FIDO2における中間者攻撃の影響のProVerifを用いた検証

○西 総一朗*, 中井 雄士*, 鈴木 幸太郎*

*豊橋技術科学大学

FIDOフレームワーク

■パスワードに依存した認証は様々なセキュリティ問題を引き起こす

- 一部のユーザは脆弱なパスワードの使用やパスワードを使い回す
- 強力なパスワードであっても，フィッシング攻撃の被害にあう可能性がある

■パスワードに依存しない認証を目指す, FIDO [1]が2013年に提唱された

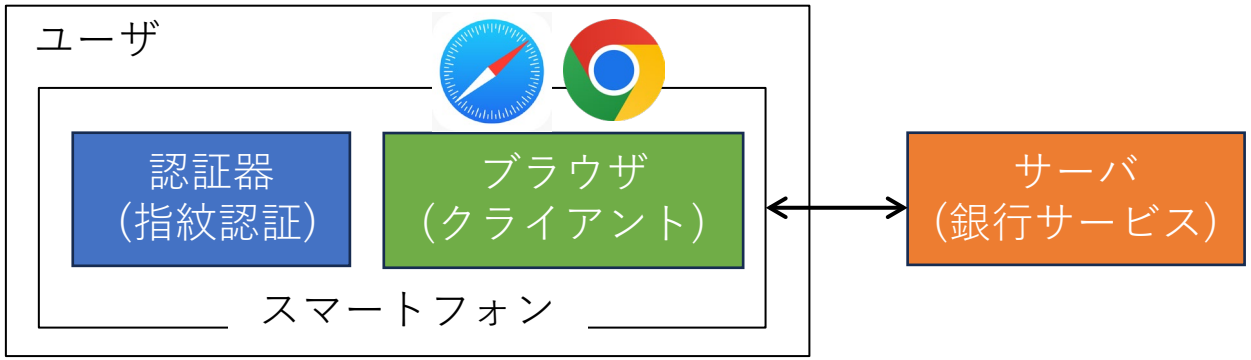
■次期バージョンとしてFIDO2 [2]が2018年に策定された

[1] Fast Identity Online (FIDO) Alliance. FIDO Authentication., https://fidoalliance.org/fido-authentication/, 2013.
[2] FIDO Alliance. FIDO2, https://fidoalliance.org/fido2/, 2018.

FIDOのユースケース

ユーザは銀行サービスにスマートフォンの指紋認証でログインしたい

1. 銀行サービスにパスワードでログイン
2. スマートフォンの指紋認証を銀行サービスに登録
3. 次回は指紋認証を使用してログイン



FIDO2プロトコル

FIDO2プロトコルはWebAuthnとCTAP2で構成

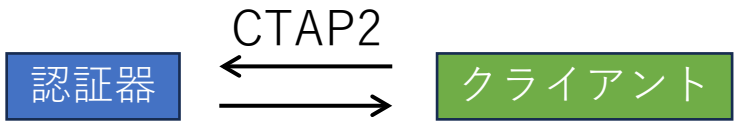
■WebAuthn [3]：登録された認証器を用いてユーザを認証する

- 認証器・クライアント・サーバの3者間プロトコル
- 認証器の登録プロセスとユーザ認証プロセス
- 用途ごとに複数のモードが存在
  - 認証器の検証をするアテステーションモード

認証器の例:

