クライアントを自由に選択可能な認証プロトコル TSSAP の提案

123430016　五島 秀典
渡邊研究室

1. はじめに

インターネットの普及に伴い，ユーザがクライアント端末を利用して遠隔地のサーバと安全に情報交換したいという要求が増えている．また，企業においては情報漏洩の防止，情報管理の徹底が重要となっている．情報漏洩の原因，3 割はノート PC 等のモバイル機器の盗難，紛失によるものと言われている．そこで社外に情報を持ち出さずに，必要に応じてクライアント PC から社内システムに安全にアクセスするリモートアクセスが注目されている．社内システムにアクセスするにはクライアントとサーバ間で正しい認証と暗号鍵の共有が必須である．セキュリティの強度を上げるために，２要素以上の認証が有効であると知られている．２要素以上の認証ではパスワードの流出，推測された場合でも認証を否定できる．このようなシステムにはユーザの視点から考えると自宅のパソコン等，異なるクライアントからでもサーバへアクセスできることが有用である．

本論文ではスマートフォンに認証情報を持つデバイスとして利用し，初期情報を一切持たないクライアントに対し、サーバから重要な情報を配送することを可能とするプロトコル TSSAP(Terminal Selectable and Secure Authentication Protocol) を提案する．

2. 既存の認証方式

既存の認証方式の例として非接触 IC カードを利用した方式がある．この方式では IC カード-クライアント間は無線通信で行われるため両者に共有鍵を埋め込む事前共有鍵を利用する方式が JICSAP により定義されている [1,2]．クライアントと IC カードの間は上記の共有鍵を使って認証と暗号通信を行う．ユーザは IC カードを所有しており，IC カード内の認証情報をクライアントから入力する認証情報で確認することにより認証する．しかし，この方式はクライアントに共有鍵を保持させているためクライアントから秘密情報が漏洩する可能性がある．また，漏洩した場合システム全体に影響を与える可能性がある．さらにセキュリティ面を考えると共有鍵を定期的に更新する必要があり，鍵の管理が煩雑になるという課題がある．

3. TSSAP の提案

3.1 想定するシステムモデル

TSSAP で想定するシステムモデルを図 1 に示す．本システムの構成要素はスマートフォン，クライアント，サーバである．スマートフォンとクライアントは Bluetooth で接続する．ユーザは秘密情報を格納したスマートフォンを所持し，クライアントを操作する．クライアントには秘密情報を一切保持させないものとする．クライアント-サーバ間は任意のネットワークで接続できる．ユーザとクライアントの通信距離が近いため，スマートフォン-クライアント間の中間者攻撃はできないものとする．

3.2 TSSAP の初期情報

初期情報としてスマートフォンにはユーザ ID，ユーザの登録したパスワードで暗号化したスマートフォン秘密鍵が格納されている．

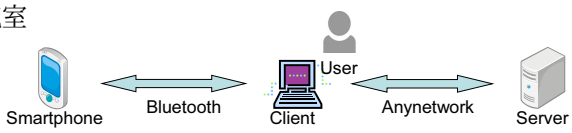


図 1: TSSAP のシステムモデル

次にサーバはサーバ秘密鍵，スマートフォン公開鍵，ユーザ ID が登録されている．クライアントには初期情報が一切ない．

3.3 TSSAP の動作

図 2 に TSSAP のシーケンスを示す．図中の記号はパケット名を示しており，パケット名後の (　) の内容はパケットの情報を示している．スマートフォン-クライアント間の Bluetooth のペアリングは完了しているものとする．即ち，この間の通信は Bluetooth の共通鍵で暗号化される．

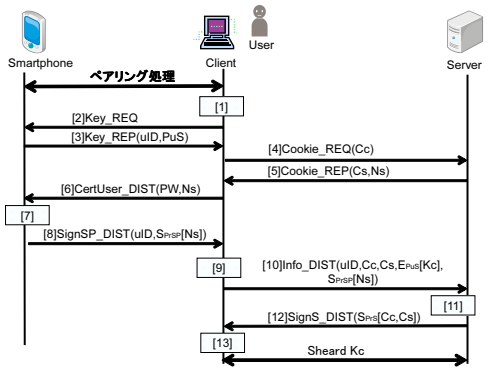


図 2: TSSAP シーケンス

1. スマートフォンへ Key_REQ を送信
   ユーザがクライアント画面上に示される開始ボタンをクリックすることによりクライアントはスマートフォンへサーバ公開鍵 PuS，ユーザ ID の情報配送を要求する．
2. Key_REP を送信
   スマートフォンはユーザ ID，サーバ公開鍵 PuS を送信する．
3. Cookie_REQ を送信
   クライアントは DoS 攻撃を防止するためのクッキー Cc を生成して，サーバへ送信する．
4. Cookie_REP の送信
   サーバはリプレイアタックに対応するための乱数 Rs と DoS 攻撃防止のクッキー Cs を生成する．この時接続してきたクライアントの IP アドレスと生成したクッキーとを対応づけて記憶させておく．また，クッキー Cs と共に乱数 Rs をクライアントへ送信する．
5. ユーザ情報 (PW) の入力
   クライアント PC の画面に PW 入力画面が表示される．ユーザはこの画面に PW を入力する．

