なることがわかる．

次に，数式に出てくるパラメータαとコンポーネントが持つ時間依存しない値((2-1)から(2-4))との紐付を考える．簡単な方法として, (2-1)から(2-4)を数値化して((2-3)については，例えば CPU に応じた脆弱性の数などを利用)ベクトルとして並べることが考えられる．このベクトルはデバイスの状態を表したベクトルと見なせるため，以後状態ベクトル $\boldsymbol{s}$ と呼ぶ．そして定義した状態ベクトルからαを計算すればよい．ただし，αはスカラーであるため $\boldsymbol{s}$ をスカラー値に変換する必要がある．そこで本稿では一例として以下のような変換関数 $q(\boldsymbol{s})$を考える．

$$q(\boldsymbol{s}) = \sum_i w_i s_i \qquad (6)$$

なお，式(6)において s_i は状態ベクトルの第 i 要素を，w_i はその要素の重みをそれぞれ意味する．$q(\boldsymbol{s})$を用いてαを表現

表 2 成熟度と閾値の対応例



成熟度	閾値 (%)
5	15
4	30
3	45
2	60
1	75

することで，信頼度の減少速度の程度にコンポーネントの状態を反映することが可能となる．最も簡単な例としては $\alpha = q(\boldsymbol{s})$である．一方，パラメータ$\beta$は Attestation によって信頼度が回復する程度を表すパラメータであるため，その値は TPM を用いた Attestation や TEE を用いた Attestation，セキュアハードウェアを用いないソフトウェア Attestation など，Attestation の確かさを基に適切に設計することが考えられるが，具体的な議論は今後の課題とする．

上記には一例を示したが，実際にはシステムの管理者によって適切にパラメータを選択することが望まれる．また，状態ベクトルの要素の数値化や，$T(t)$，$q(\boldsymbol{s})$の関数形に関する議論は今後の課題とする．

4.2.2 閾値の設計

本節では，Attestation 実行を判定するための閾値 T_{th} について議論する．T_{th} の設定に関する課題として，T_{th} を大きくし過ぎると頻繁に Attestation を実行することになり，計算負荷が増大する．一方で閾値が小さい場合は Attestation の実行までにある程度信頼度が減少している必要があるため，不正なコンポーネントの検知に時間がかかる．つまり，閾値の設定においては，計算負荷と検知に要する時間とのトレードオフが存在する．Attestation にかかわらず，古くから閾値に対するトレードオフ問題に対して議論がされており，これまで大きく 2 種類の閾値設定方式が検討[11, 12]されている．1 つは固定閾値方式である．この方式は，閾値をある適当な固定値で設定するものであり，実装が容易である．しかし，運用開始初期などの固定閾値を決定するのが困難な場合がある．もうひとつは統計的閾値方式と呼ばれ，標準偏差などから統計学に基づいて理論的に閾値を算出する方式である．この方式では，母集団として定期的に観測/収集したデータを閾値の計算に利用する．そのため，一定時間ごとに閾値の値が変化する．また，運用時に大域的な (全コンポーネントの) 情報が必要となる．しかし，スマートシティなどの大規模 IoT システムでは大域的情報を取得することは難しいことから，本稿では固定閾値方式を想定して議論を進める．

一般に，固定閾値はシステムの管理者によって設定される．つまり，管理者のコンポーネントに対する安全性の要求レベルに基づいてコンポーネント単位，もしくはシステム単位で設定する必要がある．ここで，要求レベルを決める際の指標としてコンポーネントそのものの重要度や，コンポーネント上を流通する (処理される) データの重要度などが考えられるが，これら重要度を定量化 (数値化) することが必要である．重要度の数値化を議論したものとして，NIST SP 800-26 (現在 SP 800-53A[13]の付録) や NIST SP 800-55[14]がある．これらは，システムへの質問事項からセキュリティに関して 5 段階の成熟度指標を計算することを提言している．従って，本稿でも成熟度をもとに閾値を定義することを考える．例えば表 2 の様に対応付けることが一例として考えられる．なお，成熟度は管理者がシステムの運用状況に応じて変化させてもよいものとする．ここで，上記の NIST による提言に記載されている成熟度計算に用いる指標の例をいくつか記しておく．

- メンテナンスを受けているシステムコンポーネントの割合
- 過去に情報システムを設置している施設に対して未認可の入場を許したインシデントの割合
- システムが許容できる脆弱性の割合
- 最新のパッチが適用されたコンポーネントの割合
- FIPS 140-2 認定暗号化モジュールをすべての暗号処理に使用しているコンポーネントの割合

以上のように，多くがシステムの大域的な情報を必要としている．そのため，今後はシステムの局所情報から閾値を計算する方法の検討や，そもそもの閾値の意味づけを議論する必要がある．

表 3 各 Case における攻撃への対応

手法	対応
Case1	× (検知不可)
Case2	△ (参入する頻度に依存)
Case3	○ (検知可能)

5. 議論

本節では，提案するモデルの有効性を議論する．まず，想定する脅威 (攻撃)として，マルウェアなどの悪意を持ったソフトウェアがユーザの気づかないうちに秘密裏にダウンロードされ，インストール/実行される攻撃を考える．これ以外にも，IoT システムのコンポーネントは IT システムのコンポーネントに比べて攻撃者が直接物理的に攻撃しやすい．そのため直接コンポーネントのファームウェアを書き換えることによって，動的にマルウェアをインストールすることも可能である．

