本資料について

本資料は下記文献を基にして作成されたものです。文書の内容の正確さは保障できないため、正確な知識を求める方は原文を参照してください。

著者：J. Arkko, V. Torvinen , G. Camarillo, Ericsson , A. Niemi , T. Haukka , Nokia

文献名：Security Mechanism Agreement for the Session Initiation Protocol (SIP)

種類：RFC3329

発表日：January 2003

セッション開始プロトコル(SIP)におけるセキュリティメ合意カニズム
- RFC3329 -

渡邊研究室
００Ｊ０８２
竹内　元規

RFC3329

- RFC3329では、セッション開始プロトコル(SIP)におけるユーザエージェント（UA）とその次のホップとの間で使用されるセキュリティ・メカニズムの協定のための新しい機能性を定義
- この新しい機能性は、セキュリティ・メカニズムに関する既存の方法を補足

SIP

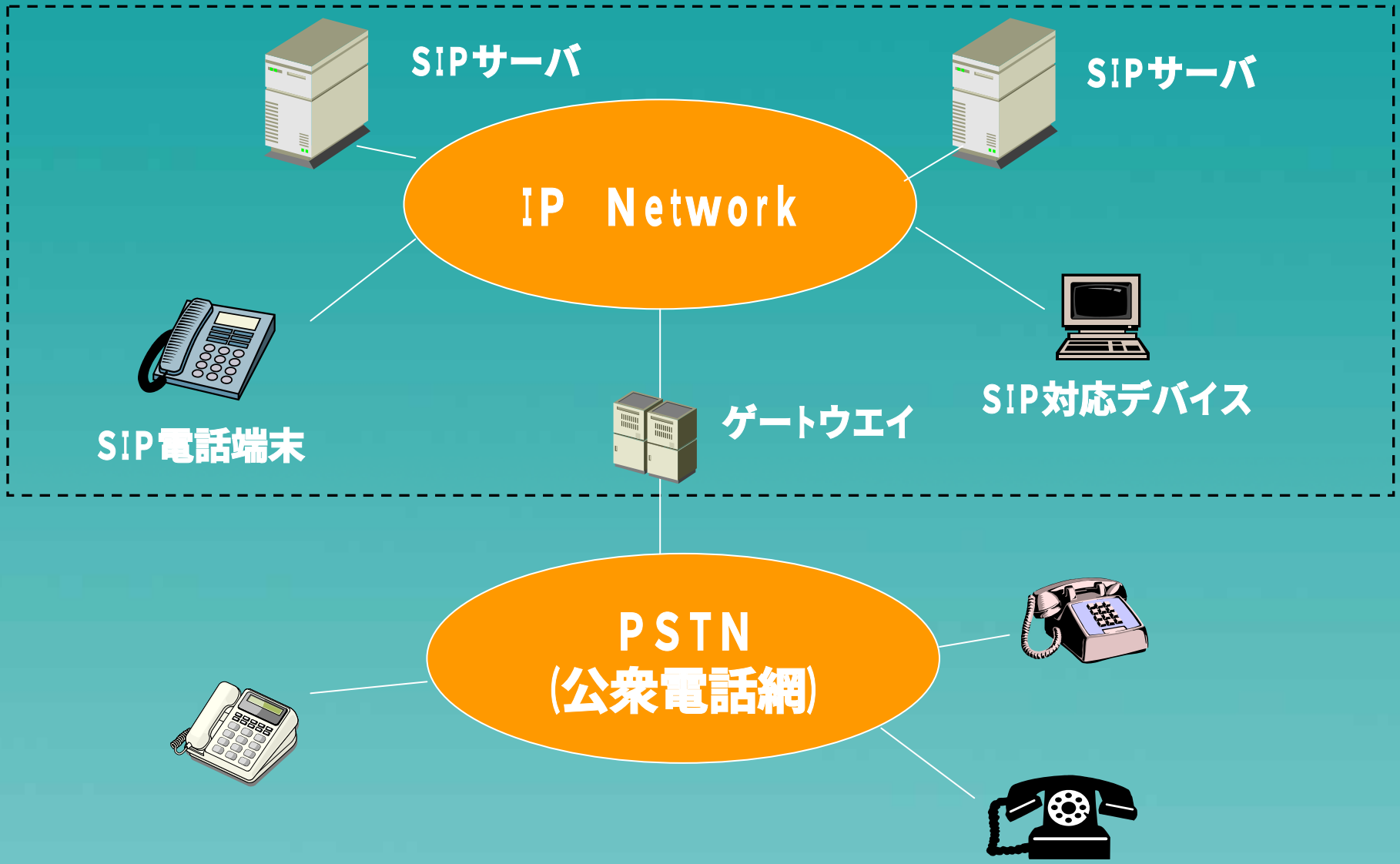
- SIPは電話、映像、チャット、ゲームなどのセッションを確立、変更、終了するためのプロトコル

