Vol.2022-GN-115 No.43
Vol.2022-CDS-33 No.43
Vol.2022-DCC-30 No.43
2022/1/21

IoTシステムへのHTIPの適用に関する研究

宮川 真一[1]　PHAM Van Cu[1]　リム 勇仁[1]　丹 康雄[1]

概要：HTIP(Home-Network Topology Identifying Protocol) とは IoT エリアネットワーク環境においてエンドノード、リンク、トポロジ等の情報を収集するプロトコルである。
本稿では、複数の IoT エリアネットワークで構成される IoT システムに HTIP を適用する研究を通して、ある条件の下で、IoT エリアネットワークに収容できるエンド端末数の収容数が最大で 719 台、最小で 77 台となる結果を得たこと、異なるプロトコル間で HTIP 情報をやり取りするネイティブフレームのフォーマットにおける問題点を指摘し、新たな提案を行ったこと、および IoT システムにおける HTIP 情報の利用の形態について述べる。

キーワード：HTIP, IoT システム、IoT エリアネットワーク、トポロジ

1. はじめに

HTIP(Home-Network Topology Identifying Protocol) とは IoT エリアネットワーク環境においてエンドノード、リンク、トポロジ等の情報を収集するプロトコルであり、2010 年に TTC(情報通信技術委員会)JJ-300.00 規格 [1] として国内標準化されている。その後、2011 年には ITU-T G.9973[2] として国際標準化も行われている。

HTIP はこれまでに個々の IoT エリアネットワークの信頼性を向上させることに対して成果を上げている一方で、多数の IoT エリアネットワークで構成される大規模な IoT システムという視点での信頼性向上の議論はなされてこなかった。IoT システムは LPWA に代表されるようにエリア的な広がりをみせ、構成も大規模化・複雑化してきている。それに伴い、IoT システムとしての信頼性向上の必要性は高まってきている。

本研究の目的は IoT エリアネットワーク管理運用技術として用いられてきた HTIP を利用し、一般的な IoT システム全体の信頼性を飛躍的に改善することである。

多数の IoT エリアネットワークが階層的に接続された IoT システムにおいて、上位層になるにつれて増加する HTIP の情報を最小限にしぼりこむことでリソースの消費を抑え、大規模な IoT システムの一元的な管理運用を可能とする。これにより IoT システムの総合的な信頼性向上をはかることを目的とする。

本研究ではまず、通信速度や帯域など制約の多い IoT エリアネットワークに HTIP を適用した際に、何台の端末が収容できるかを検討する。

次に、IoT エリアネットワーク内にプロトコルの異なる端末が存在しても、他の端末と同様に扱うことができる通信フォーマットのフレーム形式について、実装に向けた提案を行う。

最後に、大規模な IoT ネットワークの管理手法についての提案を行う。

[1] 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology

2. IoT エリアネットワーク内に収容できる最大接続数の検討

HTIP は、機器の種類や設定情報などを近隣の機器に通知するデータリンク層のプロトコルであり、IEEE802.AB[3] で標準化された LLDP(Link Layer Discovery Protocol) の拡張項目を利用している。

HTIP ではメッセージのブロードキャストにより管理情報を得ている。ホーム NW を想定した HTIP 利用時の負荷の検討 [4] はすでに行われている。一方で、IoT エリアネットワークはブロードバンドホームネットワークとは異なり、通信帯域が限られ、ノード数が多くなるなど前提条件が違っている。そのため、最悪のケースでも HTIP が利用可能な範囲であるかについて改めて検討する必要がある。

構成要素となる機器には、センサデバイス等のエンド端末と、それらを収容するネットワーク機器（以下、NW 機器と記述）とがあり、以下の特徴を持つ。

- エンド端末は末端の構成要素であり、自分自身の機器情報のみを通知する。
- NW 機器は自分自身の機器情報だけでなく、ブリッジ

としてのフォワーディングデータベース (FDB) に登録されている情報も通知する。

NW 機器が多くなればなるほど送信される FDB が増えること、またトポロジによって重複した内容が含まれることから、限られた帯域内で管理できる台数は減っていくことが予想される。

このことについて、条件として最悪となるトポロジを想定し、管理限界の定量的な評価を行う。

2.1 検討対象とするトポロジ

検討対象は、収容される台数が最も多いと想定される場合 (Star 型) と、最も少ないと想定される場合 (Phylogenetic Tree 型) の 2 種類とする。

- Star 型
  1 台の NW 機器にすべてのエンド端末が接続されている形態。イメージを図 1 に示す。
- Phylogenetic Tree 型
  NW 機器が階層的に接続され、各 NW 機器には 1 台のエンド端末が接続されている形態。イメージを図 2 に示す。

