スマートフォンアプリケーションにおいて エンド間通信を実現可能なプラットフォーム開発

内藤 克浩[1] 鈴木 秀和[2] 渡邊 晃[2] 森 香津夫[3] 小林 英雄[3]

概要：スマートフォンはユーザ自身が容易に様々なアプリケーションをインストール可能なことから, 近年急激に普及しているモバイルデバイスである. 現在のインターネットでは, NAT ルータなどがネットワーク内に設置されることも多く, 直接通信を行うことは困難な場合も多い. そのため, 多くのスマートフォンアプリケーションはクライアント・サーバーモデルに基づいて設計されている. 一方で, 近年のスマートフォンアプリケーションでは, 音声, ビデオ通話, 対戦ゲームなど, リアルタイム性を要求するアプリケーションも開発されつつある. このようなアプリケーションでは, サーバーを経由しない, 端末間で直接通信が実現可能な方式が遅延低減の観点で適している. また, ネットワーク資源の最適利用の観点からも, 端末間の最短経路を用いて通信を実現できるため直接通信は適している. 端末間で直接通信を行うことが可能な既存技術として, 移動透過技術が既に提案されているが, これらの技術は OS 内に実装することを想定して設計されており, 特定のアプリケーションにおいて, 移動透過技術を採用することは容易ではない. 著者らは, 新たな移動透過技術として Network Traversal with Mobility (NTMobile) の開発を進めてきた. NTMobile では, IPv4/IPv6 ネットワークにおける移動透過性と通信接続性を担保可能である上, 実装コードは OS の機能とは分離して設計されている. そのため, NTMobile は各アプリケーションが独自に移動透過技術を採用することが比較的容易である. 本稿では, スマートフォンアプリケーション内において NTMobile を利用するための, サーバーシステムを含めたプラットフォームの提案を行う. 提案プラットフォームでは, スマートフォンアプリケーションのアカウント管理, リアルタイム通信, 認証および鍵交換, オフライン時の非リアルタム通信によるデータ交換手段を提供可能である. 実証実験では, 提案プラットフォームが iOS および Android OS 上で動作することを確認する.

1. はじめに

近年急激に普及しているスマートフォンでは, ユーザーが任意のアプリケーションを自由にインストールできるため, 様々なモバイルアプリケーションが開発されている. これは, スマートフォン OS を開発している Apple, Google などが開発環境を提供するとともに, アプリケーションストアを通して, 容易にアプリケーションを配布可能な枠組みを提供していることが大きい. なお, 現在の多くのネットワークでは, NAT ルータがネットワーク内に設置されていることが多いため, 公開されている多くのアプリケーションは, 既存のクライアント・サーバーモデルに基づいた実装を採用している. そのため, アプリケーションに関するトラヒックはすべてサーバーを経由することとなり, ネットワークを利用するアプリケーション開発者は, 大規模なサーバー資源を準備する必要に迫られている. 結果として, アプリケーションを公開するのが容易になった一方で, ネットワークを利用する大規模なアプリケーションは, 必要なサーバー資源などの都合から, 未だに公開は困難な状況である.

スマートフォンの大きな特徴として, 3G, LTE, WiFi など複数の無線インターフェースを実装していることが多く, OS が自動的に適切な無線インターフェースを選択する [1], [2], [3]. 一般的には, 3G, LTE, WiFi などのバックボーンネットワークは独立に設計されているものであり, 事業者による枠組みなどがない場合には, 各インターフェースを切り替えるたびに IP アドレスは変化する. IP アドレスが変化する状況において, ネットワークを利用したモバイルアプリケーションを開発した場合, 同一スマートフォンからのサーバーへのアクセスが異なる IP アドレスによ

[1] 愛知工業大学 情報科学部
Faculty of Information Science, Aichi Institute of Technology
[2] 名城大学大学院 理工学研究科
Graduate School of Science and Technology, Meijo University
[3] 三重大学 工学研究科 電気電子工学専攻
Department of Electrical and Electronic Engineering, Mie University

り行われるため, 適切なセッション管理に関する実装の煩雑性が増す. 結果として, 多くのセッション管理を必要とするアプリケーションでは, 異なる IP アドレスからのアクセスを検出した場合, セッション情報をリセットすることも多く, ユーザの利便性を損なうこともある.

このような問題の解決手段として, サーバーをできるだけ経由しない枠組みと IP アドレスの変化を隠蔽する枠組みが必要と考えられる. 著者らは, 移動透過技術を用いることが, 上記課題を解決する手段として有効と考えている. 一般に移動透過技術は, IP アドレスが持つノード識別部と位置識別部を何らかの方法により分離し, ユーザは同一のノード識別子を用いて, IP アドレス変化時にも継続的な通信を実現可能である [4], [5], [6], [7]. 移動透過技術に関する実装は既に公開されているものもあるが, その多くはモビリティを想定した IPv6 を前提としたものが主流である. 一方, 現在のインターネットで利用されている IPv4 に関する実装はきわめて少ない状況であり [8], [9], [10], FreeBSD や Linux などの UNIX 系 OS 用の実装は公開されている一方で, 現在のスマートフォンなどで利用可能な実装はほぼ公開されていない状況である. なお, IPv6 は IPv4 アドレスが枯渇していることもあり, 今後も順調に利用されていくことが想定されるが, ネットワーク事業者はラージスケール NAT (LSN) を導入することも増えており, IPv4 も今後の主要な通信方式として残ると予想される [11], [12], [13].

