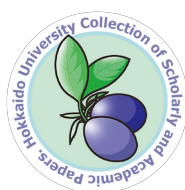




Title	DNSとPCPを用いたプライバシー配慮型IoT遠隔制御システムの設計と評価
Author(s)	相良, 隼; 金, 勇; 砂原, 悟 他
Citation	電子情報通信学会技術研究報告, 125(385), 202-207
Issue Date	2026-02-25
Doc URL	https://hdl.handle.net/2115/100411
Rights	Copyright ©2026 IEICE
Type	conference paper
File Information	NS2025-255.pdf



DNS と PCP を用いたプライバシー配慮型 IoT 遠隔制御システムの設計と評価

相良 隼[†] 金 勇^{††} 砂原 悟^{†††} 飯田 勝吉^{†††} 高井 昌彰^{†††}

[†] 北海道大学 大学院情報科学院 情報理工学コース 先端ネットワーク研究室

〒060-0811 札幌市北区北 11 条西 5 丁目

^{††} 東京科学大学 情報基盤センター

^{†††} 北海道大学 情報基盤センター

E-mail: [†]sagara.hayato.y1@elms.hokudai.ac.jp, ^{††}yongj@cii.isct.ac.jp, ^{†††}{suna,iida,ytakai}@iic.hokudai.ac.jp

あらまし IoT の普及によって多様な機器のネットワーク接続が可能となった一方で、NAT によりネットワークが分離された環境における外部からの通信の困難性や、データの機密性確保などのプライバシー保護が課題となっている。本研究では DNS と PCP を連携させ、これら双方の課題に対処するセキュアな IoT 遠隔制御システムを提案する。提案手法の設計・実装を行うとともに、提案システムが期待通りの動作をすることを仮想環境上での機能評価により確認する。

キーワード Internet of Things (IoT), プライバシー, NAT, Port Control Protocol (PCP), CoAP, OSCORE, DNS, DNS over TLS (DoT)

Design and evaluation of privacy-preserved IoT remote control system using DNS and PCP

Hayato SAGARA[†], Yong JIN^{††}, Satoru SUNAHARA^{†††}, Katsuyoshi IIDA^{†††}, and Yoshiaki TAKAI^{†††}

[†] Laboratory of Advanced Information Network,

Course of Computer Science and Information Technology, Graduate School of Information Science & Engineering,
Hokkaido University, Kita 11 Nishi 5, Kita-ku, Sapporo, 060-0811, Japan.

^{††} Center for Information Infrastructure, Institute of Science Tokyo

^{†††} Information Initiative Center, Hokkaido University

E-mail: [†]sagara.hayato.y1@elms.hokudai.ac.jp, ^{††}yongj@cii.isct.ac.jp, ^{†††}{suna,iida,ytakai}@iic.hokudai.ac.jp

Abstract While the widespread adoption of the Internet of Things (IoT) has enabled network connectivity for diverse devices, challenges remain in establishing external communication across NAT-separated environments and ensuring privacy through data confidentiality. This paper proposes a secure IoT remote control system that integrates DNS and Port Control Protocol (PCP) to address these issues. We design and implement the proposed system, and verify that the proposed system operates as expected through functional evaluation in a virtual environment.

Key words Internet of Things (IoT), Privacy, NAT, Port Control Protocol (PCP), CoAP, OSCORE, DNS, DNS over TLS (DoT)

1. はじめに

近年、Internet of Things (IoT) 技術の急速な普及に伴い、スマートホームや産業用ネットワークなどにおいて、内部に存在する多種多様な機器を外部からリアルタイムで監視・制御するニーズが高まっている。しかし、実運用における大きな障壁として、多くの組織や家庭のネットワークが Network Address

Translation (NAT) を用いて構築されている点が挙げられる。NAT はセキュリティや IP アドレス資源の有効活用の観点から必須の技術であるが、その特性上、外部ネットワークからプライベート IP アドレスを持つ内部機器への通信到達性は制限される。先行研究では、ホームネットワーク内で生成される IoT 機器のデータを、外部クライアントがセキュアに取得するための手法が実現された。しかし、実用的な IoT システムの運用に

においては、外部クライアントからホームネットワーク内の IoT 機器へセキュアに制御データを送信する遠隔制御機能が求められる。本研究では、従来の手法を拡張し、NAT 環境下においても外部クライアントによる内部機器へのセキュアかつ容易なアクセスを可能とする IoT 遠隔制御システムを提案する。

2. 関連研究

2.1 既存の遠隔制御システム

本研究に関連する先行研究として、NAT 環境下における IoT 通信の接続性を確立する手法や、セキュアな遠隔接続技術に関する研究が多数提案されている。

田中らは、通信端末同士が常に直接接続されているかのように見える通信経路を仮想的に構成することで、NAT 環境下における外部への通信制限や、通信端末の移動に伴う IP アドレス変更の影響を低減する手法を提案した。[1]。

Lu らは、IoT 環境におけるネットワークセキュリティおよび遠隔制御技術について検討し、クラウドプラットフォームを中心とした IoT 遠隔制御アーキテクチャを提案した[2]。ネットワークセキュリティ評価においては、遺伝的アルゴリズムと誤差逆伝播型ニューラルネットワークを組み合わせた GA-BPNN を用いることで、高精度なセキュリティ評価を実現できることを示している。