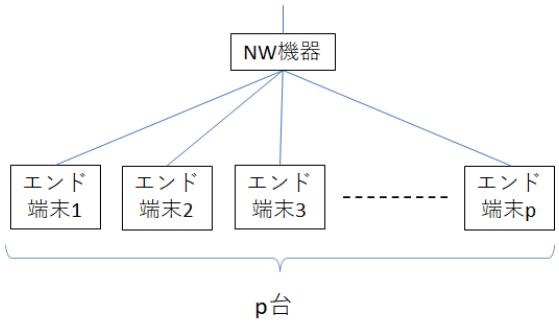

図 1 Star 型

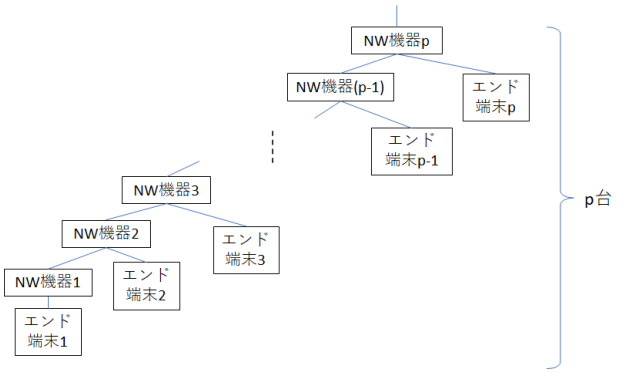

図 2 Phylogenetic Tree 型

2.2 検討における前提条件

2.2.1 対象の伝送技術

対象とする伝送技術は以下の 2 種類とする。

- Wi-SUN
- ZigBee

2.2.2 通信速度

Wi-SUN の通信速度は、50kbps, 100kbps, 150kbps, 300kbps から選択可能である。ZigBee の通信速度は 250kbps であるが、安定した通信を行うには 144kbps 程度である。

これらのことから、通信速度は 144kbps としてこの先の議論をすすめることとする。

2.2.3 HTIP の送信間隔

送信間隔は IEEE802.1AB[3] で推奨されている 30 秒とする。

ただし、HTIP のフレームが MTU サイズを超えて、分割して送信された場合には、図 3 に示すように最初に到着した LLDP フレームから 10 秒以内 (送信間隔の 1/3 以内)[5] に到着した LLDP フレームに含まれる TLV の和集合を、送信者に関する TLV 情報として扱うものとする。

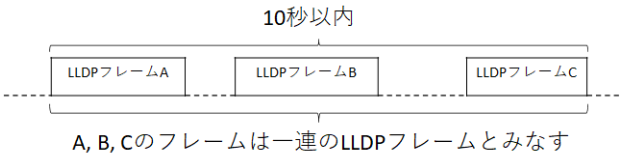

図 3 一連の LLDP フレーム

2.2.4 通信データ量

144kbps で 10 秒間安定して通信した際のデータ量を計算する。

$$144[kbps] \times 10[sec] \div [bit/byte] = 180[kbyte]$$

帯域を 50%消費するものとする。

$$180[kbyte] \times 0.5 = 90[kbyte]$$

よって、今回の条件において 1 回の通信で送信できる LLDP フレームのデータの総量は 90kbyte となる。また、LLDP フレームが MTU サイズを超えた場合は、毎回 MTU のサイズで分割できるものとする。

2.2.5 その他前提条件

そのほかの前提条件について、以下のように定める。

- NW 機器が FDB を格納するバッファの容量は十分あるものとする
- エンド端末、NW 機器からの送信は順次行われ、衝突は発生しないものとする
- HTIP 情報を集約する HTIP マネージャ配下を 1 つのトポロジとする
- トポロジ内の各ノードがブロードキャストで LLDP フレームを出力する

Vol.2022-GN-115 No.43
Vol.2022-CDS-33 No.43
Vol.2022-DCC-30 No.43
2022/1/21

2.3 机上計算

2.3.1 LLDP データサイズ

2.3.1.1 エンド端末

HTIP の仕様および稼働中の HTIP データサイズを参考に、以下の値とする。

表 1 エンド端末の LLDP データサイズ

TLV Type	説明	サイズ [kbyte]
1	Chassis ID	9
2	Port ID	9
3	Time To Live	4
4	Port Description	6
127	機器情報 区分	16
127	機器情報 メーカコード	16
127	機器情報 機種名	16
127	機器情報 型番	16
0	End Of LLDP	2
合計		94