著者らは IPv4 及び IPv6 ネットワークで利用可能な移動透過技術として, Network Traversal with Mobility (NTMobile) を提案し, 実装を進めてきた [14], [15], [16], [17]. NTMobile では, 各クライアントがノード識別子として仮想 IP アドレスを利用することにより, 物理 IP アドレス変化時にも通信を継続可能である. また, 各クライアントの情報は Direction Coordinator (DC) が管理することにより, 各クライアントが利用しているアクセスネットワークの情報管理と仮想 IP アドレスの割当を実現する. アプリケーションは仮想 IP アドレスを用いて通信を行うが, アプリケーションからの IP パケットは物理 IP アドレスを用いた User Datagram Protocol (UDP) トンネルを通して転送されるため, UDP トンネルの経路を切り替えることにより, アクセスネットワークに合わせた適切な経路をクライアント間で構築可能である. NTMobile は Linux カーネルへのインストールを想定したカーネル版実装 [14],iOS, Android などのスマートフォンアプリケーションを想定したアプリケーションモジュール版実装 [15], [16] の試作を終えている. 本稿では, アプリケーションモジュール版実装を利用するための, プラットフォームの提案を行う. 提案プラットフォームでは, 各アプリケーションモジュールの認証機能をつかさどる Account Server (AS) を導入し, 認証結果に基づいた暗号化用の鍵管理を実現する. また, 端末オフライン時などにデータ交換を仲介する Cache Server (CS) を導入することにより, 常時通信可能なことを保証できないスマートフォンにおいても，実用的なアプリケーション開発が可能な枠組みを提供する. 実装実験では, 本プラットフォームを利用した iOS 及び Android アプリケーションを作成することにより, 提案プラットフォームの実現性を確認する. 開発アプリケーションは, iOS 及び Android OS 上のアプリケーションとして動作し, 開発プラットフォームと連携することにより, 任意のクライアント間でセキュアなエンド間通信が実現可能であることを示す.

2. NTMobile の概要

既存の NTMobile のシステムモデルは, Direction Coordinator (DC), Relay Server (RS), NTMobile クライアントにより構成されていた. DC は自身の NTMobile クライアントのネットワーク情報の管理を行うとともに, 他クライアントとの通信開始時には, 各クライアントのネットワーク状況に応じて適切なトンネル構築指示を行う. また, DC は自身の RS を管理しており, トンネル構築を行いたいクライアントが共に NAT 配下のプライベートネットワークに接続されている場合, クライアントの IP プロトコルバージョンが異なる場合, RS に指示を出すことによりクライアント間のトンネル中継を実現する.

NTMobile では, DC と RS をセットとして任意のネットワーク上に配置することを想定しており, 各 DC は DNS の枠組みを用いて相互探索を行う. 具体的には, 各 DC は独自のドメインを管理する DNS サーバー機能も保持しており, DC の物理 IP アドレスは NS レコードに登録されている. また, 各 DC のクライアントは DC のドメインを持つ FQDN が割り当てられており, すべてのクライアントは FQDN を用いて, どの DC に管理されているのかを NS レコードを確認することにより知ることが可能となる. 結果として, NTMobile では, 異なる管理者により運用されている DC が自律分散的に連携して, 各 DC 配下のクライアント間のトンネル通信を実現可能である.

なお, NTMobile では, すべての DC 及び RS が NTMobile ネットワークの管理者から配布されるデジタル証明書を保持している. このデジタル証明書は, DC 及び RS 間で初めて通信が行われるときに, 相互のサーバーが正当な NTMobile ネットワークのサーバーであることを確認するとともに, サーバー間で暗号通信を行うための共通鍵の共有で利用される. また, NTMobile では, 後述する手順を用いて, DC と各クライアント間で共通鍵の共有も行う. そのため, NTMobile ネットワークで利用される通信は, すべて何らかの鍵により暗号化処理が行われる.

3. 提案プラットフォーム

3.1 システムモデル

図 1 は提案プラットフォームのシステムモデルを示す.

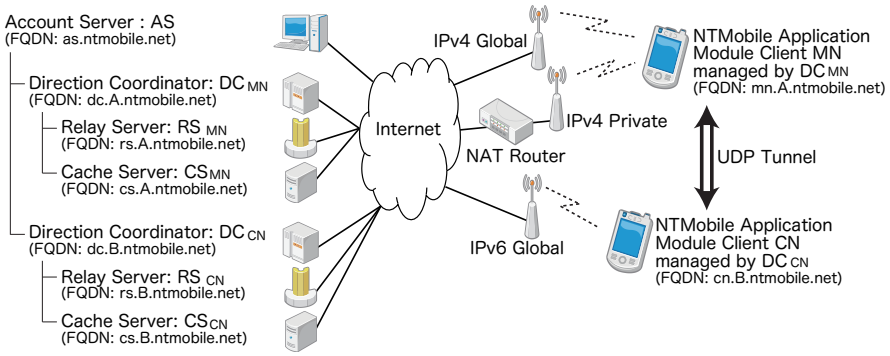

図 1　System model.

提案プラットフォームは既存の NTMobile を拡張することにより, 実用的なアプリケーション開発が可能な環境を用意することを目的としている. 本システムは, DC 及び RS に加えて, Account Server (AS) と Cache Server (CS) を導入する. AS はクライアントの認証を行うサーバーであり, すべての DC と連携して処理を行うことを想定する. また, DC は自身の RS と CS を管理することを想定しており, RS は既存の NTMobile と同様にトンネル中継処理を行い, 新たに導入する CS はクライアント間のデータ交換処理を支援する. なお, クライアントの NTMobile 機能は NTMobile アプリケーションモジュールとしてパッケージ化されており, 各スマートフォンアプリケーション内のライブラリとして実装される. そのため, NTMobile の機能を有するスマートフォンアプリケーションは一般的なアプリケーションと同様にユーザ権限でスマートフォン上にインストールすることが可能となる. なお, 本プラットフォームでは, クライアントのアプリケーションが共に起動しておりリアルタイムにデータ交換をする枠組みと, クライアントのアプリケーションが別のタイミングで起動しており, 非リアルタイムにデータ交換をする枠組みを提供する. 以下に, 構成要素の詳細を述べる.

