Computer Security Symposium 2020
26 - 29 October 2020

PUF の偽造防止技術に関する考察

大井 駿平 [1,*]　金田 北洋 [1]　岩村 惠市 [1]

概要： IoT の普及が進み，デバイス認証や本人認証技術の必要性が高まることが考えられる．そこで，微小に異なる複製困難な特徴からその個体に固有の値を出力する技術である Physical Unclonable Function(以後 PUF)が注目されている．PUF によってパスワードを覚える必要がなくなり，かつ強力なランダム性を持つ鍵を生成できる．この鍵の一致検証によって不正な個体から得た鍵では鍵の検証を通過できず，正当な個体から得た鍵のみが検証を通過するので，認証や真贋判定に利用できると期待されている．しかし,生成された鍵はあくまでもただの数値であって保存することは容易である．したがって PUF を用いて認証，真贋判定を行う場合に一度でも正しい鍵によって検証を行った者については鍵の値を保存しておくことで，鍵の偽造攻撃が可能であると考えられる．本論文では PUF の出力に含まれるノイズを除去して安定的に鍵を生成する手法であるファジーエクストラクタ(Fuzzy Extractor)やパターン照合鍵生成など関連手法についてまとめ，PUF から得た鍵を用いて検証を行うプロトコルについて前述の問題点を考え，鍵の偽造防止に関して考察を行う．

キーワード：PUF, ファジーエクストラクタ, 認証, 真贋判定

Stuady of forgery prevention for Physical Unclonable Function

Shunpei Ohi[1,*]　Kitahiro Kaneda[1]　Keiichi Iwamura[1]

Abstract: The spread of IoT increase the necessity for authentication technology. Therefore, Physical Unclonable Function (PUF), which is technology that outputs a unique value from a characteristic that is slightly difference for each individual and is difficult to copy is useful. PUF can generate keys with strong randomness. It is expected that key can be used for authentication because only the key obtained from the valid individual passes the verification. However, the generated key by PUF is just a numerical value and it is easy to save it. Thus, it is considered that a key forgery attack is possible for a person who has verified with a correct key even once. In this paper, we summarize related techniques such as fuzzy extractor and pattern matching key generation and consider the above-mentioned problems with the verification protocol to discuss the prevention for key forgery attacks.

Keywords: PUF, Fuzzy Extractor, Authentication, Authenticity determination

1. はじめに

近年，IoT(Internet of Things)を活用して人々の生活を豊かにしようという動きが話題になっている．IoT の普及が進むことで，インターネットに接続される機器は増加していき，例えば自宅の家電がインターネットに接続されるようになれば外出先からスマートフォンで家電を操作したり，センサーから得たデータの確認，また家電間の相互の情報のやり取りが実現することが考えられる．しかし，多くのシステムがインターネットに接続されることで，それらのシステムに対するサイバー攻撃のリスクが高まるという側面もある．よって，個人情報などが扱われる場合には特に IT セキュリティ技術の重要性が高まると考えられる．加えて，不正なユーザーやデバイスがシステムにアクセスできないようにするためにデバイス認証技術や本人認証技術の重要性も高まることが考えられる．現在では，ユーザーやデバイスの認証には事前に設定しておいたパスワードを用いることが一般的だが，パスワード中にユーザーの名前や生年月日が含まれている予測しやすい脆弱なパスワードを使用している人やパソコンに付箋でパスワードを書いたものを貼っている人，同じパスワードを複数の認証に用いている人は意外と多いのではないだろうか．インターネットに接続するモノが増加していく中で，パスワードの漏洩がもたらす損失はより大きなものになっていくことが考えられる．そこで人が設定した脆弱なパスワードに変わって PUF, バイオメトリクスによって個体毎に微妙に異なる特徴量から得られる固有値から鍵を生成して認証や真贋判定に利用する手法が注目されている．このような手法によって，設定したパスワード覚えておく必要がなくなり，かつ強力なランダム性を持つ鍵を利用できることが期待される．本論文では Dodis[1]らによって提案された

1 東京理科大学
Tokyo University of Science
* ohi_shunpei@sec.ee.kagu.tus.ac.jp

© 2020 Information Processing Society of Japan

Fuzzy Extractor や Paral と Devadas[2]らによって提案された Pattern Matching Key Generation に注目して，生成された鍵を使って認証を行う際に起こりうる問題点を提起しそれについての考察を行う．

コピー不可能な個体毎の特徴量を用いて鍵を生成することで，正当な個体からしか鍵を再現できないという点が PUF やバイオメトリクスの持つ安全性の根拠であるが，生成した後の鍵はあくまでもただの数値であり，保存することは容易である．個体の正当性を確かめるためには，手にした個体の特徴量から鍵を復元するだけではなくその後に，事前にその個体から生成された正しい鍵と比較して一致していることを確認するといった検証作業が必要になる．この時に鍵の復元を行った者が鍵を確認し保存できてしまうと個体が手元になくとも鍵を知っていることになってしまい鍵の安全性が損なわれ，以降使用することができなくなってしまう．よって，鍵を複数回の一致検証に使う場合の問題点について考える必要がある．