■CTAP2 [4]：WebAuthnで使用する鍵を共有

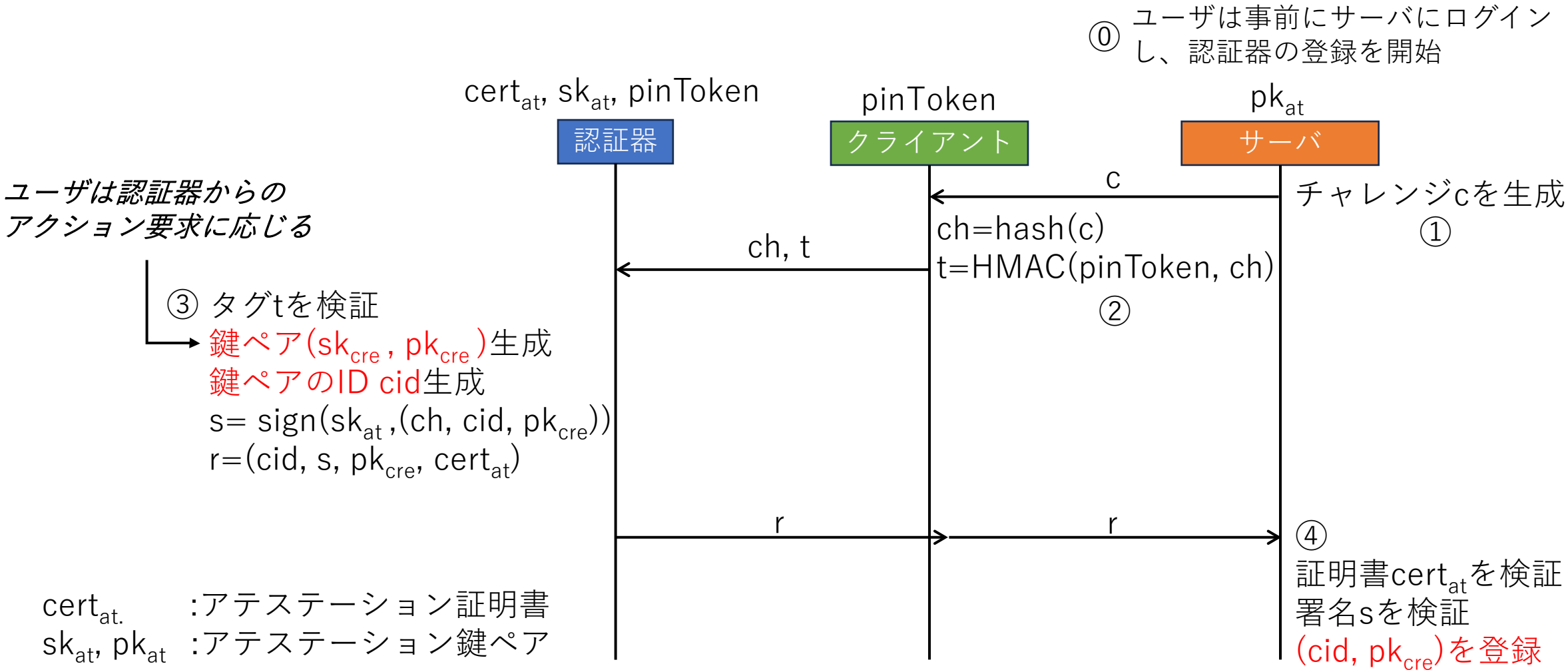
- 認証器・クライアントの2者間プロトコル



[3] W3C. Web Authentication: An API for accessing Public Key Credentials W3C Recommendation Level 2, 8 April 2021.
[4] FIDO Alliance. Client to Authenticator Protocol (CTAP) Proposed Standard, June 15, 2021.

WebAuthnの認証器登録プロセスの流れ

登録プロセス開始前に認証器・クライアント間でCTAP2を用いpinTokenを共有済みであるとする



アテステーションモード

■アテステーション証明書により認証器の登録プロセス時に，認証器が信頼できるかを検証する

- none: アテステーションを行わない
- **basic**: 製造時に埋め込まれたアテステーション証明書は**複数の認証器間で共有**
  - 市販されているUSBドングル認証器などで使用

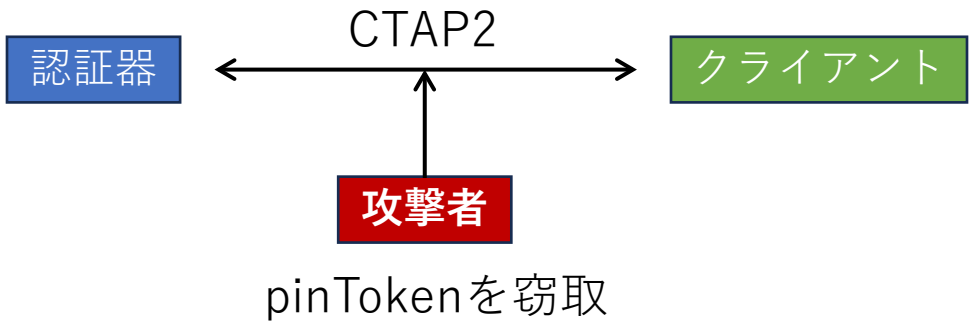
Yubikey

- self: 自己署名証明書を使用
- attCA: 認証局(CA)による署名が行われた証明書を使用
  - CAが署名を行った認証器のみをサーバに登録することができる

FIDO2の安全性に関する先行研究 (1/2)

■Barbosaら[5]によりCTAP2に中間者攻撃があることが指摘

- 未認証のDH鍵交換に基づいているため，中間者攻撃が可能
- 秘密であるべきpinTokenを攻撃者が取得できてしまう
- CTAP2.1でも修正されていない



[5] Manuel Barbosa, Alexandra Boldyreva, Shan Chen, and Bogdan Warinschi. Provable security analysis of fido2. pages 125–156, 2021.

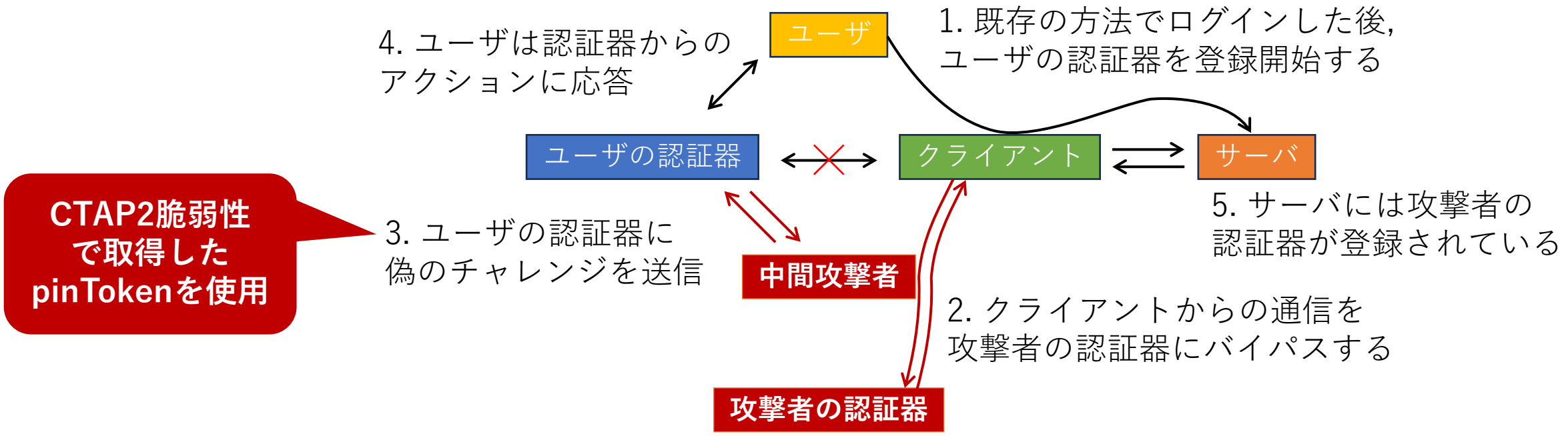
FIDO2の安全性に関する先行研究 (2/2)

