IoTデバイスのための柔軟なホスト名自動生成の提案

柳瀬 知広[1]　田中 久順[1]　鈴木 秀和[1]

概要：近年，IP を利用して通信を行う IoT（Internet of Things）デバイスが普及し始めている．今後，IPv6 ネットワークが普及することを想定すると，遠隔地から様々な通信規格の IoT デバイスに対して通信するために，FQDN（Fully Qualified Domain Name）の設定が必要不可欠になると考えられる．しかし，一般的に既存の IoT デバイスは FQDN を動的に設定する機能を有しておらず，ユーザ自身でデバイスの FQDN を設定する必要がある．本稿では，既存の通信プロトコルを利用して IoT デバイスから得られる情報に基づいて FQDN を自動生成する FNAC（Flexible Name Autoconfiguration）を提案する．提案手法では，ホームゲートウェイに IoT デバイスのホスト名自動生成機能を搭載することで，既存の IoT デバイスの仕様を変更することなく，FQDN の自動生成を実現する．FNAC のプロトタイプシステムを開発し，IoT デバイスの発見から FQDN の登録までに発生する処理時間を計測した．その結果，デバイス探索終了後，即座に自動生成した FQDN を DNS サーバに登録できることを考察した．

A Proposal of Flexible Name Autoconfiguration for IoT Devices

TOMOHIRO YANASE[1]　HISAYOSHI TANAKA[1]　HIDEKAZU SUZUKI[1]

1. はじめに

インターネット技術や各種センサー・テクノロジーの進化等を背景に，パソコンやスマートフォンなど従来のインターネット接続端末に加え，家電や自動車，ビルや工場など，世界中の様々なモノがインターネットへつながり，先進的なサービスを可能とする IoT（Internet of Things）が世界規模で注目を集めており，そのデバイスの数は爆発的に増加している [1]．世界経済を牽引する主要な産業や市場に対して重要な情報や分析を提供するイギリスの企業，IHS Markit によると，インターネットにつながるモノ，IoT デバイスの世界で普及している数は 2015 年時で 154 億台であり，2025 年には 754 億台に増加すると予測されている [2]．

IoT サービスを実現するためには，現実世界をセンシングし，あらゆるデータをサーバー空間へ送信，蓄積し，AI 技術によりデータを分析する必要がある．このような IoT サービスを構成するセンサノードやウェアラブルデバイスなどの IoT デバイスは，今後インターネットに直接接続することも考えられる．これらの IoT デバイスには，通信を行うために IP アドレスが割り当てられるが，現在主流である IPv4 グローバルアドレスは枯渇状況にあるため，普及する IoT デバイスに対応する対策として IoT 社会では，無尽蔵なアドレス空間を有する IPv6 アドレスを用いることが不可避となる [3]．IPv6 アドレスは従来の IPv4 アドレスよりアドレスサイズが大きく，アドレスの表記も IPv4 アドレスと比較して 16 進数で桁数が多い．そのため，ユーザは IPv6 アドレスを利用する際には FQDN（Fully Qualified Domain Name）のようなユーザフレンドリな識別子を利用して通信や管理を行うことが一般的である．

現在，普及している多くの IoT デバイスは一般的に FQDN を動的に設定する機能を有していないため，ユーザ自身で IoT デバイスの FQDN を一つ一つ設定して DNS サーバへ登録する必要がある．そのため，今後，IoT デバイスの普及が進み，宅内に多くの IoT デバイスが設置されることを考えると，ユーザ自身で FQDN を設定する行為は非常に煩わしいものになる．

従来の IoT デバイスの FQDN 自動生成手法に Bonjour [4] が挙げられるが，この技術はローカルネットワーク内でしか利用できない．そこで，Jeong らはグローバルネット

[1] 名城大学大学院理工学研究科
Graduate School of Science and Technology, Meijo University

ワークでも利用することが可能な IPv6 アドレスが割り当てられた IoT デバイスに対して FQDN を自動で設定する DNSNA（DNS Name Autoconfiguration）[5] を提案している．しかし，この手法では IoT デバイスに独自のプロトコルを実装する必要があり，様々な通信プロトコルが存在する IoT デバイスの現在を鑑みると，プロトコルの標準化や業界の統一基準などを取り決める必要があるなど，実現までに大きなハードルが存在する．