6. CertUser_DIST の送信
   5. で入力された PW とサーバから受け取った乱数 Rs をスマートフォンへ送る．
7. スマートフォン秘密鍵の取得
   スマートフォンは受け取った PW で Epw[PrSP] を復号する．
8. SignSP_DIST の送信
   スマートフォンはクライアントから受け取った乱数 Rs，PW にスマートフォン秘密鍵 PrSP でディジタル署名をし，クライアントへ送信する．
9. 共有鍵 Kc の生成
   クライアント自身が共有鍵 Kc を生成する．Kc をサーバ公開鍵 PuS で暗号化する．
10. Info_DIST の送信
    8. で生成した Epus[Kc] と 7. で生成した Sprsp[Rs] と共に uID，Cc，Cs をサーバへ送信する．
11. ユーザ，スマートフォン認証と署名の作成
    受信した Rs とサーバが 4. で生成した Rs を比較する．Sprsp[Rs] のディジタル署名をスマートフォン公開鍵 PuSP を用いて確認する．ここでディジタル署名が正しいと確認されれば，7. で復号したスマートフォン秘密鍵 PrSP が正しく復号されたことになるため，ユーザの入力した PW が正しいといえる．これより，ユーザ認証が完了する．また，クッキー Cc，Cs についても比較を行うことでスマートフォン認証が完了する．次に Cc，Cs にサーバ秘密鍵 PrS を用いてディジタル署名を行う．最後に，暗号化された Kc をサーバ秘密鍵 PrS で復号する．
12. SignS_DIST の送信
    シーケンス中の 10. で生成された Sprs[Cc，Cs] をクライアントへ送信する．
13. サーバ認証
    Sprs[Cc，Cs] のディジタル署名をサーバ公開鍵 PuS で確認することによりサーバ認証を行う．

以上の認証処理を行うことにより認証が完了し，クライアント-サーバ間で共有鍵を安全に共有できる．

4. 実装と評価

図 2 に実装のネットワーク構成を示す．実装には，PC1 と VMware 上に作成した PC2 とスマートフォンの 3 台を用いた．PC1 と PC2 はブリッジ接続でネットワークに接続されており，スマートフォン，PC1 は Bluetooth で接続している．PC1 にはクライアントの処理プログラムを，PC2 にはサーバの処理プログラムを実行させた．

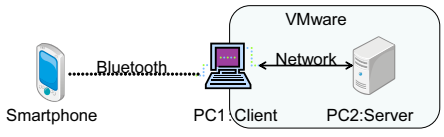


図 3: ネットワークの構成図

アプリケーションを起動し，ユーザがパスワードを入力してから認証が完了するまでの合計時間を測定する．測定方法は QueryPerformanceCounter 関数を利用し，クライアントがデータ送信し，レスポンスが返っくるまでを分割して測定する．分割したシーケンス図を図 4 に示す．また，表 1 に計測結果を示す．

①はクライアントが Key_REQ の生成から，スマートフォンから Key_REP を受信するまでの時間，②はクライアントが Cookie_REQ の生成から Cookie_REP を受信するまでの時間，③はクライアントが CertUser_DIST の生成から SignSP_DIST を受信するまでの時間，④はクライアントが Info_DIST の生成から SignS_DIST を受信するまでの時間示している．しかし，③に関してはスマートフォン内で行うディジタル署名をする部分が未実装であるため③では Bluetooth での送受信時間＋スマートフォンの処理を同内容で C 言語で実装したプログラムを PC1 で別途測定した結果を示す．すべての計測結果は 10 回計測した平均値を示している．

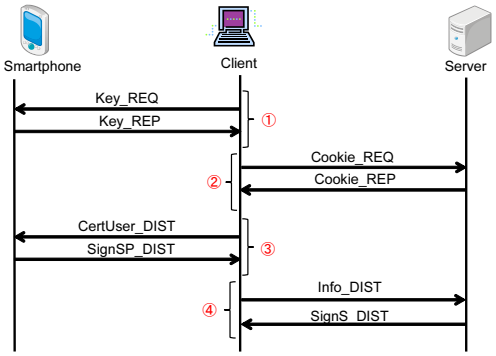


図 4: シーケンス図

表 1: TSSAP の実測値

番号	時間 (ms)
①	2.6
②	51.7
③	34.7
④	112.5
合計	200.9

本実験の測定結果はスマートフォンのやるべき処理を PC1 を利用し，C 言語で実装しているため，疑似的ではあるがかなり近似していると思われる．ユーザがパスワードを入力してから認証が完了するまでの時間が 200.9ms だったためシステムの立ち上げ時にかかる処理としては，十分許容範囲となると考える．

5. むすび

本論文ではクライアントサーバ間で重要情報を安全に交換する方式 TSSAP を提案した．クライアントが秘密情報を持たないため，クライアントからの情報漏洩の心配がなく，クライアントを自由に選べるという利点を持つ．また，スマートフォン内の秘密鍵も暗号化されているため，漏洩する心配がない．

参考文献

[1] 影井良貴，IC カードの動向，情報処理学会会誌，Vol.39，No.5，pp.429-433 .

[2] IC カードシステム利用促進協議会，JICSAP IC カード仕様書 V2.0.