この結果にイーサネットヘッダの 14byte を加えて、合計は 108byte となる。よって、ネットワーク内にエンド端末が n 台あった場合には、LLDP フレームのデータ総量は以下のようになる。

$$108 \times n \quad [byte] \tag{1}$$

2.3.1.2 NW 機器

NW 機器では、エンド端末の LLDP フレームのデータサイズに加えて接続構成情報が付加される。接続構成情報において、インタフェース種別を 1 byte、ポート番号を 1 byte、ポートに配下に n 台のエンド端末または NW 機器が接続されているとしたとき、1 ポート当たりの接続構成情報のサイズは以下となる。

$$11 + 6 \times n \quad [byte] \tag{2}$$

NW 機器のポート数が p であり、各ポートの先に接続されたエンド端末と NW 機器の数を順に $E_1, E_2, E_3, \ldots E_p$ としたとき、1 台の NW 機器が送出する LLDP フレームのデータサイズの総和は以下の式で表される。

$$108 + \sum_{k=1}^{p}(11 + 6 \times E_k) \quad [byte] \tag{3}$$

2.3.2 Star 型での計算

1 台の NW 機器に n 台のエンド端末が接続されている状態で HTIP データを送信したとき、90 kbyte 以下となる n の値を計算する。

各エンド端末が送信する LLDP フレームの合計は

$$108 \times n \quad [byte] \tag{4}$$

であり、NW 機器が送信する LLDP フレームは、

$$108 + (11 + 6 \times 1) \times n \quad [byte] \tag{5}$$

となるため、下記の式を満たす n を求めると、

$108 \times n + 108 + (11 + 6 \times 1) \times n < 90000$

$125 \times n < 89892$

$n < 719.136$

となり、719 台以下であることがわかる。

2.3.3 Phylogenetic Tree 型での計算

Phylogenetic Tree 型の特徴は、1 台の NW 機器に 1 台のエンド端末が接続したものを複数階層に積み上げた構成と考えることができる。すなわち、m 階層からなる Phylogenetic Tree 型のトポロジには、エンド端末が m 台、NW 機器も m 台収容されていることになる。

この時、エンド端末が送出する LLDP フレームの合計は以下のとおりである。

$$108 \times m \quad [byte] \tag{6}$$

NW 機器において、接続構成情報を考えない NW 機器本体のみの LLDP フレームの合計は以下のとおりである。

$$108 \times m \quad [byte] \tag{7}$$

NW 機器において、エンド端末との接続構成情報の LLDP フレームの合計は以下のとおりである。

$$(11 + 6 \times 1) \times m \quad [byte] \tag{8}$$

$$= 17 \times m \quad [byte] \tag{9}$$

NW 機器における、NW 機器との接続構成情報の LLDP フレームについて、図 4 で考える。

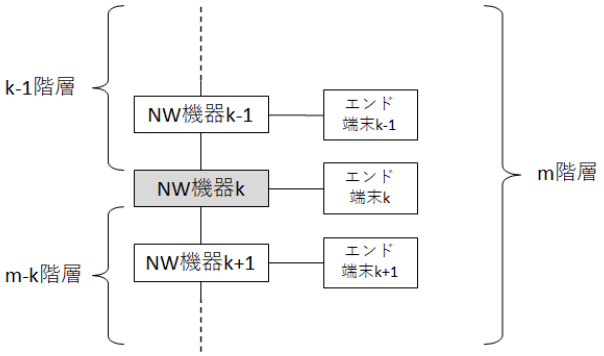

図 4 k 階層にある NW 機器

まず、m 階層あるうちの、末端でない k 階層目の NW 機器について考える。(k=2, 3, ..., $m-2$, $m-1$) このとき、k 階層より上位側は $(k-1)$ 階層あり、下位側は $(m-k)$ 階層あることになる。そのため、上位側の接続構成情報の LLDP フレームのバイト数は下記となる。1 階層につき NW 機器とエンド端末が 1 台ずつ接続されているため、ポートに接続された MAC アドレスの数は階層数× 2 となる。

$$(11 + 6 \times (k-1) \times 2) \quad [byte] \tag{10}$$

同様に下位側の接続構成情報の LLDP フレームのバイト数は下記となる。

$$(11 + 6 \times (m - k) \times 2) \quad [byte] \tag{11}$$

上記の 2 つの式を合わせて整理すると、

$$10 + 12 \times m \quad [byte] \tag{12}$$

となり、さらにこれが $(m - 2)$ 階層分あるので、下記となる。

$$(10 + 12 \times m) \times (m - 2) \quad [byte] \tag{13}$$

次に、NW 機器のうち末端の接続構成情報は、下記のとおりである。

$$(11 + 6 \times (m - 1) \times 2) \quad [byte] \tag{14}$$

これが上下に 2 台あるため、2 倍して整理すると下記のようになる。

$$24 \times m - 2 \quad [byte] \tag{15}$$

これまでの式を全部合算すると下記のようになる。

$$\begin{aligned} &108 \times m + 108 \times m + 17 \times m \\ &+(10 + 12 \times m) \times (m - 2) \\ &+24 \times m - 2 \quad [byte] \end{aligned}$$