2. 予備知識

2.1 PUF(Physical Unclonable Function)

PUF とは，Physically Unclonable Function の略称で物理複製困難関数の事であり，複製困難な物理的特徴を利用して個体毎に異なる固有の値を出力する関数である．複製困難な物理的特徴としては人間の指紋や虹彩や半導体デバイスの閾値電圧などの微妙なばらつき，紙の繊維構造，不純物を含む光透過性デバイスを通過するレーザー光の拡散パターン等が挙げられる．理想的な PUF の特性として以下の 3 つの特性がある．

・再現性：同一の個体からは同じ個体別情報が出力される
・独自性：異なる個体からは全く別の個体別情報が出力される
・予測困難性：得られる出力は予測困難である．

上記の 3 つの特性によって個体識別が可能で認証や真贋判定に利用できることが期待されている．しかし，現実に利用する場合にはノイズの影響を受けやすいという問題点がある．よって後述するようなノイズ除去技術と合わせて用いることが一般的になっている．

2.2 誤り訂正符号

誤り訂正符号とは，送信したい情報ビットに誤りを訂正するための検査ビットを加えて送信することで通信容量は大きくなってしまうが，ノイズの影響を受けて誤りが生じても受信側で整合性を調べることで誤りを検出し訂正を可能する技術である．普段，私たちが利用しているスマートフォンやパソコンなどでデータ通信を行う際の様々な伝送路で活躍している技術で，効率的な通信を実現する技術である．誤り訂正符号は情報源記号系列を一定長のブロックに区切り，各ブロックごとに符号化を行うブロック符号と情報源符号系列を一定長のブロックに区切らずに連続したままで扱う畳み込み符号の二つに大きく分けられる．ブロック符号としてはハミング符号や BCH 符号が，畳み込み符号としてはビタビ復号法が代表的な例である．

2.3 ハッシュ関数

ハッシュ関数とは要約関数とも呼ばれ，任意の長さの入力ビット列から規則性のない固定長のビット列を出力する関数である．ハッシュ関数から得られた出力をハッシュ値と一般に呼ぶ．ハッシュ関数は決定的であり，ある入力が与えられた時のハッシュ値は常に同じでなければならない．ハッシュ関数は，主にパスワードの保護やデータ比較処理の高速化，データ改ざんの検出に利用されている．ハッシュ関数の持つ特徴としては以下のものがある．

・衝突困難性：Hash(m)=Hash(n)となるような 2 つの異なる 2 入力 m,n を見つけることが困難である
・現像計算困難性：ハッシュ値 h から h=Hash(m)となるような任意の入力 m を見つけることが困難である
・第二現像計算困難性：任意の入力 m が与えられたとき，Hash(m)=Hash(n)となるようなもう 1 つの入力 n を見つけることが困難である

ハッシュ関数の代表例として，MD5，SHA-1，SHA-2 がある．

3. 関連研究

3.1 Fuzzy Extractor

PUF の持つ再現性と独自性の問題点を解決し，ノイズの影響を受けやすいデータから一様にランダムな分布を持つビット列に変換する誤り訂正符号に基づく手法が Dodis[1]らによって提案されている．論文中で Fuzzy Extractor と定義され，そのアプローチが注目されている．

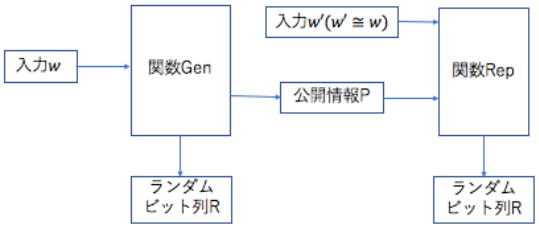

図 1　Fuzzy Extractor

Figure 1　Fuzzy extractor

図 1 に Fuzzy Extractor の簡易的なブロック図を示した．Fuzzy Extractor は鍵の生成手順と復元手順の 2 つに分けられ，生成手順の関数 Gen(Generation)と復元手順の関数 Rep(Reproduction)の 2 つの関数から構成される．ハッシュ関数と誤り訂正符号技術を構成中に含み，データのノイズ耐性と出力されるビット列の一様なランダム性を実現する

アルゴリズムである．2 つの関数は論文中で以下のように定義される．

・関数 Gen

ノイズを含むデータを入力 w として，公開可能な情報 P とランダムなビット列 R を出力する．公開情報 P は元の入力 w を復元するための補助データとして関数 Rep で入力とする．また，ランダムなビット列 R は秘密情報として鍵に利用する．

・関数 Rep

関数 Gen で生成した公開情報 P と元の入力 w に近い w' の 2 つを入力とする．入力された w'と元の入力 w のハミング距離が使用する誤り訂正符号の訂正可能範囲内である場合，関数 Gen で生成したものと同様なランダムなビット列 R を出力する．

これによって，ノイズの影響を受けやすいデータからノイズを除去し，双方の手順で鍵として利用する共通のビット列 R を得ることができる．誤り訂正符号によってノイズを除去し，鍵が復元成功する確率は一般的に10^{-6}以下になるように設計されることが多い．