仮にマルウェアがコンポーネント上で動作している場合，それはユーザから見ると意図しないプロセスとして見える．

そのため Attestation を実行すると，異常なプロセスが動いている結果 PCR 値が変化しているため，異常を検出できる．そこで，Attestation の戦略として，既存の手法を含めた以下の 3 つを考える．

Case1: 新規コンポーネントを一度 Attestation したら二度と Attestation しない
Case2: 新規コンポーネントが参入するたびに全コンポーネントを再度 Attestation する
Case3: 信頼度に基づいてコンポーネントを Attestation する (本提案)

それぞれの Case における攻撃への対応を表 3 にまとめる．Case1 の場合は，仮に一度 Attestation を行ってもシステムの運用中にコンポーネントが悪意あるコンポーネントへと変化するため，対応ができない．Case2 では，コンポーネントの新規参入のタイミングによって，この攻撃に対応できる可能性があるが，新規参入が頻発しなければ対応は難しい．しかし Case3 では，時間とともに変化する信頼度に基づいてコンポーネントの Attestation を行うため，仮にシステム運用中にコンポーネントがマルウェアに感染して悪意あるコンポーネントとなったとしても検出が可能となる．従って，段階的な信頼モデルをうまく考慮した Attestation が実現できる．

6. おわりに

スマートシティをはじめとした公共系 IoT システムでは，多種多様なコンポーネントがシステム内にある程度自由に参入可能である．これをシステムへの攻撃者の立場で考えると，不正なコンポーネントが容易にシステムへ侵入可能であることを意味する．この対策として，コンポーネントに対する Attestation があるが，既存の Attestation ではコンポーネントの内部状態が時間とともに変化することは想定していないため，一度信頼する/しないのどちらかの判定が出ると以後その結果は時間変化しないという 0/1 の信頼モデルが前提であった．しかし，IoT システムのコンポーネントではこの単純な 0/1 の信頼モデルに基づいて対応できず，多段化の信頼モデルが必要となる．そこで本研究では，IoT システム向けの信頼モデルとして，段階的信頼モデルを提案し，段階的信頼モデルに基づいた Attestation の初期検討として，コンポーネントの信頼度に基づいた Attestation 実行タイミングの設定方法を提案した．提案手法を用いることで時間とともにコンポーネントが悪意あるコンポーネントに変化したとしても，それを検出できることを議論した．これによって，IoT システムに不正なコンポーネントが侵入することを防ぐことが可能となる．今後は段階的信頼モデルの理論的考察やパラメータ設計の詳細検討，および実機やシミュレーションを用いた提案手法の有効性評価を行う必要がある．

参考文献

[1] H. Arasteh, V. Hosseinnezhad, V. Loia, A. Tommasetti, O. Troisi, M. Shafie-khah, and P. Siano``IoT-based smart cities: A survey,'' Proc. IEEE 16th EEEIC 2016, pp.1–6, 2016.
[2] L. Atzori, A. Iera, and G. Morabito, ``The internet of things: A survey,'' Comput.Netw., vol.54, no.15, pp.2787–2805, 2010.
[3] Trusted Computing Group, available at http://www.trustedcomputinggroup.org/, cited 2017.
[4] V.L. Le, I. Welch, X. Gao, and P. Komisarczuk, ``Anatomy of drive-by download attack,'' Proc. AISC 2013, vol.138, pp.49-58, 2013.
[5] W. Arthur, D. Challener, and K. Goldman, A practical guide to TPM 2.0—Using the new trusted platform module in the new age of security, Apress, 2015.
[6] GlobalPlatform white paper, ``The trusted execution environment: Delivering enhanced security at a lower cost to the mobile market,'' 26pages, 2011.
[7] N. Asokan, F. Brasser, A. Ibrahim, A.-R. Sadeghi, M. Schunter, G. Tsudik, and C. Wachsmann, ``SEDA: Scalable embedded device attestation,'' Proc. ACM-CCS2015, 2015.
[8] M. Ambrosin, M. Conti, A. Ibrahim, G. Neven, A.-R. Sadeghi, and M. Schunter, ``SANA: Secure and scalable aggregate network attestation,'' Proc. ACM-CCS2016, 2016.
[9] K.E. Defrawy, A. Francillon, D. Perito, and G. Tsudik, ``SMART: Secure and minimal architecture for establishing a dynamic root of trust,'' Proc. NDSS 2012, 15pages, 2012.
[10] J. Kong, F. Koushanfar, P.K. Pendyala, A.-R. Sadeghi, and C. Wachsmann, ``PUFatt: Embedded platform attestation based on novel processor-based PUFs,'' Proc. ACM-DAC 2014, pp.1–6, 2014.
[11] J.S. Weszka, ``A survey of threshold selection techniques,'' Computer Graphics and Image Processing, vol. 7, pp.259–265, 1978.
[12] P.K. Sahoo, S. Soltani, A.K.C. Wong, and Y.C. Chen, ``A survey of thresholding techniques,'' Computer Vision, Graphics, and Image Processing, vol. 41, pp.233–260, 1988.
[13] Joint Task Force Transformation Initiative, ``NIST Special Publication 800-53A rev.4,'' 2013.
[14] E. Chew, M. Swanson, K. Stine, N. Bartol, A. Brown, and W. Robinson, ``NIST Special Publication 800-55 rev.1,'' 2008.