- Account Server

  AS はフレームワーク内で唯一のサーバーであり, ユーザの認証情報の管理を行う. そのため, クライアントは AS に問い合わせることにより, 自身を管理する DC などの NTMobile に関連する初期情報を入手することが可能である. なお, クライアントの認証処理は DC に対して問い合わせが行われ, DC が AS に再問い合わせを行うことにより, 認証処理を実現している. これは, クライアントの再認証処理は, AS のみが持つユーザー認証情報だけを利用するのではなく, DC とクライアント間で共有している共通鍵なども用いており, 一定時間内の再認証は共通鍵のみで実現できるためである. 結果として, 各クライアントからの認証要求の多くは DC で処理することが可能となり, 初回認証時及びクライアントが長期間オフライン後に再認証を行う場合のみ, AS への問い合わせが行われる.

- Direction Coordinator

  DC は複数の NTMobile アプリケーションモジュールを利用したクライアントの管理を行う. 各クライアントは自身の物理 IP アドレスなどのネットワーク情報を DC に適時通知することにより, DC は各クライアントの所在を把握する. また, DC は RS, CS の管理も行う. RS は既存の NTMobile 同様にクライアント間で直接通信が実現できない場合のトンネル中継を行う. CS はクライアント間のデータ交換を支援することを目的としており, データ送信先のクライアントが通信不可能な状況においても, CS がデータを一時的に保管することにより, データ送信先クライアントの通信復帰後のデータ交換を実現する.

  DC はクライアントからの認証要求に対して, AS に問い合わせを行うことにより認証結果の回答を行う. そして, 認証が承認された場合には, DC とクライアント間で暗号化に利用する共通鍵を交換し, その後の再認証処理は, 共通鍵を用いて共通鍵の更新を行う. なお, スマートフォンでは任意のアプリケーションのインストールが可能なことから, NTMobile アプリケーションモジュールを採用したアプリケーションが同一クライアント上に複数インストールされる可能性がある. これらのアプリケーションは異なるポート番号を用いて DC と通信を行うため, DC は各アプリケーションの利用ポート番号も管理を行う. なお, 本フレームワークでは, 各アプリケーションはフレームワーク内で一意となるアプリケーション ID を割り当てられるものとする.

- Relay Server

  RS はクライアント間で直接トンネル構築ができない場合, RS を経由したトンネル中継を行うために導入されている. 具体的には, 両クライアントが NAT ルーター配下のプライベートネットワークに接続されている場合と, IPv4 と IPv6 のように, 各クライアントが異なる IP バージョンのネットワークに接続されている

場合となる．なお, RS の利用は DC がクライアントの接続ネットワーク情報に基づいて判断を行う．

- Cache Server

  CS はクライアントからのデータを一時的に保管する機能を持つ．そのため, CS を利用することにより, データ交換を行うクライアントが同時にアプリケーションを起動する必要はなく, 送信元クライアントのデータアップロード後に, 送信先クライアントが任意の時間にデータダウンロードを行うことが可能となる．これは, スマートフォンの利用形態では, ネットワークサービスの圏外によるオフラインが発生することが想定されるためである．なお, 本プラットフォームでは, クライアント間で交換データを暗号化するための共通鍵を DC と CS を用いて交換することが可能である．そのため, 交換データは CS 上に一時的に保管されるが, 交換データの中身は共通鍵を持つクライアント以外確認できず, クライアント間のセキュアなデータ交換を実現可能である．

3.2 リアルタイム通信

提案プラットフォームのリアルタイム通信は, 既存の NTMobile の枠組みを拡張してクライアント間の通信を実現する．そのため, メッセージ形式やシグナリングは NTMobile と互換性を持つ．なお, 既存の NTMobile の枠組みとの大きな違いは, スマートフォンのアプリケーションを呼び出すために, Apple Push Notification Service (APNS), Google Cloud Messaging (GCM) などの通知技術 [18], [19] を利用する点, 及びクライアント認証と共通鍵に関する通信手順を定義した点である．

図 2 はクライアント間のトンネル通信過程である．図では, MN がグローバル IP アドレスを利用しており, CN は NAT ルータ NAT_{CN} 配下のプライベート IP アドレスを利用している状況である．以下に, クライアント間のトンネル構築までの基本手順を述べる．

- ログイン処理

  MN 及び CN は各自のアプリケーションの利用を開始する際に, ログイン処理を行う．その結果, NTMobile アプリケーションモジュールは自身の DC に対して Login Request メッセージに認証情報を含めて送信する．DC は認証情報を含む Authentication Request メッセージを AS に送信することにより, AS に認証を依頼する．AS は認証結果を Authentication Response メッセージを用いて DC に返信することにより, DC は認証結果を得る．正規クライアントと認証された場合, 各 DC は MN 及び CN との通信で利用する共通鍵を生成し, Login Response メッセージを用いてクライアントと共通鍵の共有を行う．共有した共通鍵は, 以降の MN と DC_{MN} 及び CN と DC_{CN} 間のメッセージ交換の暗号化とメッセージ認証で利用される．