これまでに Fuzzy Extractor の安全性に関しての重要な 2 つの要素 robustness と reusability についての改良手法が提案されている．前者の安全性を満たしたものは，Robust Fuzzy Extractor[3][4]と呼ばれ、正しい入力からでも公開情報が改ざんされた場合には鍵の復元を行うことができないといった問題点を改善している．また後者の安全性を満たしたものは，Reusable Fuzzy Extractor[5][6][7][8]と呼ばれ，相関関係のある入力から発行されたいくつかのペア（鍵と公開情報）が攻撃者に明らかにされたとしても，Fuzzy extractor が安全なままであれば再利用可能に改善している．再利用性によりユーザーの一部の公開情報および鍵が侵害された場合でも認証システムが将来の使用に対して安全であることを保証することができる．

次に Fuzzy extractor の基本的な構成例について紹介を行う．

3.1.1 Fuzzy Extractor の鍵生成の構成例

Fuzzy Extractor において鍵生成を行う関数 Gen の構成例を以下の図 2 に示す．

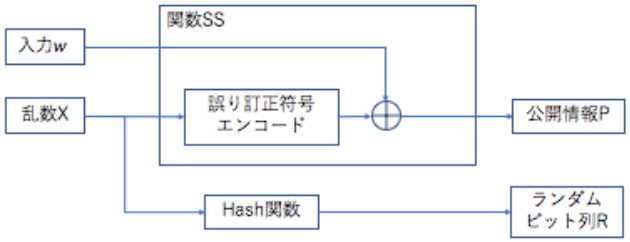

図 2　関数 Gen の構成例

Figure 2　Example of function Gen

鍵を生成するため，PUF 出力を w とし，乱数 X を用意する．関数 SS の処理は以下のように行う．

・関数 SS

① 用意した乱数 X と入力 w を入力とする

② 乱数 X に対して誤り訂正符号をエンコードする
(誤り訂正符号エンコード関数を C( )とする)

③ ②で算出した C(X)と入力 w を XOR で加算する

④ 計算結果 P= $w \oplus C(X)$を公開情報とする

鍵として利用するランダムビット列 R を乱数 X をハッシュ関数に入力して得られる出力 R= $Hash(X)$とする．

生成手順における最終的な出力は以下のようになる．

公開情報 P= $w \oplus C(X)$

非公開情報（鍵） R= $Hash(X)$

3.1.2 Fuzzy Extractor の鍵復元の構成例

Fuzzy Extractor において鍵復元を行う関数 Rep の構成例を以下の図 3 に示す．

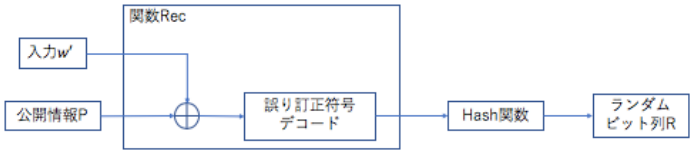

図 3　関数 Rep の構成例

Figure 3　Example of function Rep

元の入力 w にノイズの影響が加わった新たな入力 w'(w' ≅ w)と生成手順で生成した公開情報 P= $w \oplus C(X)$を組み合わせて，共通のビット列 R を復元する．関数 Rec の処理は以下のように行う．

・関数 Rec

① 公開情報 P= $w \oplus C(X)$と入力 w'を XOR で加算する

② 加算結果は$w \oplus w' \oplus C(X)$となる．$w \cong w'$であるからwとw'を XOR で加算した場合，wとw'の各ビットが異なる部分のみ 1 となり構成要素がほとんど 0 となる微小な誤差と考えられ，$w \oplus w' = \Delta$とする

③ 加算結果$w \oplus w' \oplus C(X)$に対して誤り訂正符号をデコード（誤り訂正符号デコード関数を D()とする）するとΔが使用する誤り訂正符号の訂正可能範囲内であれば D($w \oplus w' \oplus C(X)$)= D($C(X) \oplus \Delta$)= Xとなり，元の乱数 X を復元される．

復元した乱数 X をハッシュ関数に入力することで R= $Hash(X)$となり，鍵であるランダムビット列 R を復元することができる．以上の手順から元の入力 w に近い入力 w' から鍵を復元することができる．

3.2　PMKG(Pattern Matching Key Generation)

PUF を用いたパターン照合による鍵生成手法である PMKG が Paral と Devadas[2]らによって提案された．他の手法 Fuzzy Extractor などと PMKG の最大の違いは，PUF

の弱点であるノイズの除去を処理が重いとされる誤り訂正符号に頼らずにハミング距離の比較のみによって実行できる点が挙げられる．これまでに PMKG に対する様々な改良手法や攻撃手法が提案されている．代表的なものに太田，崎山[3]らによって提案されている Single-Round Circular PMKG(SC-PMKG)が複数ラウンドの操作をなくし，manipulation attack に対して有効な手法がある．以下，PMKG の原点となった手法における鍵の生成・復元手順を太田，崎山[9]らを参考に示す．