- 双方向通信の特徴や実現できるサービスの種類を決定するシグナリングの役割をする
  - リモートユーザを探し出して、対話型の通信セッションを確立、変更、切断
  - 確立したセッションでどのようなIP通信を行うのかを規定する

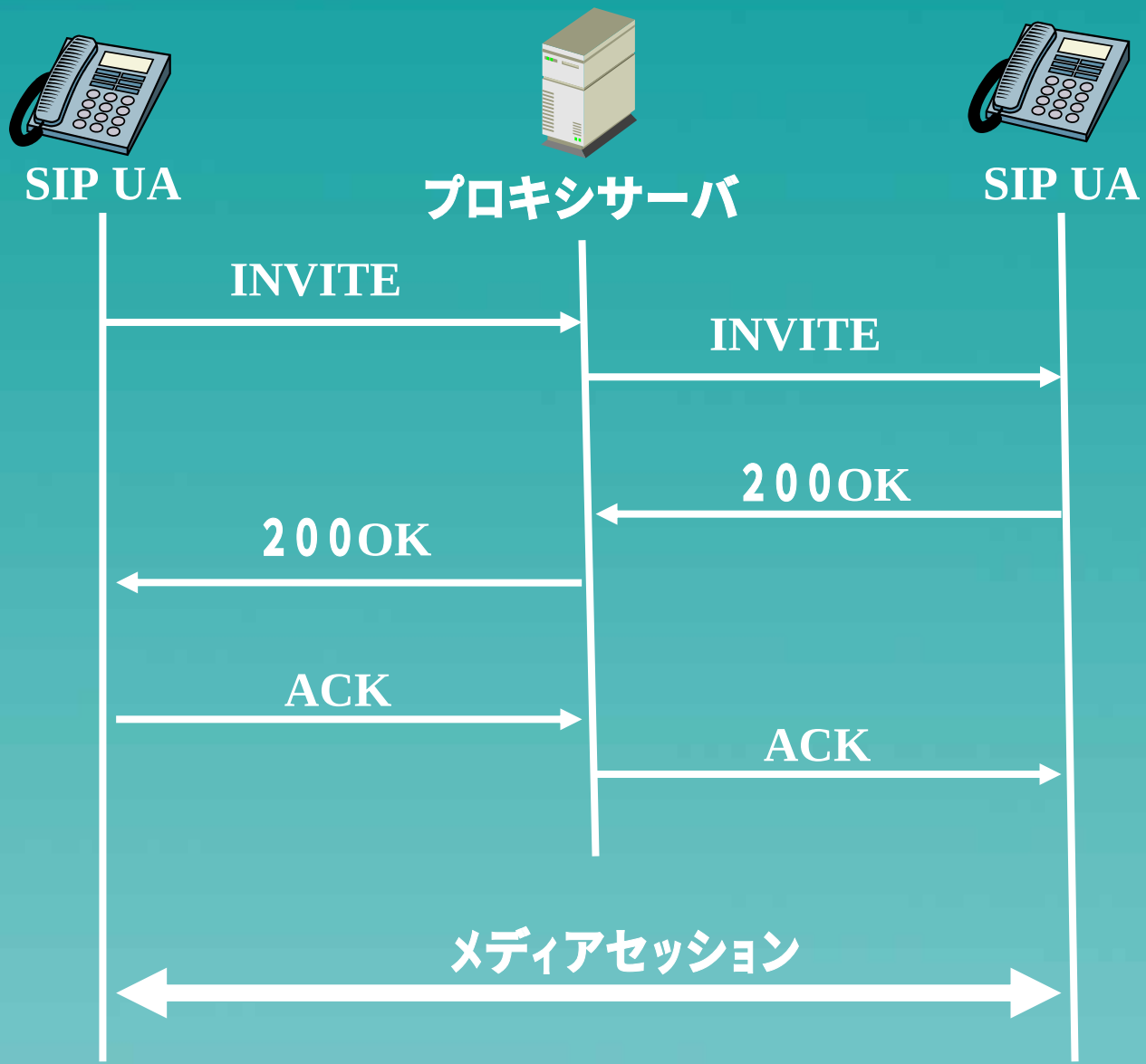
SIPの構成要素

要素	機能
ユーザエージェント (UA)	セッション確立のためのSIPリクエストの送信、メディアの送信を行う
SIPサーバ	セッションに関するリクエストを目的のUAや別のSIPサーバへ転送する
ロケーションサーバ	SIPネットワークでのデータベース

SIPネットワーク


SIPサーバ
SIPサーバ
IP Network
SIP電話端末
ゲートウエイ
SIP対応デバイス
PSTN
(公衆電話網)

セッションの確立


SIP UA
プロキシサーバ
SIP UA
INVITE
INVITE
200OK
200OK
ACK
ACK
メディアセッション

セキュリティの必要性

シグナリングの情報には、ユーザのプライバシーに関連する秘密性の高いデータが含まれている

プライバシー情報
通話履歴 ・ 通話傾向
発信元情報（電話番号・IPアドレス・ポート番号など）

これらの漏洩を防ぐためにセキュリティが必要

暗号化

- SIPでは
  - IPsec(IP Security Protocol)
  - TLS(Transport Layer Security)

  を利用してSIPメッセージを暗号化することができる
- SIPの暗号化には2つの形式がある
  - ホップバイホップ暗号化

    SIPメッセージ全体を暗号化
  - エンドツーエンド暗号化

    SIPメッセージの部分的暗号化

認証

SIPネットワークにおける認証とはUA（発信元）がSIPサーバや他のUAに対し自分の資格を証明すること

UA同士の認証
UAとサーバ間の認証
の2つの認証が必要

既存のSIP認証では
SIPダイジェスト認証 方式
が使われている

SIPダイジェスト認証

- SIPダイジェスト認証とは
  - HTTPで使用されている認証方式の一つをわずかに手直ししたもの
  - 共通シークレットを使ったチャレンジ／レスポンス方式の認証
  - リクエストを送信するUAと認証を要求するプロキシサーバ、UAとの間で行われる