Novo は、Constrained Application Protocol (CoAP)[3] と Port Control Protocol (PCP)[4] を活用した、IPv4 ネットワークと IPv6 ネットワークにそれぞれ存在する IoT 機器の間で、IP プロトコルの差異を意識させない透過的かつセキュアな通信を実現するシステムを提案した[5]。このシステムには、2つのセキュリティ上の懸念が存在する。1つ目は、IoT データの暗号化や IoT 機器の認証が行われていない点である。PCP とリソースに基づくポートマッピングは実現されているものの、ポートマッピング以降における IPv4 機器と IPv6 機器間の通信については、十分なセキュリティ対策が考慮されていない。2つ目は、Resource Directory (RD)[6] への問い合わせにおいて、認証やアクセス制御が行われていない点である。意図しない機器が RD からリソース情報を取得し、その中に存在するポートマッピング情報を利用して不正にアクセスするおそれがある。

これらの手法はクラウドプラットフォームへの依存度が高く、NAT 配下の IoT 機器に対する直接的な到達性の確保や、接続情報の分散的かつ安全な管理について十分に検討されていないという課題が存在する。これらの課題に対し、本研究では、著者らが過去に行った遠隔監視システム[7][8]を拡張することで、NAT 環境下の IoT 通信における接続性の確立、接続情報の分散的かつ安全な管理、ならびにプライバシー情報の保護を実現する IoT 遠隔制御システムの設計および実装を行った。

3. IoT 遠隔制御システムの設計と実装

3.1 IoT 遠隔制御システムが備えるべきセキュリティ要件

本研究では、IoT 遠隔制御システムがセキュリティおよびプライバシーの課題を解決するために満たすべき4つのセキュリティ要件を定義した。それぞれの要件を表1に示す。要件

(1) および要件 (2) は、不正な IoT 機器や想定外のクライアントによる不正アクセス、ならびに意図しない登録や情報取得を防止するために、IoT 機器およびクライアントをそれぞれ認証する要件である。要件 (3) および要件 (4) は、通信経路上に存在する機器や事業者による IoT データの窃取、ならびに通信情報の分析によるプライバシー情報の漏洩を防止するために、IoT データおよび通信をそれぞれ暗号化する要件である。ただし、IoT 機器は小型化や省電力性により計算資源や処理性能に制約があるため、各セキュリティ要件は IoT 機器自身による実現に限定せず、IoT システムの設計に応じて、IoT 機器の処理を補完する機器が一部または全部を担う構成も含めるものとした。

3.2 プライバシー配慮型 IoT 遠隔制御システムの設計手法

本研究では表1に示した要件を満たす IoT 遠隔制御システムを実現するために、DNS と PCP、OSCORE[9] を組み合わせた方式を新たに提案する。OSCORE は、低電力や低メモリ環境などの制約のあるデバイス向けに設計されたセキュリティプロトコルである。本システムの構成の概要を図1に示す。IoT 遠隔制御システムの主な処理の流れは次の通りである。

(1) IoT ゲートウェイが、PCP クライアントとしてホーム NAT にポートマッピングを要求する。

(2) IoT ゲートウェイがホーム NAT からマッピング応答を受け取り、マッピングされたポート番号を DNS SRV RR (Resource Record) として外部 DNS サーバに登録する。

(3) クライアントが外部 DNS サーバに SRV RR の問い合わせを行い、返答に付属する A RR と合わせてホーム NAT のマッピング情報を取得する。なお、当該 A RR にはホーム NAT のインターネット側の IP アドレスを記載する。

(4) クライアントがマッピング情報の宛先に、制御データを格納した CoAP メッセージを送信する。

(5) 制御データがホーム NAT を介して IoT ゲートウェイに送られ、その後 IoT 機器へと転送される。

提案方式では、制御データを格納した CoAP メッセージを OSCORE によって保護することで、クライアントから IoT 機器までの制御データのエンドツーエンド暗号化や、通信の秘匿化、遠隔制御時の IoT 機器およびクライアントの認証が可能となり、表1の要件(1)、要件(2)および要件(3)を満たす。また、DNS の通信に DNS over TLS (DoT)[10]を用いることにより、通信経路が暗号化で保護されるため、表1の要件(4)も満たされる。

図1に示した IoT ゲートウェイ、ホーム NAT、外部 DNS サーバ、クライアントの動作について、それぞれ以下に詳しく述べる。

IoT ゲートウェイ: IoT ゲートウェイは次の3つの処理を並列に行う。1つ目は、ポートマッピングの要求処理である。PCP クライアントとして、任意のポート番号を指定し、ホーム NAT に対してポートマッピングを要求する。指定するポート番号には、OS などが使用するシステムポート以外の番号を用いる。当該ポートマッピングの有効期限が経過した場合、IoT ゲートウェイは別のポート番号を用いて再度ポートマッピングを要求する。

表 1 IoT システムが設計において満たすべき 4 つのセキュリティ要件

各要件	要件の概要
(要件 1) IoT 機器の認証	不正な IoT 機器からの不正なデータの登録や受信を防止
(要件 2) クライアントの認証	想定外のクライアントによる意図しない登録や情報取得を防止
(要件 3) IoT データのエンドツーエンド暗号化	通信経路上に存在する機器や事業者による IoT データの窃取を防止
(要件 4) 通信の暗号化	通信情報の分析を通じたプライバシー情報の漏洩を防止