PMKG(Pattern Matching Key Generation)は生成手順と復元手順に分けられる．それぞれ以下の関数を用いる．

・PUF(Physical Unclonable Function)：入力 x に対して y という文字列を出力する

・CS(Challenge Sequencer)：インデックス ind とフラグ flag を入力とし，PUF に入力する x を出力する

・KGF(Key Generation Function)：インデックスの集合{ind}を入力とし，鍵 key を出力する

・pattern match：公開メモリ(Public-NVM)に保存された補助データである PUF 出力の一部を少しずつずらして PUF 出力に照合する．ハミング距離がある所定の閾値未満である，全ての部分文字列候補の中で距離が最小である位置を探してインデックス ind とする．

3.2.1 PMKG の鍵生成手順

生成手順の概要図を以下の図 4 に示す．

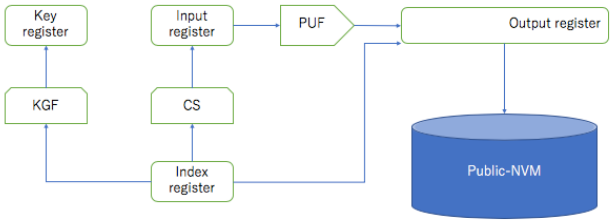

図 4　PMKG の鍵生成

Figure 4　Key generation of PMKG

生成手順では，任意の r 回の操作を繰り返す．r 回の操作を行うとすると，インデックス ind が r 個得られることになる．インデックス ind の 0 番目は CS への初期入力であり，あらかじめ公開メモリに保存される．

鍵の生成は以下の手順で進められる．

① フラグ flag を 1 にする

② 以下の手順を r 回繰り返す（i 番目の操作を示す）

(a) i-1 番目のインデックス ind_{i-1} とフラグ flag を CS に入力し，PUF への入力 x_i を得る

(b) x_i を PUF へ入力し，出力 y_i を得る

(c) Index register で i 番目の乱数 ind_i を生成する．PUF の出力 y_i のインデックス ind_i ビット目からある固定長のビット列を抜き出して z_i とし，公開メモリに保存する．

③ r 個のインデックスを KGF に入力し，鍵 key を算出する

3.2.2 PMKG の鍵復元手順

復元手順の概要図を以下の図 5 に示す．

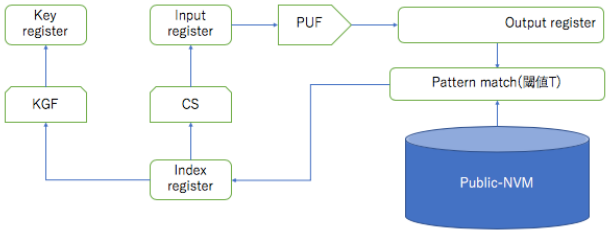

図 5　PMKG の鍵復元

Figure 5　Key reproduction of PMKG

復元手順も同様に r 回の操作を繰り返す．r 回の操作を行うことで，インデックス ind を r 個得て，それらを用いて鍵 key の復元を行う．また PUF を用いるので，出力にノイズが加わることを考慮し，生成手順とは文字を区別して表現することとする．

鍵の復元は以下の手順で進められる．

① フラグ flag を 1 にする

② 以下の手順を r 回繰り返す（i 番目の操作を示す）

(a) i-1 番目のインデックス ind'_{i-1} とフラグ flag を CS に入力し，PUF への入力 x'_i を得る

(b) x'_i を PUF へ入力し，出力 y'_i を得る

(c) 公開メモリから z_i を読み出す

(d) パターン照合によって出力 y'_i の中で z_i とのハミング距離がある所定の閾値 T 未満である，全ての部分文字列候補の中で距離が最小である位置を探してインデックス ind'_i とする．また，条件を満たすインデックスが検出されなかった場合には，インデックス ind'とフラグ flag を 0 に設定する．

③ フラグ flag が 1 であるとき，r 個のインデックスを KGF に入力し，鍵 key を復元する．flag が 1 でない場合，中断する．

パターン照合によってノイズの影響を除去し，鍵の復元が可能である．ここで重要になるのは，閾値 T の設定である．閾値 T が低い場合にはノイズの影響に対応できず一致検出ができない，また閾値 T が高い場合には部分文字列候補が多くなってしまうことが考えられる．よって閾値 T の適切な設定をどうするのかを考える必要がある．

4.　本論文で提案する問題点

PUF などから得たノイズの影響を受けやすいデータから鍵を生成する技術である Fuzzy Extractor や PMKG などの手法に注目して，その構造や問題点などを整理した．しか

し，生成手順と復元手順で同じ鍵 R（ビット列 R）を生成できることは示されているが，双方の手順で生成された鍵 R が一致しているかを検証しなければ，個体の正当性はわからない．したがって，鍵 R の一致検証を行う際に起こりうる問題点などを新たに考える必要がある．