これを整理すると、以下となる。

$$12m^2 + 243m - 22 \quad [byte] \tag{16}$$

LLDP フレーム のデータ量は、m=77 のとき 89,837 バイト、m=78 のとき 91.940 バイトとなる。このことから、Phylogenetic Tree 型で収容できるエンド端末の台数は 77 台以下であることがわかる。

2.4 計算結果の検証

Phylogenetic Tree 型に関して、机上計算結果が正しいことを証明するために、仮想環境および実機を用いて検証を行った。検証には、エンド端末として Raspberry Pi、NW 機器として Wise System[6] 社製の L2Switch を使用した。Raspberry Pi では HTIP Agent の C 言語実装である lwhtip[7] を動かし、パケットキャプチャには Wireshark[8] を用いて通信データの解析を行った。[9]

その結果、理論値通りの出力が行われていることが確認できた。

2.5 最大接続数についての考察

今回の前提条件の下では、エンド端末の収容が最大と思われる場合 (Star 型) で 719 台、最小と思われる場合 (Phylogenetic Tree 型) で 77 台となった。実際のエリアネットワークにおいては、Star 型に近い構成が多いこと、同一ブロードキャストドメイン内に接続する機器が数百台に及ぶケースは少ないことから、現実的に問題はないものと考えられる。

前提とした帯域使用率 50%という状況はアプリケーションによっては許容できないものであること、ネットワーク内でのデータの衝突は発生するものであること、通信速度もフィールドごとに条件が変わることを鑑みれば本結果よりも悪い結果になる可能性がある。

電池駆動する端末に対して定期的なブロードキャストベースのアプローチが適切なのかは要検討である。

トポロジおよび障害発見までに許容できる時間に依存して HTIP 送信間隔を調整するアルゴリズムも検討可能である。

3. ネイティブフレームフォーマットの実装に向けた提案

IoT エリアネットワークでは Ethernet フレームおよび GRE トンネリングが利用できない伝送技術が多く、これらで HTIP を利用するためにはシリアルラインとして扱うか、小さなフレームサイズに乗せるための標準形式が必要となる。標準形式に関して、先行研究 [10] では HTIP フレームから必要なデータを抽出し、それを伝送するネイティブのフレームにのせて送信する、ネイティブフレーム方式が提案されている。さらに、ネイティブフレームにおける非 Ethernet・非 IP 向けの HTIP 規格としてフレームフォーマットが提案されている。

先行研究において提案されたネイティブフレーム方式ではあるが、そこではフレームフォーマットの提案のみであり、実装や評価は行われていない。そのため、本研究においてネイティブフレーム方式のフレームフォーマットの実装に向けての提案を行う。

提案における留意点を述べる。IoT エリアネットワークにおいて利用されるエンド端末は、バッテリ駆動のデバイスのように計算資源の少ないものが多い。そのため、フレームに格納するデータは必要最低限のものとする。また、フレーム内のビットの利用について新たに定義する場合は、従来の利用方法との互換性や実装に負担にならないように留意する必要がある。対象とするプロトコルは、先行研究と同様に BLE, プライベート LoRa, LoRaWAN とする。

3.1 先行研究で提案されたフォーマット

先行研究 [10] が提案する ネイティブフレームフォーマットを示す。フォーマットはペイロード長に制限のある無線規格に対応するため、フレーム長の違う 2 種類のフォーマットが存在している。

3.1.1 uplink37 フォーマット

ペイロード長の長い uplin37 フォーマットを図 5 に示す。

フォーマットの各フィールドについて説明する。

byte	1	2	3	4	5	6	7	8	9	10	11
説明	※1	カウンター値		※2	区分	メーカコード					

byte	12	13	14	15	16	17	18	19	20	21	22	23	24
説明	機器の型番												

byte	25	26	27	28	29	30	31	32	33	34	35	36	37
説明	機種名												

図 5　先行研究の uplink37 フォーマット

- ※ 1（第 1 オクテット）
  - 0, 1 ビット　パケットの種類
    パケットの種類に関する情報を 2bit で格納する。パケットの種類の取りうる値を表 2 に示す。