■Guanら[6]はFIDO2についてProVerifを用いた安全性の検証を行い，
複数の攻撃とBarbosaらの中間者攻撃を示した

■佐藤，米山[7]はGuanらの形式化に存在したミスを修正し，
攻撃者の能力を弱めた上で新たな攻撃を発見した

[6] Jingjing Guan, Hui Li, Haisong Ye, and Ziming Zhao. A formal analysis of the fido2 protocols. page 3 – 21, 2022.
[7] 佐藤 瑞己, 米山 一樹. Fido2 の形式化の再考と複数モードの検証への拡張.日本応用数理学会, 数理的技法による情報セキュリティ研究部会, 第19回日本応用数理学会研究部会連合発表会, 2023.

本研究で発見した攻撃 – 概要

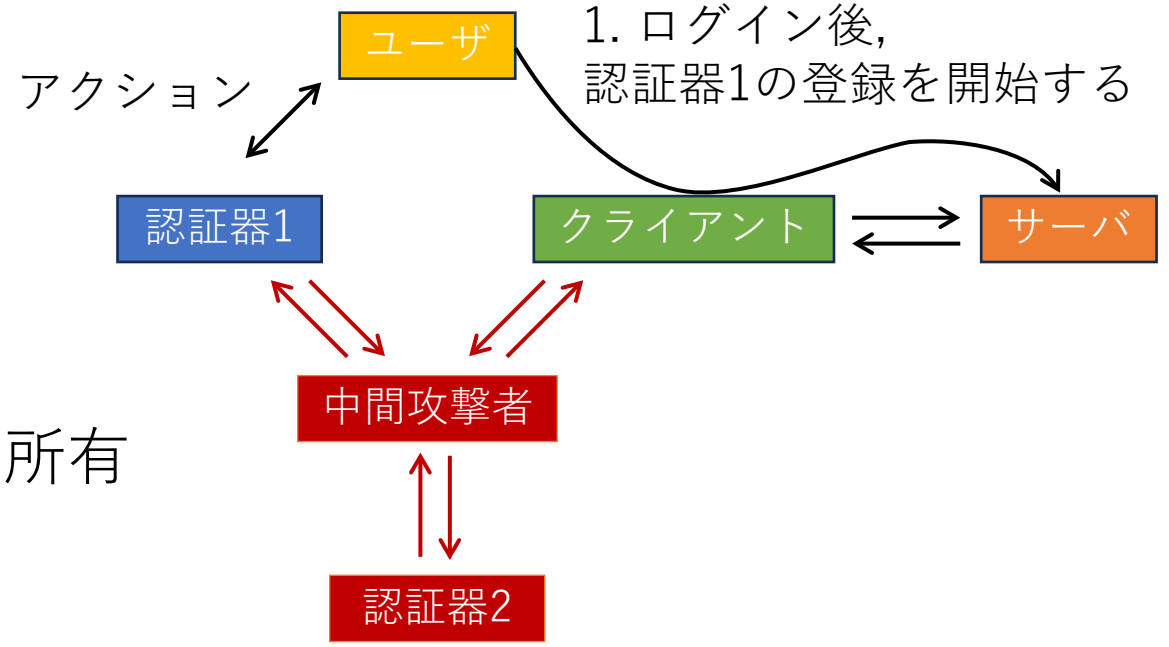


- ユーザは自身の認証器がサーバに登録されてると誤認している
- 実際には攻撃者の認証器が登録されている
- ProVerifでの検証により, 本攻撃を発見した

本研究でのFIDO2の形式モデル

■認証器が2台存在するモデルを新たに提案

- ユーザが意図した認証器がサーバに登録されているかを評価するため



■モデル設定

- ユーザは認証器1を, 攻撃者は認証器2を所有
- ユーザは認証器1の登録を開始
- 攻撃者は認証器1-クライアント間の中間者と仮定

安全性要件

■**要件**: ユーザが意図した認証器がサーバに登録されている

- 要件を検証するためには認証器の識別が必要
- しかし，basicモードでは認証器を識別可能なアテステーション証明書は複数の認証器で共有されている

■認証器に**擬似的**な識別子`AuthenticatorID`を追加

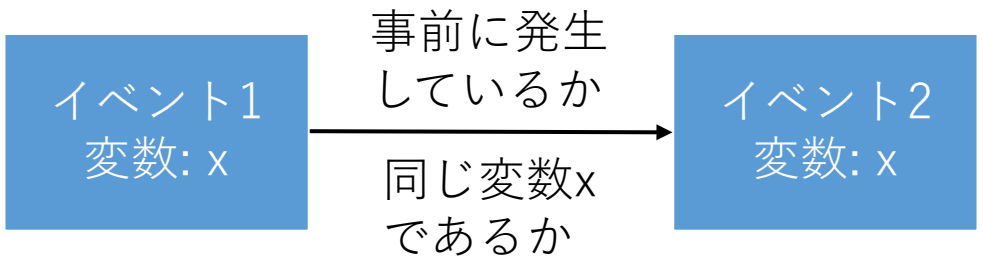
- 追加した識別子は通信に含まれず, プロトコルの動作に影響を与えない

安全性要件の形式化 (1/2)

■**要件**:登録プロセス前後のサーバ・認証器–クライアント間で
AuthenticatorIDの**認証性**が成り立つ

■**認証性**はProVerifで検証できるセキュリティ特性

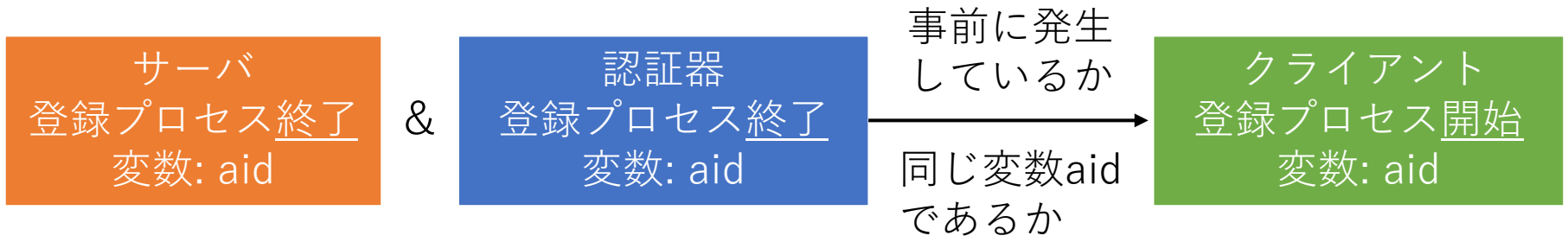