4.1 鍵の一致検証時における偽造攻撃

まず Fuzzy Extractor で生成した鍵 R の一致検証について図 6 に示した Fuzzy Extractor の構成図を用いて説明する．

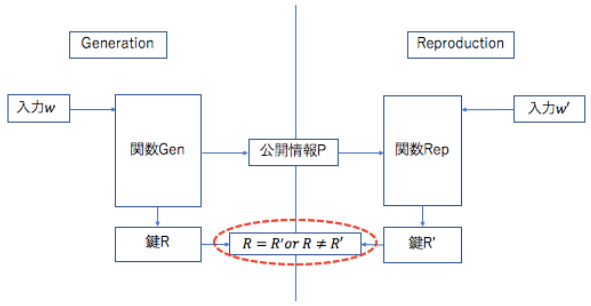

図 6　鍵 R の一致検証

Figure 6　Verification of key R

物や人物の正当性を確かめるためには Fuzzy Extractor の鍵復元処理を行った後に，最終的に図 6 の赤い丸で示したように生成手順で得た鍵 R と復元手順で得た鍵 R'が一致しているかをなんらかの方法を用いて検証する必要がある．その結果として R=R'ならば正規の個体であることが，　R≠R'ならば不正な個体であることが分かる．しかし，鍵 R'の復元を行った者にとって，自身が持つ鍵 R'が正しい鍵 R と一致したということは正しい鍵 R を知ることになる．ここで鍵 R を複数回にわたって一致検証に利用する場合に鍵 R'を保存することで偽造攻撃が可能になってしまう．また，PMKG による手法で生成された鍵についても同様の問題点が考えられる．

以下の図 7 を用いて鍵の偽造攻撃モデルについて示す．

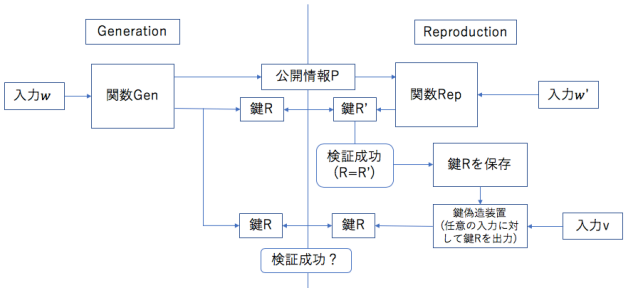

図 7　想定される攻撃モデル

Figure 7　Expected attack model

鍵の偽造攻撃の手順に関して，ここでは鍵の偽造を行う者を攻撃者とし，鍵の生成手順については省略する．

① 攻撃者は個体情報から得た入力 w'(ただし$w' \cong w$とする)と生成手順で出力された公開情報 P を関数 Rep に入力する

② 攻撃者は得られた鍵 R'を用いて正しい鍵 R と一致しているか確認し，正当性を調べる

③ 元の入力に近い w' ($w' \cong w$)から鍵 R'が生成されているので鍵が一致し，正しい個体であることが証明される

④ 証明後，攻撃者は自身が持つ鍵 R'(R'=R)を保存しておく

⑤ 任意の個体情報から得た入力を v とすると，入力 v に対して保存しておいた鍵 R が出力されるように装置などを偽造する

⑥ 鍵 R は一致してしまうので任意の個体であっても正しい個体であることが証明される

上記の一連の手順から一度でも鍵 R の一致検証を成功させている攻撃者はそれ以降，鍵 R を偽造できてしまうことがわかる．また，攻撃者によって鍵 R を拡散されてしまうことも考えられ，同じ鍵 R を複数回使って個体の正当性を証明することはできない．PUF などによって複製困難な各個体の微小なばらつき，特徴量を利用して，Fuzzy Extractor や PMKG で鍵を生成することで鍵が不正に複製されないということが最大の特徴かつ，安全性の根拠であるが，鍵を確認し保存できてしまっては悪用されてしまうので対策を考える必要がある．

4.2 鍵の偽造攻撃への考察

4.1 で示した偽造攻撃が起こらないために必要な条件は以下の 2 つがあると考えた．二つの条件のうちどちらか一方でも満たしていれば，安全に検証できると考えられる．

条件 1：生成手順と復元手順を行う者が常に同一であること

まず，条件 1 について説明する．生成手順と復元手順を行う者が同一の場合には，そもそも自身が持っている鍵 R と鍵 R'を比較して一致しているかを確認すれば良いだけであり，不特定多数の者に鍵 R が知られることがないので鍵の一致検証は安全に行えると考えられる．この条件 1 では，従来のパスワードのように余計に記憶しなければならない情報を減らすことができるというユーザーにとってのメリットは残るが，鍵 R を復元するための補助データを公開できてリモート認証ができるという強みが消えてしまうことになる．生成と復元が同一の者のみで完結しているならば，わざわざ補助データを公開するリスクを負う必要がないので，秘密情報として自身のみが保持して利用する方が安全である．よって Fuzzy Extractor や PMKG を使う必要性が低下してしまうので，条件 1 は FE や PMKG の利用にあまり適さないことがわかる．

条件 2：鍵 R が一度，一致検証を行うと効力を失う使い捨ての鍵であること

次に条件 2 について説明する．偽造される問題が発生するのは同じ鍵を使って複数回の一致検証を行う時であった

ので，そもそも一度でも鍵が一致したという検証結果が得られた時点で鍵は効力を失う使い捨ての鍵として扱う場合には，特に問題はなく安全に鍵の一致検証を行えると考えられる．実用上，商品の流通などを想定すると消費者が購入したものが正当なものであること証明したいといった状況では一度きりの鍵でも有効である．しかし，消費者が鍵の検証を行って，失敗した場合にどの中継業者で偽造品へのすり替えが発生したのか特定することはできないことになる．また，アカウント認証のようなものを想定すると認証が成功するたびに鍵を変更するための登録を何度もやり直す必要があり，ユーザーにとって手間が多く実用的でない．広く応用できるように鍵の一致検証は複数回行える方が便利なことは明確である．