クライアントを自由に選択可能な認証プロトコルTSSAPの提案

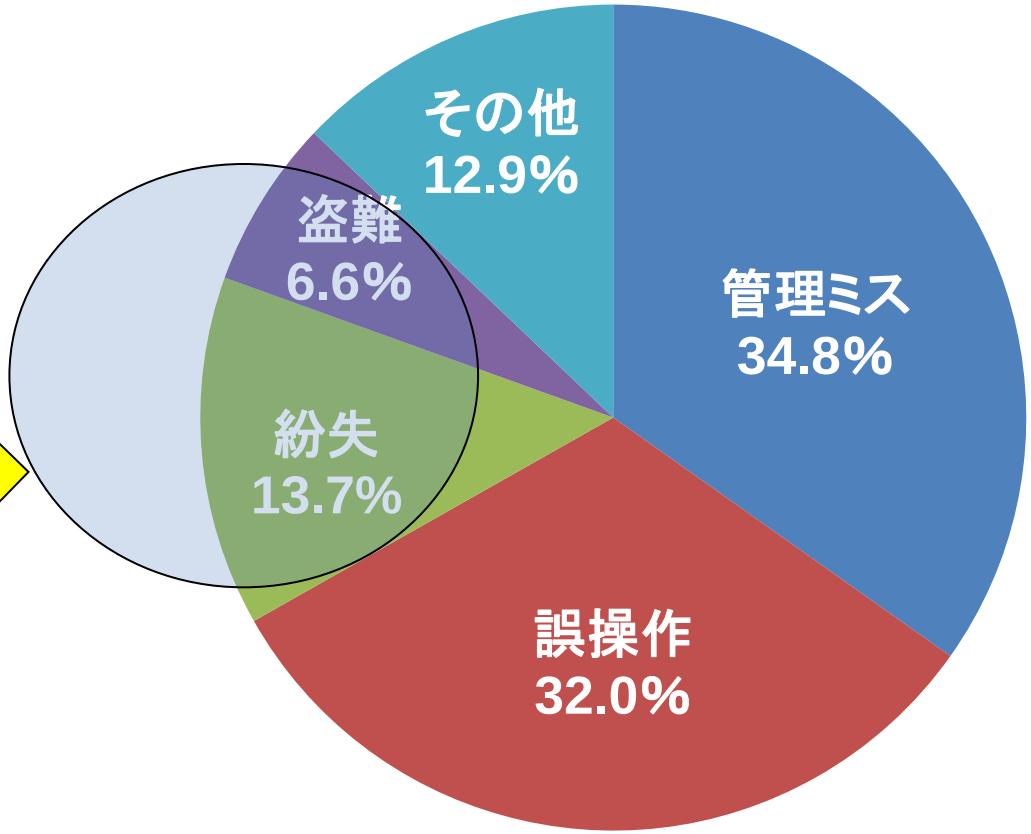
名城大学大学院
理工学研究科　情報工学専攻
渡邊研究室
123430016　五島 秀典

Watanabe Lab.

研究背景

- 企業では情報漏洩の防止が重要となっている

情報漏洩の原因

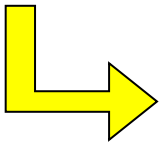


システムでカバーできる

出典：NPO 日本ネットワークセキュリティ協会

紛失/盗難

- 事例
  - 自宅に空き巣が入りPCごと盗難
  - USBメモリなどの記憶媒体の置き忘れ，紛失



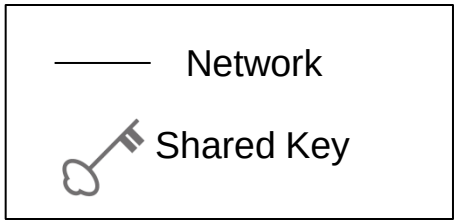
情報を社外へ持ち出すことが共通点

- 解決策
  - 社内サーバへ外部からアクセスすることで情報を持ち出さない
    - 社内サーバとクライアント間で暗号鍵を共有することによりリモート処理を可能とする

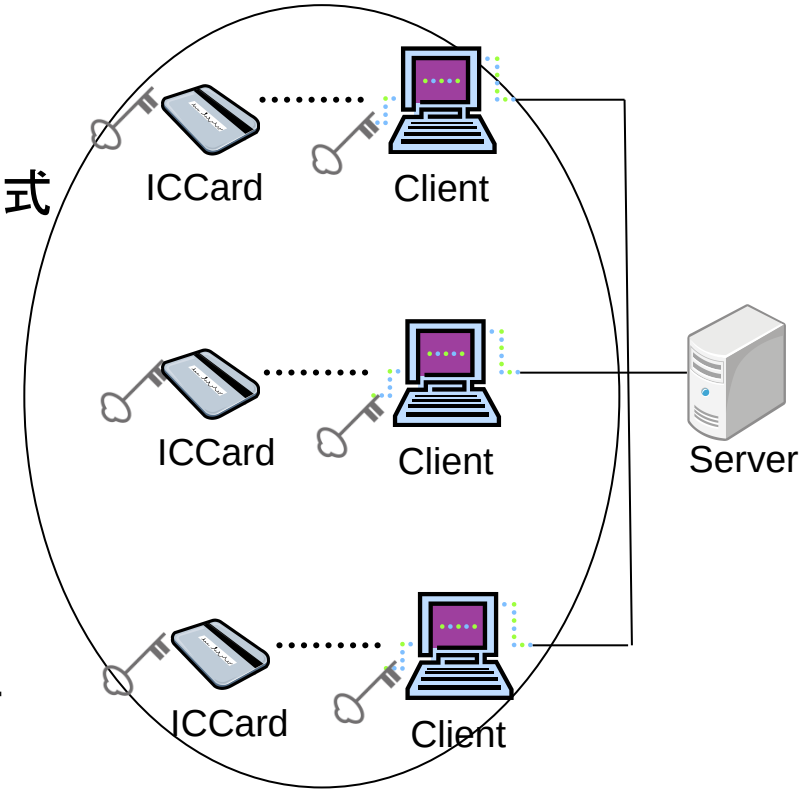
要求

- 異なるクライアントから社内サーバへアクセス
  - サーバ内の情報にどこからでもアクセス可能
- クライアント/サーバ間の安全確保
  - 認証
    - 2要素以上の認証(パスワード＋認証デバイスなど)
  - 暗号化
    - すべての通信路を暗号化
  - 各種攻撃に対応
    - DoS攻撃(サーバに負荷をかけ，サービスを不能にする)
    - 中間者攻撃(2者間の通信を中継し，盗聴，改ざんを行う)
    - リプレイアタック(以前成功したパスワードを盗聴し，再利用して認証する)

ICカードを利用した認証方式と課題



- 特徴
  - 認証デバイスとしてICカードを利用
    - パスワード＋ICカードの2要素で認証
  - ICカード/クライアント間は事前鍵共有方式
  - 共有鍵を用いて暗号化
  - クライアントとサーバ間は公開鍵で認証
- 課題
  - クライアントから共有鍵が漏洩
    - 漏洩時の影響が全体に及ぶ
  - 共有鍵を定期的に変更する必要がある
    - 同じ鍵を使い続けることでセキュリティ低下
    - 管理が煩雑になる