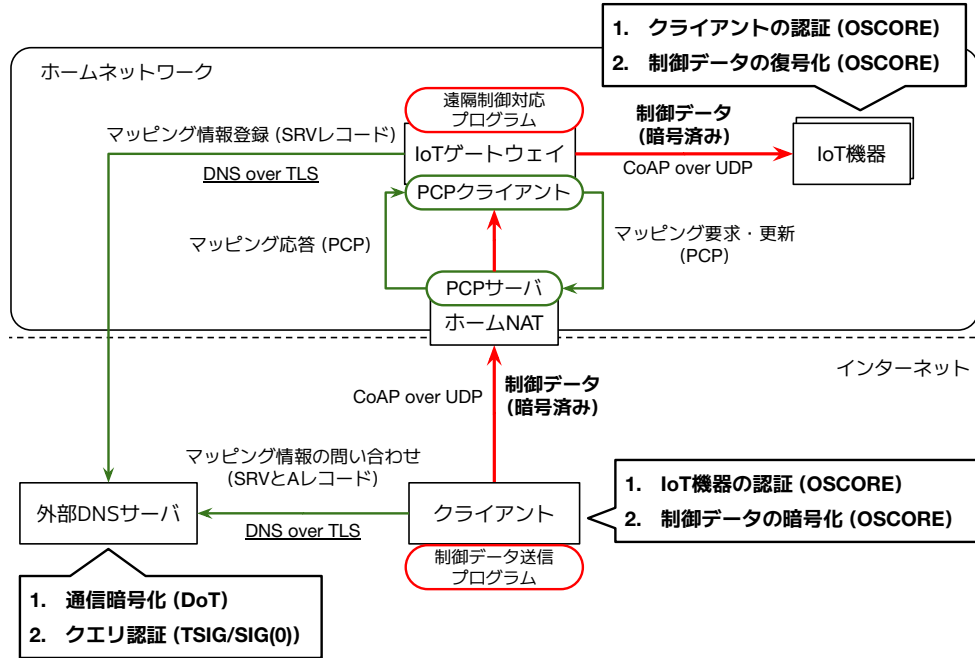


図 1 IoT 遠隔制御システムの概要

2 つ目は、ポートマッピング情報の登録処理である。ホーム NAT に対するポートマッピングで割り当てたポート番号を、SRV RR として外部 DNS サーバに登録する。登録の際は通信に DoT を、認証に TSIG または SIG(0) をそれぞれ使用する。

3 つ目は、制御データの転送処理である。ホーム NAT 経由で受信したパケットを、ホームネットワーク内の IoT 機器に転送する。さらに、当該応答パケットのホーム NAT への逆方向の転送も行う。

ホーム NAT: ホーム NAT は PCP サーバとして動作し、ポートマッピングに基づいて、ホームネットワーク外部から内部へのパケット転送や応答パケットの逆方向の転送を行う。

外部 DNS サーバ: 外部 DNS サーバは、ホーム NAT のマッピング情報を格納した SRV RR や A RR を管理し、クライアントや IoT ゲートウェイからの DNS クエリを受け付ける。問い合わせや登録の際、外部 DNS サーバは DoT による通信の暗号化を行うとともに、TSIG または SIG(0) による認証を要求する。

クライアント: クライアントはまず、外部 DNS サーバに SRV RR を問い合わせることでポートマッピング情報を得る。このとき、外部 DNS サーバ間との通信には DoT を、認証には TSIG または SIG(0) をそれぞれ使用する。次に、SRV RR とそれに付随する A RR のペイロードから、ホーム NAT のインターネット側の IP アドレス、およびマッピング中のポート番号を入手する。A RR が付属していない場合は、SRV RR に記載されたド

表 2 外部 DNS サーバの構成情報

OS	Ubuntu Desktop 24.10
DNS ソフトウェア	BIND 9.20.3

表 3 各機器の構成情報 (クライアントおよび外部 DNS サーバ以外)

OS	Ubuntu Desktop 24.04.3 LTS
DNS ソフトウェア	BIND 9.18.31
CoAP ソフトウェア	libcoap 4.3.4
Go	go1.24.1 linux/amd64
Python	Python 3.12.3
PCP サーバソフトウェア	miniupnpd 2.3.9
PCP クライアントソフトウェア	libpcpnatpmp

メイン名の A RR を直接問い合わせる。続いて、制御データを格納した CoAP メッセージを作成し、OSCORE によってメッセージを暗号化する。最後に、ホーム NAT のマッピング先に向けて CoAP メッセージを送信する。

4. IoT 遠隔制御システムの実装と検証環境

本研究の検証環境を表 2 および表 3 に示す。BIND9 [11] は DNS サーバの実装に、libcoap [12] は IoT 機器の実装に、プログラミング言語 Go [13] は遠隔制御対応プログラムと制御データ送信プログラムの実装にそれぞれ使用した。また、Python [14] は制御データ送信プログラムの実装にも一部使用した。

4.1 OSCORE の設定

OSCORE の実装に必要な鍵や ID を以下のように用意し、OSCORE の設定ファイルを記述した。

鍵について、事前共有鍵 (PSK) として実装を行うため、OpenSSL の “openssl rand” コマンドを用いてマスターシークレットとソルトをそれぞれ生成した。そして、これらの鍵情報を IoT 機器およびクライアントの両方に配置し、設定ファイルに記載した。

次に ID として、IoT 機器用の設定ファイルで送信者 ID に “IoTdev” を、受信者 ID に “client” をそれぞれ記載した。反対にクライアント用の設定ファイルで、送信者 ID に “client” を、受信者 ID に “IoTdev” をそれぞれ記載した。