共通シークレット
暗号化したユーザ名とパスワードが使われる

ダイジェスト認証方式の流れ

問題点

- ダイジェスト認証メカニズムは、メッセージの完全性や機密性がなく、メッセージ認証とリプレイ防御のみを提供する

特に

ダイジェスト認証はMitM攻撃に対して無防備となってしまう

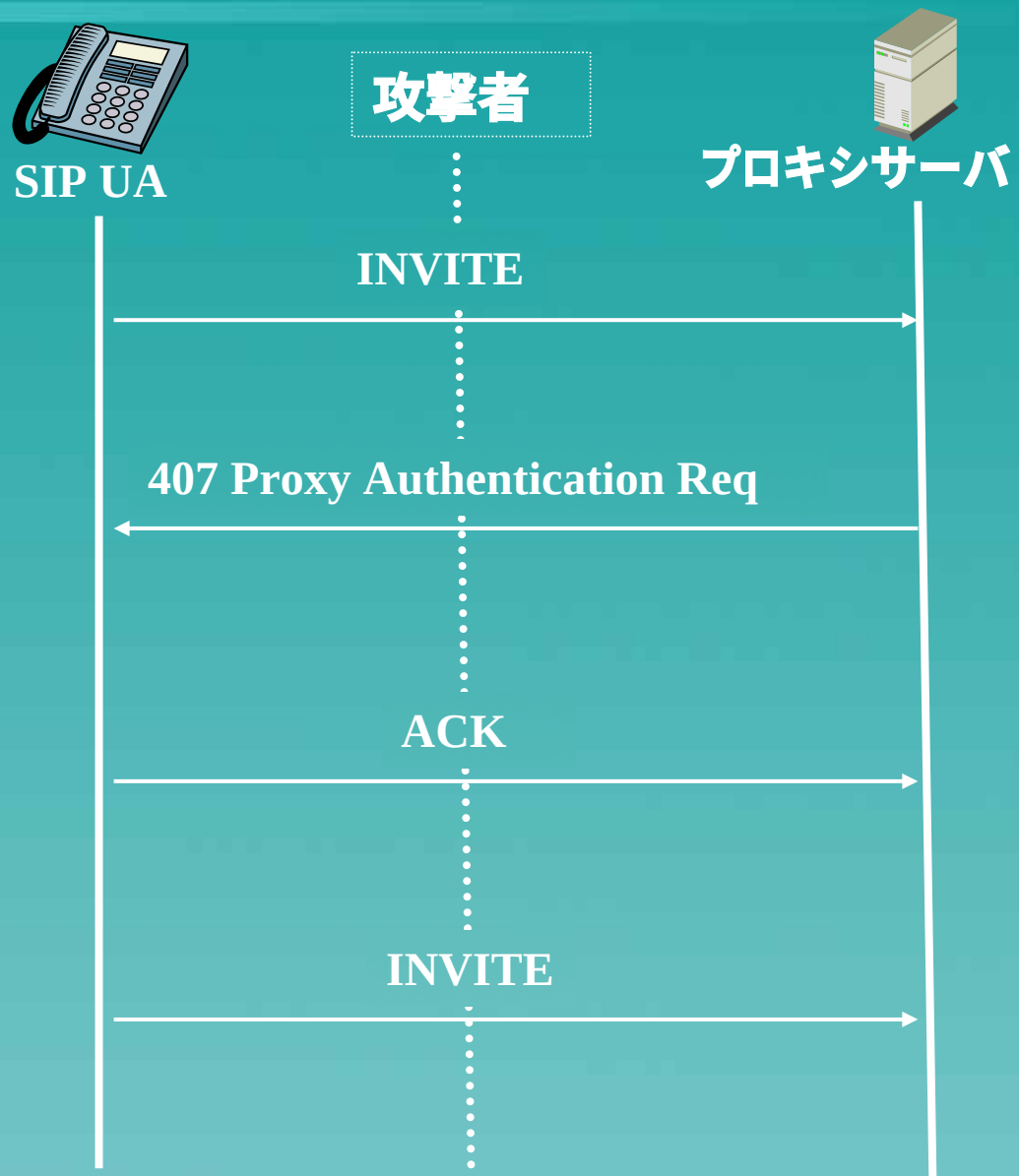
MitM攻撃：攻撃者がクライアントとサーバの間でメッセージの修正などを行うもの

MitM（Man-in-the-Middle）攻撃

