IIR

Internet
Infrastructure
Review

Dec.2025
Vol.68

定期観測レポート

IIJインフラから見たインターネットの傾向〜2025年

フォーカス・リサーチ(1)

DNSフルリゾルバbowlineの設計と実装

フォーカス・リサーチ(2)

古くて新しい衛星インターネット、Starlinkが変えていく世界

フォーカス・リサーチ(3)

生成AIによる社内RAG基盤とマルチエージェント連携への拡張
〜 実装技術と業務効率化の成果、今後の展望 〜

IIJ
Internet Initiative Japan

Internet Infrastructure Review

December 2025 Vol.68

エグゼクティブサマリ

2025年は「AIエージェント元年」と呼ばれています。2022年に公開され、世界中に衝撃を与えたChatGPTの登場から3年、2025年8月にGPT-5が発表されました。GPT-5は画像・動画といったマルチモーダルに対応しただけでなく、外部連携機能が大幅に強化され、必要に応じて外部データソースと連動した自律的な動作（エージェント）が可能になりました。更に、企業においても試験的なAIの利用に目処がたち、実際の業務に即した活用の機運が高まっていることも、AIエージェントの導入を後押ししています。

今号では、インターネットの最新動向をお伝えすると共に、生成AI及びAIエージェントに関するIIJの開発事例も紹介します。

第1章は、定期的にお伝えしているインターネットの傾向分析です。IIJバックボーンにおけるIPv6トラフィックは前年比25.2%増と大きく伸長し、全体の約24%を占めるまでになりました。モバイル端末でのIPv6有効化率も向上しています（iOS系 86.9%、Android系 35.5%）。加えて、IIJのインターネットバックボーンから見たトレンドにも触れています。

第2章では、IIJが独自開発したDNSフルリゾルバ「bowline」の設計思想と実装技術を解説します。bowlineはHaskellで実装され、オープンソースとして公開されています。Haskell実装に関する背景及び詳細は、ぜひ本文をご一読いただきたいと思いますが、こうした自社開発ソフトウェアが、IIJの提供するサービスの信頼性向上を支えています。

第3章では、近年注目を集めるLEO（低軌道）衛星インターネットの歴史と、Starlinkを中心とした最新動向を解説します。2024年元日に発生した能登半島地震の際には、その特性が活かされ、災害時の通信手段として機能しました。将来的には衛星間レーザー通信による高速・低遅延ネットワークや惑星間通信への応用など、様々な構想が広がっています。IIJグループでも2025年12月から正式にStarlinkの取り扱いを開始します。

第4章では、IIJが独自開発・運用する社内RAG（Retrieval-Augmented Generation）基盤「sbdGPT」と、そのマルチエージェント拡張について紹介します。IIJは、社内に分散している膨大なナレッジを統合し、生成AIを活用した業務効率化を推進しており、2023年夏の「sbdGPT」運用開始以降、月間約1500時間相当の業務効率化を実現しました。現在は、社外情報も組み合わせたマルチエージェント構成、高度な調査を可能にするDeepResearchとの連携、更には企画・提案・戦略を自動生成する「Panorama」の独自開発など、AI活用の幅を大きく広げています。

以上、インターネット傾向分析や自社開発によるDNSフルリゾルバといった従来の領域から、衛星インターネットや生成AI活用など最新分野に至る幅広いトピックが、読者の皆様の一助となることを願っています。IIJは「技術で社会を支える」という使命のもと、安定したサービスの提供と新しい技術への挑戦という両面で、進化を続けてまいります。

染谷 直（そめや なおし）
IIJ常務執行役員 ネットワークサービス事業本部 クラウド本部長。1998年、IIJ入社。直後にIIJテクノロジー（2010年にIIJに吸収合併）へ出向。IIJテクノロジーではSI事業の立ち上げに携わり、多くのインターネットシステムの構築やコンサルティングに従事。その後、16年よりIIJのサービス事業部門に異動し、クラウド事業の中期事業戦略を担当。19年、クラウド事業責任者に就任。今年度より「IIR」編集長に就き、IIJにおけるリアルな技術情報を横断的かつ積極的に読者の皆様へお届けしたいと考えている。

IIJインフラから見たインターネットの傾向 〜2025年

インターネットサービスを提供するIIJは、国内有数規模のネットワーク・サーバインフラの運用を通じて得られた情報からインターネットの動向を分析し、本誌IIRで毎年報告しています。今回もBGP経路、DNSクエリ分析、IPv6＆モバイル、インターネットバックボーンの各視点からここ1年の変化の傾向を分析しました。

Theme 01

BGP・経路数

最初にIIJ網から他組織に広報している「IPv4 フルルート」の情報（表-1）及び「IPv4 フルルート」に含まれるunique IPv4アドレス数の情報（表-3）を確認します。

総経路の年間増加数は+4万超にまで回復し（図-1）、総数は99万超となりました。なお本稿執筆時点（2025年10月上旬）では既に100万経路を突破していますが、他の観測点と同様に特段の騒ぎも起こらず無事に節目を越えられた模様です。プレフィクス長ごとで見ると/24、/23に加えて/16及び/20の経路数増加が目立つ結果となりました。/18経路数も表-1からは減少に見えますが、集約経路のみ（より短いプレフィクス長の情報がある経路を除去。表1-a）であれば逆に3桁の増加であり、これらが近年のアドレス移転で需要が高いものと言えそうです。unique IPv4アドレス数は2年連続の減少傾向から一転して7400万弱（/8ブロック×4.4）の大幅増となりました。2022年の値も超えて本定期観測開始後の最高値を更新しています。

表-1「IPv4フルルート」に含まれるプレフィクス長ごとの経路数の推移

年月	/8	/9	/10	/11	/12	/13	/14	/15	/16	/17	/18	/19	/20	/21	/22	/23	/24	total
2016年9月	16	13	36	101	267	515	1050	1767	13106	7782	12917	25229	38459	40066	67270	58965	335884	603443
2017年9月	15	13	36	104	284	552	1047	1861	13391	7619	13385	24672	38704	41630	78779	64549	367474	654115
2018年9月	14	11	36	99	292	567	1094	1891	13325	7906	13771	25307	39408	45578	88476	72030	400488	710293
2019年9月	10	11	37	98	288	573	1142	1914	13243	7999	13730	25531	40128	47248	95983	77581	438926	764442
2020年9月	9	11	39	100	286	576	1172	1932	13438	8251	14003	25800	40821	49108	101799	84773	473899	816017
2021年9月	16	13	41	101	303	589	1191	2007	13408	8231	13934	25276	41915	50664	106763	91436	497703	853591
2022年9月	16	13	39	101	298	592	1208	2064	13502	8292	13909	25051	43972	52203	109071	96909	536520	903760
2023年9月	16	14	39	102	298	577	1196	2064	13490	8245	13809	25059	43863	51012	109514	98178	550621	918097
2024年9月	16	16	37	93	295	573	1165	2059	13224	8220	13718	24624	43786	51827	111483	99239	579274	949649
2025年9月	16	14	41	92	298	576	1200	2125	13768	8257	13491	24718	44863	52244	112330	102820	616490	993343

表-1-a「IPv4 フルルート」に含まれるプレフィクス長ごとの経路数（集約経路のみ）

年月	/8	/9	/10	/11	/12	/13	/14	/15	/16	/17	/18	/19	/20	/21	/22	/23	/24	total
2024年9月	16	9	34	88	264	487	1024	1667	10277	4417	6815	14827	19079	22624	59317	42062	283007	466017
2025年9月	16	9	41	85	263	485	1031	1655	11026	4460	6943	14831	20459	22686	59763	43628	304448	491829
（増減数）	0	0	4	-3	-1	-2	7	-12	749	43	128	4	1380	62	446	1566	21441	25812

表-2「IPv6フルルート」に含まれるプレフィクス長ごとの経路数の推移

年月	/16-/28	/29	/30-/31	/32	/33-/39	/40	/41-/43	/44	/45-/47	/48	total
2016年9月	153	1294	216	8110	3092	1445	371	1492	1006	14291	31470
2017年9月	158	1757	256	9089	3588	2117	580	1999	1983	18347	39874
2018年9月	168	2279	328	10897	4828	2940	906	4015	2270	24616	53247
2019年9月	192	2671	606	12664	6914	3870	1566	4590	4165	34224	71462
2020年9月	205	3164	641	14520	9063	4815	2663	5501	4562	45160	90294
2021年9月	223	3628	705	20650	13050	10233	4170	11545	5204	61024	130432
2022年9月	298	4247	895	21926	15147	12509	4108	13840	6994	73244	153208
2023年9月	316	4357	923	23228	17427	14828	5518	16453	9579	86881	179510
2024年9月	322	5360	934	24739	20198	17657	4672	19418	12470	95628	201398
2025年9月	271	5089	1062	25643	22627	20719	5430	21021	14522	101103	217487

次に「IPv6 フルルート」の情報（表-2）及び「IPv6 フルルート」に含まれるunique IPv6/64ブロック数の情報（表-3）を確認します。

総経路の年間増加数は約1.6万でした。これは2019年以降の最低値であり、本定期観測で初めて前年比増加率1割未満（+8%）となりました。しかしながら集約経路のみを見るならば前年と同程度、本定期観測全体では4位の増加数（+0.9万）であり、またunique/64ブロック数が1兆を突破した（+23%）ことからもIPv6の導入、IPv6ネットワークの拡大は引き続き順調であると考えられます。プレフィクス長ごとで見ると/16～/28ブロック及び/29の経路数が大きく減少しましたが、前者の減少はプレフィクス長のより短い経路情報が他に存在する経路の減少に起因するものであり、集約経路のみであれば逆に+6の増加となっています（表2-a）。

最後に「IPv4/IPv6 フルルート」広報元AS（Origin AS）数を確認します（表-4）。なおこの一年の間にAPNICに対し2048の32-bit only AS番号が追加割り振りされています。

表-2-a「IPv6 フルルート」に含まれるプレフィクス長ごとの経路数（集約経路のみ）

年月	/16-/28	/29	/30-/31	/32	/33-/39	/40	/41-/43	/44	/45-/47	/48	total
2024年9月	223	5311	521	22942	6861	4776	1773	5348	3712	38335	89802
2025年9月	229	5037	546	23628	7820	6265	2282	6635	4235	42376	99053
（増減数）	6	-274	25	686	959	1489	509	1287	523	4041	9251

表-3「IPv4フルルート」に含まれるunique IPv4アドレス総数及び「IPv6フルルート」に含まれるunique IPv6/64ブロック総数の推移

年月	IPv4 アドレス数	IPv6 /64ブロック数
2016年9月	2,824,538,880	26,432,856,889
2017年9月	2,852,547,328	64,637,990,711
2018年9月	2,855,087,616	258,467,083,995
2019年9月	2,834,175,488	343,997,218,383
2020年9月	2,850,284,544	439,850,692,844
2021年9月	3,036,707,072	461,117,856,035
2022年9月	3,068,374,784	532,578,391,219
2023年9月	3,055,604,992	700,359,397,494
2024年9月	3,033,333,504	896,502,953,452
2025年9月	3,107,136,512	1,102,106,091,904

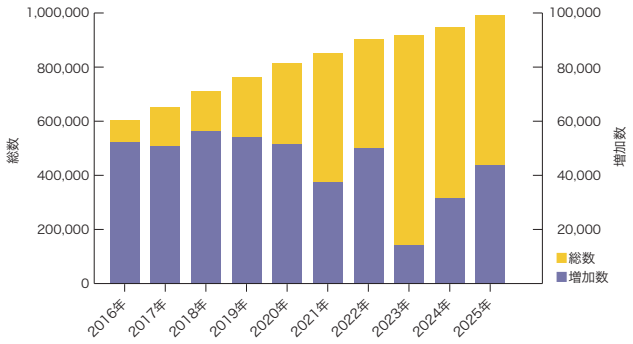


図-1「IPv4 フルルート」経路の総数及び年間増加数の推移

表-4「IPv4/IPv6フルルート」の広報元AS数の推移

AS番号	16-bit（1～64495）					32-bit only（131072～4199999999）				
広報経路	IPv4+IPv6	IPv4のみ	IPv6のみ	total	(IPv6 -enabled)	IPv4+IPv6	IPv4のみ	IPv6のみ	total	(IPv6 -enabled)
2016年9月	9116	33555	158	42829	(21.7%)	2406	9391	146	11943	(21.4%)
2017年9月	9603	32731	181	42515	(23.0%)	3214	12379	207	15800	(21.7%)
2018年9月	10199	31960	176	42335	(24.5%)	4379	14874	308	19561	(24.0%)
2019年9月	10642	31164	206	42012	(25.8%)	5790	17409	432	23631	(26.3%)
2020年9月	11107	30374	229	41710	(27.2%)	7653	19668	574	27895	(29.5%)
2021年9月	11465	29219	302	40986	(28.7%)	9514	21108	5242	35864	(41.1%)
2022年9月	11613	28398	369	40380	(29.7%)	10816	22211	5764	38791	(42.7%)
2023年9月	11770	27617	460	39847	(30.7%)	12640	22128	2067	36835	(39.9%)
2024年9月	12068	26720	476	39264	(31.9%)	13905	22737	2386	39028	(41.7%)
2025年9月	12239	25835	438	38512	(32.9%)	15319	23223	2568	41110	(43.5%)

16-bit AS番号 Origin AS数の減少は10年連続かつ本定期観測開始後で最多となりました。「IPv4のみ」の減少数は今回も同32-bit only ASの増加数を上回ったため、全体の「IPv4のみ」Origin AS数は3年連続の減少となっています（図-2）。また今回は「IPv6のみ」Origin AS数が初の2桁減となりました。32-bit only AS番号Origin AS数は前回と同程度増加し、遂に全Origin ASの過半を占めるに至りました。一方で「IPv4のみ」「IPv6のみ」の増加数は前回を大きく下回っており、全増加数における「IPv4+IPv6」Origin AS数の割合は前回の58%から68%に増加しています。過去にも観測されたIPv4とIPv6を別扱い（別AS）とする状況が更に解消に向かっていると感じられる結果であり、次回以降も継続されるのか注視していきたいと思います。

Theme 02

DNSクエリ解析

IIJでは利用者がDNSの名前解決を利用できるようフルサービスリゾルバを提供しています。この項目では名前解決の状況を解説し、IIJで2025年10月22日に行ったフルサービスリゾルバの一日分の観測データのうち、主にコンシューマサービス向けに提供しているサーバのデータに基づいて分析と考察を行います。