4.2 IoT 機器の実装

IoT 機器は、libcoap の “coap-server” コマンドを改変なく用いて実装した。具体的には、“coap-server -d 4096 -E <IoT 機器の OSCORE 設定ファイル>” を実行し続ける。

4.3 ホーム NAT の実装

ホーム NAT の実装には、PCP サーバソフトウェアである miniupnpd [15] を使用した。具体的には、“miniupnpd -d” コマンドを実行して PCP サーバを立ち上げる。このコマンドによって、PCP クライアントからの PCP 要求の受け付けや、Linux のパケットフィルタリングツールと連携したポートマッピングの管理が可能となる。

4.4 遠隔制御対応プログラムの実装

IoT ゲートウェイ上の遠隔制御対応プログラムの実装について説明する。このプログラムは、ホーム NAT へのポートマッピング、ポートマッピング情報の外部 DNS サーバへの登録、およびホーム NAT 経由での外部からの受信待機と IoT 機器へのパケット転送をそれぞれ行う。DNS 関連の処理の実装には、Go 言語の dns パッケージ [16] を使用する。また、PCP クライアントの処理の実装には、PCP クライアントソフトウェアである libpcpnatpmp [17] の “pcpnatpmpc” コマンドを使用する。遠隔制御対応プログラムを図 2 に示す。このプログラムの具体的な処理の流れは下記の通りである。

(1) 最初に別プロセス (サブプロセス 1) を立ち上げる。

(2) メインプロセスでは、“socat -TIMEOUT UDP4-LISTEN:PORT,fork UDP4:SERVERIP:SERVERPORT” を実行し続け、ホーム NAT 外部からのパケットを順次 IoT 機器に転送する。ここで、TIMEOUT は “socat” コマンドのフォークプロセスのタイムアウト時間、PORT は待ち受けるポート番号、SERVERIP および SERVERPORT は転送先の IP アドレスとポート番号をそれぞれ指定する。本実装では、TIMEOUT は暫定的に 60 秒で記述する。また PORT および SERVERPORT に CoAP の標準ポート番号である 5683 を記述し、SERVERIP に CoAP サーバ (IoT 機器) の IP アドレスを記述する。

(3) サブプロセス 1 では、以下の処理を行う。

(a) “pcpnatpmpc -d -f -s PCPSERVER -l LIFETIME -i INT-PORT -e 0:EXTPORT -u” を実行し、ホーム NAT に PCP でポートマッピングを要求する。ここで、PCPSERVER は PCP サーバ (ホーム NAT) の IP アドレスを、LIFETIME にはポートマッピ

ングの持続時間を、INTPORT は割り当てる内部のポート番号を、EXTPORT は割り当てる外部のポート番号をそれぞれ指定する。本実装では INTPORT に、“socat” コマンドのパラメータ PORT と同じ値 (5683) を記述する。また EXTPORT には、サービスポート以外でランダムに生成したポート番号を記述する。なお、LIFETIME は遠隔制御対応プログラムの実行時引数として任意の値を指定する。

(b) ポートマッピング終了後、マッピングしたポート番号を SRV RR に格納する。

(c) DNS サーバへの動的更新を行うための更新メッセージを作成する。更新メッセージには、登録先ドメインに存在する既存 SRV RR の削除、および登録先ドメインへの新規 SRV RR の追加を指定する。

(d) 更新メッセージを TSIG または SIG(0) で署名し、外部 DNS サーバに対して TLS 上で送信する。なお TLS ハンドシェイクにおいて、DNS サーバから受け取る証明書単体を Go 言語の標準パッケージ x509 [18] の VerifyPeer 関数で検証する。

(e) LIFETIME 秒経過後、手順 3-a に戻る。これにより、ポートマッピングの期限切れ時に自動更新を行う。

4.5 制御データ送信プログラムの実装

クライアント上の制御データ送信プログラムの実装について説明する。このプログラムは、キャッシュサーバへの名前解決、外部 DNS サーバへのポートマッピング情報を格納した SRV RR の取得、IoT 機器への制御データの登録をそれぞれ行う。DNS 関連の処理の実装には Go 言語の dns パッケージを、OSCORE を伴う CoAP 関連の処理の実装には Python の aiocoap パッケージ [19] をそれぞれ使用する。制御データ送信プログラムを図 3 に示す。このプログラムの具体的な処理の流れは下記の通りである。

(1) キャッシュサーバに対し、外部 DNS サーバの A RR を問い合わせる名前解決を行う。

(2) 外部 DNS サーバに対して、所望のホームネットワークの SRV RR を問い合わせる。そして、SRV RR と返答に付属する A RR から、ホーム NAT に割り当てられたポートマッピング情報を得る。A RR が付属していない場合、SRV RR に記載されたターゲット (サービスを提供するホストのドメイン名) に対応する A RR について、外部 DNS サーバへ再度問い合わせを行う。これらの問い合わせにおいて、外部 DNS サーバとの通信には DoT を使用し、TSIG または SIG(0) による認証も行う。さらに TLS ハンドシェイクの過程で、x509 パッケージの VerifyPeer 関数を用いて証明書単体の検証を行う。

(3) 各ポートマッピングの候補先に対して順に、Python プログラムを呼び出す。正常に制御データを送信できた場合はメインプロセスを終了し、送信できなかった場合は次の候補のポートマッピングを試す。