- イベント1の前に同じ変数でイベント2が発生しているとき認証性が成り立つ



安全性要件の形式化 (2/2)

■**要件**:登録プロセス前後のサーバ・認証器–クライアント間で AuthenticatorID (aid)の**認証性**が成り立つ

- 登録プロセスにおいてサーバ・認証器がaidで終了しているならば, クライアントもaidで登録開始していなければならない
  - クライアントがプロトコルを開始するため, クライアントを事前条件として表現



安全性要件のProVerifによる記述

■**要件**:登録プロセス前後のサーバ・認証器–クライアント間で

AuthenticatorID (aid)の**<u>認証性</u>**が成り立つ

- 登録プロセスにおいてサーバ・認証器がaidで終了しているならば,

  クライアントもaidで登録開始していなければならない

認証器での登録プロセス
<u>終了時</u>に発生

```
query aid: AuthenticatorID;
event(ServerFinishReg(aid)) && event(AuthenticatorFinishReg(aid))
      ==> event(ClientInitReg(aid)).
```

サーバでの登録プロセス
<u>終了時</u>に発生

クライアントでの登録プロセス
<u>開始時</u>に発生

擬似的な識別子 (1/2)

■2つの認証器プロセスの引数にそれぞれ固有のaid (authr1, authnr2)を与えることで，異なる認証器であることを表す

- authrn1: ユーザが所有する認証器
- authrn2: 攻撃者が所有する認証器

■クライアントプロセスの引数にauthrn1を与える

- クライアントがauthrn1で登録プロセスを開始したことを表す

```
const authnr1, authnr2 : AuthenticatorID.
…
Client(authnr1, …) | (* クライアントはユーザの認証器を登録しようとする *)
Authenticator(authnr1, …) | Authenticator(authnr2, …) | (* ユーザ・攻撃者の認証器 *)
```

擬似的な識別子 (2/2)