攻撃者は

メッセージを盗聴して、メッセージに含まれるプライバシー情報を取得

メッセージを修正して、セッションの確立を妨げる



動機

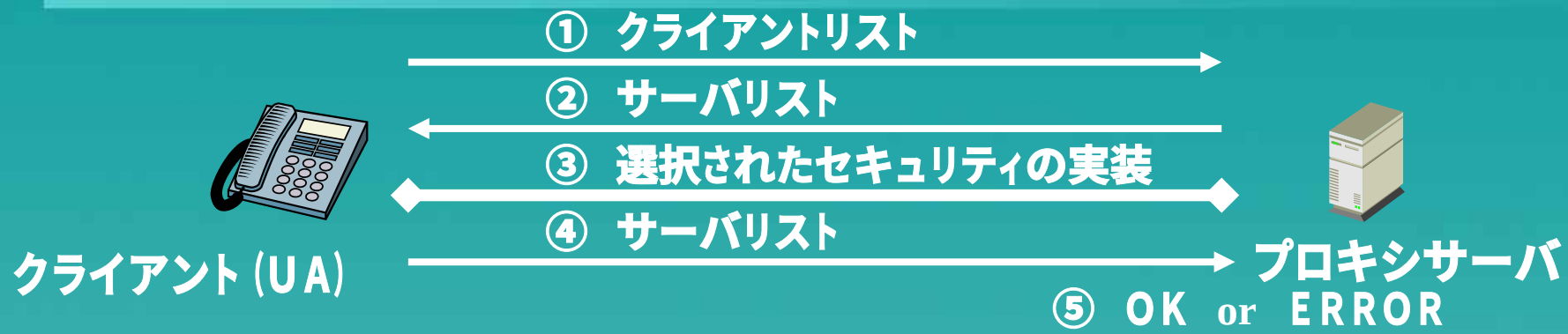
- 活動的な攻撃者がSIPリクエストおよび応答を修正することを防ぐために、ダイジェスト認証で提供される防御手段に加え、暗号化方式などの防御手段をとる必要がある
- 複数ある暗号化方式を選択することでさらにセキュリティを向上させる