鍵の一致検証を安全に行える条件について考えたが，広く応用ができる認証技術としては正しい個体情報を持つあらゆる人が鍵の復元をすることができ，同じ鍵を使って複数回の一致検証ができることが望ましいことがわかった．そこで，復元した鍵を保存できないように鍵の復元を行う者が鍵の値を知らずに正しい鍵と一致するかのみを知ることができる構成を実現できれば誰でも，何度でも鍵の一致検証が行えると考えた．鍵の偽造攻撃によって，認証や真贋判定を行うシステムとしては鍵の偽造ができてしまうということは大きな問題である．

5. 従来の認証プロトコル

4 章で述べた問題点について，これまでに提案されている認証プロトコルでは起こりうるのかを検討する必要がある．具体的なプロトコルにおいて問題点が発生するかを考え，その結果から問題点を一般化することを目的に考察を行った．

5.1 BlockPro

Javaid[10]らによって提案されている BlockPro という手法について考える．BlockPro はブロックチェーンベースのデータの出所と整合性の保証を提供する PUF を用いた IoT 環境向けのプロトコルである．本手法では，ブロックチェーンプラットフォームの一つであるイーサリウムのスマートコントラクトと PUF を利用して，データ来歴の保証と不変のストレージプラットフォームを提供し，以下のセキュリティを提供することを目的としている．

・データの整合性を維持するためのブロックチェーンベースの手法により，未登録デバイスの不正な参加をブロックする

・制御と信頼のない IoT 環境向けの分散型ネットワークモデルを実現

・IoT デバイスに一意の ID を提供する

また，重要な前提の一つに PUF と IoT デバイスはシステムオンチップであることが記載されている．

BlockPro のネットワークモデルを以下の図 8 に示す．

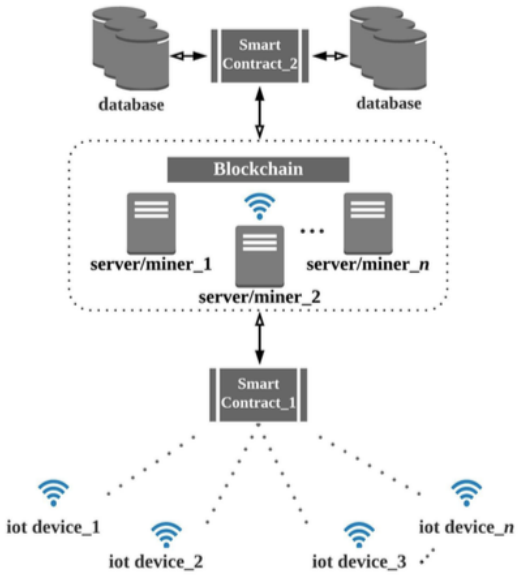

図 8　BlockPro ネットワークモデル[10]

Figure 8　The BlockPro network model

・Smart Contract 1

IoT デバイスと相互作用して，それらが正当なデバイスであることや送信されたデータが信頼できる発行元からのものであることを確認する．

・Smart Contract 2

ブロックチェーン上のデータの保存と取得を担当する．

BlockPro における認証プロトコルについて説明する．まず，事前処理として登録したいデバイスの PUF チャレンジレスポンスペア(CRP)とイーサリウムアドレス(ID)をブロックに保存する．これらの登録された情報は Smart Contract 1 内にリスト化して保持される．

次に PUF を使ったチャレンジレスポンスプロトコルの概要図を以下の図 9 に示す．

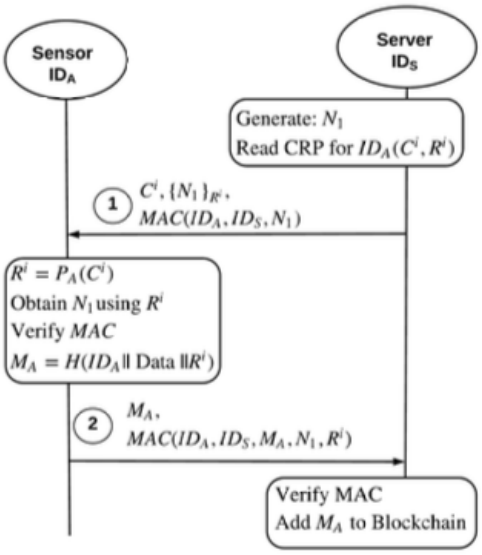

図 9　PUF チャレンジレスポンスプロトコル[10]
Figure 9　PUF challenge-response protocol

PUF チャレンジレスポンスプロトコルは以下の手順で行われる．以下の説明でのチャレンジレスポンスペア(C,R)とイーサリウムアドレス(ID)は事前処理で登録されたものであり，固定の値である．