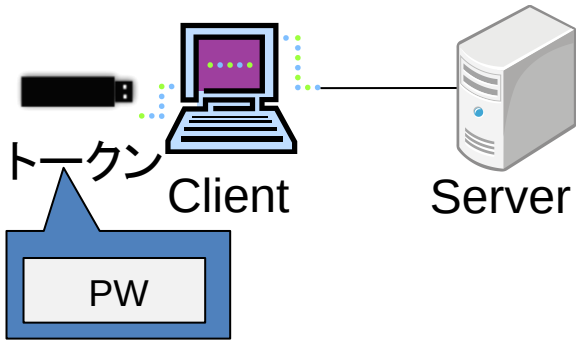


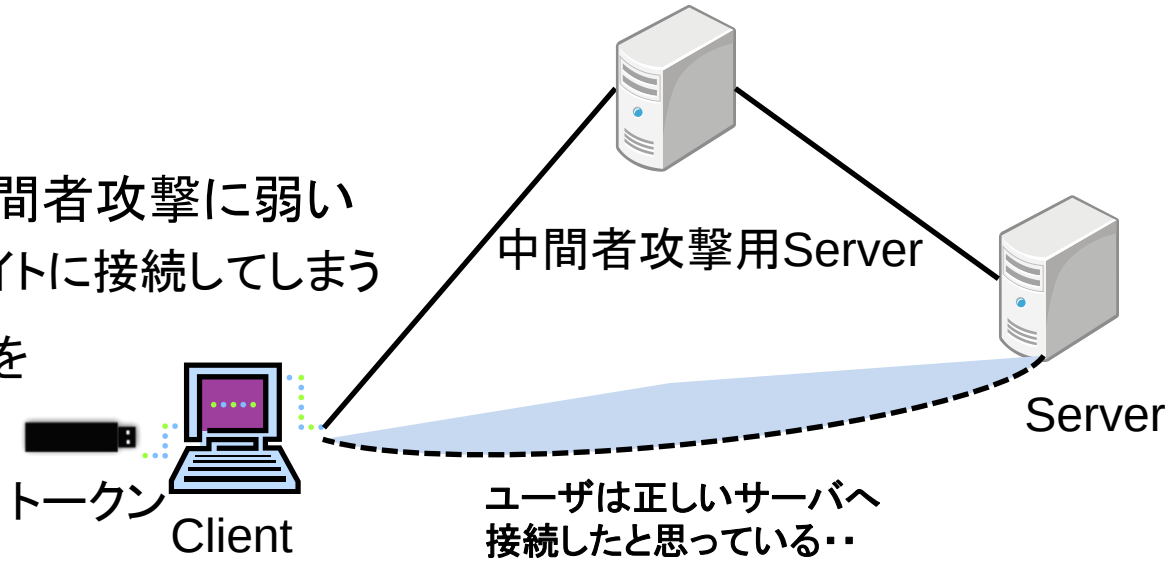
出典：日本ICカードシステム利用促進協議会(JICSAP)

ワンタイムパスワードを利用した認証方式と課題

- 特徴
  - 認証デバイスとしてトークンを認証に利用
    - パスワード＋トークンの2要素で認証
  - トークンでワンタイムパスワードを生成
    - トークンに液晶があり，そこに表示される
    - サーバとトークン間で関数や時間を同期している
  - クライアントとサーバ間はSSLを利用
  - クライアントに初期情報を必要としない
- 課題
  - トークンが必要となる
  - フィッシングを利用した中間者攻撃に弱い
    - 正規サイトに似せた偽サイトに接続してしまう

ユーザがサーバの正当性を
確認できないことが原因





TSSAPの提案

TSSAP (Terminal Selectable and Secure Authentication Protocol)

- 特徴
  - スマートフォンを認証デバイスとして利用
    - スマートフォンを利用することにより利便性の向上
  - クライアントに初期情報を持たせる必要がない
  - 接続先のサーバの正当性を確認できる
  - 特別なハードウェアを必要としない
  - クライアントを自由に選択できる
    - 自宅PC,会社のPCなど
- 実現に向けての課題
  - スマートフォンからの秘密情報漏洩防止
    - スマートフォンには耐タンパ性がないため