表 2　パケットの種類

パケットの種類	説明
0b00	読み込み可能センサ
0b10	再書き込み可能センサ
0b01	HTIP 情報

  - 2-7 ビット　コマンド長
    パケット長から 1 を引いた数を byte 単位で格納する。HTIP 情報の場合、後述する機種名の長さが可変長となるためコマンド長は機種名に依存し、最大長は 36 となる。
- カウンター値（第 2,3 オクテット）
  通信時のパケットエラー率をゲートウェイで算出するためのカウンタ値を格納する。カウンタ値は機器の再起動時には 0 となり、これ以降パケット送信ごとに 1 ずつインクリメントされる。これにより、連続に送信されたパケットの順序や受信漏れを検出することができる。カウンタ値が 2byte の最大値の 65535 でカウントオーバーフローした場合はエラーとし、これ以上カウントさせない。
- ※ 2（第 4 オクテット）
  - 0-3 ビット　バッテリ残量
    デバイスの電源電圧もしくは蓄電電圧を、バッテリ残量としてパーセンテージで表現した値を 4bit で格納する。バッテリ残量の取りうる値を表 3 に示す。

表 3　バッテリ残量の値

バッテリ残量値	説明
0x0	バッテリ残量 0-10%
0x1	バッテリ残量 10-20%
0x2	バッテリ残量 20-30%
・・・	・・・
0x9	バッテリ残量 9-100%
0xa	バッテリ残量 100%
0xf	測定範囲外 (マイナス値など)

  - 4-7 ビット　Status
    デバイスの動作にかかわる情報を 4bit で格納する。Status の定義を表 4 に示す。

表 4　Status の値

対象ビット	説明
0b000x	0: 光発電による動作 (照度 2000lx >) 1:EDLC による動作 (照度 2000lx <)
0b00x0	0: 外部 I2C 通信ポート無 1: 外部 I2C 通信ポート有
0b0x00	0: 外部 UART 通信ポート無 1: 外部 UART 通信ポート無

- 区分（第 5 オクテット）
  デバイスの種類を 1 byte で格納する。このとき、端末区分情報リスト [11] を参照して該当する区分を記述するが、端末区分情報リストでの端末区分一覧では値が文字列となっており、1 byte で入力できるようなコード化がされていない。そのため、実装の際には情報の送り手と受け手の間でコード化のルールを定める必要がある。
- メーカコード（第 6-第 11 オクテット）
  デバイスのメーカコードを 6 文字の ASCII で記述する。メーカコードは IEEE に登録されたカンパニー ID のみを記述する。使用可能な文字は以下である。
  [a-fA-F0-9]
- 機種の型番（第 12-第 24 オクテット）
  型番を 13 文字の ASCII で記述する。使用可能な文字は以下である。
  [a-zA-Z0-9]|[-’()+,./:=?;!*#@$_%]
- 機種名（第 25-最大第 37 オクテット）
  機種名を最大 13 文字までで記述する。

3.1.2　uplink11 フォーマット

ペイロード長の短い uplin11 フォーマットを図 6 に示す。

フレーム形式①

byte	1	2	3	4	5	6	7	8	9	10	11
説明	※3	カウンター値		※2	区分	メーカコード					

フレーム形式②

byte	1	2	3	4	5	6	7	8	9	10	11
説明	※3	カウンター値		機器の型番							

フレーム形式③

byte	1	2	3	4	5	6	7	8	9	10	11
説明	※3	カウンター値		機種名							

図 6　先行研究の uplink11 フォーマット

フォーマットの各フィールドについて説明する。説明がないフィールドは、uplink37 と同一である。

- ※ 3　(第 1 オクテット)
  - 0, 1 ビット　パケットの種類
    パケットの種類にかかわる情報を 2bit で格納する。HTIP 情報では、0b01 を格納する。

Vol.2022-GN-115 No.43
Vol.2022-CDS-33 No.43
Vol.2022-DCC-30 No.43
2022/1/21

– 2, 3 ビット　フレームインデックス
フレームの種類にかかわる情報を 2bit で格納する。フレームインデックスの定義を表 5 に示す。

表 5　フレームインデックス

フレームインデックス	説明
0b00	区分・メーカコード
0b01	機器の型番
0b10	機種名

– 4-7 ビット　コマンド長
パケット長から 1 を引いた数を byte 単位で格納する。

- 機種名（第 4-第 11 オクテット）
機種名を最大 8 文字で記述する。

3.2　先行研究のフレームフォーマットの妥当性

3.2.1　機種名について

先行研究で提案されたネイティブフレームフォーマットについて妥当性を検討する。

まず機種名が含まれている理由について先行研究によれば、”実装必須”であるからとのことであった。HTIP の仕様を定めている JJ-300.00[1] によれば、機種名を含む機器情報 (TTC Subtype=1) 自体は”実装必須”となっているものの、機器情報を構成する各項目に関して、機種名は”実装推奨”となっており、”実装必須”とはなっていない。