(a) Python プログラムにおいて、OSCORE の鍵情報や設定情報を読み込む。

(b) ポートマッピングされた宛先に、OSCORE を適用した CoAP メッセージを送信する。これにより、ホーム NAT を介して IoT 機器 (CoAP サーバ上のリソース) に制御データが送

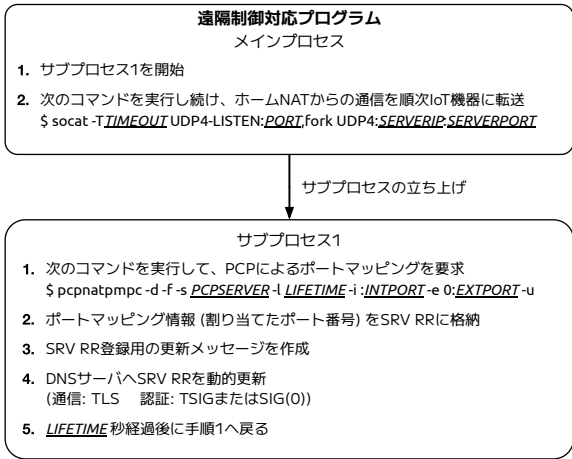


図2 遠隔制御対応プログラムの処理の流れ

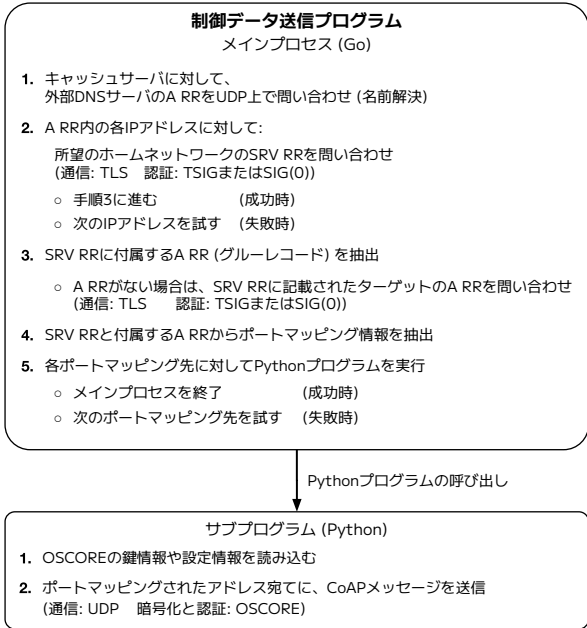


図3 制御データ送信プログラムの処理の流れ

られる。なお、本実装では制御データとして、PUT メソッドで現在日時のテキスト情報を登録する。

5. IoT 遠隔制御システムの機能評価

提案システムの評価を行うための実験環境として、図1に示すクライアント、ホームNAT、外部DNSサーバ、IoT機器、IoTゲートウェイの他、名前解決用として外部DNSサーバの上位の権威DNSサーバとなるルートDNSサーバおよびトップレベルドメイン (TLD) サーバ、さらにキャッシュサーバを、VirtualBoxを用いたホストOSの仮想環境上に構築した。本節では提案システムが設計通りに動作することの確認を行うため、構築した実験環境を用いて機能評価を実施する。

5.1 ポートマッピング情報のセキュアな通知と取得

IoTゲートウェイによる外部DNSサーバへのセキュアなポートマッピング情報の登録と、クライアントによる当該情報のセキュアな取得が有効に動作することを確認する。

まず、遠隔制御対応プログラムおよび制御データ送信プログラムを実行したときの、外部DNSサーバへの登録または問

No.	Time	Source	Destination	Source Port	Dest Port	Protocol	Length	Info
3	0.013078	192.168.1.100	192.168.1.254	39640	853	TCP	74	39640 → 853 [SYN] Seq=0 Min=4248 Len=0 MSS=1460 SACK
4	0.014153	192.168.1.254	192.168.1.100	853	39640	TCP	74	853 → 39640 [SYN, ACK] Seq=0 Ack=1 Min=65232 Len=0 MSS=
5	0.014185	192.168.1.100	192.168.1.254	39640	853	TCP	66	39640 → 853 [ACK] Seq=1 Ack=1 Min=64256 Len=0 TSval=3
6	0.017191	192.168.1.100	192.168.1.254	39640	853	TLSv1.3	1529	Client Hello
7	0.018064	192.168.1.254	192.168.1.100	853	39640	TCP	66	853 → 39640 [ACK] Seq=1 Ack=1 Min=63872 Len=0 TSva
8	0.132767	192.168.1.254	192.168.1.100	853	39640	TLSv1.3	1399	Server Hello, Change Cipher Spec, Application Data, A
9	0.132825	192.168.1.100	192.168.1.254	39640	853	TCP	66	39640 → 853 [ACK] Seq=1464 Ack=1334 Min=62976 Len=0 T
10	0.134891	192.168.1.100	192.168.1.254	39640	853	TLSv1.3	138	Change Cipher Spec, Application Data
11	0.134754	192.168.1.254	192.168.1.100	853	39640	TCP	66	853 → 39640 [ACK] Seq=1334 Ack=1528 Min=63872 Len=0 T
12	0.137189	192.168.1.100	192.168.1.254	39640	853	TLSv1.3	317	Application Data
13	0.138668	192.168.1.254	192.168.1.100	853	39640	TCP	66	853 → 39640 [ACK] Seq=1334 Ack=1779 Min=63744 Len=0 T
14	0.149792	192.168.1.254	192.168.1.100	853	39640	TLSv1.3	576	Application Data, Application Data
15	0.187698	192.168.1.254	192.168.1.100	853	39640	TLSv1.3	208	Application Data
16	0.187744	192.168.1.100	192.168.1.254	39640	853	TCP	66	39640 → 853 [ACK] Seq=1779 Ack=1986 Min=62336 Len=0 T
17	0.189888	192.168.1.100	192.168.1.254	39640	853	TLSv1.3	90	Application Data
18	0.198486	192.168.1.254	192.168.1.100	853	39640	TLSv1.3	90	Application Data
19	0.198486	192.168.1.254	192.168.1.100	853	39640	TCP	66	853 → 39640 [FIN, ACK] Seq=2010 Ack=1803 Min=63744 L
20	0.191365	192.168.1.100	192.168.1.254	39640	853	TCP	66	39640 → 853 [FIN, ACK] Seq=1983 Ack=2011 Min=62336 L
21	0.193897	192.168.1.254	192.168.1.100	853	39640	TCP	66	853 → 39640 [ACK] Seq=2011 Ack=1804 Min=63744 Len=0 T