TSSAP初期情報

- ユーザID
- $E_{h[PW]}$[SP秘密鍵]
- サーバ公開鍵

なし

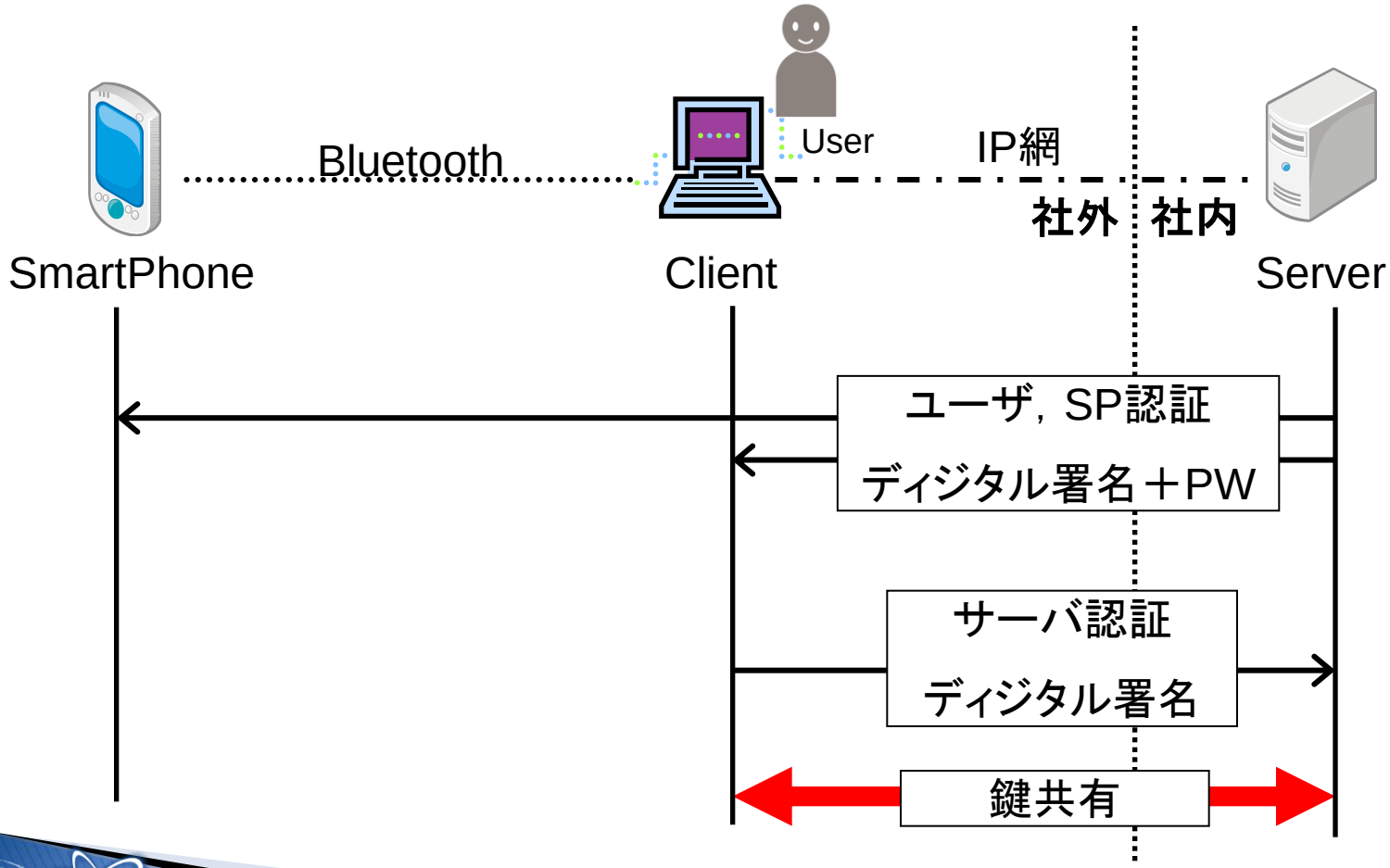
- サーバ秘密鍵
- ユーザID
- SP公開鍵

- 初期情報は事前にオフラインで設定する
- クライアントには初期情報が必要ない
- SP秘密鍵をパスワードのハッシュで暗号化する

※h[A]　Aのハッシュ値