- ネットワーク情報の登録処理

  MN 及び CN はログイン処理後に APNS 又は GCM サーバーに問い合わせを行うことにより, アプリケーションの通知で利用するデバイストークンを取得する．次に, 取得したデバイストークンと自身の物理 IP アドレスなどのネットワーク情報を Registration Request メッセージを用いて DC_{MN} 及び DC_{CN} に通知を行う．DC_{MN} 及び DC_{CN} は Registration Response メッセージを返信することにより, MN 及び CN にネットワーク情報の更新が行われたことを通知する．なお, 接続ネットワークの切り替えなどにより, 物理 IP アドレスが変化した場合には, Registration Request メッセージの送信を改めて行う．これは, 通信継続性を実現するためには, 新たな IP アドレスを DC_{MN} 及び DC_{CN} が把握する必要があるためである．

- 通信先 DC の発見処理

  MN が CN への通信を行いたい場合, MN は DC_{MN} に向けてトンネル構築の依頼を示す Direction Request メッセージを送信する．NTMobile では, 各 DC が自身の DNS ドメインを管理しており, すべてのクライアントは FQDN により管理されている．そのため, DC_{MN} は CN を管理している DC_{CN} を発見するために, CN のドメイン管理サーバーを示す NS レコードを確認する．その結果, DC_{MN} は DC_{CN} のホスト名を取得する．次に, DC_{MN} は DC_{CN} に対して NTM Information Request メッセージを送信することにより, CN のネットワーク情報の交換を依頼する．DC_{CN} は NTM Information Response メッセージ内に CN のネットワーク情報を含めて返信をする．DC_{MN} は MN と CN のネットワーク情報を確認することにより, 構築するトンネル経路を判定し, 本例では CN から MN に向けてトンネル構築を開始する判断を行う．これは, CN は NAT ルータ配下のネットワークに接続しており, MN から CN への接続が不可能なためである．

- トンネル構築の準備処理

  構築するトンネル経路判定後, DC は構築トンネル用の共通鍵を交換するために利用する, 一時共通鍵を生成する．次に, DC_{MN} は Route Direction メッセージを DC_{CN} に送信することにより, CN へのトンネル構築に関する指示と一時共通鍵を送信する．DC_{CN} は CN のデバイストークンを用いて APNS, GCM などの通知技術を用いて CN の該当アプリケーションを呼び出す．通知技術により呼び出されたアプリケーションは, Route Direction Request メッセージを DC_{CN} に送信することにより, 自身宛の Route Direction メッセージの送信依頼を行う．DC_{CN} は CN へのトンネル構築に関する指示と一時共通鍵を Route Direction

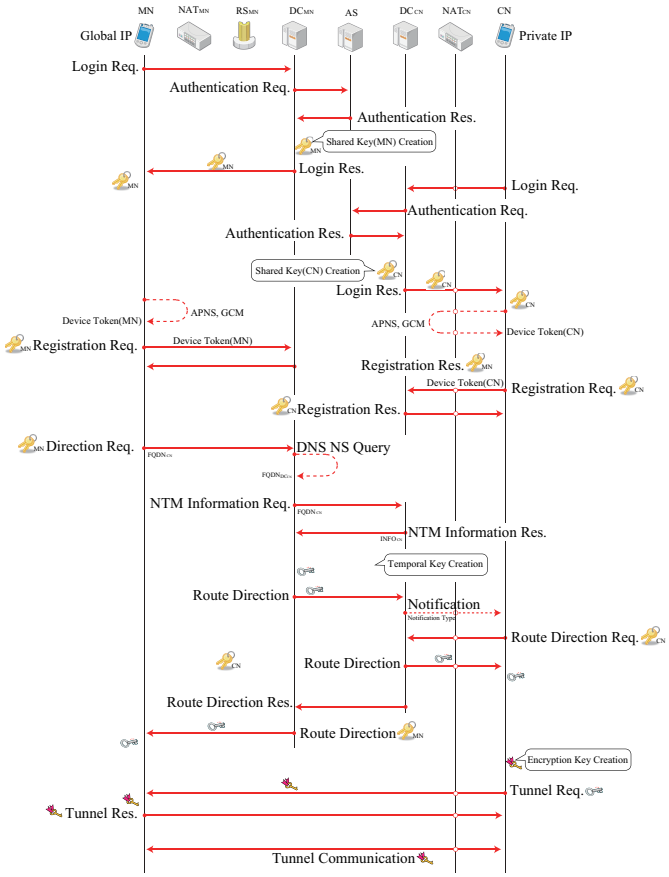

図 2 Tunnel creation process for realtime communication(Direct).

Response メッセージを用いて送信する．また，DC_{CN} は DC_{MN} に Route Direction Response メッセージを返信することにより，CN を正常に呼び出したことを通知する．次に，DC_{MN} は MN に向けてトンネル構築に関する指示と一時共通鍵を Route Direction メッセージを用いて送信する．

- トンネル構築処理

  MN 及び CN は Route Direction メッセージの指示に従いトンネル構築処理を開始する．なお，トンネルの構築を開始する端末はトンネル通信の暗号化及びメッセージ認証で利用するトンネル通信用共通鍵を生成する．本例では，CN が MN に向けてトンネル構築を行うため，CN がトンネル通信用共通鍵の生成を行う．また，CN は MN に向けてトンネル通信用共通鍵が含まれている Tunnel Request メッセージを送信する．なお，Tunnel Request メッセージは，DC_{MN} から配布された一時共通鍵を用いて暗号化を行う．次に，MN は CN に向けて Tunnel Response メッセージを返信することにより，トンネル構築の確認を行う．なお，Tunnel Response メッセージはトンネル通信用共通鍵を用いて暗号化を行うことにより，トンネル構築の確認とトンネル通信用共通鍵の共有完了の確認を行う．