本稿では，IoT デバイスの仕様を変えず，IoT デバイスから得られる情報を基に FQDN を自動生成し，かつ部屋などの場所に関する情報などを付与してユーザが認識しやすい形式に柔軟に変更可能な FNAC（Flexible Name AutoConfiguration）を提案する．FNAC では，ホームゲートウェイ（HGW）に IoT デバイスのホスト名自動生成機能を搭載することにより，IoT デバイスの FQDN を自動生成することを実現する．FNAC のプロトタイプシステムを実装し，性能評価を行った．その結果，HGW が IoT デバイスを探索すると即座に IoT デバイスの FQDN を自動生成して DNS サーバに登録できることを確認した．

以下，2 章で既存研究とその課題，3 章で提案方式について述べる．4 章で実装および動作検証，5 章で提案方式の評価結果について述べ，6 章でまとめる．

2. 既存研究

2.1 Bonjour

2.1.1 概要

パソコンやプリンタなどのデバイスを自動的にネットワークに接続することを可能とする一連の技法として，Zeroconf（Zero Configuration Networking）[6,7] がある．Bonjour は Apple が開発した Zeroconf 技術であり，ユーザが機器をネットワークに接続するだけで IP アドレスとホスト名の割り当てやホスト名の探索を自動で行うプロトコルである．本来，これらの機能を利用するためには DHCP サーバや DNS サーバが必要になるが，Bonjour ではこれらのサーバを必要としない．

2.1.2 ホスト名の構成

Bonjour におけるホスト名の構成は “machine_name.service_type.protocol.domain_name” に従って生成される．“machine_name” はユーザがデバイスに対して設定した名前を表す識別子である．“service_type” はデバイスが提供するサービス名を表す識別子である．例えば，HTTP サービスである場合は “._http”，音楽共有アプリケーションである場合は “._music” となる．“protocol” はデバイスが利用するプロトコル名を表す識別子であり，TCP/IP の場合は “._tcp”，UDP の場合は “._udp” となる．“domain_name” はローカルネットワークのドメイン名を表す識別子であり，“.local” が組み込まれる．

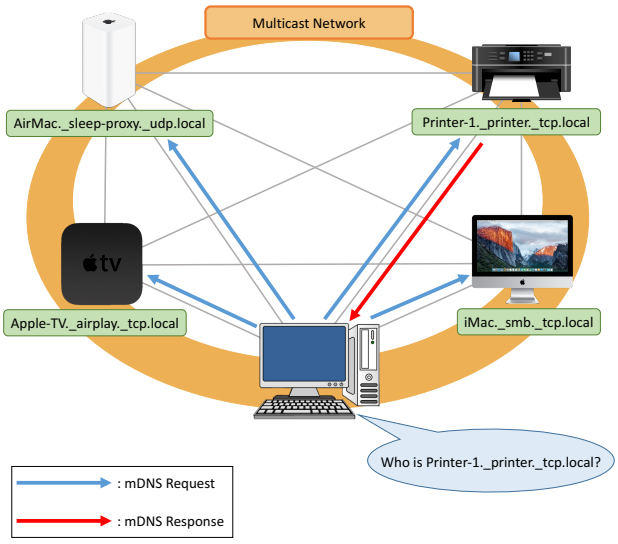

図 1 Zeroconf ネットワークの概要

2.1.3 Bonjour における名前解決法

図 1 に Zeroconf ネットワークの概要を示す．IP アドレスの自動割り当てには，IPv4 アドレスならば APIPA（Auto Private IP Addressing）[8] の技術が使われ，169.254.0.0/16 の中から空いているアドレスを自ら割り当てる．また，IPv6 アドレスならば LAN インタフェースに自動で割り当てられるリンクローカルアドレスを利用する．

ホスト名の探索には mDNS（Multicast DNS）[9] が用いられる．LAN 上の各デバイスはそれぞれ自身の A レコードや MX レコードなどの DNS リソースレコードを有しており，mDNS のマルチキャストグループに参加している．ある mDNS クライアントがデバイスのホスト名からその IP アドレスを知りたい場合，mDNS クライアントは規定のマルチキャストアドレスを利用して名前解決要求を LAN 上のデバイスにマルチキャスト送信する．該当するホスト名を有するデバイスはその IP アドレスを含めて応答することにより DNS サーバを使わずに名前解決を行うことができる．

2.1.4 課題

Bonjour ではローカルネットワーク内においてデバイスの IP アドレスとホスト名を生成し，Bonjour の機能を搭載したデバイス同士で名前解決を行うことによりデバイス間の通信を可能にしている．しかし，この技術はローカルネットワーク内だけでしか利用できないため，グローバルネットワークから FQDN を指定してデバイスを制御できない．また，デバイスが Bonjour の機能を搭載していないと FQDN を自動生成できないため，利用できるデバイスに制約がある．