図4 SRV RR 登録時のパケット (IoTゲートウェイと外部DNSサーバ)

No.	Time	Source	Destination	Source Port	Dest Port	Protocol	Length	Info
1	0.008808	172.16.0.1	172.16.0.100	50709	853	TCP	66	50709 → 853 [SYN] Seq=0 Min=65535 Len=0 MSS=1460 WS=2
2	0.009241	172.16.0.100	172.16.0.1	853	50709	TCP	66	853 → 50709 [SYN, ACK] Seq=0 Ack=1 Min=65232 Len=0 MSS=1
3	0.001818	172.16.0.1	172.16.0.100	50709	853	TCP	66	50709 → 853 [ACK] Seq=1 Ack=1 Min=65280 Len=0
4	0.001943	172.16.0.1	172.16.0.100	50709	853	TLSv1.3	1517	Client Hello
5	0.001986	172.16.0.100	172.16.0.1	853	50709	TCP	64	853 → 50709 [ACK] Seq=1 Ack=1464 Min=63232 Len=0
6	0.007827	172.16.0.100	172.16.0.1	853	50709	TLSv1.3	1387	Server Hello, Change Cipher Spec, Application Data, Appl
7	0.009261	172.16.0.1	172.16.0.100	50709	853	TCP	66	50709 → 853 [ACK] Seq=1464 Ack=1334 Min=65280 Len=0
8	0.009799	172.16.0.1	172.16.0.100	50709	853	TLSv1.3	118	Change Cipher Spec, Application Data
9	0.009799	172.16.0.1	172.16.0.100	50709	853	TLSv1.3	214	Application Data
10	0.011489	172.16.0.100	172.16.0.1	853	50709	TLSv1.3	564	Application Data, Application Data
11	0.012735	172.16.0.100	172.16.0.1	853	50709	TLSv1.3	320	Application Data
12	0.013519	172.16.0.1	172.16.0.100	50709	853	TCP	66	50709 → 853 [ACK] Seq=1608 Ack=2110 Min=64512 Len=0
13	0.015520	172.16.0.1	172.16.0.100	50709	853	TLSv1.3	78	Application Data
14	0.014419	172.16.0.100	172.16.0.1	853	50709	TLSv1.3	78	Application Data
15	0.015148	172.16.0.1	172.16.0.100	50709	853	TCP	66	50709 → 853 [FIN, ACK] Seq=1712 Ack=2134 Min=64512 Len=0
16	0.016438	172.16.0.100	172.16.0.1	853	50709	TCP	64	853 → 50709 [FIN, ACK] Seq=2134 Ack=1713 Min=63104 Len=0
17	0.017300	172.16.0.1	172.16.0.100	50709	853	TCP	66	50709 → 853 [ACK] Seq=1713 Ack=2135 Min=64512 Len=0

図5 SRV RR 取得時のパケット (クライアントと外部DNSサーバ)

い合わせのパケットをキャプチャし、Wiresharkで可視化した。それらの結果について、前者を図4に、後者を図5にそれぞれ示す。これらのキャプチャ結果から、SRV RRの登録と取得のどちらにおいても、SRV RRのペイロードと登録先ドメインがDoTによって秘匿化されていることを確認した。

次に、偽のTSIG共通鍵またはSIG(0)秘密鍵を用いて遠隔制御対応プログラムを実行した。どちらの場合においても、プログラムの署名処理の実行にエラーが生じ、外部DNSサーバにはSRV RRが登録されなかった。同様に、偽の鍵を用いて制御データ送信プログラムを実行した場合でも、実行中にエラーとなりSRV RRの取得が行われなかった。

以上の結果から、外部DNSサーバへのポートマッピング情報の登録時や問い合わせ時における、通信の暗号化と認証が有効であることを確認し、ポートマッピング情報のセキュアな通知や取得を実現できていることが示された。