設計目標

UAとその次のホップとの間での次のことができるようにする

1. どのセキュリティを適用するかを正確に選択できるようにする
2. セキュリティの選択を安全に行う
3. セキュリティ合意プロセスが成功したか、失敗したかを示す

解決



1. クライアントがサポートするセキュリティメカニズムのリストを送る
2. サーバにサポートされたセキュリティメカニズムおよびパラメーターを送る
3. 共通にサポートするセキュリティメカニズムを実装する
4. 今選択されたセキュリティを使用してサーバリストを返す
5. 返されたリストが修正されていなかったことを自分自身のリストと確認

構文

- 3つの新しいヘッダーフィールドを定義する

ヘッダーフィールド	使われる場所	説明
Security-Client セキュリティクライアント	リクエスト UA→サーバ	前述の① UAがサポートするセキュリティなどが含まれる
Security-Server セキュリティサーバ	レスポンス UA←サーバ	前述の②　暗号化して送信 サーバがサポートするセキュリティなどが含まれる
Security-Verify セキュリティ確認	リクエスト UA→サーバ	前述の④ サーバがサポートするセキュリティなどが含まれる

拡張BNF表記のヘッダーフィールド

security-client (セキュリティクライアント)

= “Security-Client” : sec-mechanism *(, sec-mechanism)

security-server (セキュリティサーバ)

= "Security-Server" : sec-mechanism *(, sec-mechanism)

security-verify (セキュリティ確認)

= "Security-Verify" : sec-mechanism *(, sec-mechanism)

sec-mechanism (セキュリティメカニズム)

= mechanism-name *(; mech-parameters)

mechanism-name (メカニズム名)

= ("digest" / "tls" / "ipsec-ike" / "ipsec-man" / token)

mech-parameters (パラメータ)

= (preference / digest-algorithm /digest-qop / digest-verify /extension)

セキュリティメカニズム

- tls
  - ＳＳＬを改良した、ＴＬＳ暗号化方式
- digest
  - ダイジェスト認証方式
- ipsec-ike
  - ike方式の鍵を利用したIPsec
- ipsec-man
  - マニュアル鍵を利用したIPsec

運用

クライアント（UA）

① Security-Client →

② 

プロキシサーバ

← ③ Security-Server (暗号化)