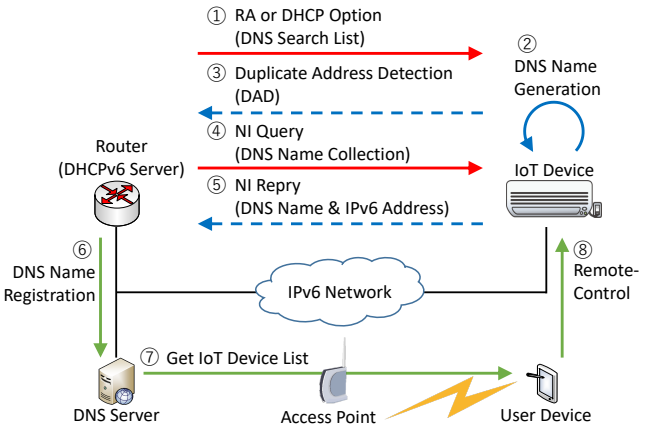

図 2　DNSNA のシステム構成

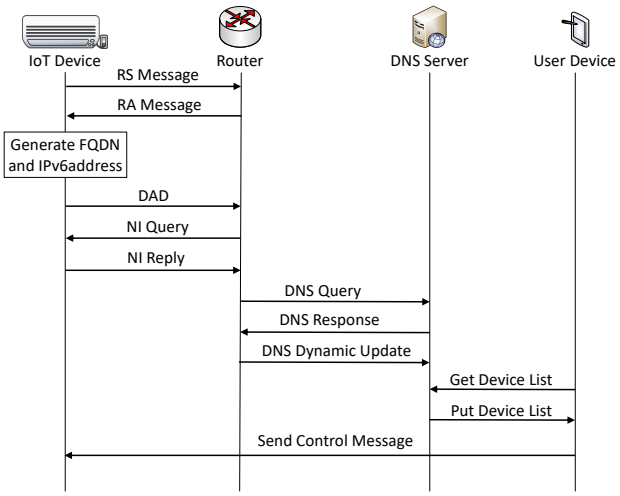

図 3　DNSNA における FQDN の生成および DNS 登録シーケンス

2.2　DNSNA

2.2.1　概要

DNSNA は，グローバルネットワークまたはローカルネットワークに存在する IoT デバイスの FQDN を動的に生成し，DNS サーバに登録するシステムである．図 2 に IPv6 ネットワークにおける DNSNA システムの構成を示す．DNSNA は IoT デバイスに FQDN を自動で生成する機能を搭載する．IoT デバイスはモデル名や製造メーカを表す機器情報とルータから得られるネットワークのドメイン情報を組み合わせることにより，自身の FQDN を生成する．生成された FQDN は IoT デバイスの IPv6 アドレスと共に AAAA レコードとして DNS サーバに登録されるため，ユーザは FQDN を用いて宅外から IoT デバイスにアクセスすることが可能となる．

2.2.2　FQDN の構成

DNSNA における FQDN の構成は “unique_id.object_identifier.OID.domain_name” となる．“unique_id” は FQDN の一意性を保証するための識別子で，製品名の末尾にシーケンス番号を加えたものなどが使われる．“object_identifier” はデバイスの機器情報を表す識別子で，製造業社 ID，デバイスのモデル ID，シリアル ID，拡張 ID の 4 つの内容を組み合わせて構成される．“OID” は “object_identifier” が利用されていることを表す識別子である．“domain_name” は IoT デバイスが存在するネットワークのドメインを表す識別子である．

また，FQDN に位置情報を含めることも可能となっており，その場合の名前は “unique_id.object_identifier.OID.mic_loc.mac_loc.LOC.domain_name” となる．新たに追加された “mic_loc” はデバイスが存在する場所の詳細な箇所を示す識別子で，部屋の中央や淵といった内容が入る．“mac_loc” はデバイスが存在する場所を表す識別子で，キッチンやリビングルームといった部屋の名称であったり，道路であれば交差点などの内容が入る．“LOC” は “mic_loc” と “mac_loc” が利用されていることを表す識別子である．

2.2.3　FQDN の自動生成と登録

図 3 に DNSNA における FQDN の生成，DNS サーバへの登録およびユーザが IoT デバイスの FQDN を取得するまでのシーケンスを示す．なお，このシーケンスは IoT デバイスをネットワークに接続した直後から始まる．まず，IoT デバイスはネットワークに接続すると自身の IPv6 リンクローカルアドレスを生成し，ローカルリンク上に存在するルータに RS（Router Solicitation）メッセージを送信する．RS メッセージを受信したルータは，RA（Router Advertisement）メッセージをブロードキャストする．RA メッセージにはネットワークのドメイン情報が含まれているため，IoT デバイスは受け取った RA メッセージより IPv6 アドレスと FQDN を自動生成する．その後，IoT デバイスは自動生成した IPv6 アドレスがネットワーク上で一意か確認するためにルータに対して DAD（Duplicate Address Detection）[10] を実施する．