フルサービスリゾルバは利用者端末からのDNS問い合わせに応じて必要な名前解決機能を提供します。具体的には、名前を解決するためrootと呼ばれる最上位のゾーン情報を提供する権威サーバのIPアドレスを手がかりとして問い合わせを行い、適宜権威サーバをたどって必要なレコードを探します。フルサービスリゾルバが毎回このように他のサーバに問い合わせをしていると負荷や遅延の影響が問題となるため、得られた情報はしばらくキャッシュしておいて再び同じ問い合わせを受けた場合にはそのキャッシュから応答しています。最近はこの他にも家庭用ルータやファイアウォールなど、通信経路上の機器にもDNS関連の機能が実装されており、こうした機器がDNS問い合わせの中継や制御ポリシーの適用に関わっている場合があります。また、Webブラウザなど一部のアプリケーションでは独自の名前解決機能を実装している場合があり、OSの設定とは異なるポリシーで名前解決を行っている場合もあります。

ISPは接続種別に応じたPPPやDHCP、RA、PCOなどの通知手段を利用してフルサービスリゾルバのIPアドレスを利用者に伝え、利用者端末が名前解決用のネームサーバを自動設定できるようにしています。ISPは複数のフルサービスリゾルバを利用者に伝えられるほか、利用者は自身でOSやWebブラウザなどの設定を変更して利用するネームサーバを指定することもできます。利用者端末に複数の名前解決用ネームサーバが設定されている場合、どれを利用するかは端末の実装やアプリケーションに依存するため、フルサービスリゾルバ側では利用者が総量としてどの程度の問い合わせを行っているか分かりません。このことから、利用者側の挙動や状態が変わると突然あるフルサービスリゾルバに問い合わせが偏ることも考えられるため、フルサービスリゾルバでは問い合わせ動向を注視しながら、常に処理能力に余裕を持たせた運用を心がける必要があります。

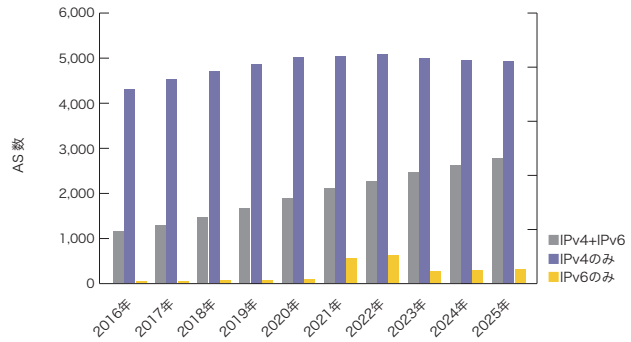


図-2「IPv4/IPv6 フルルート」の広報元AS数（合算）の推移

IIJが提供するフルサービスリゾルバの観測データを見てみると、利用者の利用傾向を示すように時間帯によって問い合わせ量が変動し、朝3時30分ごろに問い合わせ元のIPアドレス当たり最小の0.13query/sec、昼12時25分ごろにピークを迎えて0.27query/sec程度になっています。昨年に比べると、最小値は0.02ポイント、最大値は0.05ポイント程度減少しています。問い合わせ傾向を通信に使われたIPv4とIPv6のIPプロトコル別に見てみると、昨年からはIPv6が16ポイント程度増えており、IPv4を通信に使った問い合わせが全体の約43%、IPv6が約57%となり、観測開始以降初めてIPv6での問い合わせの方が多くなりました。問い合わせプロトコルに注目すると、UDPが97.67%でほとんどがUDPでの問い合わせになっています。ただ、TCPでの問い合わせは2021年が0.189%、2022年が0.812%、2023年が1.419%、2024年が1.561%、2025年が2.335%であり、近年TCPでの問い合わせ割合が緩やかに増加してきています。DNS over TLS (DoT)での問い合わせが増えてきているほか、TCPでの問い合わせを疎通や動作確認など、何らかの目的で利用する実装があるのではないかと推測しています。

近年の特徴的な傾向として、朝方の正時などキリの良い時刻に一時的に問い合わせが増加することが挙げられます。今年もこれまでと同様に増加が見られました。また、こうした正時の14秒前と9秒前の問い合わせも前年同様増加していることが観測できました。これは近年見られている傾向で、正時に増加する問い合わせ量では急な増加後、緩やかに問い合わせ量が減っていくのに比べて、正時前の増加では急な増加の直後にそれまでの問い合わせ量程度に戻っています。つまり多くの端末がきれいに同期して問い合わせを行っていることから、何かすぐに完了する軽量なタスクが実行されているのではないかと推測しています。昨年は朝8時から夜22時までの正時には逆に問い合わせが減少して、そこから徐々に増加していましたが、今年はすべての正時で増加していました。名前解決を利用している端末の実装に何らかの変更があったと推測しています。

問い合わせレコードタイプに注目すると、ホスト名に対応するIPv4アドレスを問い合わせるAレコードとIPv6アドレスを問い合わせるAAAAレコード、そしてWebサービスの解決に用いられるHTTPSレコードが全体の98%を占めています。AとAAAAの問い合わせ傾向は通信に利用されるIPプロトコルで違いが見られ、IPv6での問い合わせではより多くのAAAAレコード問い合わせが見られます。IPv4での問い合わせでは、全体の71%程度がAレコード問い合わせ、11%程度がAAAAレコード問い合わせです(図-3)。一方IPv6での問い合わせでは、全体の42%程度がAレコード問い合わせ、34%程度がAAAAレコード問い合わせです(図-4)。

昨年と比べるとIPv4では+9ポイント、IPv6でも+2ポイント程度Aレコードの問い合わせ割合が増加しています。これに代わって、2020年から観測され始めたHTTPSレコードのDNS問い合わせが2024年の観測に引き続き減少しています。IPv4で16%、IPv6で22%程度となっており、昨年と比べるとIPv4で-1ポイント、IPv6では-2ポイントと減少していました。2022年から観測され始めたSVCBレコードは、IPv4で0.28%、IPv6では0.58%と、まだ全体に対する比率は少

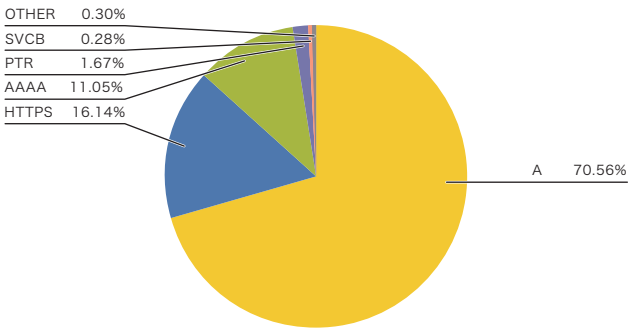


図-3 クライアントからのIPv4による問い合わせ

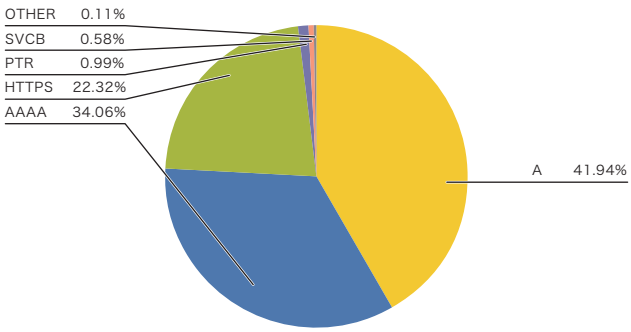


図-4 クライアントからのIPv6による問い合わせ

ないながらも順調に推移しています。これは、Discovery of Designated Resolvers（DDR）という、クライアントが暗号化に対応したフルサービスリゾルバを検出するための実装が利用されているためと推測しています。

Theme 03

IPv6＆モバイル

今回もIIJバックボーンのIPv6トラフィック、送信元AS番号、主なプロトコルについて見ていきます。また、昨年同様モバイルサービスの端末OS別のIPv6有効化率などについても調査します。

■ トラフィック

IIJのコアPOP（東京3ヵ所、大阪2ヵ所、名古屋2ヵ所）のバックボーンルータで計測したトラフィックを図-5に示します。集計期間は2025年2月1日から9月30日までの8ヵ月間です。

インターネットトラフィック量の期中の推移は、総量で前年比5.6%増となりました。内訳としてはIPv6が25.2%増、IPv4は0.6%増でした。昨年はIPv6、IPv4共に横ばいでしたが、2025年はIPv6が顕著な伸びとなりました。

図-6に、2025年2月1日を100とした指数化グラフを示します。IPv4は100近辺で推移しているのに対し、IPv6は100から140の間、おおむね120あたりで推移していることが見て取れます。

次に、トラフィック全体に占めるIPv6の比率を図-7に示します。期間中は最小20.5%から最大26.9%で推移し、平均は23.9%でした。昨年と比較すると約4ポイント増加しており、昨年は停滞していたため今年もあまり伸びないと予想していましたが、良い意味でその予想を裏切る結果となりました。

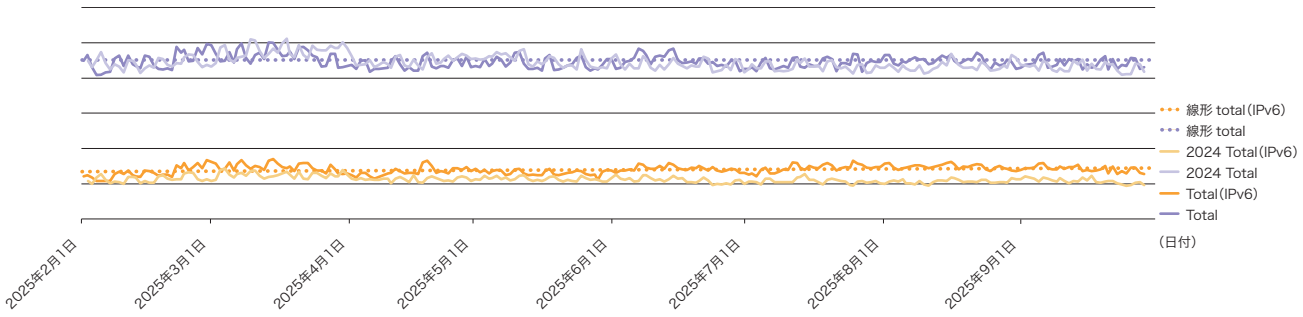


図-5 IIJコアPOPのバックボーンルータで計測したトラフィック

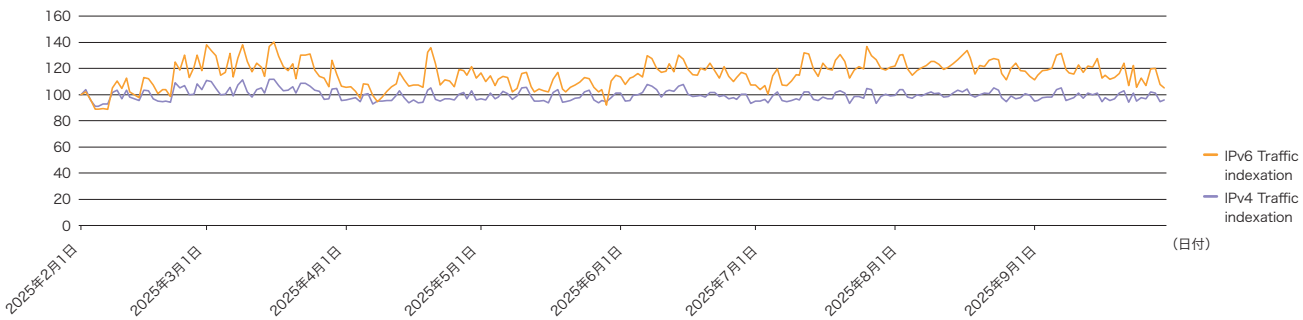


図-6 2025年2月1日を100としたときの変動状況

表-5に2017年からのIPv6比率の推移を示します。定点観測を始めた当初は4%程度しかなかったIPv6トラフィックですが、全体の約24%を占めるほどに利用が拡大してきました。

■ 送信元組織(BGP Source AS)

2025年2月1日から2025年9月30日までの、IPv6とIPv4のトラフィック送信元組織(BGP Source AS番号)の上位を図-8と図-9に示します。

IPv6では、IIJ内部(AS2497<=>AS2497)の通信が69%を占めます。昨年は66%だったので、3ポイントほど伸長しています。

IIJ以外のASについて見てみると、昨年2位だった日本の大手コンテンツ事業者A社が、昨年首位の米検索大手B社を抜きトラフィック比率6%で1位となりました。2位は逆転されたB社で4%、3位は米通販及びクラウドサービス大手のC社で2%でした。その他の顔触れはあまり変わっていませんが、昨年同様飛び抜けて多いところがあるわけではなく、順位は観測時期により変わってくるものと思います。

なお、IPv6で1位のA社ですが、IPv4では5位に位置していますので、IPv6対応サービスを積極的に構築・提供しているのだろうと想像します。

表-5 IPv6比率の推移(2017年～)

	2017年 IIR Vol.37	2018年 IIR Vol.41	2019年 IIR Vol.45	2020年 IIR Vol.49	2021年 IIR Vol.53	2022年 IIR Vol.57	2023年 IIR Vol.61	2024年 IIR Vol.65	2025年 IIR Vol.68
IPv6比率	4%	6%	10%	10%	11.2%	15.1%	20.1%	20.16%	23.9%

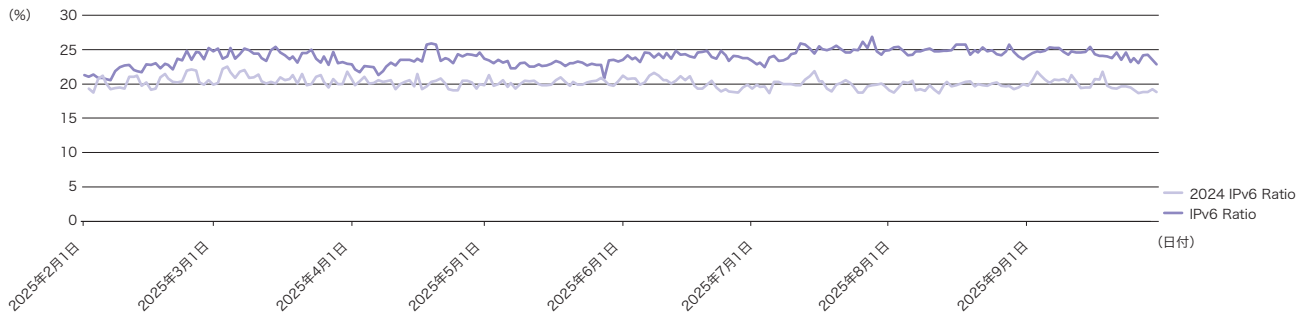


図-7 トラフィック全体に占めるIPv6の比率

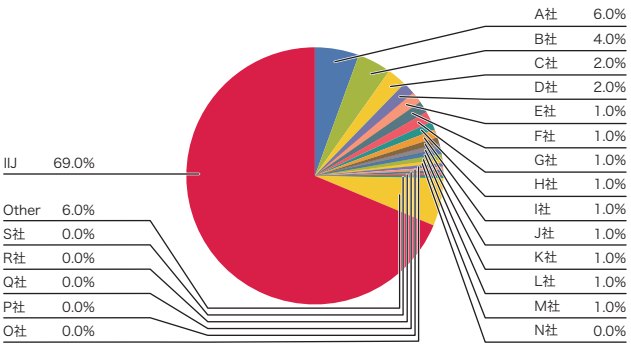


図-8 IPv6のトラフィック送信元組織(BGP Source AS番号)