図 3 は RS を用いる場合のクライアント間のトンネル通信過程である．図では，MN は NAT ルータ NAT_{MN} 配下のプライベート IP アドレスを利用しており，CN は NAT ルータ NAT_{CN} 配下のプライベート IP アドレスを利用している状況である．以下に，クライアント間のトンネル構築までの基本手順を述べる．なお，ログイン処理などは図 2 と同様のため省略する．

- 通信先 DC の発見処理

  図 2 と同様のプロセスにより，MN は DC_{MN} にトンネル構築の依頼を行い，DC_{MN} は DC_{CN} を探索することにより，CN のネットワーク情報の交換を依頼する．図 3 では，MN と CN が共に NAT ルータ配下のネットワークに接続されているため，互いに Tunnel Request メッセージを受け入れることができない．そこで，DC_{MN} は RS を利用するトンネル中継が必要と判断する．

- RS の中継準備処理

  DC_{MN} は RS と MN 及び CN が通信を行う際に利用する RS 用の共通鍵を生成し，Relay Direction メッセージにより RS にトンネル中継を指示する．RS は Relay Response メッセージを返信することにより，RS が MN と CN のトンネル中継の準備が終わったことを DC_{MN} に通知する．

- トンネル構築の準備処理

  DC_{MN} は MN と CN 間のトンネル通信で利用するトンネル通信用共通鍵を交換するための一時共通鍵を生成する．図 2 と同様のプロセスにより，DC_{CN} は CN の該当アプリケーションを通知技術により呼び出し，Route Direction メッセージを用いてトンネル経路の構築手法を指示する．RS 利用時と非利用時の違いは，RS 利用時は RS 用共通鍵も送付する必要がある点である．

- トンネル構築処理

  MN 及び CN は Route Direction メッセージの指示に従いトンネル構築処理を開始する．また，通信開始側端末である MN は，MN と CN 間のトンネル通信で利用するトンネル通信用共通鍵を生成する．次に，MN と CN は Tunnel Request メッセージを RS に向けて送信する．なお，Tunnel Request メッセージは RS 用共通鍵により暗号化されている．また，MN からの Tunnel Request メッセージに含まれるトンネル通信用共通鍵は，DC_{MN} から配布された一時共通鍵により暗号化される．RS は MN 及び CN から Tunnel Request メッセージを受信した場合，MN からの Tunnel Request メッセージの中身を CN に Tunnel Response メッセージとして転送し，CN からの Tunnel Request メッセージの中身を MN に Tunnel Response メッセージとして転送する．その結果として，一時共通鍵で暗号化さ

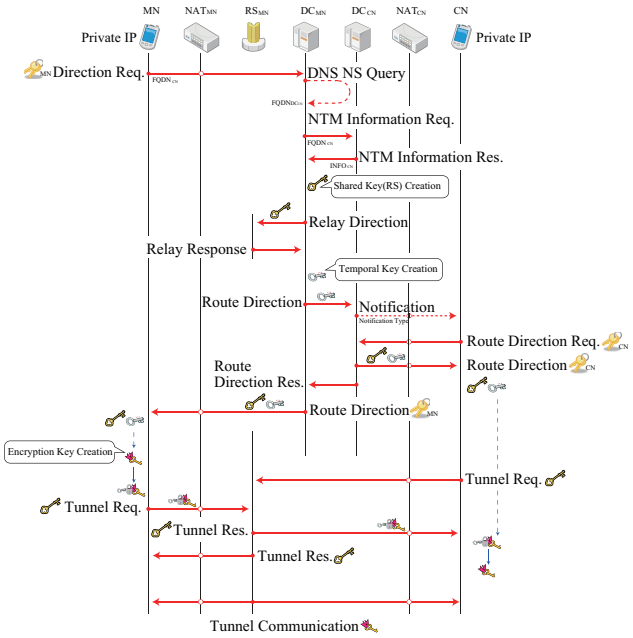

図 3　Tunnel creation process for realtime communication(RS).

れたトンネル通信用共通鍵を MN と CN 間で共有する．その後のトンネル通信では，トンネル通信用共通鍵を用いて暗号化を行う．

3.3　非リアルタイム通信

スマートフォンはサービスエリア圏外などによりオフラインとなることがあるため，提案プラットフォームでは，非リアルタイムでデータ交換を行う CS を導入する．サーバーを介したデータ交換の場合，多くのサービスではネットワーク部の暗号化のみを実施しており，サーバーでは非暗号のデータを処理していることが多い．その結果，サーバー管理者はすべてのデータ交換内容を把握することも技術的には可能である．提案プラットフォームでは，データ交換内容をサーバー管理者すら確認できない方法を目指しており，各クライアントが交換データ暗号用の共通鍵を予め交換する方式を提案する．提案方式では，各クライアントのみが持つ共通鍵を用いてデータを暗号化した後，CS にデータ交換を依頼する．そのため，CS に保管されるデータは暗号化されたデータとなり，クライアント以外はデータの内容を把握することができない．

3.4　鍵交換手順

NTMobile ではすべてのデータはクライアント間で暗号化されることを想定している．非リアルタイム通信では，クライアントからのデータが CS を経由するが，CS に一時保管されるデータを予めクライアントが暗号化することにより，データの安全性を担保している．

図 4 は非リアルタイム通信における鍵交換手順を示す．図では MN が CN にデータを送信する状況を示し，データ送信の前に MN が CN に共通鍵を渡す手順を示す．