IPv6 アドレスの一意性が保証された後，IoT デバイスの FQDN を DNS サーバに登録するために，ルータは IoT デバイスに対して NI Query（Node Information Query）[11] を送信し，IoT デバイスの FQDN を問い合わせる．NI Query を受け取った IoT デバイスは，自動生成した自身の FQDN と IPv6 アドレスのペア情報が含まれた NI Reply（Node Information Reply）をルータに返信する．IoT デバイスの FQDN と IPv6 アドレスの情報を入手したルータは IoT デバイスが生成した FQDN の一意性を確認するために，DNS サーバに対して DNS Query を送信する．DNS サーバからの応答から FQDN が一意であることを確認できたら，DDNS（Dynamic Domain Name System）[12] の機能を利用し，IoT デバイスの FQDN と IPv6 アドレスのペアを AAAA レコードとして DNS サーバに登録する．

以上により，ユーザは FQDN を用いて IoT デバイスと

通信することができる．

2.2.4 課題

Bonjour がローカルネットワークでしか利用できないことに対し，DNSNA はグローバルネットワーク上の IoT デバイスに対しても FQDN の自動生成を行うことが可能である．しかし，この DNSNA には次に示す 2 つの課題が存在する．

第一に，DNSNA を利用するためには IoT デバイスとルータに機能追加が必要になることである．そのため，既存の IoT デバイスでは DNSNA を利用することができない．また，ルータにも NI Query を利用するために DHCPv6 サーバを兼ねたり，DNS サーバに Dynamic Update を行う機能を追加したりする必要がある．通信プロトコルが統一されていない現状の IoT デバイスを鑑みると，ユーザが気軽に DNNSA を利用することは不可能で，標準化などの作業が必須になると考えられる．

第二に，ユーザにわかりにくい FQDN を変更することが難しい．FQDN の構成にはデバイスのシリアル番号などのユーザが通常認識しない情報が含まれているため，ユーザフレンドリでない．仮にユーザが認識しやすい FQDN に変更したとしても，IoT デバイスは DNSNA で定められている手続きに従って所定の FQDN を自動生成してしまう．そのため，ルータが再度自動生成された FQDN で DNS サーバの AAAA レコードを上書きして更新してしまうため，ユーザが設定した FQDN は解決できなくなってしまう．

3. 提案方式

3.1 概要

2 章で示した既存技術の課題を解決するために，本稿では IoT デバイスの仕様を変えず，デバイスから得られる情報を基に FQDN を自動生成し，かつ自動生成後の FQDN をユーザが認識しやすい形式に柔軟に変更可能な FNAC（Flexible Name Autoconfiguration）を提案する．**図 4** に提案方式におけるシステム構成と仕組みを示す．ルータ配下のホームネットワークに HGW および各通信規格の IoT デバイス，グローバルネットワーク上に DNS サーバを設置する．DNSNA では IoT デバイスに搭載させていた FQDN 自動生成機能を，提案方式では HGW にその機能を搭載する点が異なる．HGW は既存の IoT デバイスで使用されている DLNA（Digital Living Network Alliance）[13] や ECHONET Lite [14]，HomeKit [15] などで定義されている機器探索メッセージ等を用いて IoT デバイスの機器情報を取得する．取得できた機器情報に基づいて，ユーザが認識しやすい形式をした IoT デバイスの FQDN を自動生成し，DNS サーバへ動的に登録する．この方式では自動生成された IoT デバイスの FQDN をユーザ自身で変更可能にすることで，より認識しやすい FQDN を設定することが可能である．