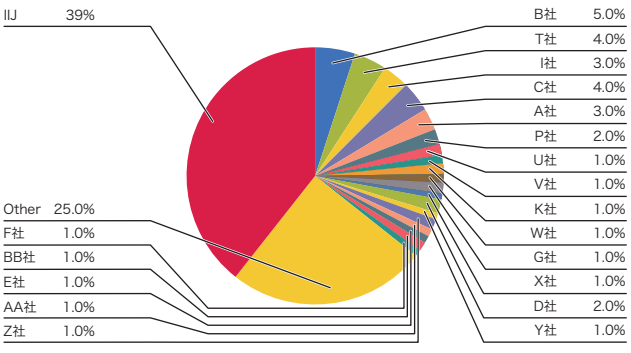


図-9 IPv4のトラフィック送信元組織(BGP Source AS番号)

■ 利用プロトコル

IPv6トラフィックのProtocol番号(Next-Header)と送信元ポート番号で解析したグラフを図-10に、IPv4トラフィックのProtocol番号と送信元ポート番号のグラフを図-11に示します。期間は2025年9月29日(月)から10月5日(日)の1週間です。

IPv6は、昨年と同じくHTTPS、QUIC、NAT Traversal、ESPと続き、これら上位4つのプロトコルで利用率92.4%を占めています。なお、HTTPSは76.3% (+2.3昨年比)、QUICは8.7% (-0.3)、HTTP 0.8% (-0.2)となっており、HTTP系プロトコルで約85.9% (+1.9)と着実に増加していると共に、暗号化の実装が進んでいることが見て取れます。

IPv4のHTTP系プロトコルは、HTTPS 55.8% (+0.6)、QUIC 6.7% (+1.0)、HTTP 4.2% (-1.1)で、HTTP系プロトコル合計で66.8% (+0.6)となっています。こちらもIPv6同様に暗号化の実装が進んでいるようです。IPv6と比較すると、HTTP系プロトコルの割合の低さ、otherに分類されるトラフィックの多さから、IPv4の方が利用方法(プロトコル)のバリエーションが多く、IPv6対応できないアプリやサーバが多いと推察します。

トラフィック傾向は昨年と大きく変わらず、IPv6は夜間の利用が多く、日中と比較すると倍近く利用されているように見えます。ただ、週末になると日中の利用が増加し、夜間利用が若干減少しているようです。IPv4と比較すると、山の形が大きく異

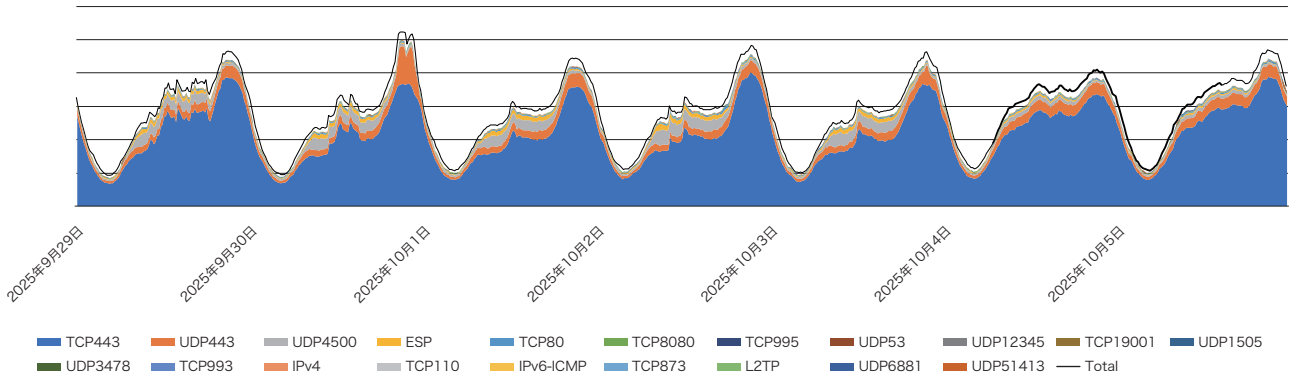


図-10 IPv6トラフィックの送信元ポート解析

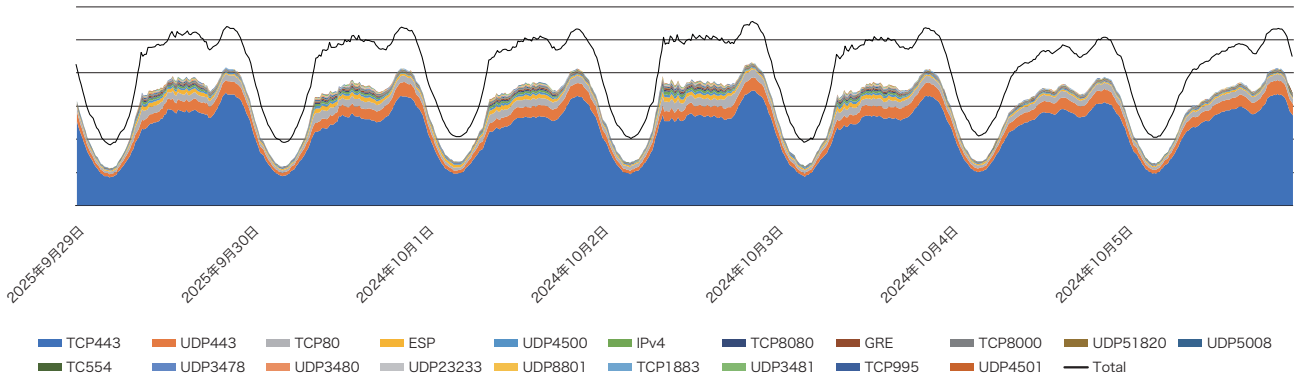


図-11 IPv4トラフィックの送信元ポート解析

なり、IPv4は人間が活動する時間帯にまんべんなく利用されていますが、IPv6はプライベート時間帯（勤務時間外）により多く利用されているように見受けられます。

■ モバイルのIPv6接続状況

今回も個人向けモバイルサービス（IIJmioモバイルサービス）の接続におけるIPv6有効化率を調査します。また、端末OS種別による違いと端末メーカによる違いの有無も見てみることにします。

IIJmioモバイルサービスに接続している端末のIPv6有効化率は62.9%でした。昨年は60.6%、一昨年は58.73%でしたので、毎年2ポイント程度の増加となっています。端末OS別に見ると、Apple iOS（iPadOSなどAppleの他のモバイル向けOSを含む）のIPv6有効化率は86.9%、AndroidのIPv6有効化率は35.5%でした。AndroidのIPv6有効化率は2年連続で、昨年比+5ポイント上昇していて、全体のIPv6有効化率増加に寄与しています。

次に、IIJmioモバイルサービスでIPv6が有効になっている接続について、接続数の多い順にメーカ別に見てみます。図-12の円グラフですが、Apple（iPhone、iPadなど）の接続数が大半を占め、74%です。次いでGoogle Pixelが10.2%、3番目にモトローラが8.5%と健闘しています。日本メーカでは、FCNTやSharp、

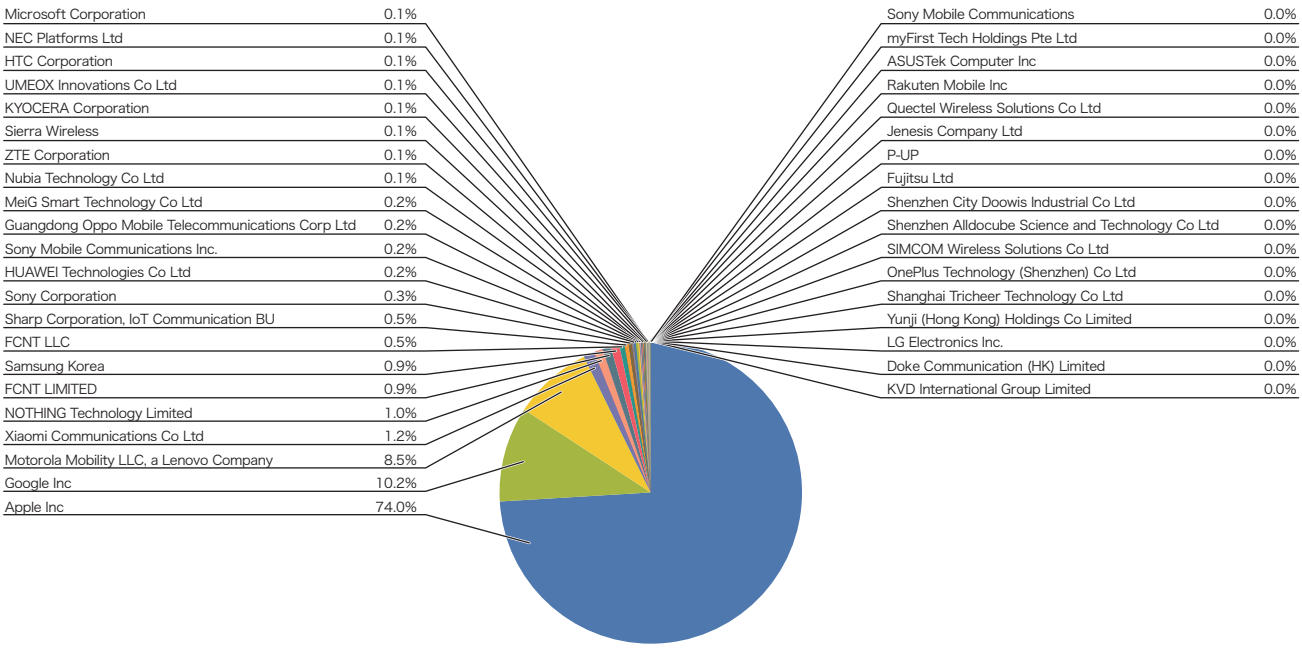


図-12 UEメーカIPv6有効化状況

Sony、Kyocera、NEC Platformsなどがありますが、接続数はそれぞれ少なく、またIPv6が有効になっていない端末も多いので、それぞれ1%を割る状況となっています。昨年の記事でFCNTのarrows We2はデフォルトでIPv6が有効になっているようだと報告しましたが、今後更に多くの機種でIPv6標準対応が進むことを期待します。

■ まとめ

- 2025年のIIJバックボーンにおけるIPv6トラフィックは前年比25.2%増と大きく伸長し、全体のトラフィック量も前年比5.6%増加しました。特にIPv6の成長が顕著であり、インターネット基盤のIPv6化が着実に進んでいることが示されました。
- トラフィック全体に占めるIPv6比率は平均23.9%となり、過去最高を記録しました。2017年の4%から約8年で約24%まで拡大しており、IPv6が定着しつつあります。
- 送信元AS番号別では、IIJ内部通信を除くと、A社が6%で1位、B社が4%で2位、C社が2%で3位となりました。A社はIPv6対応サービスの積極展開がうかがえます。
- プロトコル別ではHTTPSが主流で、IPv6トラフィックの76.3%を占めています。QUICやNAT Traversal、ESPも含め、暗号化HTTP系プロトコルとVPNプロトコルで9割以上を占めており、セキュアな通信が標準となっています。
- モバイル端末のIPv6有効化率は62.9%で、iOS（iPadOS含む）端末が86.9%、Android端末が35.5%と、Androidの伸びが全体の上昇に寄与しています。メーカ別ではAppleが圧倒的多数を占めますが、Android陣営ではGoogle Pixelやモトローラ製端末が健闘しています。

Theme 04

インターネットバックボーンのトレンド

IIJのインターネットバックボーンインフラの相互接続におけるインタフェースの状況と、RPKIに関するトレンドを紹介します。

■ 相互接続インタフェースのトレンドと要件

インターネットにおける相互接続を行う上では、インタフェース種別を事業者間で統一させる必要があります。2025年10月現在、IIJでは相互接続のインタフェースとして400G-FR4、100G-LR4及び10G-LRを主に利用して相互接続を行っています。ここ数年のトレンドとして見られる10Gを使った事業者間の相互接続を見直す動きは今年も続いています。相互接続のインタフェースを決定する要素は、トラフィックの流量やお互いのインタフェースの保有数に依存しており、事業者間で条件の合意が形成されれば100Gへ移行しているといった状況が継続しています。

図-13、図-14は、現状のIIJバックボーンにおける相互接続のインタフェースの割合です。2024年のデータについては2025年の抽出条件に合わせてデータを作成しています。本誌IIR Vol.65（https://www.iij.ad.jp/dev/report/iir/065/01.html#anc04）と割合が異なることに注意してください。

400Gインタフェースの利用は相互接続においてはまだ限定的です。グラフ上では100G/10Gの数量が増えたため、0%へ繰り下げとなってしまっていますが、実は少しだけ存在しています。10Gインタフェースは全体の個数から減少が見て取れます。これはここ1年でInternet Exchange Point（IXP）のインタフェースを10Gから100Gへ移行したことが大きな要因だと考えられます。

10Gでは、Ling Aggregation（LAG）で接続していたインタフェースは、機器の更改に合わせて100Gへ移行しています。併せて、機器のアップグレード後に事業者間で100Gが用意できたところは100G化を進めています。増強後の10Gインタ