5.2 制御データのセキュアな送信

OSCOREによる制御データのエンドツーエンド暗号化と通信の秘匿化から、制御データのセキュアな送信が実現されていることを確認する。

制御データ送信プログラムを実行し、そのときに送られた制御データの packets をIoTゲートウェイ上でキャプチャした。その結果を図6に示す。図6において、クライアント(172.16.0.1)の packets がホームNATを介してIoTゲートウェイ(192.168.1.100)に届き、さらにIoT機器(192.168.1.2)へと転送されている。これらの通信において、遠隔制御に関するデータはACKの返答以外の packets に格納されているが、Wireshark上では“can't decrypt”と表示され、リソース名やペイロードを解読することはできない。さらに、クライアントから送信されるCoAPメッセージのメソッドが全てPOSTに偽装されている。これはOSCOREの仕様によるものであり、要求時における外部コードには、原則としてPOSTが設定される[9]。そのため、クライアントがどのような操作を行ったかなどについて、通信上から解読することはできない。

以上の結果から、中継機器上においても制御データの暗号

No.	Time	Source	Destination	Source Port	Dest Port	Protocol	Length	Info
5	1.787341	172.16.0.1	192.168.1.100	61456	5683	OSCORE	93	CON, MID:33118, POST, TKI:52 5d
6	1.793475	192.168.1.100	192.168.1.2	49734	5683	OSCORE	93	CON, MID:33118, POST, TKI:52 5d
7	1.796943	192.168.1.2	192.168.1.100	5683	49734	CoAP	60	ACK, MID:33118, Empty Message
8	1.797907	192.168.1.2	192.168.1.100	5683	49734	OSCORE	71	CON, MID:9845, 2.04 Changed, TKI:52 5d
9	1.798276	192.168.1.100	172.16.0.1	5683	61456	CoAP	46	ACK, MID:33118, Empty Message
10	1.803332	192.168.1.100	172.16.0.1	5683	61456	OSCORE	71	CON, MID:9845, 2.04 Changed, TKI:52 5d
11	1.809149	172.16.0.1	192.168.1.100	61456	5683	CoAP	60	ACK, MID:9845, Empty Message
12	1.815944	172.16.0.1	192.168.1.100	61456	5683	OSCORE	103	CON, MID:33111, POST, TKI:52 5e
13	1.817529	192.168.1.100	192.168.1.2	49734	5683	CoAP	46	ACK, MID:9845, Empty Message
14	1.820390	192.168.1.100	192.168.1.2	49734	5683	OSCORE	103	CON, MID:33111, POST, TKI:52 5e
15	1.824296	192.168.1.100	172.16.0.1	5683	49734	CoAP	60	ACK, MID:33111, Empty Message
16	1.824591	192.168.1.100	172.16.0.1	5683	61456	CoAP	46	ACK, MID:33111, Empty Message
17	1.833922	192.168.1.2	192.168.1.100	5683	49734	OSCORE	60	CON, MID:9846, 2.04 Changed, TKI:52 5e
18	1.837040	192.168.1.100	172.16.0.1	5683	61456	OSCORE	59	CON, MID:9846, 2.04 Changed, TKI:52 5e
19	1.877998	172.16.0.1	192.168.1.100	61456	5683	CoAP	60	ACK, MID:9846, Empty Message
20	1.877998	172.16.0.1	192.168.1.100	61456	5683	OSCORE	73	CON, MID:33112, POST, TKI:52 5f
21	1.879598	192.168.1.100	192.168.1.2	49734	5683	CoAP	46	ACK, MID:9846, Empty Message
22	1.880244	192.168.1.100	192.168.1.2	49734	5683	OSCORE	73	CON, MID:33112, POST, TKI:52 5f
23	1.892828	192.168.1.2	192.168.1.100	5683	49734	CoAP	60	ACK, MID:33112, Empty Message
24	1.892828	192.168.1.2	192.168.1.100	5683	49734	OSCORE	80	CON, MID:9847, 2.04 Changed, TKI:52 5f
25	1.893246	192.168.1.100	172.16.0.1	5683	61456	CoAP	46	ACK, MID:33112, Empty Message
26	1.895744	192.168.1.100	172.16.0.1	5683	61456	OSCORE	80	CON, MID:9847, 2.04 Changed, TKI:52 5f
27	1.900346	172.16.0.1	192.168.1.100	61456	5683	CoAP	60	ACK, MID:9847, Empty Message
28	1.900680	192.168.1.100	192.168.1.2	49734	5683	CoAP	46	ACK, MID:9847, Empty Message

図6 IoT ゲートウェイ上でキャプチャした制御データの packets

化が維持されることや、CoAP メッセージの秘匿化による通信の秘匿化が有効に機能していることを確認し、制御データのセキュアな送信が実現されていることが示された。

5.3 遠隔制御時の IoT 機器とクライアントの認証

OSCORE による遠隔制御時の IoT 機器とクライアントの認証が、それぞれ有効に動作することを確認する。

まず、鍵の一部を書き換えた偽の IoT 機器用 OSCORE 設定ファイルを用意し、当該設定ファイルで IoT 機器を立ち上げた。その後、正規の鍵を指定した制御データ送信プログラムを実行し、IoT 機器に制御データを送信した。すると “4.00 Bad Request” が返され、制御データの登録に失敗した。

次に、正規の鍵で IoT 機器を立ち上げた後、鍵の一部を書き換えた偽のクライアント用 OSCORE 設定ファイルを用意し、当該設定ファイルで制御データ送信プログラムを実行した。その結果、IoT 機器上で動作する coap-server のログに “OSCORE: Decryption Failure” が記録され、クライアントから送信された CoAP メッセージは受理されなかった。

以上より、正規の PSK 鍵を持たない不正な IoT 機器やクライアントによる遠隔制御の失敗が確認され、遠隔制御時の IoT 機器やクライアントの認証が有効に機能することが示された。