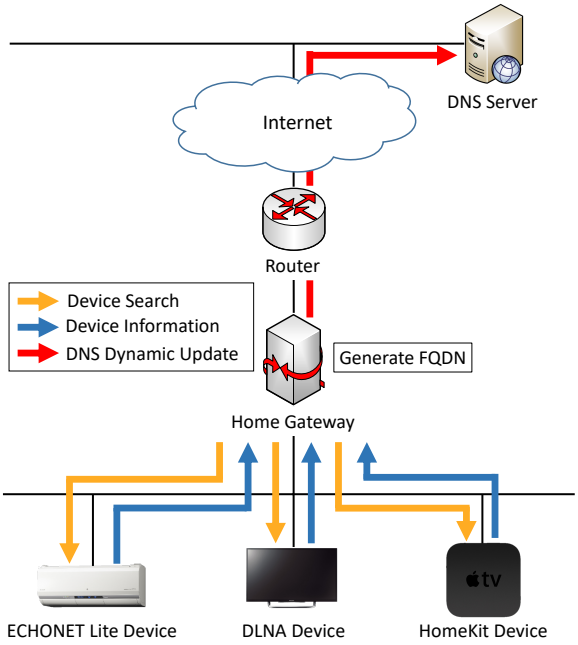

図 4 提案方式におけるシステム構成と仕組み

3.2 FQDN の構成

FNAC における FQDN の構成は “unique_id.maker_name.room_name.home_domain” のように定義する．“unique_id” は，FQDN の一意性を保証するための識別子で，デバイスの型番やカテゴリ名に加えてシリアル番号の一部や通し番号などが付与される．“maker_name” はデバイスの製造メーカの名称を表す識別子である．“room_name” はデバイスが設置されている部屋の名称を表す識別子である．“home_domain” はデバイスが存在するホームネットワークのドメインを表す．

FNAC の FQDN における “unique_id” と “home_domain” は必須の識別子であが，“maker_name” と “room_name” はプロトコルによっては IoT デバイスから取得できない場合があるため省略することが可能である．

3.3 FQDN の自動生成と登録

図 5 に IoT デバイスの FQDN を生成し，DNS サーバに登録するまでの流れを示す．なお，IoT デバイスはホームネットワークに接続した際，IPv6 アドレスを自動生成しているものとする．また，FNAC の機能を搭載した HGW にはホームネットワークのドメイン名 “myhome” が設定されているものとする．

HGW は IoT デバイスの情報を取得するために，DLNA や ECHONET Lite，HomeKit などで定義されている機器探索パケットをマルチキャストする．機器探索パケットを受信した IoT デバイスは，型番や製造メーカ名などの機器情報を含んだ応答メッセージを返信する．HGW は受信し