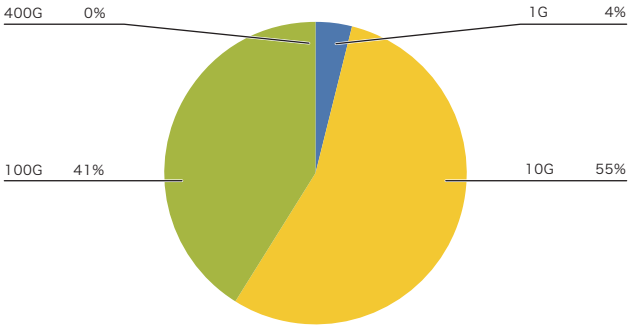


図-13 IIJインターネットバックボーンにおける相互接続インタフェースの割合（2024年10月）

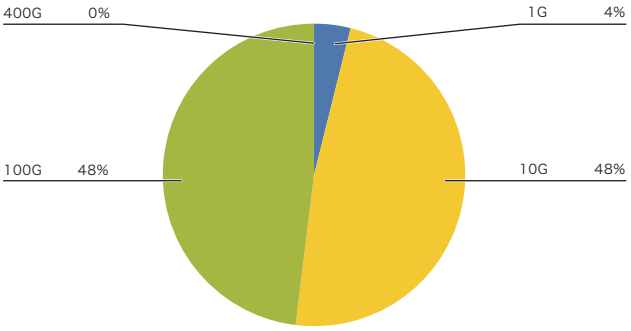


図-14 IIJインターネットバックボーンにおける相互接続インタフェースの割合（2025年10月）

フェースについては、100GインタフェースからMPOケーブルを使ったBreakoutにより物理インタフェースを束ねて提供することが多くなりました。機器によっては単独の10Gインタフェースを持たず、高速インタフェースから低速インタフェースへ分割して利用する機能を搭載しています。IIJでも機器更改後は、10Gインタフェースの利用を継続する場合、高速インタフェースから低速インタフェースへ分割して利用する機能を導入する機会が増えてきました。

100Gの割合も引き続き増えており、現状の相互接続のインタフェースにおいて主力となっています。100Gインタフェースの種別は100G-LR4を引き続き利用しています。100G-LRのシングルラムダは現状導入には至っていません。

■ RPKI関連の状況

RPKI関連の状況について確認してみましょう。自組織が保有するIPアドレスの正当性を署名として登録されるROAの状況を見てみます。ROAの登録情報を把握するために、IIJ Labで保有するRPKI ROVによるインターネット経路の検証結果データから解析を行いました(図-15、図-16)。インターネット上の全IPv4経路におけるValid(ROAが登録され、経路検証済)の割合が57.18%、Not-Found (ROAの登録がまだ実施されていないもの)の割合が42.68%、Invalid (ROAの登録状況に差異があり、不正経路と扱われるもの)の割合が0.14%でした。IPv6においてはValidが63.32%、Not-Foundが36.25%、Invalidが0.44%でした。

IPv4、IPv6共にValidの割合が2024年10月よりも増加していることが分かります。更にNot-Found、Invalidの割合が少なくなっているため、IPアドレスのROA発行が進んだと推測されます。特にInvalidの割合がかなり改善しているためここ1年でインターネット上の経路のROA見直しが進んでいる点は評価できます。

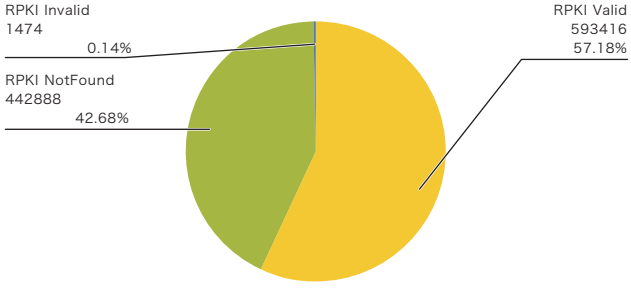


図-15 RPKI MonitorによるROAの当登録状況(IPv4)2025年10月

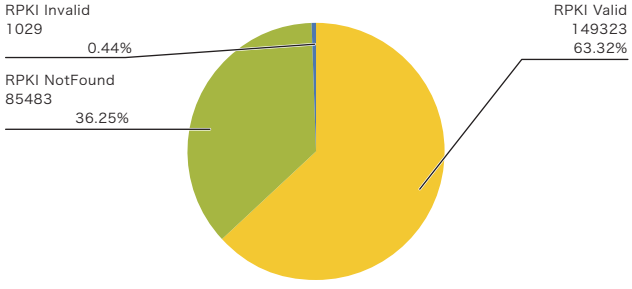


図-16 RPKI MonitorによるROAの当登録状況(IPv6)2025年10月

一方、IIJが生成・広告しているインターネット経路のROA登録状況を見てみましょう（表-6）。IIJはAS2497というGlobal ASを保有してインターネットへ参加しており、IIJから経路広告される場合のOriginASはAS2497となります。2025年10月現在において、AS2497がOriginとなる経路のROA登録状況としては、Validの割合が43.65%となります。全体の半分弱においてROA登録されており、経路の正当性が担保されている状況です。一方で、ROAが登録されていないNot-FoundとしてROV判定されている経路は50%以上存在します。ROA登録の敷居が高い理由としては、IPアドレスを保有する組織が自らROA登録、発行する必要があるからと言えます。IIJが保有するIPアドレスは特殊な場合を除いてROA登録はほぼ完了している状況ですが、IIJへIPアドレスを持ち込まれているユーザにおいてはまだROA登録が進んでいないことが確認できます。IIJでの代行登録ができない現状ではユーザ自身に登録してもらう必要があります。IIJがサポートしますので、ぜひAS2497のROA登録率を上げていけるよう、引き続きご協力をお願いいたします。

また、IIJがお客様の経路をトランジットしている割合を含めると、全経路の56.72%はROVでValidとして判定されています。この数値は昨年度の統計結果よりも増えており、日本国内においてもROAの発行が進んでいることが改めて確認できます。

表-6 IIJが生成・広告しているインターネット経路のROA登録状況

	Origin AS2497である経路数	IIJがTransitし、インターネットへ広告している経路数
Valid	79	4405
Unknown	102	3757
Invalid	0	4
Valid rate	43.65%	56.72%

執筆者：

1.BGP・経路数
倉橋 智彦（くらはし ともひこ）
IIJ ネットワークサービス事業本部 基盤エンジニアリング本部 運用技術部 技術開発課

2.DNSクエリ解析
松崎 吉伸（まつざき よしのぶ）
IIJ ネットワークサービス事業本部 基盤エンジニアリング本部 運用技術部 技術開発課

3.IPv6＆モバイル
佐々木 泰介（ささき たいすけ）
IIJ モバイルサービス事業本部 MVNO事業部 基盤開発部

4.インターネットバックボーン
蓬田 裕一（よもぎた ゆういち）
IIJ ネットワークサービス事業本部 基盤エンジニアリング本部 ネットワーク技術部 ネットワーク技術1課

DNSフルリゾルバbowlineの設計と実装

IIJでは、DNSフルリゾルバ(別名キャッシュDNSサーバ、以下単にフルリゾルバ)を独自に開発しています。ソフトウェアの名前はbowlineです。「結びの王様」である「もやい結び」から名付けました。現時点では、ロギングやモニタリングなどISPでの運用に必要な機能のほとんどを実装し終えており、試験的な運用を通じて安定性を検証しています。この記事では、bowlineの設計と実装について述べます。

IIJは、フルリゾルバの複数の実装を用いて、キャッシュDNSサーバを提供しています。bowlineの狙いの1つは、独立した実装を増やし、攻撃に対して耐性を向上させることです。また、IIJが完全にコントロールできることもポイントです。コントロール下にあれば、新たな攻撃方法に対する対策を素早く実装し、稼働しているサーバを置き換えることが可能なはずです。

bowlineはオープンソースとして公開しており、実装言語はHaskellです。Haskellを利用する理由は、「HaskellによるQUICの実装」[*1]の3.2節「Haskellで実装する理由」を参照してください。一番大切なのは、Haskellでは軽量スレッド(以下単にスレッド)が提供されているので、イベント駆動プログラミングに比べて、ソフトウェアを柔軟かつ見通しよく構成できることです。

2.1 開発の経緯とライブラリ群の構成

著者は、2010年にアンチスパムの取り組みとして、SPF、Sender ID、DomainKeys、及びDKIMを統合するフレームワークをHaskellで実装しました。これらの技術を利用するには、DNSを検索する機能が必須です。当初は、C言語で書かれた有名なDNSスタブリゾルバ・ライブラリを他言語関数インタフェース(FFI)を通じて使用していましたが、Haskellの高度な並行処理の下では、表明違反が多発し、うまく動かないことが判明しました。

そこで、このライブラリの利用を諦め、完全にHaskellのみで書かれたDNSスタブリゾルバ・ライブラリ(名称はdns)を開発しました。すべてをHaskellで書けば、言語の特性から、高度な並行性は自動的に実現されます。事実、その実用性が複数のインターネット・サービスを通じて実証されました。

bowlineの開発は、同僚の日比野と共に2022年から始めました。フルリゾルバの主要な機能である反復検索、キャッシュ、及びDNSSEC検証は日比野が担当しました。著者は2013年からHaskellで、HTTP/2、TLS 1.3、QUIC、及びHTTP/3のライブラリを開発しており、それらをDNSに応用することに興味があったので、主にトランスポートを開発しました。

bowlineの原型ではdnsライブラリを利用していましたが、これまでの経験から、dnsライブラリには拡張性がなく、またメモリが断片化する問題のあることが分かっており、これらの欠点を下位互換性を気にすることなく解決できるよう、新たにDNSライブラリ群を開発することにしました。機能ごとに分割されたライブラリ群は、すべてdnsextという接頭辞から始まり、それぞれ以下のような機能を提供しています。

- dnsext-types:拡張可能で断片化しない基本データ型及び基本RR(Resource Record)の符号器/復号器
- dnsext-dnssec:DNSSEC関連のRRの符号器/復号器、DNSSECの検証器
- dnsext-svcb:最近仕様が決まったSVCB(Service Binding)RRの符号器/復号器
- dnsext-utils:ログやキャッシュなどユーティリティ関数
- dnsext-do53:クライアント側のDNS over UDP、TCP
- dnsext-dox:クライアント側のDNS over HTTP/2、HTTP/3、TLS、QUIC
- dnsext-iterative:反復検索アルゴリズム、サーバ側のDNS over UDP、TCP、HTTP/2、HTTP/3、TLS、QUIC
- dnsext-bowline:フルリゾルバbowline、DNS検索コマンドdug[*2]、暗号化されたDNSサーバの探索を実現するデーモンddrd[*3]

dnsext-dnssecやdnsext-svcbは、dnsext-typesが拡張できることの例となっています。

＊1 Internet Infrastructure Review(IIR)Vol.52、「HaskellによるQUICの実装」(https://www.iij.ad.jp/dev/report/iir/052/03.html)。
＊2 IIJ Engineers Blog:「DNS検索コマンドdugの紹介」(https://eng-blog.iij.ad.jp/archives/27527)。
＊3 IIJ Engineers Blog:「暗号化されたDNSサーバの探索」(https://eng-blog.iij.ad.jp/archives/31843)。

2.2 スレッドの構成

一般的に、フルリゾルバは、以下のように動作することが期待されています。

- スタブリゾルバから、再帰検索(Recursion Desiredフラグがオン)の要求を受け取る
- 要求に対してキャッシュを検索し、「肯定応答」あるいは「否定応答」が存在すれば、スタブリゾルバに返す
- 存在しなければ、権威サーバに対して反復検索(Recursion Desiredフラグがオフ)を繰り返し実行し、肯定応答あるいは否定応答を得てスタブリゾルバに返し、新たなエントリをキャッシュに登録する

ネットワークを利用する反復検索は、メモリ操作であるキャッシュの検索よりも圧倒的に時間がかかります。そこで、反復検索が他の要求や応答の処理に悪影響を及ぼさないようにソフトウェアを設計する必要があります。

歴史的にDNSのトランスポートには主にUDPが使われており、同一スタブリゾルバから複数の要求を受けた場合、解決できたRRから応答を返します。一方、コネクション型のトランスポートでは、スタブリゾルバが送った順番どおりに、フルリゾルバが要求を受け取ります。応答を順番どおりに返そうとすると、ある要求に対する反復検索が後続の要求をブロックしてしまうことがあります。これを防止するために、コネクション型のトランスポートでは、先に解決できた要求に対する応答から返すことが求められています(パイプライニング)。つまり、コネクション型のトランスポートでも、UDPと同じように振る舞う必要があるのです。

仮に、1つのスレッドがキャッシュと反復検索の両方を担当するように設計したとしましょう。このスレッドは、反復検索でブロックされる可能性があります。そのため、後続の要求を滞りなく処理するためには、要求ごとに、このスレッドを生成する必要があることが分かります。このような設計では、主機能を司るこのスレッドの数が膨大になる可能性があり、脆弱です。

そこで、キャッシュを検索するスレッド(以下、キャッシュ検索器)と反復検索をするスレッド(以下、反復検索器)を別々に用意することにしました。キャッシュ検索器は、キャッシュの検索に失敗した場合、反復検索を反復検索器に任せます。キャッシュ検索器はブロックされませんが、反復検索器はブロックされる可能性があります。キャッシュ検索器と反復検索器は、サーバの起動時に固定数が生成されます。反復検索器の数がキャッシュ検索器の数よりも十分に大きければ、スムーズなパイプライニングが実現できます。

以上の考察の下に設計したスレッドの構成を図-1に示します。二点鎖線の四角がbowline全体を表し、灰色の四角がスレッドを表現しています。トランスポートを担当するのは受信器と送信器です。この組は、UDPに対してはネットワーク・インタフェースの数だけ常駐します。コネクション指向のトランス

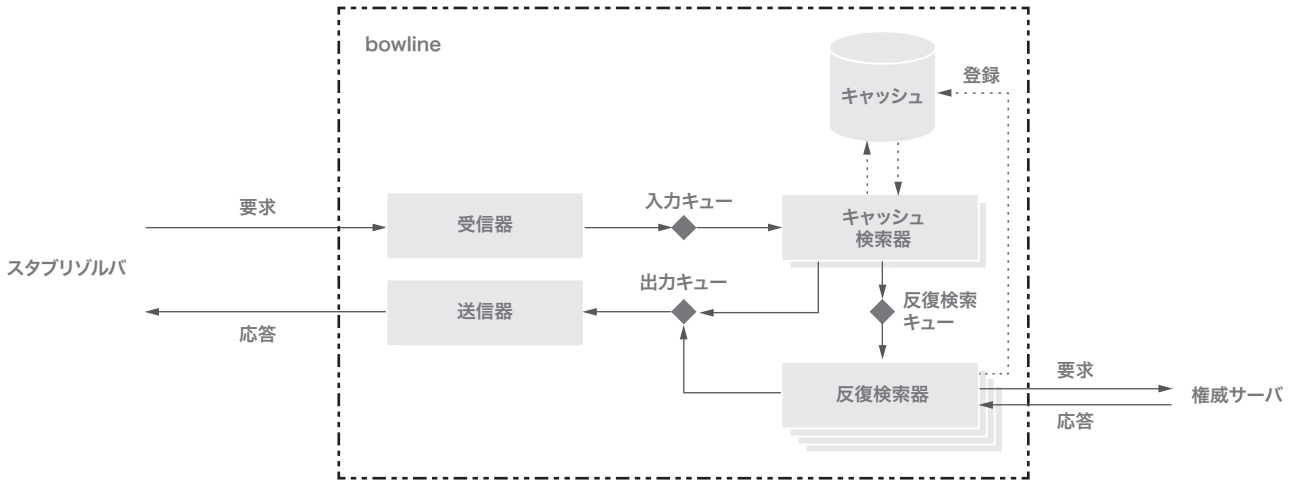


図-1 bowlineのスレッドの構成

ポートでは、ネットワーク・インタフェースの数だけ待ち受けスレッドが常駐し、コネクションが作成されるたびに受信器と送信器の組が生成されます。HTTP/2、QUIC、HTTP/3に関しては、補助的なスレッドも起動されます。