5.4 機能評価のまとめ

3.1 節の表 1 で示した通り、IoT 遠隔制御システムには 4 つのセキュリティ要件が存在する。5.3 節の評価で要件 (1) と要件 (2) が、5.2 節の評価で要件 (3) が、そして 5.1 節の評価で要件 (4) が満たされていることの確認ができた。これらの結果から、提案手法は 4 要件を全て充足可能なシステムと言える。

6. まとめと今後の課題

本研究では、NAT 環境下に存在する IoT 機器の遠隔制御を可能にしながら、IoT 機器への接続情報をセキュアかつスケーラブルに外部へ配信し、さらにセキュリティ対策とプライバシー保護も実現するような、IoT 遠隔制御システムを提案した。

提案システムでは DNS、PCP および OSCORE を組み合わせた設計を行うことで、NAT 環境下における柔軟なポートマッピングの実現や、アクセス情報のセキュアかつスケーラブルな配信、および各種セキュリティ機能の導入を実現した。また、提案システムの実装と機能評価を通じて、遠隔制御の実行や各種セキュリティ機能が有効に動作することを示した。

今後の課題として、評価面では、複数のクライアントが同時

に複数の IoT 機器を制御する場合や、実ネットワークにおけるパフォーマンスなどの各種性能評価の実施が挙げられる。

また設計面では、IoT ゲートウェイによる転送時のセキュリティリスクが挙げられる。転送処理におけるホームネットワーク内部への侵入リスクを考慮すると、IoT ゲートウェイ上においても追加のセキュリティ対策を講じることが望ましい。

さらに実装面では、ポートマッピングの更新方法の拡充が挙げられる。本論文の実装では、単一の有効期限のみに起因するポートマッピングの失効や更新を行っている。セキュリティリスクの低減や実運用性の向上を実現するために、例えばクライアントからの明示的な指示を受けてポートマッピングの更新や有効期限の調整を行うなど、より柔軟なアクセス制御が求められる。

文 献

- [1] H. Tanaka, H. Suzuki, A. Watanabe, and K. Naito, “Evaluation of a secure end-to-end remote control system for smart home appliances,” Proc. IEEE Int’l Conf. Consumer Electronics (ICCE), pp.1–2, Jan. 2018.
- [2] H. Lu, H. Wu, and R. Jing, “Evaluation of internet of things computer network security and remote control technology,” Open Computer Science, vol.14, no.1, p.20240004, July 2024.
- [3] Z. Shelby, K. Hartke, and C. Bormann, “The Constrained Application Protocol (CoAP),” IETF RFC 7252, June 2014. DOI: 10.17487/RFC7252
- [4] D. Wing, S. Cheshire, M. Boucadair, R. Penno, and P. Selkirk, “Port Control Protocol (PCP),” IETF RFC 6887, April 2013. DOI: 10.17487/RFC6887
- [5] O. Novo, “Making constrained things reachable: A secure IP-Agnostic NAT traversal approach for IoT,” ACM Transactions on Internet Technology, vol.19, no.1, pp.1–21, Oct. 2018.
- [6] C. Amsuss, Z. Shelby, M. Koster, C. Bormann, and P. Van der Stok, “Constrained RESTful Environments (CoRE) Resource Directory,” IETF RFC 9176, April 2022. DOI: 10.17487/RFC9176
- [7] 相良隼, 金勇, 飯田勝吉, 高井昌彰, “DoT を用いたプライバシー配慮型 IoT データ照会システムの検討,” 電子情報通信学会・技術研究報告, vol.123, no.85, pp.56–61, June 2023.
- [8] H. Sagara, Y. Jin, K. Iida, and Y. Takai, “Privacy preserved IoT data inquiry system using DoT,” Proc. IEEE Conf. Cloud & Internet of Things (CIoT), pp.1–6, Nov. 2024.
- [9] G. Selander, J.P. Mattsson, F. Palombini, and L. Seitz, “Object Security for Constrained RESTful Environments (OSCORE),” IETF RFC 8613, July 2019. DOI : 10.17487/RFC8613
- [10] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P.E. Hoffman, “Specification for DNS over Transport Layer Security (TLS),” IETF RFC 7858, May 2016. DOI: 10.17487/RFC7858
- [11] Internet Systems Consortium, “BIND 9”. <https://www.isc.org/bind/> (accessed 2026-01-27).
- [12] O. Bergmann, “libcoap”. <https://libcoap.net/> (accessed 2026-01-27).
- [13] Google, “The Go programming language”. <https://go.dev/> (accessed 2026-01-27).
- [14] Python Software Foundation, “Welcome to Python.org”. <https://www.python.org/> (accessed 2026-02-08).
- [15] T. Bernard, “MiniUPnP Project HomePage”. <http://miniupnp.free.fr/> (accessed 2026-01-29).
- [16] M. Gieben, “dns”. <https://pkg.go.dev/github.com/miekg/dns> (accessed 2026-01-27).
- [17] Cisco Systems, “libpcpnatpmp”. <https://github.com/libpcpnatpmp/libpcpnatpmp> (accessed 2026-01-29).
- [18] Google, “x509”. <https://pkg.go.dev/crypto/x509> (accessed 2026-01-27).
- [19] Christian Amsuss, “aiocoap – The Python CoAP library”. <https://aiocoap.readthedocs.io/en/0.4.17/> (accessed 2026-01-29).