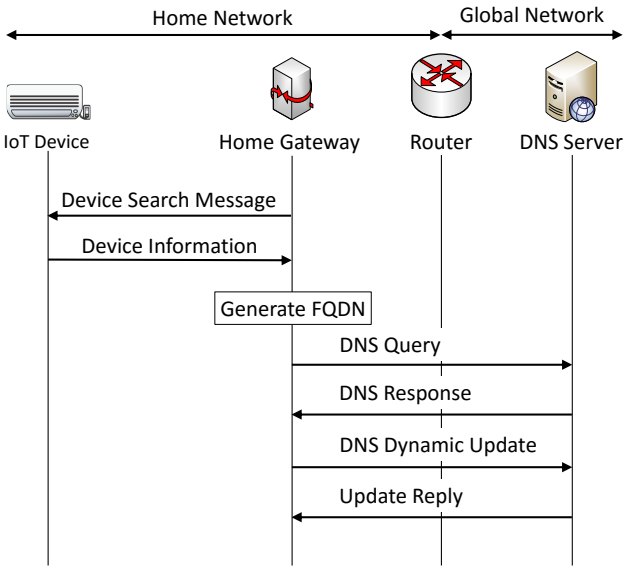

図 5　提案方式における FQDN の自動生成および DNS 登録シーケンス

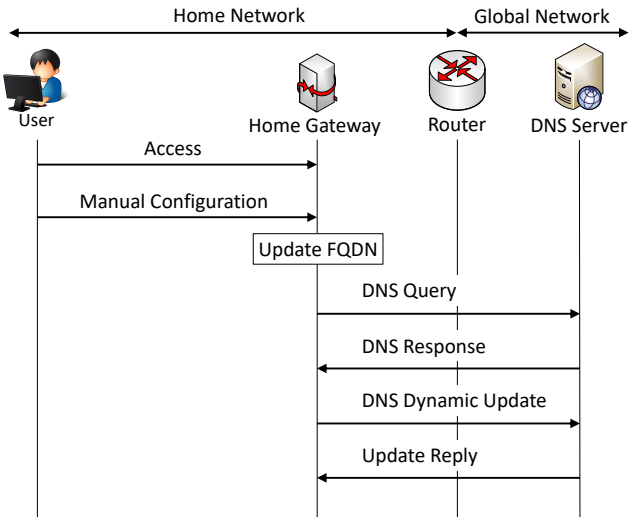

図 6　提案方式における自動生成後の FQDN 変更シーケンス

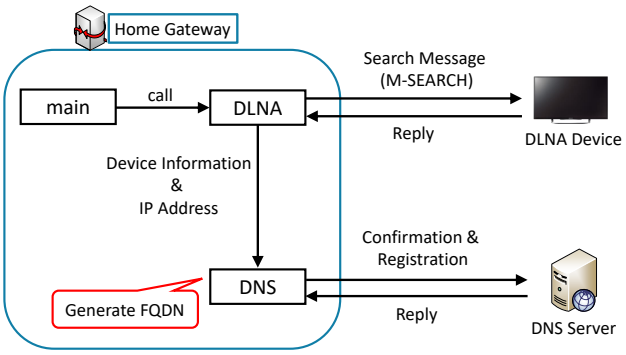

図 7　プロトタイプのモジュール構成図

た機器情報を基に，FQDN を自動生成する．例えば，東芝製エアコンであれば “airconditioner-1.toshiba.myhome” のようになる．

自動生成した FQDN がインターネット上で一意であるかを確認するために，HGW は DNS サーバに生成した FQDN の AAAA レコードを問い合わせる．DNS サーバからの応答で IPv6 アドレスが取得できなかった場合は自動生成した FQDN に重複がないということが確認できるため，IoT デバイスの FQDN と IPv6 アドレスのペアを DDNS の機能を使って DNS サーバに登録する．なお，DNS サーバからの応答で IPv6 アドレスが取得できた場合は自動生成した FQDN に重複があるため，FQDN のドメインである “unique_id” の通し番号を変え，一意なものになるまで繰り返す．

以上の処理により，既存の IoT デバイスをホームネットワークに接続するだけで，初期 FQDN が自動生成され，DNS サーバに登録される．

3.4　自動生成された FQDN の変更

自動生成された IoT デバイスの FQDN をユーザが変更し，DNS サーバに再登録を行うまでの流れを図 6 に示す．ユーザは IoT デバイスの FQDN を変更するため，ブラウザ等を用いて HGW にアクセスする．HGW にはホームドメインの他，宅内の部屋の情報等を登録したり，各種通信プロトコルで取得した IoT デバイスの情報を確認することが可能で，IoT デバイスの FQDN と IoT デバイスが設置されている部屋を関連付ける．その結果，設定した部屋情報を反映した FQDN に自動更新することが可能である．

また，HGW には DNS サーバに登録されている IoT デバイスの FQDN の一覧が記録されているため，設定変更を行いたいデバイスの FQDN を選択し，ユーザ自身でホームネットワークドメイン以外の名前を任意で変更することが可能である．変更した FQDN に重複がなければ，HGW は変更後の FQDN を DNS サーバに再登録する．例えば，リビングルームにエアコンが一台だけ設置されており，その FQDN が “airconditioner-1.toshiba.myhome” とのように自動生成されていたとする．ユーザが HGW にこのエアコンがリビングルームに設置されていることを設定することにより，“airconditioner.livingroom.myhome” といった FQDN に変更することが可能である．

以上の通り，FNAC ではユーザが IoT デバイスを操作管理する際により認識しやすい名前へ変更することが可能である．

4.　実装・検証

4.1　実装

DLNA 家電機器を対象として，動作する FNAC のプロトタイプを C 言語を用いて実装した．図 7 にプロトタイプのモジュール構成図を示す．作成したプロトタイプには，各通信規格モジュールの処理を行う main モジュール，DLNA 機器からの探索パケットを送信する機能を有

した DLNA モジュール，応答によって入手可能な機器情報を基に FQDN を生成する機能，および DNS サーバに登録する機能を有する DNS モジュールを実装した．HGW の OS は Ubuntu 17.04 であり，DLNA 通信ライブラリに mUPnP [16] を，DNS Dynamic Update を利用するために libbind-6.0 [17] を使用した．

4.2 動作検証

図 8 に動作検証を行った環境を示す．本稿における動作検証では，DLNA 通信ライブラリに利用した mUPnP が IPv6 通信に対応していなかったため，IPv4 ネットワークでの検証を行った．大学研究室内の IPv4 プライベートネットワークには DLNA 家電機器であるソニー株式会社製のテレビ “BRAVIA KDL-32W700B” と QNAP 社製の NAS（Network Attached Strage）“TS-259 Pro Turbo NAS”，ホームネットワークのドメイン名 “myhome.ucl.meijo-u.ac.jp” を設定，および FNAC のプロトタイプを実装した仮想マシンを HGW と見立て設置した．DNS サーバはさくらレンタルサーバで契約している VPS（Virtual Private Server）を利用し，グローバルネットワーク上に設置した．DNS サーバの OS は Debian 7.2 であり，BIND 9 を利用した．また，DNS サーバは HGW からの DNS リソースレコードの動的更新要求を受け入れるよう設定した．