※E_A[B]　BをAで暗号化

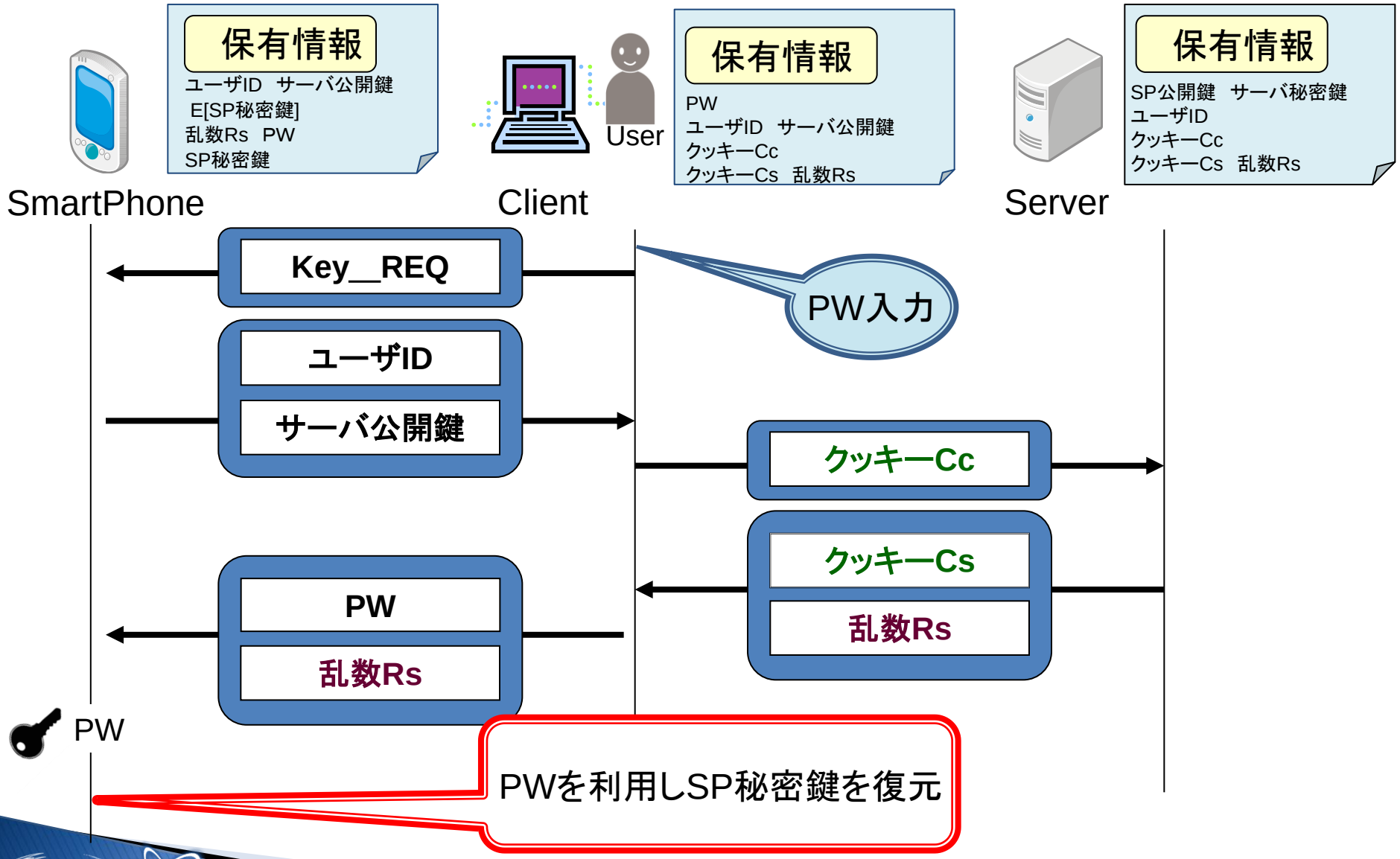
TSSAPの構成と認証

- ユーザ認証はサーバ型認証を利用する
  - ユーザ認証，スマートフォン認証をサーバ側で一括して行う

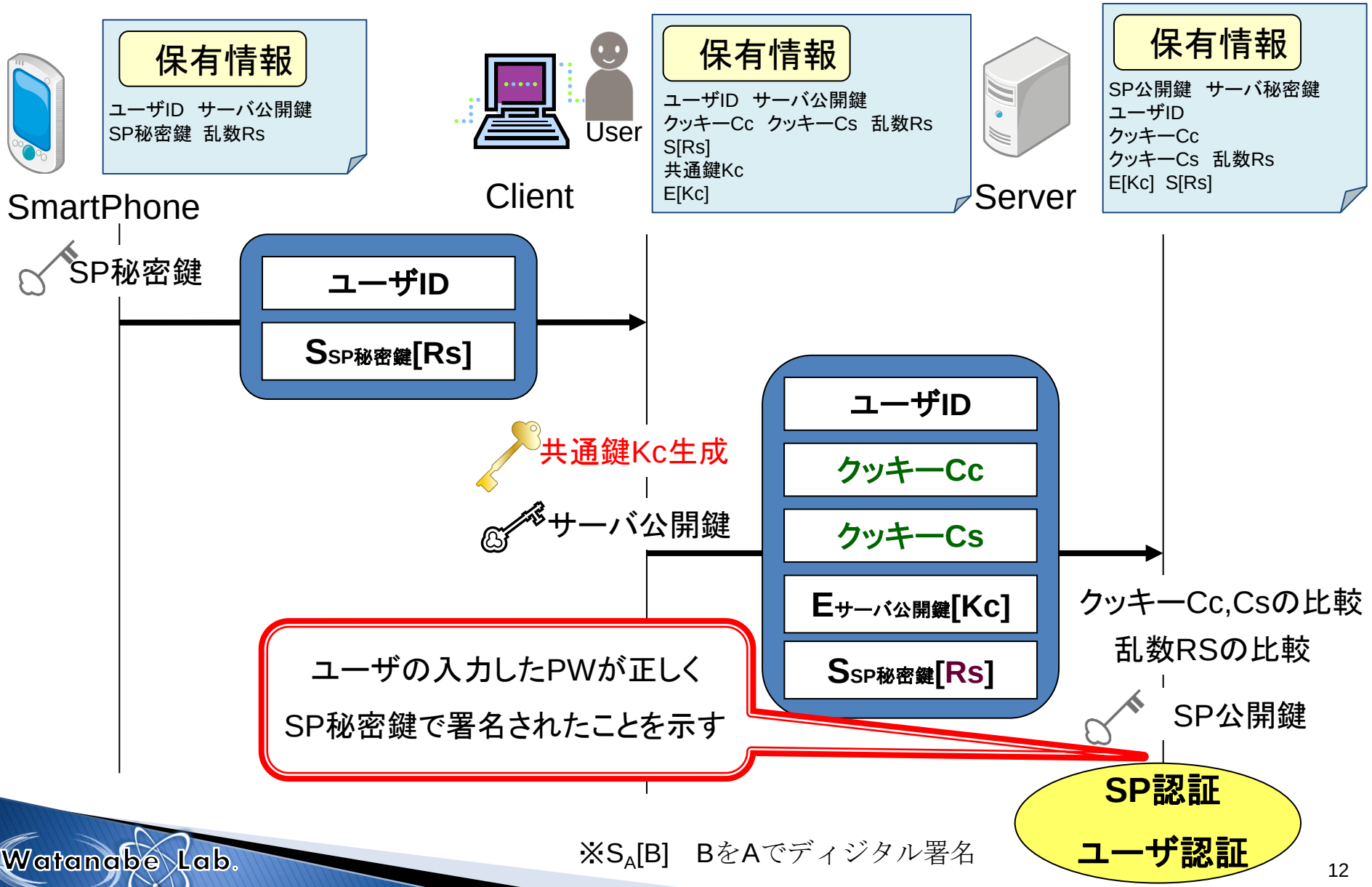


動作

TSSAP動作(ユーザ,SP認証1/2)