受信器は、スタブリゾルバからのDNS要求を復号し、InputというHaskellのデータ型に変換後、グローバルな入力キューに入れます。Inputにはその受信器に対応する送信器の出力キューへの参照が格納されています。

キャッシュ検索器は、グローバルな入力キューからInputを受け取り、キャッシュを検索します。この動作はブロックされません。検索に成功すれば、結果をOutputというデータ型で表現して、Inputから参照されている出力キューへ格納します。失敗すれば、グローバルの反復検索キューにInputをリレーします。bowlineのデフォルトでは、キャッシュ検索器の数は4個です。

反復検索器は、反復検索キューからInputを読み出し、反復検索を試みます。この動作はブロックされる可能性があります。すべての反復検索器がブロックされると反復検索の機能全体がブロックされるので、反復検索器の数は十分に大きくする必要があります。反復検索器は、反復検索で得られた結果をOutputで表現し、出力キューに格納します。bowlineのデフォルトでは、反復検索器の数は128個です。

送信器は、自分用の出力キューからOutputを読み出し、DNS応答へ符号化した後、スタブリゾルバへ送信します。

2.3 反復検索アルゴリズム

DNSの権威サーバは、クライアントからの要求に対して以下のように振る舞います。

- 自分が管理するゾーンのドメイン名であり、問い合わせRR型のRRが存在すれば、その値を返す
- 自分が管理するゾーンのドメイン名であり、問い合わせ名が存在しない、あるいは問い合わせRR型のRRが存在しない場合は、SOA RRを返して、否定応答をキャッシュするためのTTLを提示する
- 自分が管理するゾーンのドメイン名であり、下位ゾーンの委任情報が存在するなら、リゾルバが反復検索を続けられるように、下位ゾーンの委任情報（NS RR）やグルー（A RRやAAAA RR）を返す

反復検索では、フルリゾルバは、権威サーバに対するクライアントとして振る舞います。RFC 1034で定義される反復検索では、それぞれの問い合わせ名と問い合わせRR型は、スタブリゾルバが指定した値に固定されていました。

例えば、スタブリゾルバからwww.example.jpのTXT RRの解決を依頼されたとします。フルリゾルバは、最終的に"example.jp."の権威サーバにwww.example.jpのTXT RRを問い合わせるのと同様に、ルート（"."）の権威サーバや、中間の"jp."の権威サーバにも同じ要求を使います。

この仕組みは、インターネットの盗聴者に対して、オリジナルの問い合わせ名や問い合わせRR型を盗聴させる機会を多く与えていると言えます。そこで、プライバシ保護の観点から、「問い合わせ名最小化」が提案されました。問い合わせ名最小化では、オリジナルのドメイン名の必要な部分のみを用いて権威サーバに問い合わせます。また、中間で利用する問い合わせRR型には、RFC 7816ではNS RR、RFC 9156ではA RRかAAAA RRを用います。

表-1に、www.example.jp.のTXT RRを解決する際の問い合わせ名と問い合わせRR型をまとめます。bowlineは、反復検索としてRFC 9156で定義される「問い合わせ名最小化」のみを用います。表-1のRFC 9156の列を以下で詳細に説明します。

1. "."の権威サーバに"jp."のA RRを問い合わせ、"jp."の権威サーバを得る。
2. "jp."の権威サーバに、"example.jp."のA RRを問い合わ

せ、"example.jp."の権威サーバを得る。
3. "example.jp."の権威サーバに、"www.example.jp."のARRを問い合わせると、答えが得られるため、更なる委任がないことが分かる。
4. "example.jp."の権威サーバに、"www.example.jp."のTXT RRを問い合わせる。

最後に冗長に思える検索が起こるのは、オリジナルの問い合わせRR型を隠しながら、委任がないかを調べるためです。オリジナルの問い合わせRR型とアルゴリズムで用いられるRR型がたまたま一致した場合は、検索回数が1回減ります。

問い合わせ名が正常に存在するのであれば、どの段階でも得られる権威サーバ名は1つ以上あり、各々の権威サーバには1つ以上のA/AAAA RRが存在します。前述のように、グルーとしてA/AAAA RRが、NS RRと一緒に返される場合があります。

2.3.1 反復検索の実装

bowlineがスタブリゾルバから要求を受け取り、クライアントとして権威サーバに対して反復検索を実行する際は、まずルート・プライミングを実行します。ルート・プライミングとは、あらかじめ組み込まれている"."の権威サーバの候補(ヒント情報)に対して、"."の最新のNS RRを問い合わせ、A/AAAA RRも同時に解決することです。この結果は、キャッシュされます。よって有効期限内であれば、これ以降の要求からはキャッシュの値が利用されます。

この特殊な処理が完了すれば、以降は最終の答えを得るまで、以下のステップを繰り返します。まず、IPアドレスが分かっている権威サーバと、分かっていない権威サーバのグループを作ります。

次に、IPアドレスが分かっている権威サーバに対し、IPアドレスをランダムに並べ替え、権威サーバの名前が重ならないように2つを取り出します。そして、問い合わせ名最小化のアルゴリズムに従った要求を、権威サーバに並列実行で問い合わせます。

我々の実装では、このクライアントの機能はdnsext-do53ライブラリで提供されており、複数のサーバに対して問い合わせスレッドをそれぞれ生成して競争させ、一番最初に返ってきた応答を採用することができます。

問い合わせに失敗した場合は、次の2つの候補に移ります。IPアドレスが分かっている権威サーバの問い合わせにすべて失敗した場合は、IPアドレスが分かってない権威サーバに移ります。この段階では、ランダムに並べ替えた権威サーバの先頭1つに対して、ランダムにA RRかAAAA RRを選んで、権威サーバの名前に対する新規の反復検索を実行します。解決できなければ、次の名前に進みます。解決できたら、そのIPアドレスの1つに対して、このステップの目的の問い合わせを実行します。

すべてが失敗したら、全体として検索の失敗なので、スタブリゾルバにエラーを返します。いずれかが成功した場合、完全マッチで対象のRRが手に入れば、そこで反復検索は完了です。そうでなければ、下位の権威サーバの名前が手に入っているので、一段長い問い合わせ名を用いて、このステップを繰り返します。

	権威サーバ	RFC 1034	RFC 7816	RFC 9156
1	.	www.example.jp. TXT	jp. NS	jp. A
2	jp.	www.example.jp. TXT	example.jp. NS	example.jp. A
3	example.jp.	www.example.jp. TXT	www.example.jp. NS	www.example.jp. A
4	example.jp.		www.example.jp. TXT	www.example.jp. TXT

表-1 問い合わせ名最小化の例

2.3.2 反復検索のエラーケース

NODATAは、ドメイン名は存在するがRR型に該当する値がなく（他のRR型の値は存在する）、次に問い合わせるべき権威サーバもないエラーです。問い合わせ名全体を渡すRFC 1034のアルゴリズムでは、NODATAが返れば、それが最終結果です。しかし、問い合わせ名最小化のアルゴリズムでは、問い合わせ名を長くして検索を続ける必要があります。例えば、".jp."の権威サーバに、"ad.jp."を問い合わせるとNODATAですが、"iij.ad.jp."という問い合わせに対しては権威サーバの情報が返ってきます。これは、"jp."と"ad.jp."が同一のゾーンであるためです。

NXDOMAINは、ドメイン名が存在しない場合のエラーです。ある中間ドメインがNXDOMAINとなれば、RFC 8020が定める仕様としては、それより以下にドメインが存在しません。しかしながら現実的には、下位のドメインを検索するとドメインが存在する場合があります。このためbowlineでは、中間ドメインがNXDOMAINとなってもドメイン名を長くしながら検索を続けます。最終的に、問い合わせ名がNXDOMAINとなった場合にのみ、このエラーと判定します。

SERVFAILはサーバの障害、REFUSEDは何らかの理由で問い合わせが拒否されたこと、FORMERRは書式違いを表すエラーです。これらのエラーが起こった場合、次のIPアドレスの候補にフォールバックして検索を続けます。

2.3.3 dugによる可視化

前述のdugは、2つのモードを持つDNS検索コマンドです。一方のモードは、単なるスタブリゾルバとして、フルリゾルバに再帰検索を依頼します。権威サーバに対して、Recursion Desiredフラグをオフにして問い合わせることもできます。

他方のモードは、bowlineと同じ反復検索の実装を使って、その様子を可視化します。右カラムの枠内は、dugを使った反復検索の例です。-iが反復検索モード、-vvが表示量を増やすオプション、+cdflag（check disabled）がDNSSECの検証を止めるフラグです。

```
% dug -i -vv iij.ad.jp. txt +cdflag
...
root-priming: query "." NS
    query @2001:7fe::53#53/UDP "." NS
    query @198.97.190.53#53/UDP "." NS
    query @2001:7fe::53#53/UDP "." NS: win
root-priming: verification success - RRSIG of NS: "."
    "a.root-servers.net." 198.41.0.4#53 2001:503:ba3e::2:30#53
    "b.root-servers.net." 170.247.170.2#53 2801:1b8:10::b#53
    "c.root-servers.net." 192.33.4.12#53 2001:500:2::c#53
    "d.root-servers.net." 199.7.91.13#53 2001:500:2d::d#53 (※1)
    "e.root-servers.net." 192.203.230.10#53 2001:500:a8::e#53
    "f.root-servers.net." 192.5.5.241#53 2001:500:2f::f#53
    "g.root-servers.net." 192.112.36.4#53 2001:500:12::d0d#53
    "h.root-servers.net." 198.97.190.53#53 (※2) 2001:500:1::53#53
    "i.root-servers.net." 192.36.148.17#53 2001:7fe::53#53
    "j.root-servers.net." 192.58.128.30#53 2001:503:c27::2:30#53
    "k.root-servers.net." 193.0.14.129#53 2001:7fd::1#53
    "l.root-servers.net." 199.7.83.42#53 2001:500:9f::42#53
    "m.root-servers.net." 202.12.27.33#53 2001:dc3::35#53
iterative: query "jp." A
    query @2001:500:2d::d#53/UDP "jp." A (※1)
    query @198.97.190.53#53/UDP "jp." A (※2)
    query @2001:500:2d::d#53/UDP "jp." A: win
delegation - no DS, check disabled: "." -> "jp."
zone: "jp.":
    "a.dns.jp." 203.119.1.1#53 2001:dc4::1#53
    "b.dns.jp." 202.12.30.131#53 2001:dc2::1#53 (※4)
    "c.dns.jp." 156.154.100.5#53 (※5) 2001:502:ad09::5#53
    "d.dns.jp." 210.138.175.244#53 2001:240::53#53
    "e.dns.jp." 192.50.43.53#53 2001:200:c000::35#53
    "f.dns.jp." 150.100.6.8#53 (※3) (※6) 2001:2f8:0:100::153#53
    "g.dns.jp." 203.119.40.1#53
    "h.dns.jp." 161.232.72.25#53 2a01:8840:1bc::25#53
iterative: query "ad.jp." A
    query @150.100.6.8#53/UDP "ad.jp." A (※3)
    query @2001:dc2::1#53/UDP "ad.jp." A (※4)
    query @150.100.6.8#53/UDP "ad.jp." A: win
delegation - no delegation: "jp." -> "ad.jp."
cache-soa: no verification - check-disabled: "jp."
iterative: query "iij.ad.jp." A
    query @156.154.100.5#53/UDP "iij.ad.jp." A (※5)
    query @150.100.6.8#53/UDP "iij.ad.jp." A (※6)
    query @156.154.100.5#53/UDP "iij.ad.jp." A: win
delegation - no DS, check disabled: "jp." -> "iij.ad.jp."
zone: "iij.ad.jp.":
    "dns0.iij.ad.jp." 210.130.0.5#53 2001:240::105#53 (※8)
    "dns1.iij.ad.jp." 210.130.1.5#53 (※7) 2001:240::115#53
resolve-exact: query "iij.ad.jp." TXT
    query @210.130.1.5#53/UDP "iij.ad.jp." TXT (※7)
    query @2001:240::105#53/UDP "iij.ad.jp." TXT (※8)
    query @2001:240::105#53/UDP "iij.ad.jp." TXT: win
no verification - check-disabled: "iij.ad.jp."
;; HEADER SECTION:
;Standard query, NoError, id: 0
;Flags: Recursion Available

;; QUESTION SECTION:
;iij.ad.jp.              IN          TXT

;; ANSWER SECTION:
iij.ad.jp.   3600(1 hour) IN          TXT
"20f10da4-fb66-42ac-941e-133d9c6c09ba"
iij.ad.jp.   3600(1 hour) IN          TXT
"v=spf1 include:spf.iij.ad.jp include:spf.dox.jp -all"
iij.ad.jp.   3600(1 hour) IN          TXT
"_globalsign-domain-verification=qSNEOr9rmlIgx-CLJgTN-uypossXQTKD4GuWN-Jwp0"

;; AUTHORITY SECTION:

;; ADDITIONAL SECTION:

;; 172usec
```

候補の権威サーバのIPアドレスのうち、実際に利用されたものは"(※番号)"を加筆しています。"win"は2つの競争する検索のうち、どちらが勝ったかを表します。委任があればIPアドレスが一覧表示され、そうでなければ"no delegation"と表示されます。

2.4 DNSSECの反復検索アルゴリズム

DNSSECでは、公開鍵暗号技術の1つである電子署名を用いて、委任情報に対する認証の連鎖を構築します。あるゾーン内でのDNSSECに関連するRRは、以下の2つです。