以上のことから、先行研究で提案された機種名を追加したフレームフォーマットである uplink37 については、そもそもリソースの少ない IoT 機器での運用を考えると、妥当とは言えないものと考えられる。

3.2.2　フレームインデックスについて

次に uplink11 のフレームインデックスについて検討する。uplink11 は 11 バイトのフレームであるため、コマンド長は 4 ビットあれば表現可能である。そのため、パケットの種類とコマンド長では未使用の 2, 3 ビット目をフレームインデックスとして割り当てた。しかしながらネイティブフレームの受信側として、uplink11 かそうでないかの判別処理を考えたとき、その区別はコマンド長で行うのが自然である。そうなると第 1 オクテットの 3 ビット目の意味が uplink11 と uplink37 とで食い違うことになり、混乱を引き起こしてしまう。

3.3　ネイティブフレームフォーマットの新たな提案

前述のとおり先行研究の提案では問題があるため、本研究ではネイティブフレームフォーマットについて新たな提案を行う。変更内容は以下の 2 点である。

(1) フレームフォーマットに機種名は追加しない。
(2) uplink11 のフレームインデックスは第 1 オクテットの第 2 ビットのみとする。

これらの提案についての理由を以下で説明する。

3.3.1　機種名について

機種名は JJ-300.00[1] によると実装推奨であるが、実装必須ではない。IoT デバイスの少ないリソースを有効に利用する観点からも、フレームフォーマットの拡張や種類の追加は不適切である。また、機種名をなくすことで uplink11 のフォーマットの種類を 3 種類から 2 種類に減らすことができる。

3.3.2　フレームインデックスについて

uplink11 において、複数のフレーム種類を見分けるための何かしらの識別子は必要である。先行研究においては、3 種類のフレームを識別するために第 1 オクテットの第 2、第 3 ビットの 2 ビットを利用する提案となっていた。しかしながら受信側での処理を考えたとき、uplink11 と uplink37 では第 1 オクテットの第 3 ビットが複数の意味を持つため、支障が生じてしまっていた。

uplink11 のフレームインデックスは、3 種類のフレームフォーマットを見分ける必要があったため 2 ビット必要であった。機種名を送信しないとするとフォーマットは 2 種類となり、フレームインデックスは 1 ビットあれば区別できるようになる。第 1 オクテットの第 2 ビット目は uplink37 でも使用していないため、フレームインデックスとして第 1 オクテットの第 2 ビットを割り当てる。そうすることによって、前に述べた障害を回避することができるようになる。

3.3.3　uplink24 フォーマット

機種名を削除したため、uplink24 として新たに提案するフレームフォーマットを図 7 に示す。

byte	1	2	3	4	5	6	7	8	9	10	11
説明	※4	カウンター値		※2	区分	メーカコード					

byte	12	13	14	15	16	17	18	19	20	21	22	23	24
説明	機器の型番												

図 7　新たに提案する uplink24 フォーマット

フォーマットの各フィールドについて、これまでとの差分を説明する。

- ※ 4　(第 1 オクテット)
– 0, 1 ビット　パケットの種類
パケットの種類にかかわる情報を 2bit で格納する。HTIP 情報では、0b01 を格納する。
– 2 ビット　フレームインデックス
フレームの種類にかかわる情報を 1bit で格納する。フレームインデックスの定義を表 6 に示す。

表 6　フレームインデックス

フレームインデックス	コマンド長
0b0	区分・メーカコード
0b1	機器の型番

– 3-7 ビット　コマンド長
　パケット長から 1 を引いた数を byte 単位で格納する。
- 機種の型番（第 12-最大第 24 オクテット）
　任意長で HTIP における型番を最大 13 文字までの ASCII で記述する。

3.3.4 uplink11 フォーマット

本研究が提案する uplink11 フレームフォーマットを図 8 に示す。フォーマットの各フィールドについては、これまで説明してきたとおりである。

フレーム形式①

byte	1	2	3	4	5	6	7	8	9	10	11
説明	※4	カウンター値		※2	区分	メーカコード					

フレーム形式②

byte	1	2	3	4	5	6	7	8	9	10	11
説明	※4	カウンター値		機器の型番							

図 8　新たに提案する uplink11 フォーマット

4. 大規模 IoT システムの効率的な管理

これまでは IoT エリアネットワーク内についての議論であったが、ここからは複数の IoT エリアネットワークで構成された IoT システムに対象を広げる。IoT エリアネットワークの境界にはゲートウェイ (以下、GW) が位置し、GW-サーバ間は、Ethernet、LTE またはプライベート LoRa で接続されているものとする。