- 一時共通鍵のアップロード

  データ送信元となる MN は一時共通鍵を作成する．この一時共通鍵は Temporal Key Upload Request メッセージを用いて MN の DC である DC$_{MN}$ に送信される．DC$_{MN}$ は CN の FQDN から DC$_{CN}$ を探索し，Temporal Key Exchange Request メッセージを用いて一時共通鍵を送信する．DC$_{CN}$ は一時共通鍵を保管し，Temporal Key Exchange Response メッセージを返信する．DC$_{MN}$ は Temporal Key Upload Response を MN に返信することにより，一時共通鍵の CN への配送準備が終わったことを確認する．
- データ暗号用共通鍵のアップロード

  MN はデータの暗号化で利用するデータ暗号用共通鍵を作成する．作成されたデータ暗号用共通鍵は，一時共通鍵を用いて暗号化され，Encryption Key Upload Request メッセージを用いて CS に送信される．CS は CN の FQDN から DC$_{CN}$ を探索し，Encryption Key Exchange Request メッセージを DC$_{CN}$ に送信する．DC$_{CN}$ は APNS，GCM などの通知技術を用いて CN の該当アプリケーションを呼び出す．また，DC$_{CN}$ は Encryption Key Exchange Response メッセージを返信する．DC$_{MN}$ は Encryption Key Upload Response メッセージを MN に返信することにより，データ暗号用共通鍵の交換準備が終わったことを確認する．
- 一時共通鍵のダウンロード

  通知技術により呼び出しを受けた CN は自身の DC である DC$_{CN}$ に Temporal Key List Download Request メッセージを送信する．DC$_{CN}$ は Temporal Key List Download Response メッセージに一時共通鍵を取得するため情報を返信する．次に，CN は Temporal Key Download Request メッセージを DC$_{CN}$ に送信することにより，一時共通鍵を返信を依頼する．DC$_{CN}$ は Temporal Key Download Response メッセージを用いて一時共通鍵を返信する．
- データ暗号用共通鍵のダウンロード

  CN は Cache List Download Request メッセージを DC$_{CN}$ に送信することにより，CN 宛のデータ暗号用共通鍵を取得するための情報を問い合わせる．DC$_{CN}$ は Cache List Download Response メッセージを用いて，データを保管している CS$_{MN}$ の FQDN とデータをダウンロードするためのハッシュ値を返信する．CN は Encryption Key Download Request メッセージを CS$_{MN}$ に送信し，CS$_{MN}$ は Encryption Key Download Response メッセージを返信することにより，一時共通鍵で暗号化されたデータ暗号用共通鍵を CN に配信する．CN は既にダウンロード済みの一時共通鍵を用いてデータ暗号用共通鍵を復号する．

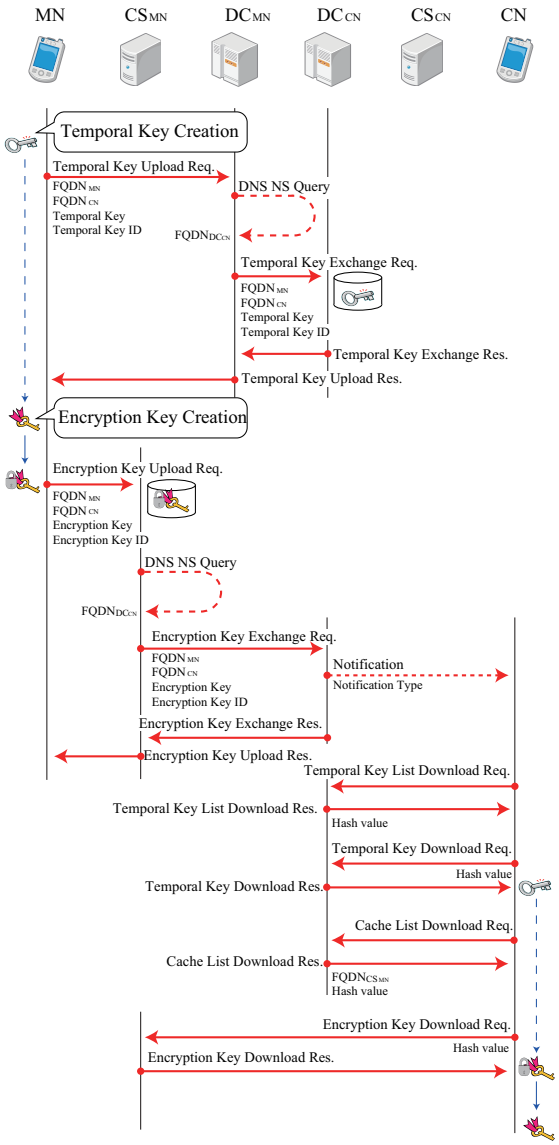

図 4　Key exchange process.

3.5　データ交換手順

MN 及び CN は交換したデータ暗号用共通鍵を用いてデータの暗号化を行う．データ交換手順の詳細を以下に述べる．

- 暗号化データのアップロード

  MN は送信したいデータをデータ暗号用共通鍵を用いて暗号化を行い，自身の CS である CS_{MN} に Encrypted Data Upload Request メッセージを用いてアップロードを行う．CS_{MN} は CN の FQDN を用いて DC_{CN} を探索し，Encrypted Data Exchange Request メッセージを送信することにより，CN 宛のデータがアップロードされたことを通知する．DC_{CN} は APNS, GCM などの通知技術を用いて CN にデータがアップロードされたことを通知する．次に，DC_{CN} は CS_{MN} に Encrypted Data Exchange Response メッセージを返信し，CS_{MN} は MN に Encrypted Data Upload Response メッセージを返信することにより，アップロード処理が完了したことを通知する．