このような動作環境において，DLNA デバイスに対する M-SEARCH メッセージのマルチキャストや FQDN の自動生成および DNS サーバへの登録処理の様子を Wireshark を用いて確認した．M-SEARCH メッセージをマルチキャストした HGW は 2 種類の DLNA 対応機器から応答パケットを受信し，自動生成した FQDN の重複確認処理および DNS サーバへの登録処理を実行していることを確認した．なお，DLNA はデバイスが設置されている部屋情報を保持していないため，自動生成された FQDN は “BRAVIA-KDL-32W700B-1.Sony-Corporation.myhome.” と “QNAP-NAS-1.Microsoft.myhome.” のように “room_name” を省略した形式となっていた．

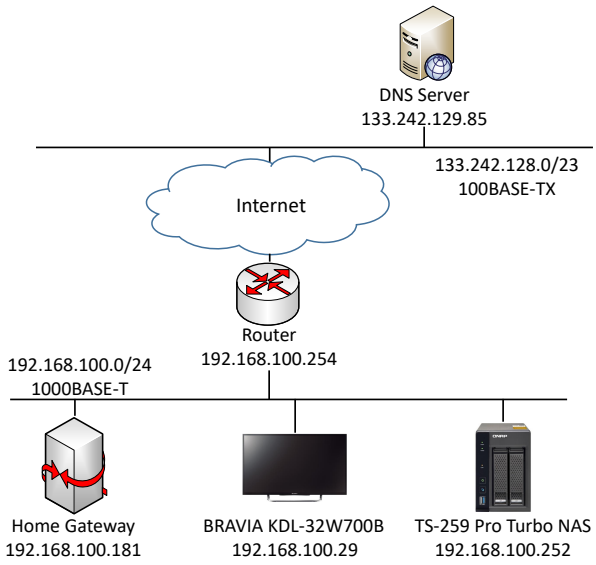

図 8 動作検証で使用した環境

表 1 既存研究との比較

	Bonjour	DNSNA	FNAC
グローバルネットワークでの名前解決	×	○	○
既存 IoT デバイスのサポート	×	×	○
IoT デバイスへの機能実装	必要	必要	不要
認識しやすい FQDN	△	△	○
FQDN の変更	△	×	○

5. 評価

5.1 既存研究との比較

表 1 に提案方式である FNAC と既存技術である Bonjour および DNSNA との比較を示す．2 章で述べた通り，Bonjour ではデバイスに制約があり，グローバルネットワークからの名前解決ができない．仮に動作検証で使用したテレビが Bonjour に対応していた場合，“BRAVIA-KDL-32W700B._upnp._udp.local.” のような名前が自動生成されるがユーザが通常意識しない情報が含まれており，ユーザはこの FQDN を認識しづらい．DNSNA ではグローバルネットワークからの IoT デバイスの制御を実現しているが，既存の IoT デバイスではこのシステムを利用できない．また，FQDN は “BRAVIA-KDL-32W700B1.0.2.481.1.100.3030.10011.OID.center.livingroom.LOC.myhome.” のような名前が自動生成されるがユーザに認識しづらく，かつ自動生成後の FQDN をユーザが変更することができない．

FNAC では，HGW が IoT デバイスから取得した機器情報を基に自動生成した FQDN を DNS サーバに登録することにより，どこからでも IoT デバイスの名前解決が可能で，FQDN を用いた IoT デバイスの制御が可能である．IoT デバイスへの実装が必要がないため，既設の IoT デバイスが設置されたネットワークであっても FNAC を導入することが可能である．また，FQDN の構成は短く，デバイスごとに対応した形式に変化し，自動生成された後の FQDN は HGW にアクセスすることで IoT デバイスへのアクセスに関するユーザ自身で変更することが可能である．例えばリビングに設置された TV であれば “tv.livingroom.myhome.” のような FQDN を設定できるため，既存技術で生成される FQDN と比較してユーザビリティが高いと考えられる．

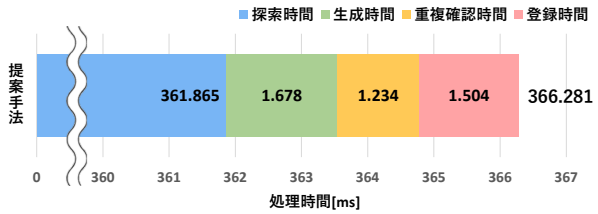

図 9　処理時間の測定結果

5.2　処理時間の検証

5.2.1　測定方法

提案システムの実用性を評価するため，図 8 と同じ構成を利用して本システムにおける DLNA 機器の探索時間，FQDN の生成時間，FQDN の重複確認時間，FQDN 登録時間を測定した．試行回数は 10 回で各処理時間の平均値を算出した．