IoT システムの構成例を図 9 に示す。

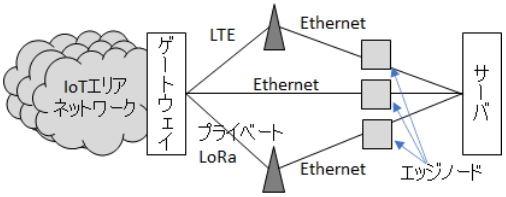

図 9　IoT システムの構成例

4.1 HTIP マネージャの配置

エンド端末や NW 機器から HTIP 情報を収集し、トポロジを作成する機能もしくはそれを行う端末を HTIP マネージャという。この HTIP マネージャ機能を持たせるノードとして下記の 3 通り考えられ、それぞれの特徴を記す。

- GW
　GW にマネージャ機能を持たせることで、IoT エリアネットワーク内の状況が把握しやすくなり、障害発生時の調査や切り分けが容易になる。GW からサーバへの HTIP 情報の送信は、HTIP マネージャが生成した JSON 形式のトポロジ情報で渡すことができる。この場合エリアネットワーク内の HTIP の送信間隔と、エリアネットワーク外への HTIP 情報の送信間隔を別々にすることができるため、GW-サーバ間の通信データ量を削減することができる。一方で、トポロジの生成などの処理行うため、GW に計算資源が必要となる。
- サーバ
　GW は、エリアネットワーク内の装置から受け取った HTIP 情報にタイムスタンプおよびサーバの IP アドレスを付与してサーバに送信する。サーバはすべての HTIP 情報を集約し、トポロジの生成などの HTIP マネージャの処理を行う。サーバで IoT システム内の全端末を一元的に管理できるが、サーバにかかる負荷は大きくなる。
- GW-サーバ間のエッジノード
　GW とサーバの間にエッジノードを置き、トポロジの構築や履歴の蓄積などサーバ処理を一部肩代わりさせる構成である。サーバの負荷を減らすことができる。

4.2 GW-サーバ間のデータ量の検討

GW から出力される HTIP 情報のサイズを計算する。エリアネットワーク内のエンド端末数を n としたとき、GW から出力される HTIP に関する情報のデータサイズは、HTIP マネージャの JavaScript 実装であり、オープンソースソフトウェアである node-htip[12] による実機検証を通して、以下の値となることが判明している。

- TLV 形式の HTIP 情報
  – n 台のエンド端末が出力する HTIP 情報

$$94 \times n \quad [byte] \tag{17}$$

  – NW 機器が出力する HTIP 情報

$$160 + 17 \times n \quad [byte] \tag{18}$$

　GW からの出力時に、各端末からの HTIP 情報にタイムスタンプ (14 バイト) と IP ヘッダ (20 バイト) が付与される。すべてを合わせると、以下の式となる。

$$194 + 145 \times n \quad [byte] \tag{19}$$

- トポロジ (JSON) 形式の HTIP 情報

$$911 + 594 \times n \quad [byte] \tag{20}$$

　GW からの出力時に、IP ヘッダ (20 バイト) が付与される。よって、以下の式となる。

$$931 + 594 \times n \quad [byte] \tag{21}$$

GW が HTIP の情報を 30 秒間隔で LTE を使用してサーバに送信するとする。総務省が定めたガイドライン [13] に基づき、携帯キャリア 3 社が計測した実効速度によれば、LTE の上りの最小値は 1Mbps となっている。この値を使用してエンド端末数が 1000 台の時の HTIP データの帯域使用率を計算すると、以下のようになる。

- TLV 形式の HTIP 情報
  $(194 + 145 \times 1000) \times 8 \div 30 \div (1 \times 10^6) \simeq 3.87$ [%]
- トポロジ (JSON) 形式の HTIP 情報
  $(931 + 594 \times 1000) \times 8 \div 30 \div (1 \times 10^6) \simeq 15.86$ [%]

実機での検証の結果、エンド端末、NW 機器が出力する TLV 形式の HTIP 情報の総和よりも、HTIP マネージャで構築されたトポロジ (JSON) 形式の HTIP 情報のサイズが 5 倍近く大きくなっている。しかしながら、1000 台構成で、LTE の最小の速度の場合でも帯域の消費は 16%弱であり、実際の利用における影響は少ないと考えられる。

4.3 サーバでの HTIP 情報の利用

サーバにおける HTIP 情報の利用について、これまで GW が TLV 形式の HTIP 情報を送信する形態、および GW がトポロジ情報を送信する形態を扱ってきた。これらのほか、サーバが主導権を持ち各 GW に問い合わせる形態も考えられる。これらは、システムに応じて適宜使い分けることができる。それぞれの特徴を示しイメージを図示する。