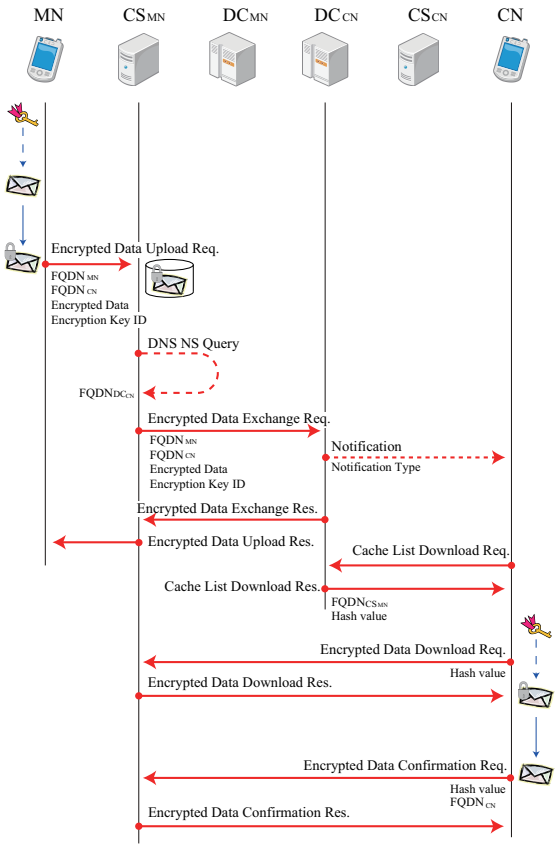

図 5　Data exchange process.

- 暗号化データのダウンロード

  CN は Cache List Download Request メッセージを DC_{CN} に送信することにより，自身宛のデータリストを確認する．DC_{CN} は Cache List Download Response メッセージを返信することにより，CN 宛のデータが保管されている CS の FQDN とハッシュ値を通知する．CN は Encrypted Data Download Request メッセージを CS_{MN} に送信し，CS_{MN} は Encrypted Data Download Response メッセージを用いて暗号化されたデータを通知する．CN は予め交換してあるデータ暗号用共通鍵を用いて暗号化されたデータを復号し，CS_{MN} に Encrypted Data Confirmation Request メッセージを送信し，データを正常に受信したことを通知する．CS_{MN} は Encrypted Data Confirmation Response メッセージを返信することにより，データ交換処理を終了する．

4.　実装

図 6 に実装モデルを示す．提案プラットフォームの実装では，NTMobile の機能を C 言語で記述するとともに，HTTP に関する機能は perl で記述した．また，Cent OS 6.4 上に mySQL のデータベースを構築することにより，関係す

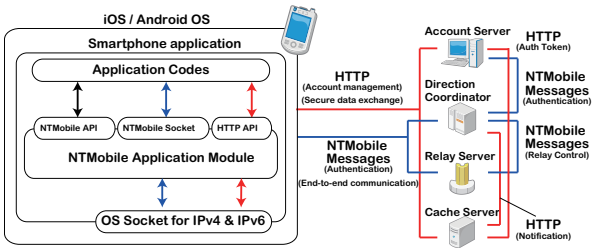

図 6　Implementation model.

る情報を管理した．NTMobile アプリケーションモジュールでは，アカウント認証とソケット通信を行う API をアプリケーション開発者に向けて準備している．そのため，アプリケーション開発者は NTMobile の詳細に関与することなく，移動透過技術と接続性技術を利用可能となる．また，データ交換に関する実装は，スマートフォン開発環境で様々な API が準備されている HTTP を用いて実装を行う．

4.1　NTMobile の拡張

提案プラットフォームを実現するために，既存の NTMobile の実装を下記の点で拡張した．

- 認証機能
  AS の導入に伴い，すべての DC は AS に問い合わせを行うことにより認証を行う．そこで，クライアントからのログイン認証処理と AS への問い合わせ機能を実装した．
- 認証用トークンの配布
  CS 利用時には認証用トークンを用いてクライアント認証を実現している．認証用トークンは AS により生成され，DC を経由して認証処理時に配布される機能を実装した．
- 通知技術
  提案プラットフォームでは，iOS 及び Android OS のスマートフォンを想定してアプリケーションモジュールの開発を行った．そのため，APNS と GCM を想定した通知技術の機能を実装した．通知技術を利用するためには，クライアントを識別する情報を管理する必要があるため，DC を拡張することにより，クライアントへ通知するためのデバイストークンなどを管理する機能を実装した．
- 一時共通鍵の保管
  提案プラットフォームでは，CS を経由する非リアルタイム通信によるデータ交換時に，共通鍵を交換する必要がある．この際，CS とは異なる経路で一時共通鍵を配布する必要があることから，DC に一時共通鍵の保管機能を実装した．

4.2　非リアルタイム通信

提案プラットフォームでは，CS を利用した非リアルタイム通信を実現している．非リアルタイム通信では，多くのスマートフォンにおいて API が準備されている HTTP を用いて実装を行う．実装では下記の情報を付加して非リアルタイム通信を実現する．

- バージョン
  NTMobile アプリケーションモジュールのバージョン情報を示す．
- 認証トークン
  認証トークンは CS がクライアントを認証するために利用される．認証トークンは，クライアントが DC に認証要求を行う際に配布される．
- 自身の FQDN
  NTMobile では，すべてのクライアントが FQDN により識別されている．そのため，データの送信元を示す自身の FQDN を付加する．
- 宛先の FQDN
  非リアルタイム通信では，複数の宛先に同時にデータを送信可能である．そのため，宛先の FQDN のリストを付加する．
- アプリケーション ID
  アプリケーションを識別するために，アプリケーション ID を付加する．
- サービス ID
  アプリケーション内のサービスを識別するため，サービス ID を付加する．
- サービスアイテム ID
  サービス内の項目を識別するため，サービスアイテム ID を付加する．
- 通知メッセージ
  APNS, GCM などの通知技術利用時の表示メッセージを付加する．