5.2.2　評価結果

図 9 に算出したプログラムの各処理時間の平均値を示す．各処理時間は，DLNA 機器の探索時間が 361.865[ms]，FQDN の生成時間が 1.678[ms]，FQDN の重複確認時間が 1.234[ms]，FQDN の登録時間が 1.504[ms] であり，プログラム全体が終了するまでの平均時間は 366.281[ms] であった．今回は通信ライブラリの制約により IPv4 環境での動作検証および性能評価となったが，IPv6 環境と処理内容は同じであり，異なる点は IP アドレスの長さが異なること，DNS サーバへの問い合わせおよび登録処理で使用するレコードの種類が A と AAAA で違うことだけである．したがって，IPv6 環境で FNAC を適用した場合においても，図 9 と同程度の処理時間になると考えられる．

以上の結果より，提案システムを用いることで，IoT デバイスをホームネットワークに接続し，HGW が機器探索を行えば，IoT デバイスの FQDN は瞬時に自動的に生成され，DNS サーバに登録されるため，ユーザは即座に IoT デバイスの操作，管理に FQDN を利用することが可能である．

6.　まとめ

本稿では，IoT デバイスのための柔軟な名前自動生成システムとして FNAC を提案した．提案方式では，HGW に IoT デバイスの FQDN 自動生成機能を搭載し，既存の IoT デバイス向けプロトコルを利用することにより，既存技術の課題を解決できることを示した．また，自動生成される FQDN の構成も既存技術よりもわかりやすく，部屋の情報やユーザの設定などにより柔軟に変更可能な仕様となっているため，ユーザビリティに優れていることを述べた．

提案システムの実用性を考察するために，DLNA 機器に対して FQDN を自動生成し DNS サーバに登録を行うシステムのプロトタイプ実装を行い，IPv4 ネットワークでの動作確認及び処理時間の評価を行った．その結果，HGW が機器探索を開始し，DLNA 機器に FQDN を割り当てるまでの時間から，提案方式では IoT デバイスをホームネットワークに接続すると直ちに FQDN の動的な生成，登録が行えると考えられる．

今後は，mUPnP ライブラリの IPv6 対応と他の通信プロトコルの対応のほか，自動生成した IoT デバイスの FQDN を変更するシステムの実装を行う．

参考文献

[1] 総務省：平成 29 年版情報通信白書，pp.125 (2017). http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h29/pdf/n3300000.pdf.
[2] IoT platforms: enabling the Internet of Things (2016). https://cdn.ihs.com/www/pdf/enabling-IOT.pdf.
[3] IPv6 によるインターネットの利用高度化に関する研究会 第四次報告書 ～IoT 時代を拓く新たな戦略～ (2015). http://www.soumu.go.jp/main_content/000388694.pdf.
[4] Bonjour Concepts - Apple Developer. https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/NetServices/Articles/about.html.
[5] Lee, S., Jeong, J. P. and Park, J. S.: DNSNA: DNS name autoconfiguration for Internet of Things devices, *2016 18th International Conference on Advanced Communication Technology (ICACT)*, pp. 410–416 (online), DOI: 10.1109/ICACT.2016.7423412 (2016).
[6] Zero Configuration Networking (Zeroconf). http://www.zeroconf.org/.
[7] Guttman, E.: Autoconfiguration for IP networking: enabling local communication, *IEEE Internet Computing*, Vol. 5, No. 3, pp. 81–86 (online), DOI: 10.1109/4236.935181 (2001).
[8] Cheshire, S., Aboba, B. and Guttman, E.: Dynamic Configuration of IPv4 Link-Local Addresses, RFC 3927, IETF (2005).
[9] Cheshire, S. and Krochmal, M.: Multicast DNS, RFC 6762, IETF (2013).
[10] Moore, N.: Optimistic Duplicate Address Detection (DAD) for IPv6, RFC 4429, IETF (2006).
[11] Crawford, M., Fermilab and Haberman, B.: IPv6 Node Information Queries, RFC 4620, IETF (2006).
[12] Vixie, P., Thomson, S., Rekhter, Y. and Bound, J.: Dynamic Updates in the Domain Name System (DNS UPDATE), RFC 2136, IETF (1997).
[13] DLNA. https://www.dlna.org/.
[14] ECHONETLite CONSORTIUM. https://echonet.jp/.
[15] HomeKit - Apple. https://www.apple.com/jp/ios/home/.
[16] GitHub - cybergarage/mupnp: mUPnP for C. https://github.com/cybergarage/mupnp.
[17] Libbind — Internet Systems Consortium. https://www.isc.org/downloads/libbind/.