④

⑤ Security-verify →

⑥

① UAはリクエストにSecurity-Clientを付加して送信

② サーバは①を受信し、Security-Serverのパラメータ（優先度など）を設定

③ サーバはレスポンスに暗号化されたSecurity-Serverを付加して送信

④ UAは③を受信し、共通にサポートしているセキュリティの中で優先度がもっとも高いものを選択し、実装

⑤ UAは④で実装されているセキュリティを使用して、③で受信したSecurity-Serverをコピーしたを送信

⑥ サーバは⑤を受信し、自分自身のもつSecurity-Server と比較し、セキュリティ合意が成功したか、失敗したかを判断する

セキュリティ考察

①

暗号化などのセキュリティが適用されているので修正するのは不可

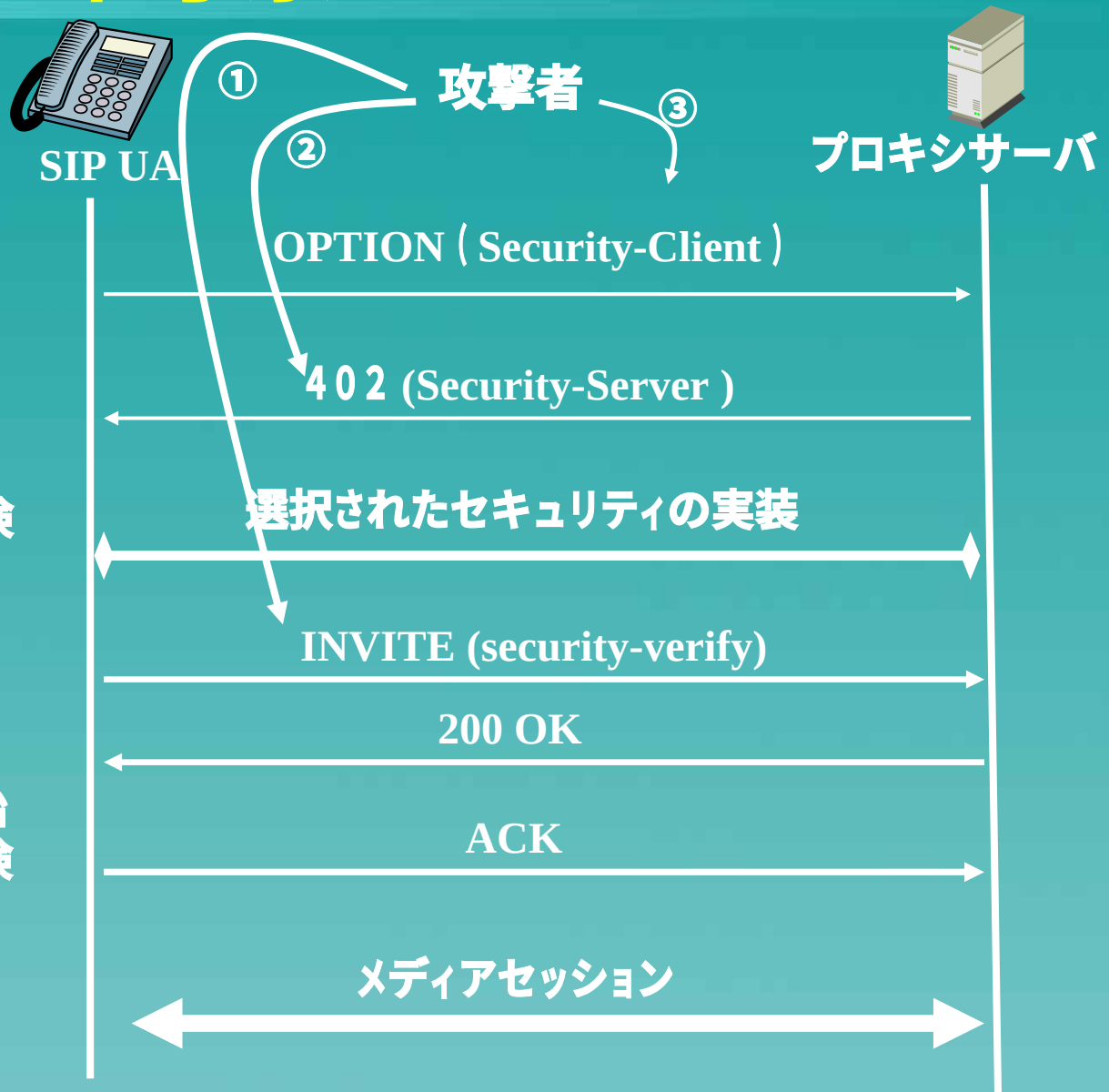
②

security-verifyによって攻撃を検知することが可能

③

セキュリティの実装に関する修正が行われた場合、通信を開始したときに潜在的な攻撃として検知される

セキュリティの実装に関する修正でない場合は影響なし



参考

- ソフトフロンティア
  - http://www.softfront.co.jp/tech/sip.html
- RFC3261
  - SIP: Session Initiation Protocol
- RFC2119
  - Key words for use in RFCs to Indicate Requirement Levels

終わり