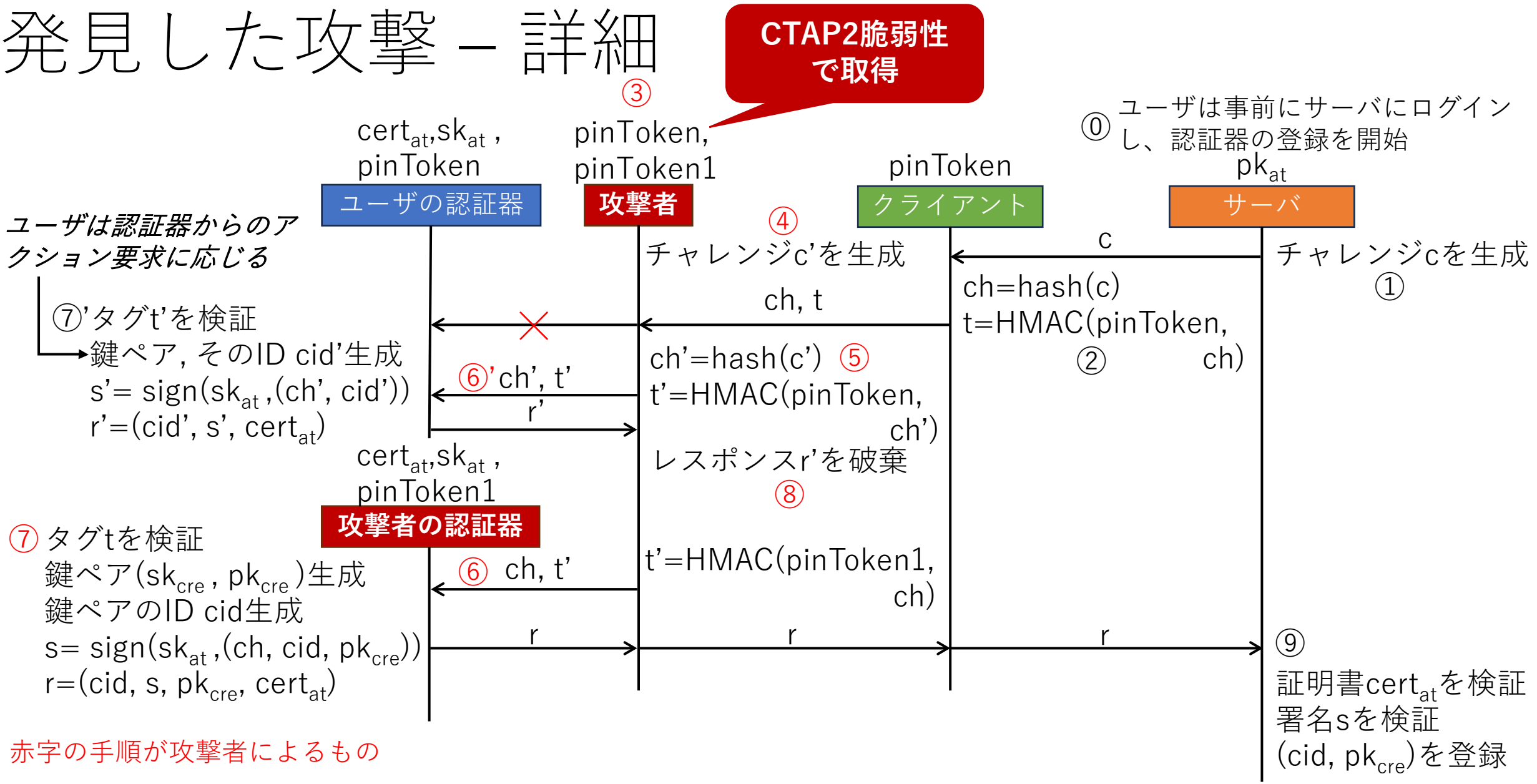
■以下2つの理由で, aidはプロトコルの実行に影響を与えない

- クライアント・認証器プロセスの引数aidはevent発生のみに使用
- サーバプロセスでは認証器プロセス内でテーブルに保存されたaidは, event発生のみに使用

```
let Client(aid: AuthenticatorID, ...) = (
        event ClientInitReg(aid);
        ...
).
let Authenticator(aid: AuthenticatorID, ...) = (
        ...
        insert AuthenticatorID_Record(aid);
        event AuthenticatorFinishReg(aid)
).
```

```
let Server(...) = (
  ...
  get AuthenticatorID_Record(aid) in
  ...
  event ServerFinishReg(aid)
).
```