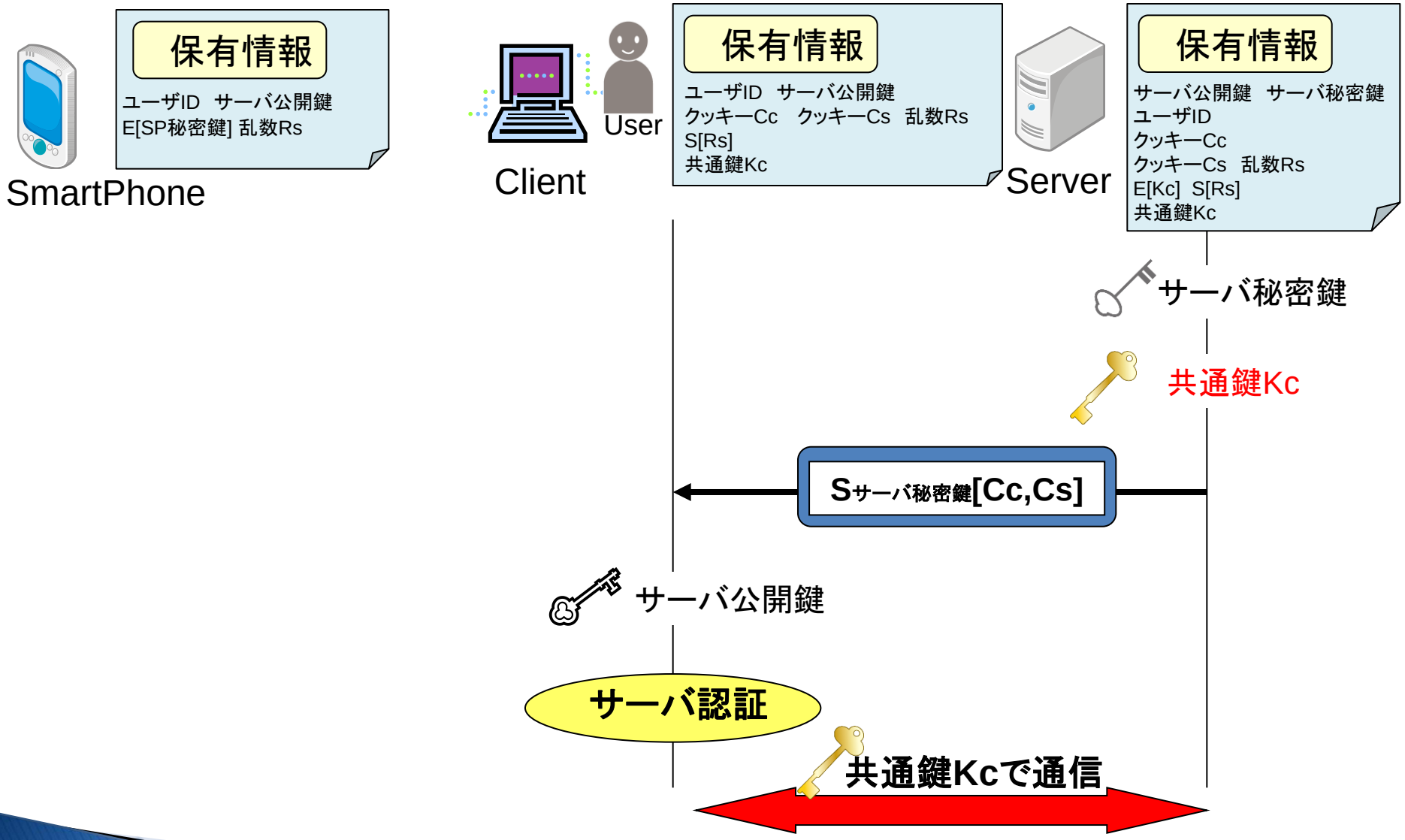


TSSAP動作(ユーザ,SP認証2/2)



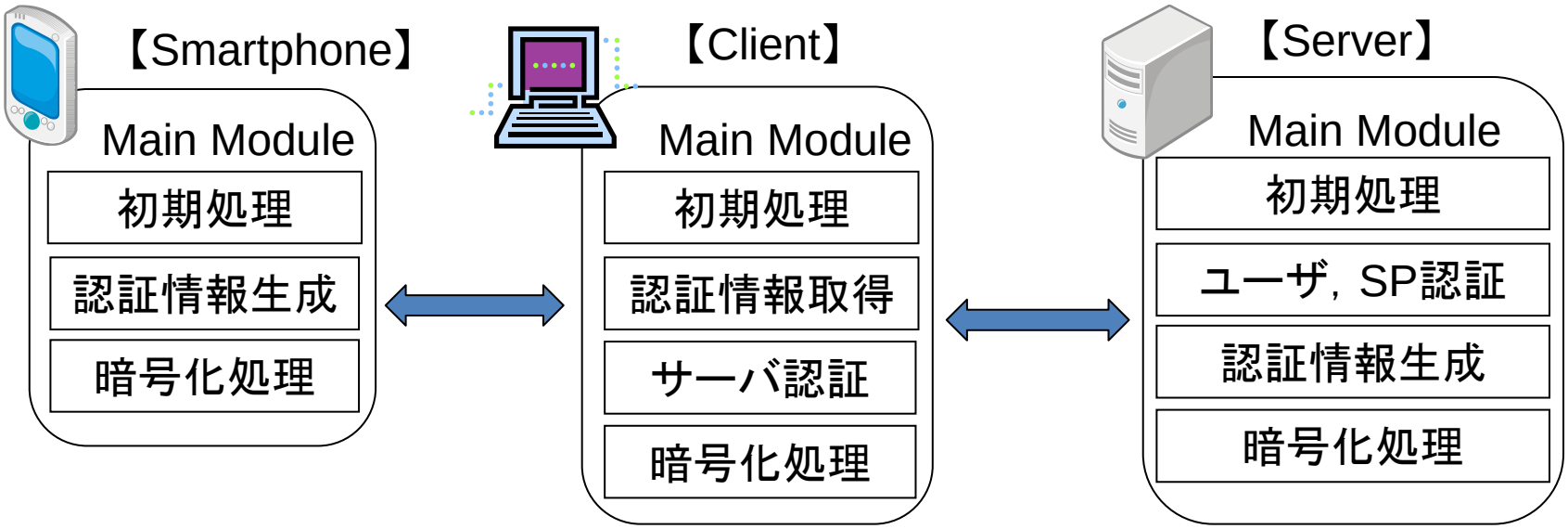
※$S_A[B]$　BをAでディジタル署名

TSSAP動作(サーバ認証)