- DNSKEY:そのゾーンから提供される電子署名(RRSIG)を検証するための公開鍵(DNSKEYには、鍵署名鍵とゾーン署名鍵が含まれますが、この記事では区別しません)
- RRSIG:そのゾーンが管理しているRRに対する電子署名

DNSKEY RRを使ってRRSIG RRを検証すれば、データに改ざんがないことが分かります。しかし、名乗っているドメイン名が本当に正当であるのかは分かりません。このため、上位のドメインから委任されていることを証明する必要があります。そこで、以下のRRが用意されています。

- DS:あるゾーンのDNSKEYに対する暗号学的ハッシュ値を上位のゾーンに登録するためのRR

DO(DNSSEC OK)フラグがオンの反復検索の要求に対して、権威サーバは以下のように振る舞います。

- 下位ゾーンへの委任がある場合:NS RRなど委任情報を返す場合は、DS RRも返します
- 下位ゾーンへの委任がない場合:返すRRに対してRRSIG RRがあれば、それも返します

問い合わせ名最小化が相まった委任の検証は若干複雑なので、汎用的な解説ではなく、図-2を用いて、bowlineが"www.example.jp"のA RRをどのように解決するのか簡略的に説明します。なお、"."のDS RRは、トラストアンカーとして事前に提供されているとします。

- あらかじめ組み込まれている"."の権威サーバの候補に、"."のDNSKEY RR(b)を問い合わせ、トラストアンカーとして与えられた"."のハッシュ値(a)に合致するDNSKEY RRを選択します。これにはDNSKEY RRに対するRRSIG RR(c)も一緒に返されます。"."のDNSKEY RRに含まれている公開鍵で、RRSIG RRの中の署名を検証します。検証に成功すれば、"."のDNSKEY RRを信頼します

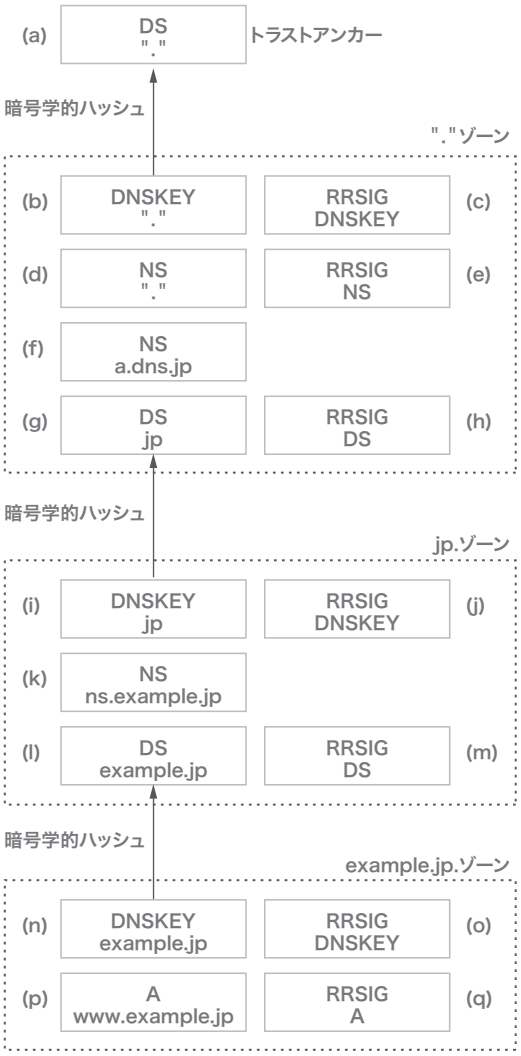


図-2 DNSSECの信頼の連鎖

- あらかじめ組み込まれている"."の権威サーバの候補に、"."のNS RR(d)を問い合わせ、"."の権威サーバの一覧を入手します。NS RRに付随するRRSIG RR(e)の署名を"."の公開鍵で検証し、"."の権威サーバを信頼します(ルート・プライミング)
- "."の権威サーバに、"jp."のA RRを問い合わせます。"jp."の権威サーバを示すNS RR(f)やA/AAAA RRが返ってきますが、これに対応するRRSIG RRはありません。"."ゾーンで管理している情報ではないからです。"jp."のDNSKEY RRのハッシュ値が格納されたDS RR(g)も返されます。これには、RRSIG RR(h)が付いています。"."の公開鍵で、この署名を検証します。成功すれば、"." → "jp."の委任を信頼します
- "jp."の権威サーバに、"jp."のDNSKEY RR(i)を問い合わせ、"jp."のハッシュ値に合致するDNSKEYを取り出します。応答には、"jp."のDNSKEY RRに加えて、DNSKEY RRのRRSIG RR(j)も含まれています。DNSKEY RRの中の"jp."の公開鍵を使って、この署名の検証に成功すれば、"jp."のDNSKEY RRを信頼します
- "jp."の権威サーバに、"example.jp."のA RRを問い合わせます。"example.jp."の権威サーバを示すNS RR(k)やA/AAAA RRが返ってきます。"example.jp."のDNSKEY RRのハッシュ値が格納されたDS RR(l)も返されます。これには、RRSIG RR(m)が付いています。"jp."の公開鍵で、この署名を検証します。成功すれば、"jp." → "example.jp."の委任を信頼します
- "example.jp."の権威サーバに、"example.jp."のDNSKEY RR(n)を問い合わせ、"example.jp."のハッシュ値に合致するDNSKEYを取り出します。これにはDNSKEYに対するRRSIG RR(o)が含まれています。"example.jp."のDNSKEYに含まれている公開鍵で、RRSIG RRの中の署名を検証します。検証に成功すれば、"example.jp."のDNSKEY RRを信頼します
- "example.jp."の権威サーバに、"www.example.jp."のA RR(p)を問い合わせると、RRSIG RR(q)も返ってきます。この署名を検証し、最終的な答えを信頼します

すべてのゾーンがDNSSECに対応しているわけではないので、認証の連鎖は途中で切れることがあります。bowlineでは、認証の連鎖が切れた場合、それ以降の反復検索ではDOフラグをオフにします。

2.4.1 親子同居問題

前述のように権威サーバは、下位ゾーンへの問い合わせに対して、検索が継続できるようにDS RRを返します。このように、DS RRを明示的に検索しなくても良い場合がほとんどです。しかしながら、認証の連鎖が切れてないにもかかわらず、DS RRが返らない場合があります。

それは、親のゾーンと子のゾーンが、同一の権威サーバで管理されているときです。例として、親のゾーンを"a."、子のゾーンを"b.a."、そしてこの2つのゾーンが同一の権威サーバで管理されているとします。

"a."の権威サーバに対して、"b.a."のA RRを検索したとしましょう。もし、親子が同居していなければ、"b.a."への委任情報、

つまりNS RRやDS RRが返ります。しかし、この場合は親子が同居しており、DNSの要求にはどのゾーンを対象としているのかという情報がないため、最長マッチが働き"b.a."のゾーンが対象であると解釈されます。A RRがあればそれが、なければSOA RRが返り、DS RRは得られません。

この場合、実際には委任が存在するが、DS RRは返ってこなかったことを判定し、明示的にDS RRを問い合わせる必要があります。bowlineで採用している判定方法は以下のとおりです。

- "b.a."に対するA RRが存在すると、A RRに加えて、RRSIG RRが返る。RRSIG RR内の署名者名フィールドが"b.a."であれば親子が同居している
- "b.a."に対するA RRが存在しないと、SOA RRが返るので、そのドメイン名(RRのNAME部分)を取り出し、"b.a."であれば親子が同居している

2.4.2 スタブリゾルバが指定するDOフラグ

DOフラグは、スタブリゾルバがDNSSECに対応しているか、フルリゾルバに伝えるためにも使われます。このフラグに対して、フルリゾルバは以下のように振る舞います。

DOフラグがオフの場合、できる限りDNSSECを検証し、DNSSEC関連のRRを差し引いて返します。返答のAD(Authentic Data)フラグは立てません。

DOフラグがオンの場合、DNSSEC関連のRRもすべて返します。すべての検証が成功したら、返答のADフラグを立てます。途中からDNSSECの委任がなくなったら、返答のADフラグを立てません。どこかで検証に失敗したらSERVFAILを返します。

2.4.3 不在証明

DNSSECの話題としては、不在証明(NSEC/NSEC3 RR)もあります。技術的な内容は、この記事の範囲を超えるので、詳しくは「DNSフルリゾルバの実装へのDNSSECの組み込み - NSEC/NSEC3による否定応答の証明」[*4]をご覧ください。

2.5 キャッシュのデータ構造

キャッシュのデータ構造としては、探索木と優先度付きキューの両方の性質を持つPSQ(Priority Search Queue)を用いています。キーには要求(ドメイン名、RR型など)、優先順位にはTTL(Time To Live)、値には「反復検索の結果」を使います。すなわち、要求から「反復検索の結果」を効率よく検索可能であると共に、TTLに従ってキャッシュ・エントリを削除できます。

肯定応答は、DNSSECの署名がない場合、署名があっても検証していない場合、署名の検証に成功した場合に分けられます。キャッシュの保存期間は、基本的にRRのTTLの値です。ただしDNSSECの場合、RRSIG RRのTTLや、署名の有効期間によっても制限されます。

否定応答のNODATAとNXDOMAINは区別せずに、SOA RRから得られたTTLの値でキャッシュします。DNSSECで署名されているなら、NSEC/NSEC3 RRとRRSIG RRが得られます。これはNODATAあるいはNXDOMAINの不在証明の証拠なので、SOAと一緒にキャッシュします。

＊4 IIJ Engineers Blog:「DNSフルリゾルバの実装へのDNSSECの組み込み - NSEC/NSEC3による否定応答の証明」(https://eng-blog.iij.ad.jp/archives/24512)。

SERVFAIL、REFUSED、FORMERRの場合は、SOA RRが手に入りません。可能なら、肯定応答あるいはNODATAやNXDOMAINの否定応答を得たいので、次のIPアドレスの候補にフォールバックします。候補が尽きた場合には、同じ問い合わせが繰り返し使われる攻撃を防ぐために、この種の否定応答をキャッシュします。TTLには設定ファイルで指定された既定値を使用します。

応答にはランキングと呼ばれる優先順位があります。AA（Authoritative Answer）フラグがオフの応答から得たグルー情報は、権威付き（AAフラグがオン）の応答から得た情報よりも優先順位が低いです。ですので、前者のキャッシュ・エントリは後者が得られた時点で上書きされます。フルリゾルバはスタブリゾルバに対して、グルー情報がキャッシュに存在しても、それは返さず、権威付きの情報を得てから返します。

2.6 自由にコントロールできたか？

bowlineの目標の1つは、問題が発見されたときに自分たちで迅速に対応できるソフトウェアとすることでした。この節では、迅速に対応できた事例を挙げて、その証左とします。

まず脆弱性に関してですが、bowlineの開発中に運用部隊からドメイン圧縮回数に関する攻撃を説明されたり、外部情報としてKeyTrap攻撃を知ったりしました。これらの脆弱性はbowlineにも存在していたため、すぐに修正しました。

bowlineの試験運用で発見された問題としては、スタブリゾルバがトランスポートとしてQUIC及びHTTP/3を用いた場合、うまく通信できない場合があることが運用部隊から報告されました。その時点のHaskell quicライブラリでは、性能を向上するために、UDPの接続済みソケットを用いていました。接続済みソケットでは、途中のNATがポート番号を変更すると、パケットが届かなくなります。quicライブラリでは、マイグレーションの機能を備えており、QUICコネクションの確立後にポートが変更された場合は、新しい接続済みソケットを使って、コネクションを維持できます。

しかし、この現象を引き起こしたNATは、コネクションを確立させる最中にポートを変えていました。これに対処するために、性能は劣るものの、UDPの利用では一般的な未接続ソケットを使うように大幅な改良を加えました。

また、これはdugの話になるのですが、暗号化されたDNSサーバのいくつかは、TLSのセッション・チケットを複数返していました。この時点でのHaskell tlsライブラリは、セッション・チケットの数は1つであるという仮定に基づいて実装していました。暗号化されたDNSサーバの返すセッション・チケットから1つを選んで、セッションを再開すると、失敗する場合があることが分かりました。おそらくTLSの終端が複数あって、そのすべてのセッション・チケットを返しており、1つだけ選ぶと、受け取ったTLSの終端が期待しないセッション・チケットである場合があるのでしょう。そのため、すべてのセッション・チケットを利用できるように、tlsライブラリを改良しました。

2.7 おわりに

bowlineのサイトは、以下のとおりです。LinuxやmacOSのバイナリへのリンク、Docker Hubからの利用方法、そしてDebianでのインストール方法が記述されています。

- https://iijlab.github.io/dnsext/bowline.html

bowlineのソースコードは、以下のURLが示すGitHubから入手できます。将来はHaskellのライブラリ登録サイトであるHackageに登録することで、Haskellのビルドシステムでも簡単にビルドできるようにする予定です。

- https://github.com/iijlab/dnsext

bowlineプロジェクトの進捗は、以下で報告されています。

- https://www.iijlab.net/projects/Underpinning/dns.html

最後になりましたが、この記事の草稿に対して、様々な意見をいただいたIIJの同僚の方々に感謝します。

執筆者:

山本 和彦（やまもと かずひこ）
IIJ 技術研究所 技術開発室 室長
2022年にIIJの正社員から契約社員となり、故郷の山口県に移住後、リモートワークで勤務。最近は、瀬戸内でサワラやアオリイカを追いかけている。

古くて新しい衛星インターネット、Starlinkが変えていく世界

Starlinkの登場はこれまでの衛星[*1]インターネットが持つイメージを変えてしまいました。Starlinkに触発された新たなプレーヤーも次々に登場しています。本フォーカス・リサーチでは衛星インターネットの歴史を振り返り、Starlinkについて解説します。そして衛星インターネットはどこに向かっていくのか、可能性を探っていきたいと思います。

3.1 衛星インターネット

3.1.1 衛星軌道の種類と特徴

地球を周回している衛星の軌道は高度を基準にGEO・MEO・LEOの3種類に分類できます(図-1)。それぞれの特徴を整理すると、GEO(Geostationary Earth Orbit) は、約36,000kmの高さに位置し、地球の自転と同期した静止軌道となります。GEO衛星は地上局からの位置が固定され、広域をカバーしますが、遅延が大きいのが欠点です。MEO (Medium Earth Orbit) は約2,000 ～35,000kmの高さで、GPS衛星などに用いられ、中程度の遅延とバランスの取れたカバー範囲を提供します。LEO(Low Earth Orbit)は約160 ～2,000kmの高さで、低遅延と高帯域が強みですが、地球全体をカバーするためには多数の衛星を必要とし運用コストがかさみます(表-1)。