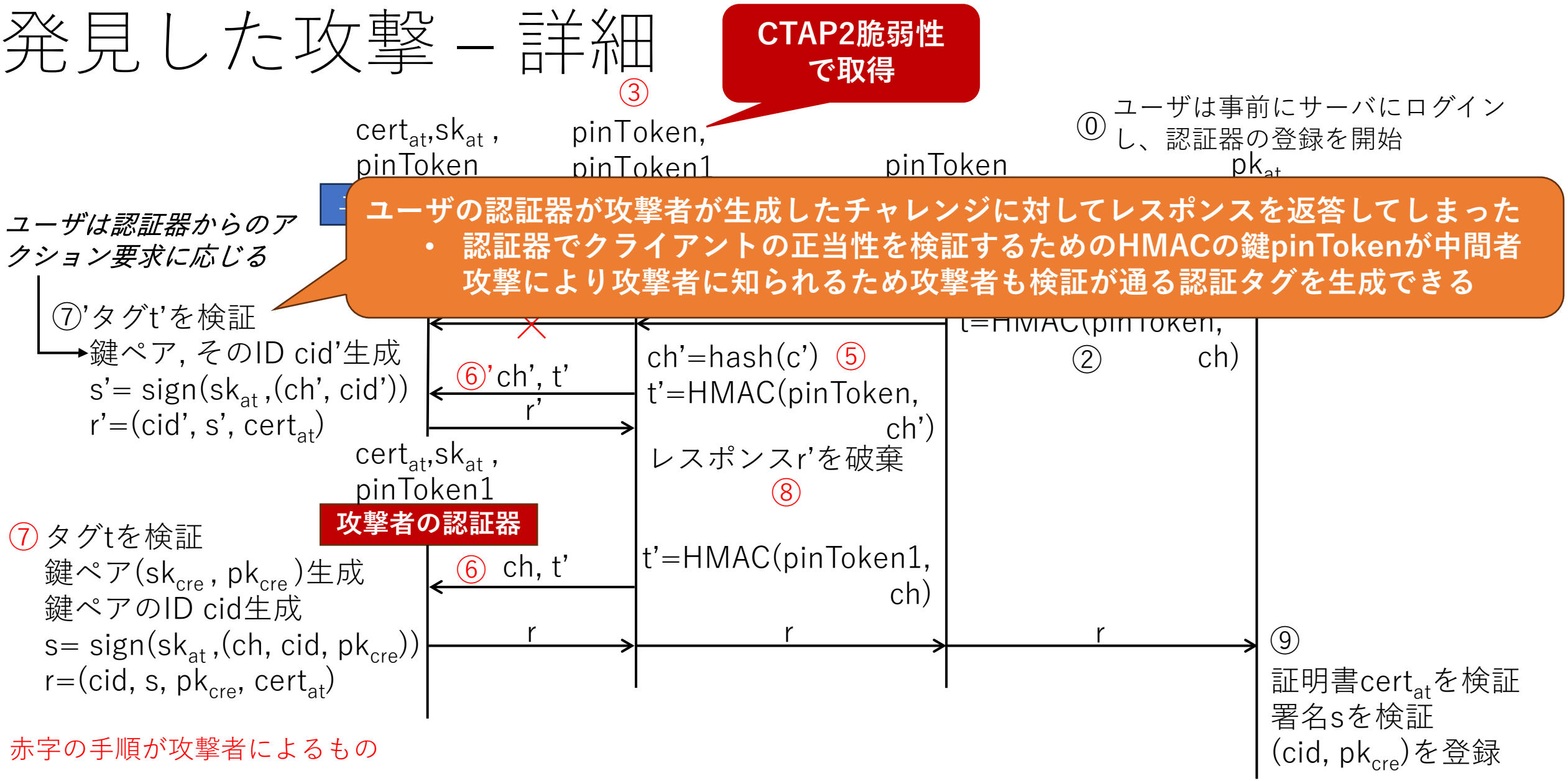
発見した攻撃 – 詳細



赤字の手順が攻撃者によるもの

<u>ユーザは自身の認証器を登録したつもり</u>だが，実際には**<u>攻撃者の認証器が登録</u>**されている

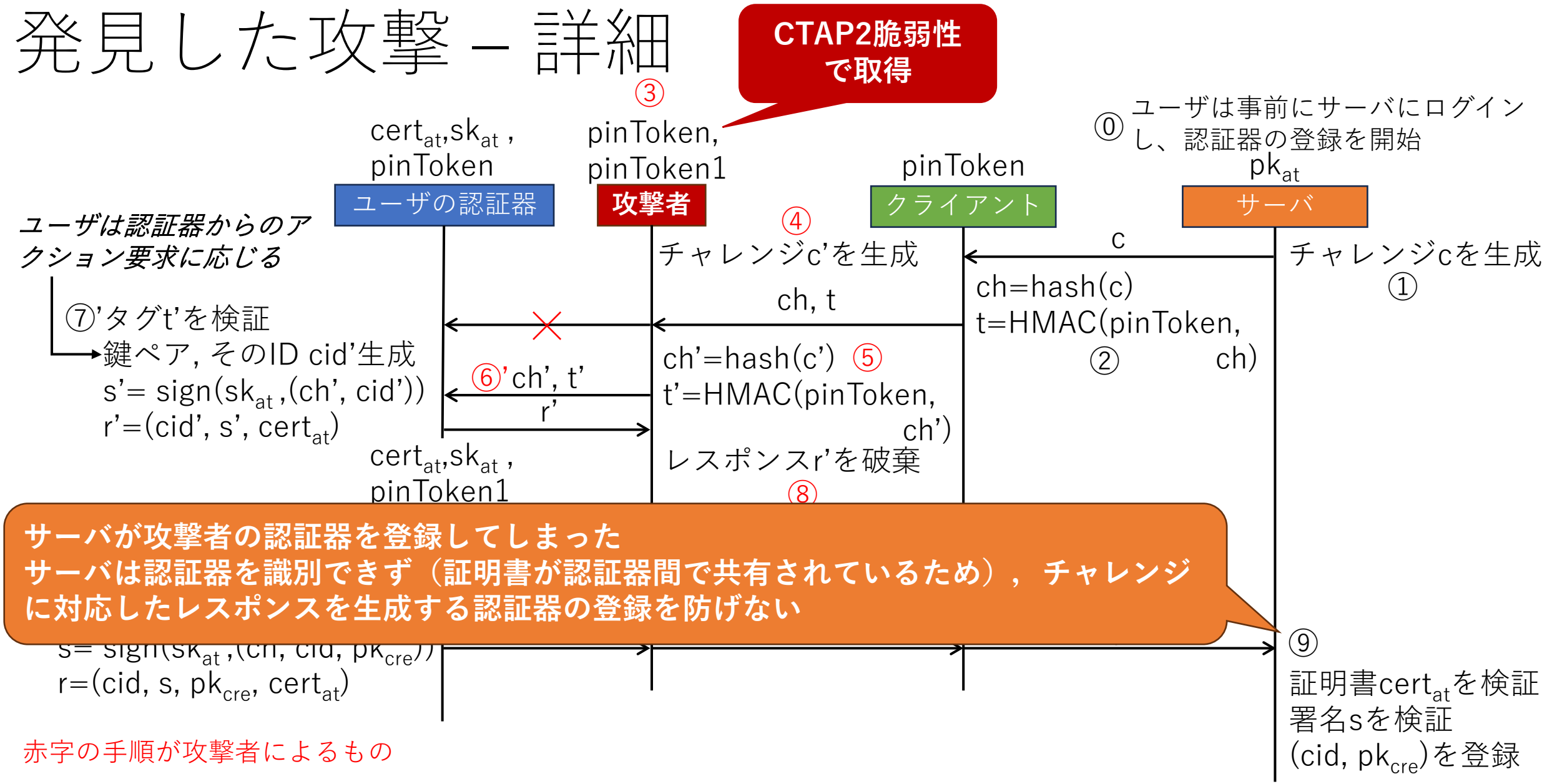
発見した攻撃 – 詳細



赤字の手順が攻撃者によるもの

ユーザは自身の認証器を登録したつもりだが，実際には**攻撃者の認証器が登録**されている

発見した攻撃 – 詳細



<u>ユーザは自身の認証器を登録したつもり</u>だが，実際には**<u>攻撃者の認証器が登録</u>**されている

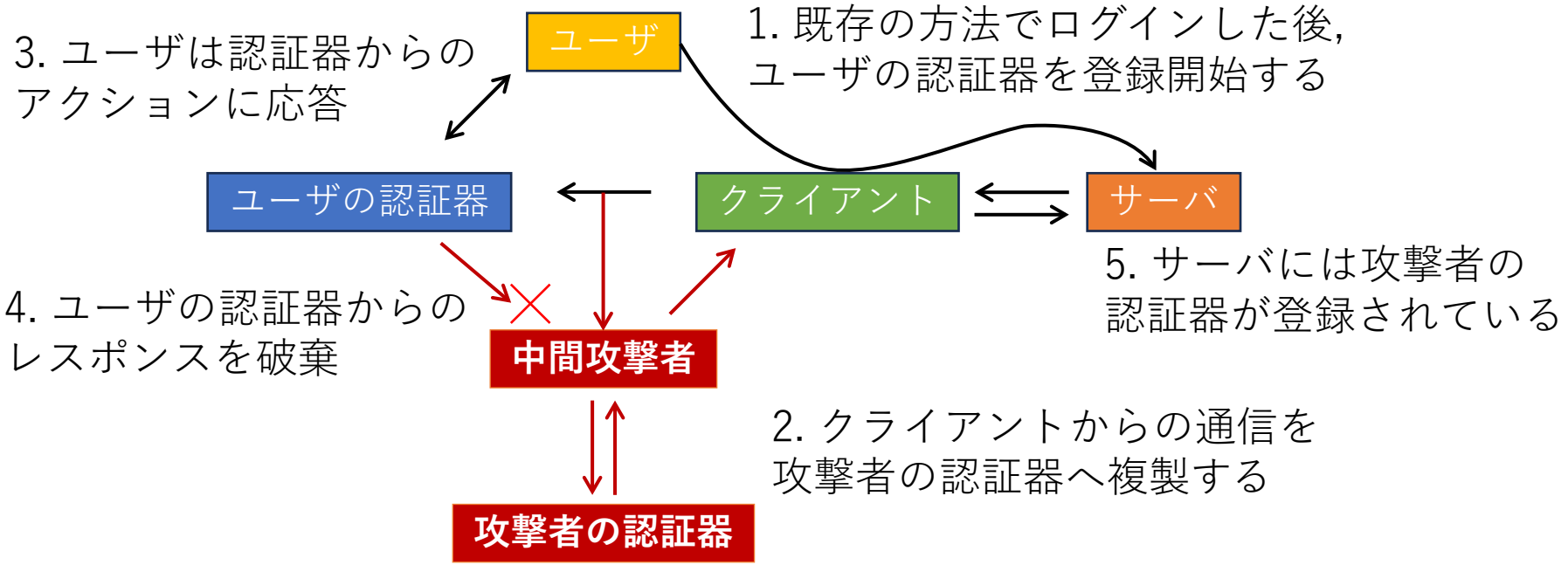
CTAP2の脆弱性が修正された場合

■ CTAP2の脆弱性が修正された場合で検討を行ったところ同様の攻撃が可能

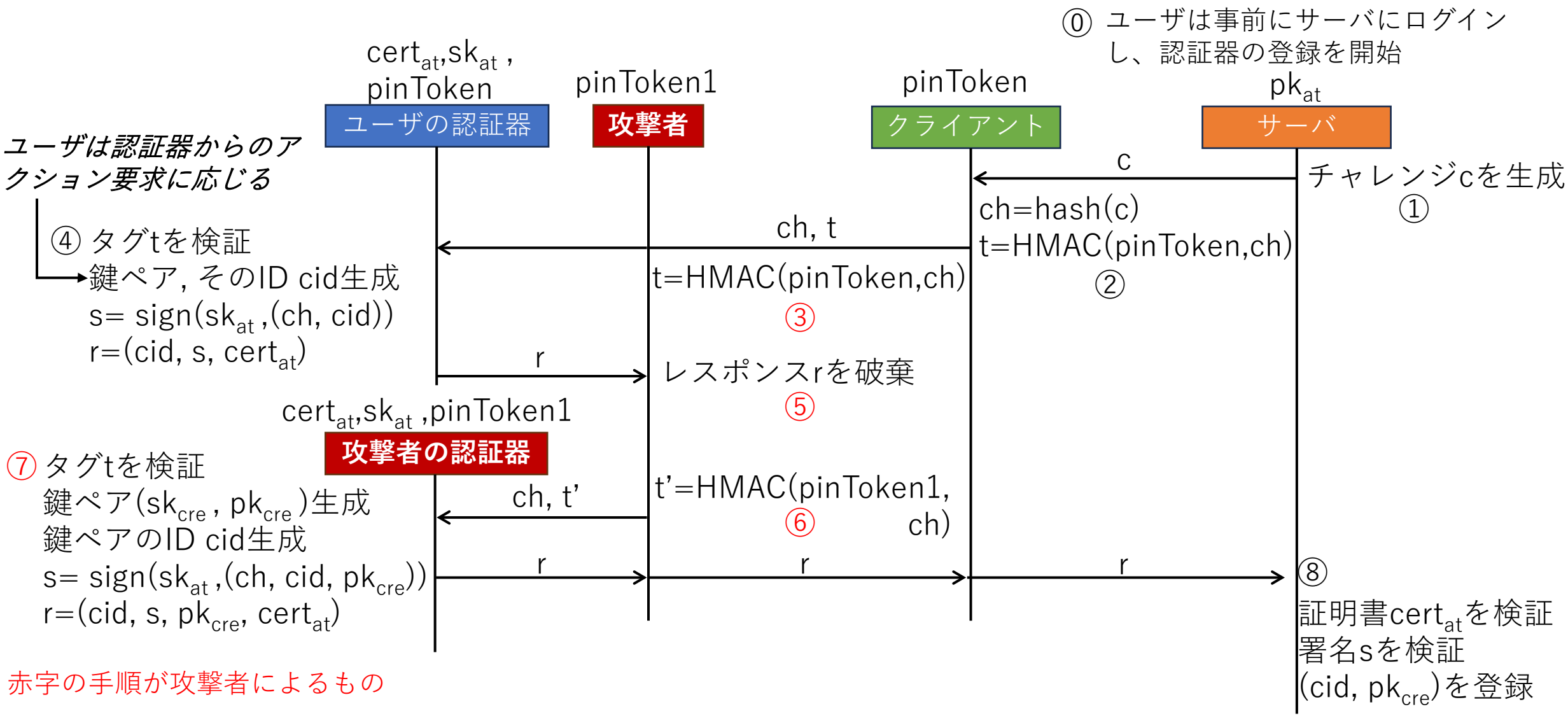