① ID_Sを持つネットワーク内のサーバーは，ID_Aを持つデバイスのチャレンジレスポンスペア CRP(C,R)を読みとり，ナンス N1 を生成する
② ID_Sを持つサーバーは，レスポンス R を使用して暗号化されたナンス N1 とチャレンジ C をメッセージ 1 としてID_Aを持つデバイスに送信する
③ メッセージ 1 を受け取ったID_Aを持つデバイスは，PUF を使用してチャレンジ C に対するレスポンス R を取得する
④ レスポンス R を取得した後，ID_Aを持つデバイスは次の手順を実行する
(a)レスポンス R を秘密鍵として使用してナンス N1 を復号する
(b)パラメータを使用して物理認証子 MAC を検証する
(c) ID_A，送信データ，レスポンス R をハッシュ関数に入力し，得られたハッシュ値 M_A をメッセージ 2 でID_Sを持つサーバーに送信する
⑤ ID_Sを持つサーバーがメッセージ 2 を受診すると MAC 及びハッシュ値 M_A をチェックし，検証する．双方が有効である場合，データの送信リクエストは承認される．それ以外の場合，リクエストは却下される

5.2 物体指紋を利用した認証

佐古，石山らによって物体指紋技術[11]を用いたサプライチェーン管理手法が提案されている[12].

・物体指紋技術

物体指紋技術とは，物体が持つ固有の物理的特徴を画像から抽出し，認証に用いる技術の一つである．人間の指紋のように，モノを一つ一つ識別するような特徴量を抽出し，照合する．

上記の技術とブロックチェーンを組み合わせた管理システムとなっている．物体が持つ固有の物理的特徴を用いているので，広い意味で PUF と考えられる．消費者の評判の高い生産者の生産物を購入したいニーズや小売業者の消費者に正しい情報を提供することによる付加価値をつけることができるインセンティブ，生産者は自信のある生産物を登録しないと評判が落ちるリスクがある．本システムは消費者が正しい生産者を確認できることを支援するシステムである．

生産者がブロックチェーンに生産物を登録するフェーズと消費者が購入しようとする生産物の生産者を確認するフェーズがある．登録フェーズは以下の手続きからなる．

① 生産者が登録用アプリで，登録したい生産物の画像を撮影する．これを登録生産物画像と呼ぶ
② 登録用アプリは生産者を認証し，登録生産物画像にユニークな採番をする．これを生産物番号と呼ぶ
③ 登録用アプリは生産者の秘密鍵を使用し，生産物番号，登録生産物画像，生産者 ID，生産物情報（品種・産地・収穫日など）に対してデジタル署名を付与して登録スマートコントラクトに登録を依頼する
④ 登録スマートコントラクトは，デジタル署名が正しいこと，生産物番号に重複がないことを確認できたら，登録生産物画像から特徴量を抽出し，生産物番号を鍵として生産者 ID と生産物情報と特徴量をブロックチェーンに登録する
⑤ 生産者は登録用アプリに表示された登録済みの生産物番号を記録し，生産物に番号札を添付し，市場に卸す

照合フェーズは以下の手続きからなる．

① 消費者が照合アプリで，生産物の画像を撮影し，番号札に記載の生産物番号を入力
② 照合アプリは，入力画像と生産物番号をブロックチェーン上の照合スマートコントラクトに照合を依頼
③ 照合スマートコントラクトは入力画像から特徴量を抽出し，生産物番号で登録されている特徴を読み出し，正当性を検証する．類似度が設定した閾値以上であれば，「照合成功」と生産者 ID 及び生産物情報を照合アプリに通知する
④ 照合アプリは，スマートコントラクトの処理結果を受け取り，表示する

提案システムの代表的な特徴は以下のものがある．

・登録用アプリが生産者を認証するのでなりすましを防止できる
・ブロックチェーンによってデータ改ざん耐性が高い
・照合にスマートコントラクトを採用しているので，照合アルゴリズムが不正に改ざんされることがない
・不正に改ざんされた照合アプリが撮影していないで照合依頼をすることを防げない

5.3 従来の認証プロトコルに対する考察

BlockPro では，理想的な PUF を想定している．事前に登録していたチャレンジ C をデバイス側で改めて PUF に入力した場合には，サーバーに登録されたレスポンス R と同じものが得られることはノイズの影響があるので考えにくい．よってノイズ除去の仕組みである Fuzzy Extractor や PMKG などを組み込む必要がある．これは多くの理論ベースのプロトコルに言えることであるが，これは問題点というわけではなく．ただ実用上では，Fuzzy Extractor や PMKG といった手法を使う必要性があることを示してい

る．

本論文で述べた問題点としては，正しいデバイスの認証に成功した者が鍵を知ってしまうことで以降，鍵の偽造を行える可能性がある点である．BlockPro では，鍵はレスポンス R に相当する．正しいデバイスの認証を行った者は，そのデバイスについて登録されているレスポンス R を知っていることになる．サーバーに事前処理で登録されているアドレス，チャレンジ C とレスポンス R を使い続けることで正しいデバイスつまり，正しい PUF が手元になくともレスポンス R を用意できることになり認証の手順を進めることができる．BlockPro では，デバイスと PUF が切り離すことができないシステムオンチップである前提がある．物体指紋技術を用いた提案システムの特徴として，不正に改ざんされた照合アプリが撮影していないで照合依頼をすることを防げないと記述されていました．つまり一度でも過去に照合成功させている者は照合成功となる画像を知っていることになり，毎回その画像を出力するような装置を用意できれば，照合結果を偽造できてしまう．