(1) GW が TLV 形式の HTIP 情報を出力する形態
GW のマネージャ機能の有無にかかわらず、GW からサーバへの TLV 形式の HTIP 情報を送信する形態。エンド端末や NW 機器が出力する HTIP 情報が、GW でタイムスタンプを付与されてサーバに届けられる。サーバでトポロジの構築や HTIP 情報の保存、分析を行う。

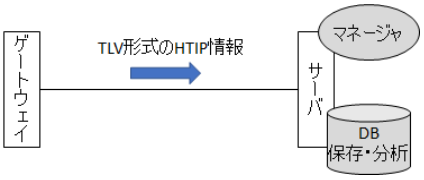

図 10 HTIP 情報の利用形態 1

(2) GW がトポロジ情報を出力する形態
GW が HTIP マネージャ機能を持ち、そこで生成したトポロジ情報をサーバに送信する形態。この場合、GW はトポロジ情報を HTIP の送信間隔である 30 秒に 1 回送る必要はなく、変化があった場合などの任意のタイミングで送信することができる。それにより、GW-サーバ間の通信量を減らすことができる。

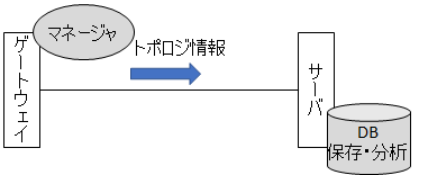

図 11 HTIP 情報の利用形態 2

(3) サーバが GW に問い合わせる形態
GW にはマネージャ機能と生成したトポロジを蓄積する DB を持たせ、サーバは REST API などにより GW に問い合わせを行う形態。問い合わせ方法としては、トポロジ全体や特定の端末だけを取得することができる。

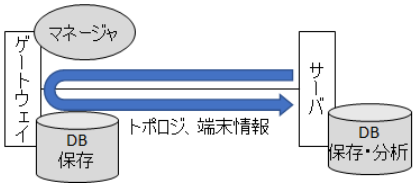

図 12 HTIP 情報の利用形態 3

5. おわりに

本研究では、IoT エリアネットワーク内での収容台数の上限の算出、ネイティブフレームを使った HTIP 情報の送信について実装面からの提案、さらに IoT システムの管理方法として、いくつかの形態とその特徴を整理した。

これにより、大規模な IoT システムにおいても HTIP を適用することで、IoT デバイス 1 台 1 台を管理し、IoT システム全体の信頼性を高めることができるようになる。

参考文献

[1] (一社) 情報通信技術委員会 (TTC), "JJ-300.00 ホーム NW 接続構成特定プロトコル", 第 3.0 版.
[2] ITU-T.:"G.9973:Protocol for identifying home network topology", 2011. https://www.itu.int/rec/T-REC-G.9973-201110-S/en(2021 年 12 月 21 日閲覧)
[3] IEEE Computer Society: "802.1AB-2009: Local and Metropolitan Area Networks - Station and Media Access Control Connectivity Discovery", September 2009.
[4] 美原義行, 山崎毅文, 佐藤敦, 市森峰樹：HTIP 利用時における LAN 内ネットワーク負荷の検討, 情報処理学会第 74 回全国大会, 3E-5, 3-77, 2012
[5] (一社) 情報通信技術委員会 (TTC), "TR-1061 JJ-300.00 機能実装ガイドライン", 第 2.0 版.
[6] Wise System, https://wisesystem.jp/(2021 年 12 月 21 日閲覧)
[7] lwhtip, https://github.com/Tan-Lab/lwhtip(2021 年 12 月 21 日閲覧)
[8] Wireshark, https://www.wireshark.org/(2021 年 12 月 21 日閲覧)
[9] (一社) 情報通信技術委員会 (TTC), "TR-1086 HTIP 評価ツールおよび構築ガイドライン", 第 1.0 版.
[10] 山本遥平. "低コスト IoT 機器の管理運用技術に関する研究", 修士論文, 北陸先端科学技術大学院大学, 2019
[11] (一社) 情報通信技術委員会 (TTC), "JJ-300.01 端末区分情報リスト", 第 2.1 版.
[12] node-htip,https://github.com/Tan-Lab/node_htip(2021 年 12 月 21 日閲覧)
[13] 総務省, 移動系通信事業者が提供するインターネット接続サービスの実効速度計測手法及び利用者への情報提供手法等に関するガイドライン. https://www.soumu.go.jp/main_content/000371346.pdf(2021 年 12 月 21 日閲覧)