実装

- モジュール構成

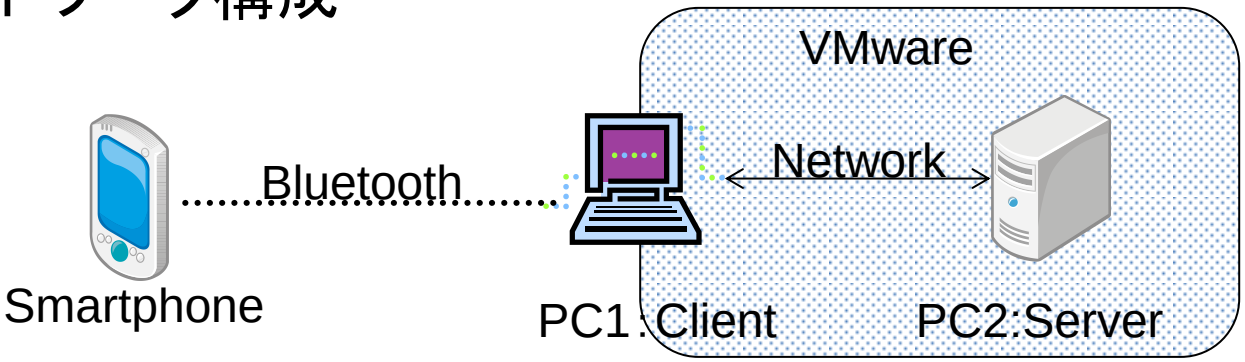


PC上の暗号化処理にはOPENSSLのソースを利用
暗号鍵はRSAの鍵を1024bit，AESの鍵を256bitを使用

※OPENSSL:暗号化関数と様々な
ユーティリティ関数を実装しているオープンソース

実験環境

- ネットワーク構成

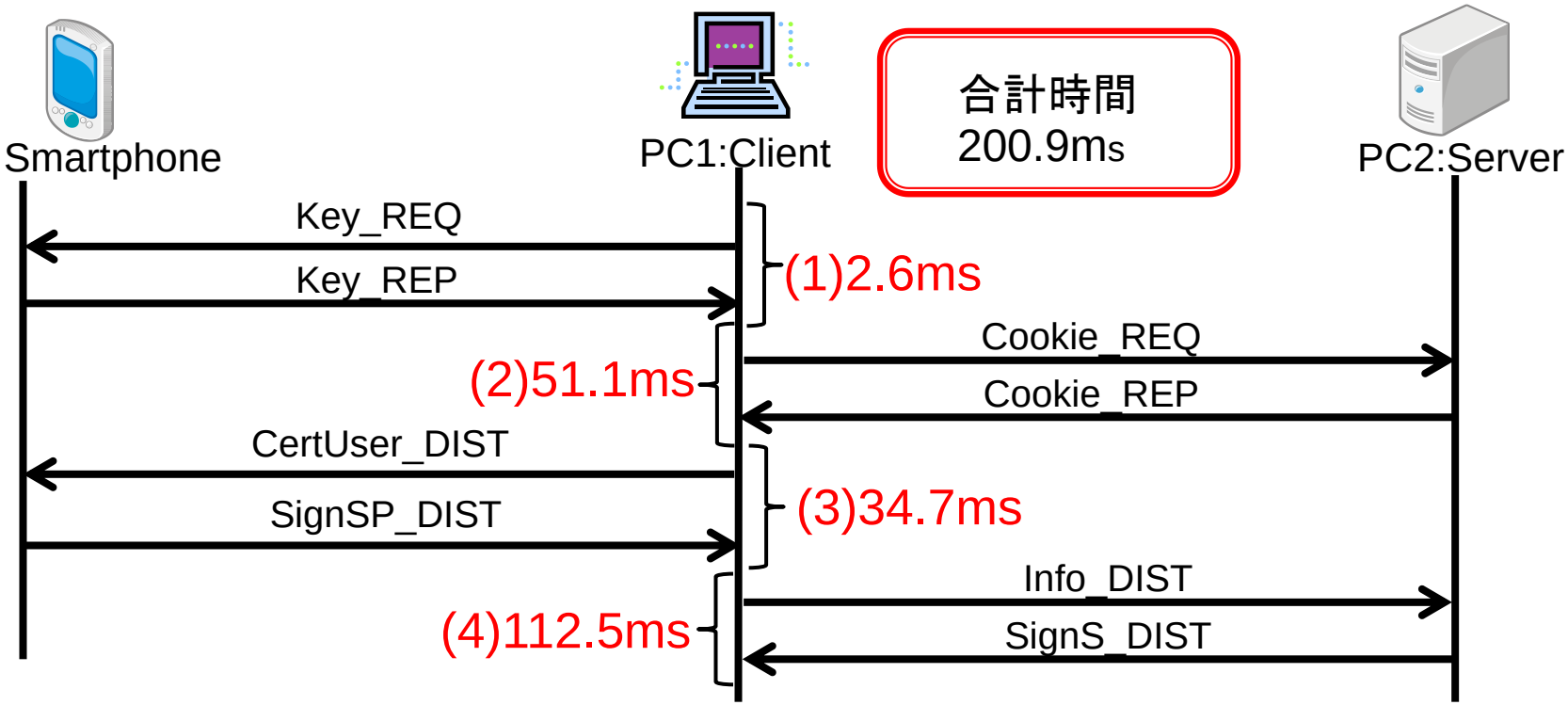


- 装置仕様

	スマートフォン	PC1(クライアント)	PC2(サーバ)
OS	Android 4.0	Windows7 64bit	Windows7 64bit
CPU	Snapdragon S4 1.5GHz	Core2 Quad 2.8GHz	Core2 2.8GHz
Memory	1GB	8GB	2GB
言語	Java	C++	C++

性能評価

- 測定方法はQueryPerformanceCounter関数で測定
- (3)の処理時間はスマートフォンの処理内容をPC1：クライアントで実行した場合の処理時間＋Bluetooth送受信時間で示す
- 合計時間200.9ms→システム立ち上げ時の処理時間として十分許容の範囲



まとめ

- 本発表では
  - 重要情報を配送するためのプロトコルTSSAPを提案
  - 実装および、評価を行った
- 今後
  - 現在実装途中のスマートフォンの実装
  - サーバがマルチ対応した場合の負荷

終