ここで問題視したいことは認証プロトコルにおいて，事前に登録した固定のチャレンジレスポンスペアや鍵を単純に複数回にわたって使う手法では，広く応用することは難しいと考えられることである．そのような PUF による認証は調査した範囲の中だけでも数多く提案されている．その中で Fuzzy Extractor や PMKG を利用して公開可能な補助データを作成し，4.2 で示した条件 1 を満たさない場合，つまりリモート認証を複数回可能にすることは難しいと考えられる．一方で，条件 2 を満たす形でチャレンジレスポンスペアや鍵を同一の個体に対して複数個用意しておく PUF による認証は可能である．あらかじめ認証回数が定まっているならばその回数分だけペアや鍵を用意しておけば良いが，不特定多数の認証が想定される場合には難しいものになる．

6. まとめ

個体ごとに微小に異なる特徴から，その個体に固有の値を出力する PUF を用いて個体の正当性を証明する鍵を生成する技術である Fuzzy Extractor や PMKG に注目し，調査を行った．鍵生成の手法について，鍵の一致検証を通過した場合に鍵が保存されてしまうことを考えると同一の鍵を用いて複数回の一致検証がなされる場合に鍵の偽造攻撃の可能性があることを示した．また，PUF を用いた認証プロトコルについて示したような鍵の偽造攻撃が起こりうるかを考察した．広く応用ができる認証技術としては正しい個体情報を持つあらゆる人が鍵の復元をすることができて，同じ鍵を使って複数回の一致検証ができることが望ましく，鍵の復元を行う者が鍵の値を知ることができずに正しい鍵と一致するかのみを知る構成を実現できれば誰でも，何度でも鍵の一致検証が行えると考えられる．

謝辞　本研究を行うにあたり，研究を進める上で実験や討論に協力していただいた研究室の皆様に深く感謝いたします．

参考文献

[1] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith, “How to Generate Strong Keys from Biometrics and Other Noisy Data” EUROCRYPT 2004,LNCS 3027, pp.523-540,2004

[2] Zdenek Paral, and Srinivas Devadas, “Reliable and efficient PUF-based key generation using pattern matching” in Proceedings of the 2011 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2011, pp.128-133,USA, June 2011

[3] Yevgeniy Dodis, jonathan Katz, Leonid reyzin, and Adam Smith, “Robust Fuzzy Extractor and Authenticated Key Agreement from Close Secrets” CRYPTO 2006, LNCS 4117,pp232-250, 2006

[4] Yunhua Wen, Shengli Liu, “Robustly Reusable Fuzzy Extractor from Standard Assumptions” ASIACRYPT 2018,LNCS 11274, pp459-489, 2018

[5] Jung Hee Cheon, Jinhyuck Jeong, Dongwoo Kim, and Jongchan Lee, “A Reusable Fuzzy Extractor with Practical Storage Size:Modifying Canetti et al.’s Construction” Springer International Publishing AG, part of Springer Nature 2018 ACISP 2018, LNCS 10946, pp.28-44, 2018

[6] Ran Canetti, Benjamin Fuller, Omer Paneth, Leonid Reyzin, and Adam Smith, “Reusable Fuzzy Extractor for Low-Entropy Distribution” EUROCRYPT 2016, Part 1, LNCS 9665, pp.117-146, 2016

[7] Yunhua Wen, Shengli Liu, Shuai Han, “Reusable fuzzy extractor from the decisional Diffie-Hellman assumption” Springer Science+Business Media, LLC, part of Springer Nature 2018

[8] Yunhua Wen, and Shengli Liu, “Reusable Fuzzy Extractor from LWE” Springer International Publishing AG, part of Springer Nature 2018 ACISP 2018, LNCS 10946, pp.13-27, 2018

[9] Yuichi Komano, Kazuo Ohta, Kazuo Sakiyama, Mitsugu Iwamoto, and Ingrid Verbauwhede, “Single-Round Pattern Matching Key Generation Using Physically Unclonable Function” Hindawi Security and Communication Networks Volume 2019, Article ID 1719585, 13pages.

[10] Uzair Javaid, Muhammad Naveed Aman, Biplab Sikdar, “BlockPro: Blockchain based Data Provenance and Integrity for Secure IoT Environments” BlockSys’18, November 4, 2018, Shenzhen, China, ACM, New York, USA, 6pages

[11] 石山塁,”物体指紋”を用いたモノの個体認証，日本セキュリティ・マネジメント学会誌，第 33 巻第 2 号 2019 年 9 月，pp.19-26, 2019.

[12] Kazue Sako, Keisuke Kajigaya, Rui Ishiyama, Toru Takahashi, “A Blockchain-based Supply Chain Management System using Object Fingerprints” SCIS2020 Symposium on Cryptography and Information Security Kochi, Japan, Jan.28-31, 2020