3.1.2 研究ネットワークの時代

SATNET (Satellite Network、またはAtlantic Packet Satellite Network)は、1970年代に米国国防高等研究計画局(DARPA)によって開発された衛星ベースのネットワークです。これは、ARPANET(Advanced Research Projects Agency Network)を衛星(Intelsat IV)経由でヨーロッパに接続する実験プロジェクトであり、衛星インターネットの先駆けと言えます。

日本の研究ではWIDE Project(Widely Integrated Distributed Environment Project) によるAI3 (Asian Internet Interconnection Initiatives) プロジェクトがあります。アジア太平洋地域、特に東南アジアの研究機関を衛星リンクで接続し、インターネットインフラの構築とネットワーク技術の研究開発を目的とし、日本のJSAT(スカパーJSAT)の衛星を使っています。SATNETの研究から約20年、インターネットが研究ネットワークとして世界中に普及していく中での活動であり、単方向リンク(UDL、Uni Directional Link)を開発・展開し、UDLR(Uni Directional Link Routing)技術の標準化に貢献しています。このUDLでIPv6とマルチキャストを運用し、アジアのより広いエリアの接続に貢献しました。

3.1.3 商用サービスの発展

GEO衛星を使った商用インターネットサービスは1990年代後半に本格化しています。1996年、Hughes Network SystemsがDirecPCを初の消費者向け衛星ブロードバンドとして開始しました。2004年にはWildBlueがKaバンド衛星を使った高速サービスを提供しましたが、2009年にViasatに買収されています。現在はHughesnet、Viasat、Konnect、SES Astra、Inmarsat、ExBird、SKY Perfect JSA(EsBird)、China Satcomといった事業者がサービスを提供しています。

MEO衛星を使った商用インターネットサービスは、O3b Networksが2007年に設立され、2013年に4機の衛星を打ち上げ、サービスを開始しました。O3bは2016年にSESに買収され、現在はSES NetworksとしてO3b mPOWER(第2世代、2022年打ち上げ開始、2024年4月運用開始)を提供しています。衛星は現在10機が稼働しています。

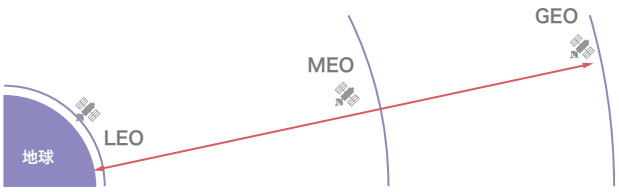


図-1 地球と衛星の位置関係
LEOは地上スレスレにあることが分かる

表-1 衛星軌道の特徴

項目	GEO(静止軌道)	MEO(中軌道)	LEO(低軌道)
高度	約36,000 km	約2,000～35,000km	約160～2,000km
遅延	約500～600ms	約100～300ms	約25～80ms
見かけの動き	静止(常に同じ位置)	ゆっくり移動	高速で移動
周回周期	24時間(地球自転と同期)	数時間	約90～120分
全球カバー	数機(3～4機)	数十機	数百～数千機
カバー範囲	広い(1機で地球の約1/3)	中程度(複数で全球カバー)	狭い(多数必要)

*1　この記事の「衛星」は月など天体としての衛星ではなく、人工衛星を指します。

LEO衛星を使った商用サービスでは、Iridiumが1998年11月、Globalstarは1999年2月にサービスを開始しています。両社は一旦破産していますが、その後復活し、音声通信を中心にサービスを続けています。Teledesicのブロードバンド構想は1990年代初頭(1990年設立)に始まり、Iridium/Globalstarの競合として1997年にはFCC承認まで進みましたが、2003年に中止となっています。

LEOの飛躍はOneWebから始まります。そして大きく成功したのがStarlinkです。Starlinkに続けとProject Kuiperが準備中で、中国勢も追いかけようとしています。

OneWebは、グレッグ・ワイラー氏が2012年に構想を始め、2019年に運用を開始しました。資金難や地政学的課題を乗り越え、2023年にEutelsatと合併しています。現在はEutelsat OneWebとして、648機の衛星を運用し、欧州・米国を中心にサービスを拡大中です。

Starlinkはイーロン・マスク氏の構想から始まりました。2015年に公表されたこのプロジェクトは、衛星の大量生産と再利用ロケット技術を活用し、現在8,500機を超える衛星を展開しています。ユーザは750万を超え、150ヵ国以上の国にサービスを展開しています。

Project Kuiperは2019年に設立されたAmazonの子会社で、2025年末のサービス開始に向けて準備を進めています。軌道にある衛星は約129機とされています。

Qianfan(千帆、Thousand Sails)は、Shanghai Spacecom Satellite Technology(SSST)が主導する中国の衛星です。Starlinkを意識しており、2024年8月に初打ち上げ、軌道にあるのは約90機とされています。

GuoWang(国家網、Guowang)は、China SatNetが主導する中国の国家プロジェクトで、これもStarlinkを意識しています。2024年12月に初打ち上げ、軌道にあるのは約95機とされています。

3.2 Starlink

3.2.1 Starlink概要

Starlinkはアメリカの民間企業SpaceX(Space Exploration Technologies Corp.)のサービスです。2015年に創業者イーロン・マスク氏がアイデアを発表。2019年に60機の衛星で限定ベータサービスを開始して以来急速に成長しています。2025年10月現在、地球上空を周回する衛星は約8,500機を超え150カ国を超える国、約750万人のユーザが利用しています。日本では2022年10月からサービスを提供しています。

Starlinkのアーキテクチャを示します(図-2)。

通信は、User Terminal → Starlinkコンステレーション → Gateway → POPを通ってインターネットにつながります。

Starlinkにつなげるにはは User Terminalが必要です。公式サイトや正規の量販店から購入する以外にも、様々な入手方法が存在します。

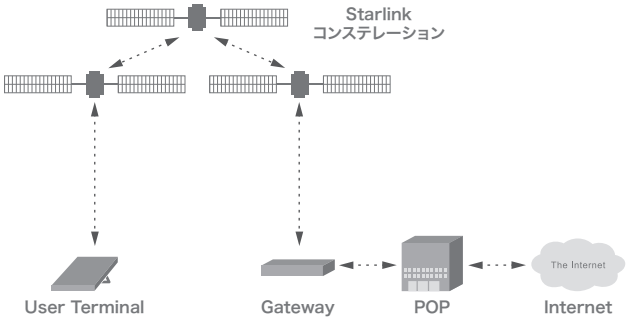


図-2 Starlink Architecture

User Terminalには複数の種類があり、Standardの他に超小型のMINIや、厳しい環境や海上利用を想定したPerformanceが存在します（図-3）。

User Terminalは自分の位置情報とStarlink衛星の軌道データを使って、通信可能なStarlink衛星を選択します。電波の指向性はフェーズドアレイを使って電子的に自動的追従するので、ユーザが調整する必要はありません。Starlink衛星は地球の低軌道を覆っており、コンステレーションを形成しています。現在約8,500機が稼働しているそうですが、日々増強やメンテナンスが行われているため、正確な数はSpaceXにしか分かりません。

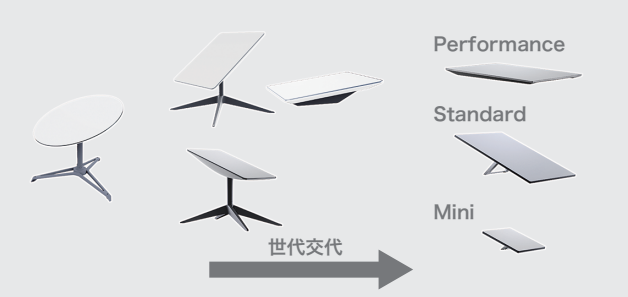


図-3 User Terminalの様々な世代
可動スタンドからキックスタンドに変わっていった

衛星コンステレーションや衛星の確認にはsatellitemap.spaceが良くできています。このサイトはspace-track.orgの公開データから軌道を基に衛星コンステレーションや衛星を可視化してくれます（図-4、図-5、図-6）。

可視化に使っているアルゴリズムをSpaceXは公開していないので、あくまで参考情報ですが、イメージは伝わるかと思います。

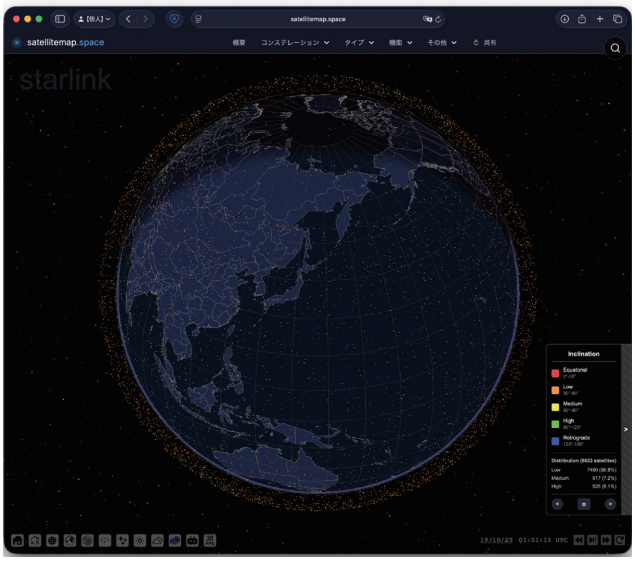


図-4 地球全体に分散しているStarlink衛星
この時点で8,622機の衛星があるようだ

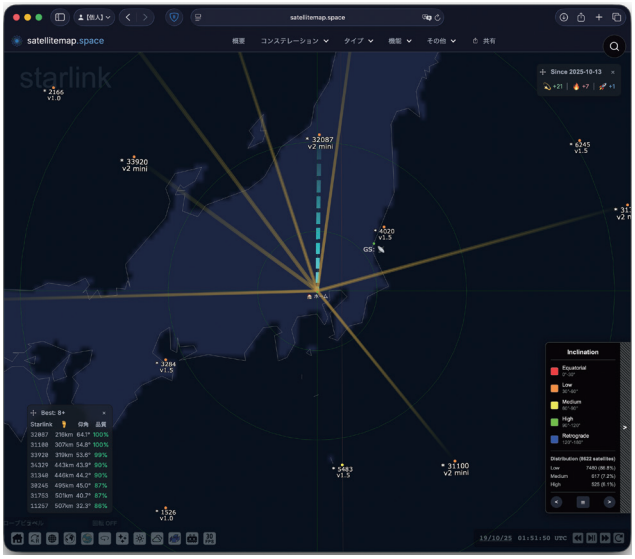


図-5 筆者がいる東京から見たとある時刻の衛星の配置
User Terminalが近くの衛星を選択している様子。
選択アルゴリズムは非公開なので、これはあくまで想像

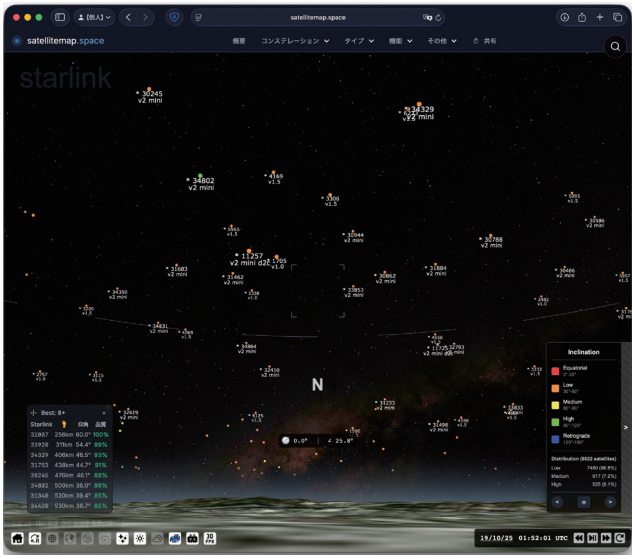


図-6 同じく東京の地上から見える衛星の配置
視界には数10の衛星が入っていることが分かる

Starlink衛星の型ですが、現在V1.5とV2miniという衛星が稼働しています。次世代衛星としてV3が計画されており、SpaceXの新しいロケットStarshipを使って投入が計画されています(図-7)。V3は従来の10倍の性能を持つと言われており、新型のUser Terminalとの組み合わせで1Gbpsの通信速度をサポートする計画になっています。

SpaceXは衛星の数を増やすこと、衛星を新型に入れ替えていくことで、コンステレーション全体の能力向上を図っていく計画です。

Gatewayは、衛星と地上インターネットの橋渡し役です。衛星からデータを中継し、POPに送ります。日本では、北海道、秋田、茨城、山口の4ヵ所に設置されていることが確認されています(図-8)。

POP(Point of Presence)はインターネットへの接続点でUser TerminalからPOPまでがL2ネットワークのように見えています。IPv4での接続はPOPからDHCPを使ってISP Shared Address(100.64.0.0/10)が配布されます。この場合ユーザはCGNAT経由で共用IPv4アドレスインターネットにつながります。一部のメニューではパブリックIPv4をDHCPで取得して通信ができます。IPv6はパブリックIPv6アドレスがDHCPv6-PDで切り出されてくるのでこれを使って通信することができます。

3.2.2 Starlinkの活躍

2022年1月15日に発生したトンガの火山噴火は観測史上最大級の水中噴火でした。海底ケーブルが切断され、トンガは通信的にも孤立してしまいましたが、Starlinkが迅速に介入し、復旧の鍵となりました。2022年2月のロシアによるウクライナ侵攻ではStarlinkが破壊された地上通信インフラの代替として急速に導入され、ウクライナの軍事・民間通信を支えました。2024年1月1日に発生した能登半島地震ではKDDIやSoftBankやDocomoがStarlinkを投入し被災地の通信回復を支えました。その後も、世界各地で起きている災害の現場でStarlinkは活躍しています。

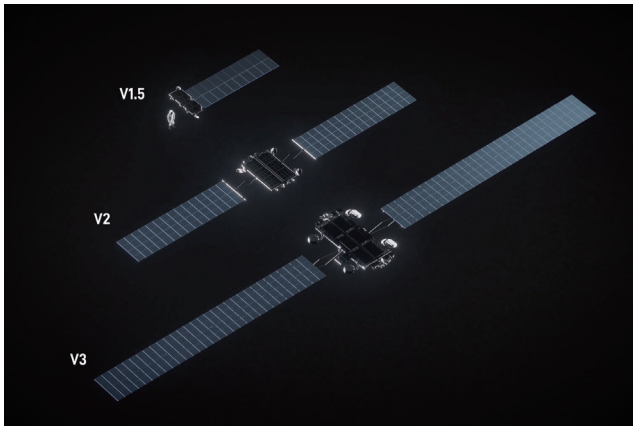


図-7 Starlink衛星のサイズ比較[*2]
V3はStarship運用後に投入されることになる

※地図の出典:国土地理院ウェブサイト(https://maps.gsi.go.jp/)
※写真はすべて筆者撮影

図-8 日本のGateway
北海道、秋田、茨城、山口の4ヵ所に設置され、通信は東京のPOPに集まる

*2 SpaceX、2025年10月24日のX投稿(https://x.com/SpaceX/status/1977873370688700846)。

3.2.3 Starlinkの強み

創業者イーロン・マスク氏のビジョンは「人類を多惑星種にする」ことです。ターゲットは火星です。火星までのロードマップを達成するには巨額の資金、技術的なブレークスルーが必要になります。Starlinkは技術開発のための資金源なのです（図-9）。