4.3　評価

提案プラットフォームの動作を確認するため，iOS と Android OS 用のサンプルアプリケーションを開発した．サンプルアプリケーションでは，UDP によるリアルタイム通信と HTTP による非リアルタイム通信を実装した．評価では，iPhone5 (iOS 6.1) と Galaxy Nexus (Android 4.2) をデバイスとして利用した．これらのデバイスを IPv4 プライベートネットワーク，IPv4 グローバルネットワーク，IPv6 グローバルネットワークに WiFi を用いて接続した．実験では，すべてのネットワークにおいてリアルタイム通信が実現可能であり，アクセスネットワーク切り替え時にも数秒以内でコネクションが復旧することを確認した．また，非リアルタイム通信についても，HTTP を用いてデータ交換が正常に行われることを確認した．

5. まとめ

本稿では, スマートフォンアプリケーションをユーザ権限でインストール可能なエンド間通信フレームワークを提案した. 提案フレームワークでは, NTMobile を拡張することによりリアルタイム通信を実現し, 新たにデータを一時保存する CS を導入することにより, 非リアルタイム通信によるデータ交換も実現した. また, リアルタイム通信, 非リアルタイム通信ともに, すべてのデータを暗号化する鍵交換手法を提案することにより, セキュアなエンド間通信を実現した. 実装評価では, 開発した NTMobile アプリケーションモジュールを iOS と Android OS で利用した場合について動作確認を行い, 適切な動作が可能であることを確認した.

謝辞

本研究は科研費 (23700075, 26330103), 総務省 SCOPE2013 の助成を受けたものである. 記して謝意を表する.

参考文献

[1] M. Buddhikot, G. Chandranmenon, S. Han, Y. W. Lee, S. Miller and L. Salgarelli, “Integration of 802.11 and third-generation wireless data networks,” Proceedings of the IEEE INFOCOM 2003, Vol. 1, pp. 503-512, 2003.

[2] Q. Zhang, C. Guo, Z. Guo and W. Zhu, “Efficient mobility management for vertical handoff between WWAN and WLAN,” IEEE Communications Magazine, Vol. 41, No. 11, pp. 102-108, 2003.

[3] L. A. Magagula and H. A. Chan, “IEEE802.21-Assisted Cross-Layer Design and PMIPv6 Mobility Management Framework for Next Generation Wireless Networks,” Proc. IEEE WIMOB’08, pp. 159-164, Oct. 2008.

[4] D. Le, X. Fu and D. Hogrere, “A Review of Mobility Support Paradigms for the Internet,” IEEE Communications surveys, 1st quarter 2006, Volume 8, No. 1, 2006.

[5] C. Perkins, “IP Mobility Support for IPv4, Revised,” RFC 5944, IETF (2010).

[6] M. Ishiyama, M. Kunishi, K. Uehara, H. Esaki, and F. Teraoka, “LINA: A New Approach to Mobility Support in Wide Area Networks,” IEICE Trans. Comm., Vol.E84-B, No.8, pp.2076– 2086, 2001.

[7] http://www.mip4.org, retrieved: February , 2012.

[8] S. Salsano, C. Mingardi, S. Niccolini, A. Polidoro and L. Veltri ”SIP-based Mobility Management in Next Generation Networks,” IEEE Wireless Communication, Vol. 15, Issue 2, April 2008.

[9] M. Bonola, S. Salsano and A. Polidoro, “UPMT: universal per-application mobility management using tunnels,” In Proc. of the 28th IEEE conference on Global telecommunications (GLOBECOM’09) 2009.

[10] M. Bonola and S. Salsano, “S-UPMT: a secure Vertical Handover solution based on IP in UDP tunneling and IPsec,” GTTI Riunione Annuale 2010, (online), `http://www.gtti.it/GTTI10/papers/gtti10\_submission\_29.pdf`, 2010.

[11] E. Fogelstroem, A. Jonsson, and C. Perkins, “Mobile IPv4 Regional Registration,” RFC 4857, June 2007.

[12] H. Levkowetz and S. Vaarala, “Mobile IP Traversal of Network Address Translation (NAT) Devices,” RFC 3519, April 2003.

[13] G. Montenegro, “Reverse Tunneling for Mobile IP, revised,” RFC 3024, January 2001.

[14] Katsuhiro Naito, Kazuma Kamienoo, Takuya Nishio, Hidekazu Suzuki, Akira Watanabe, Kazuo Mori, Hideo Kobayashi, “Proposal of Seamless IP Mobility Schemes: Network Traversal with Mobility (NTMobile),” IEEE GLOBECOM 2012, December 2012.

[15] Kazuma Kamienoo, Hidekazu Suzuki, Katsuhiro Naito, Akira Watanabe, “Implementation and Evaluation of NTMobile with Android Smartphones in IPv4/IPv6 Networks,” IEEE Global Conference on Consumer Electronics (GCCE2012)，pp.125 － 129, October 2012.

[16] Kazuma Kamienoo, Hidekazu Suzuki, Katsuhiro Naito, Akira Watanabe, “Development of Mobile Communication Framework based on NTMobile,” ICMU 2014, Jan. 2014.

[17] `http://www.ntmobile.net`

[18] `http://developer.apple.com/library/mac/\#documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Introduction.html`

[19] http://developer.android.com/google/gcm/