- 認証器のレスポンスに認証タグが付けられていないため,

  クライアントは異なる認証器が応答してきたことを検出できない

  - 認証器のレスポンスにpinTokenによる認証タグを付加すると防ぐことが可能



CTAP2の脆弱性が修正された場合での攻撃



赤字の手順が攻撃者によるもの

両攻撃への対策

使用するアテステーションモードにより異なる

■attCAモードを使用する場合

- サーバに登録可能な認証器を制限することで, 両攻撃を防げる

■basicモードを使用する場合は以下2点の修正を行う必要がある

- CTAP2脆弱性の修正
- 認証器のレスポンスにpinTokenによる認証タグを付加し, クライアントで検証する

まとめ

■背景

- FIDO2はパスワードレス認証の新たな標準として注目されており，WebAuthnとCTAP2の2つのサブプロトコルにより構成
- 先行研究で，CTAP2に存在する中間者攻撃の脆弱性が指摘されたが修正は確認されていない

■貢献

- 本研究では，この中間者攻撃がFIDO2に与える影響を調査するために，形式検証ツールProVerifによる安全性検証を行った
- 攻撃者が正規ユーザに認証器を登録したと誤認させつつ，攻撃者の認証器を登録する攻撃を発見した

補足 –リプレイ攻撃についての検証

■**要件**:登録プロセス前後のサーバ・認証器–クライアント間で

AuthenticatorID (aid)の**Injectiveな認証性**が成り立つ

- 登録プロセスにおいてサーバ・認証器がaidで終了しているときに限り,

  クライアントもaidで登録開始していなければならない

認証器での登録プロセス終了時に発生

```
query aid: AuthenticatorID;
inj-event(ServerFinishReg(aid)) && inj-event(AuthenticatorFinishReg(aid))
      ==> inj-event(ClientInitReg(aid)).
```

サーバでの登録プロセス終了時に発生

クライアントでの登録プロセス開始時に発生

- この場合で検証した結果も同様な攻撃が発見された