Starlink成功の鍵は衛星を運ぶために開発したロケット（Falcon 9）でした（図-10）。このロケットの特徴は1段目とフェアリングが再利用できることです。使用済みのロケットは打ち上げ場所まで戻し、海上に配置したドローン船に着陸・回収し再整備されます。

ロケットを回収し、整備後に再利用できるようにするまでにかかる時間は10日間程度です。ブースターの中には30回を超えて再利用されているものもあり、再利用性が非常に高いことが分かります。ロケットは現在30機程度が稼働しており、2024年は132回の打ち上げを実施、2025年は170回を計画しています。1回当たり26機のStarlink衛星を軌道に投入しているので、年間4400機を超える衛星の投入能力があることになります。

軌道に投入する大量の衛星自体を生産しているのもSpaceXです。ロケットには再利用していない2段目のロケットもあるわけですが、その生産をしているのもSpaceXです。数千機規模の衛星を自社生産できる能力、それを打ち上げるインフラ、衛星コンステレーションを運用できる体制、新しいロケットの開発・テスト。衛星からロケットまで垂直統合し、Starlinkや他社向けの打ち上げサービスの収益で運用していく方策。これまで誰も成し得なかったことにSpaceXは挑戦していることになります。

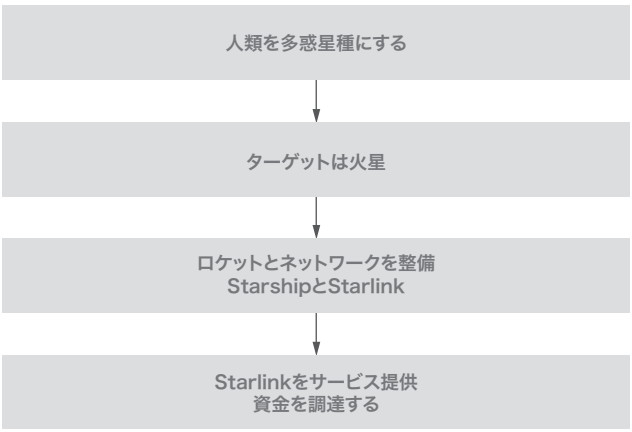


図-9 SpaceXの目的

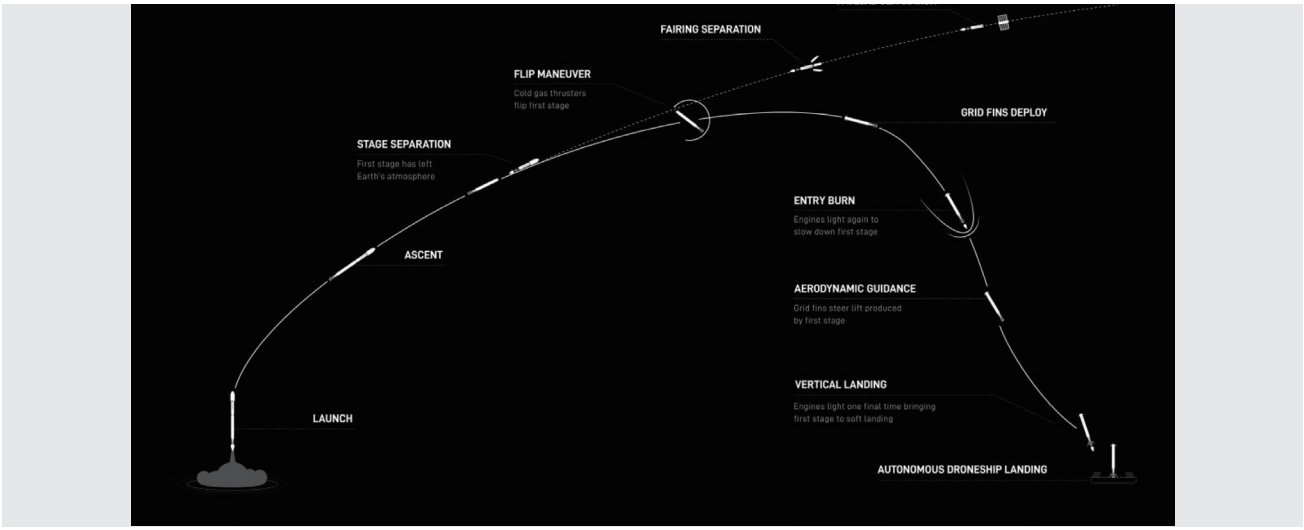


図-10 Falcon9によるミッションの流れ
ロケットは自動的にドローンシップに着陸する[*3]。

*3　SpaceX 公開資料（https://www.spacex.com/assets/media/falcon-users-guide-2025-05-09.pdf）。

3.3 衛星インターネットの未来

3.3.1 インターネットのインフラが変わる

現在のインターネットを支えているのは、地上で整備された通信網です。光ファイバーを用い、地上はもちろん、海底ケーブルを敷設して大陸間をつなぎ世界中をつなげています。携帯電話の無線ネットワークにしても、基地局をつないでいるのはこれらの通信路です。ところが、この地上ネットワークの敷設と維持には多大な費用がかかっています。

低軌道衛星を使った通信網は、新たなインフラとなる可能性が出てきました。これまで光ファイバーを使って設置していた部分は、衛星間の光リンクに置き換えることができます。衛星はインターサテライトリンク(ISL)と呼ばれるレーザー通信機を装備しており、複数のStarlinkでメッシュネットワークを構築して、地上からの通信を効率よく処理します。SpaceXによって衛星打ち上げのコストが劇的に下がったので、こんなことが考えられるようになったのです。

3.3.2 通信は宇宙経由の方が速くなる？

衛星間をつなぐ光リンクは光ファイバーよりも有利な面があります。光が真空中を進む速度は秒速30万kmですが、光ファイバーの中だと20万kmの速度まで下がってしまいます。例えば北アメリカ大陸で、東海岸から西海岸までは約8,000kmあります。地上から衛星までの距離を550kmとすると、全体の距離は9,100kmになる計算です。地上の8,000kmを光ファイバーで進むと40msの時間がかかります。衛星を介せば距離は9,100kmに延びますが、30msで到達する理屈になります。片道10ms、往復で考えれば20ms速いとなれば、衛星インターネットに有利な面は多々ありそうです(図-11)。

Starlinkの衛星間レーザーリンクは、既に200Gbpsの速度を実現しています。地上ではこれよりも速い通信がいくらでもありますが、衛星が充実してメッシュで通信路を確保できる世界になれば、衛星インフラの方が有利な面も増えてくるのではないかと思います。将来的に、地上にいるユーザもレーザー通信を搭載した端末を大量生産・利用するようになり、安くて超高速なネットワークは、衛星インフラで実現するようになるかもしれません。

3.3.3 惑星間通信への拡張

SpaceXの火星計画において、Starlinkも更に進化していく計画があります。地球から火星までの通信時間は、惑星距離に応じて3分から22分、往復で考えると6分から44分かかります。これを従来のTCP/IPで吸収するのは難しいです。衛星インターネットの世界では多惑星の未来に向かって今後も研究が進むでしょう。どんな研究成果が出てくるのか。地球での生活にどんな影響を与えていくのか。これからが楽しみです。

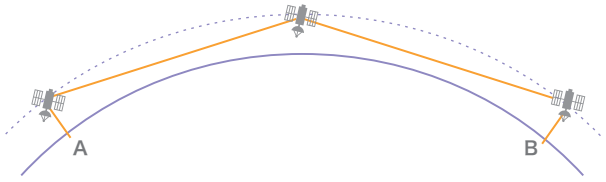


図-11 衛星間通信と地球上の距離の関係
地球が曲面である事、常に衛星まで最短距離で繋がるわけではない事、海底ケーブルのたわみ、中継機の遅延などさまざまな影響があるため、単純な比較は難しい。

執筆者:

谷口 崇(たにぐち たかし)
IIJ マーケティング統括本部 マーケティング本部 フェロー 宇宙事業推進室 兼務
1995年IIJ入社、個人向けサービスのインフラ構築・運用などに携わる。2006年からゲーム業界に出向し、AAAタイトルやソーシャルゲームのシステム開発・運用に取り組んだのち、2022年にIIJ復帰。Starlinkの登場を契機に、かねてから強かった宇宙分野への関心を深め、これまでの経験を礎に果敢な挑戦を展開中。格闘ゲームで世界チャンピオンの従兄弟がいる。

生成AIによる社内RAG基盤とマルチエージェント連携への拡張 ～ 実装技術と業務効率化の成果、今後の展望 ～

4.1 はじめに

近年、OpenAIやGoogle、Anthropic社をはじめとする各社によってAI技術、とりわけ大規模言語モデル(LLM)の進化は著しく、企業における知識活用の方法は大きく変化しています。IIJにおいても2023年夏から、LLMを活用した社内専用のRAG基盤「sbdGPT」の運用を開始しました。本稿では、この社内RAG基盤の開発背景や目的、システム構成に加え、マルチエージェント化による拡張、更に社内RAG基盤と連携した提案書生成ツールについて述べます。

4.2 社内RAG開発の背景と目的

IIJでは100を超える法人向けサービスを展開しており、社内にはマニュアル、ナレッジ共有サイト、ニュース、問い合わせメールなど、多種多様な情報が大量に蓄積されています。これらの情報資産は複数の異なるプラットフォームに分散管理されているため、そこへ横断的にアクセスし、必要な情報を迅速に検索・取得することは、営業担当やエンジニアにとって大きな課題となってきました。その結果、繰り返し発生する社内問い合わせ作業や、情報探索に要する時間が現場の業務効率化を阻害していました。

こうした状況を背景に、社内に点在するドキュメントデータを収集・加工し、一元的にデータベースとして集約すると共に、生成AIを活用して質問内容に即した具体的な回答を提供できることを目指して、社内RAG基盤の開発に着手しました。

4.3 社内RAG基盤の構成とデータ最適化

社内RAG基盤は複数のデータソースを統合し、ベクトルデータ化してデータベースに格納しています。LLMや埋め込みモデル以外は自社サービス及びOSSを活用して、コストを抑えつつ開発アップデートや運用管理を自律的かつ柔軟に行える構成としています。

構成概要は図-1のとおりです。

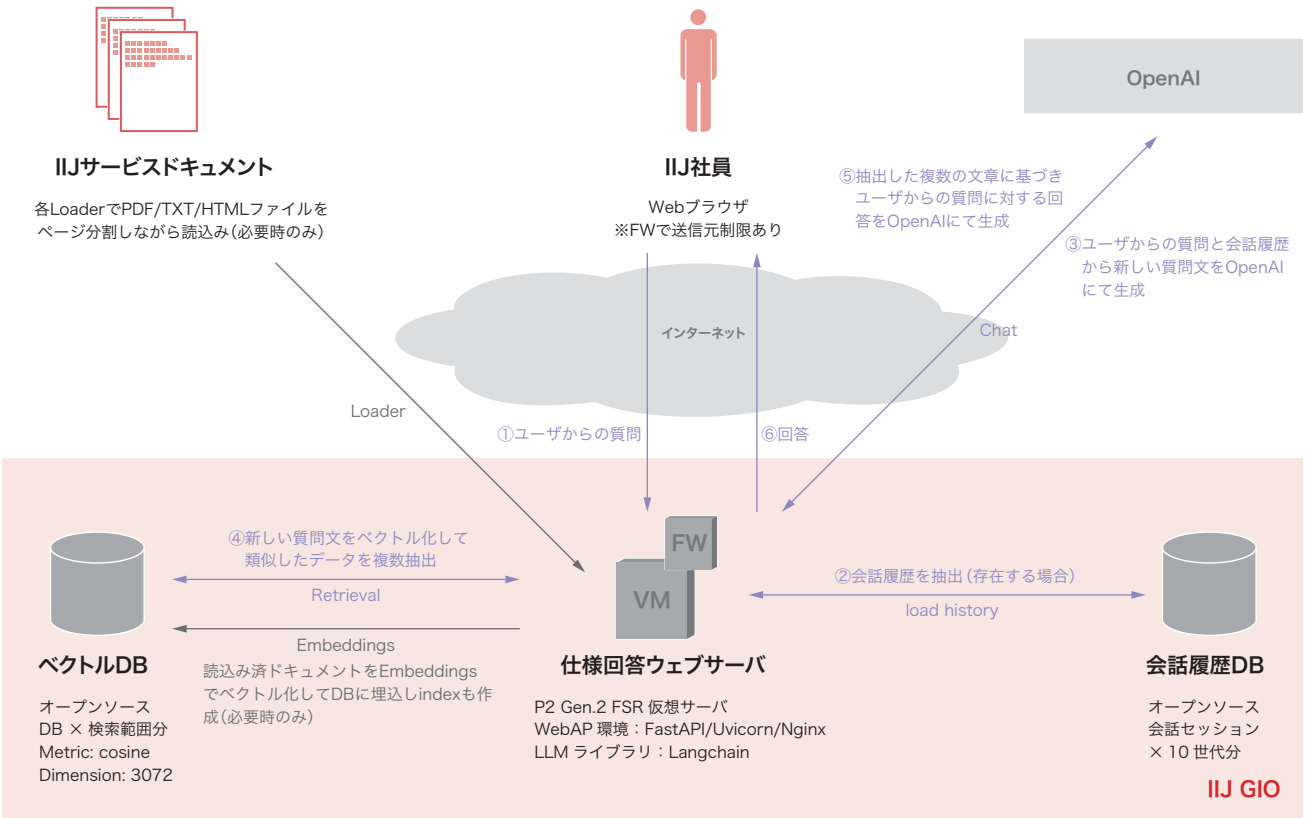


図-1 社内RAG構成概要図

- 構成要素

ナレッジベース(IIJサービスドキュメント)

- サービス詳細資料(PDFファイル)
- オンラインマニュアル(HTMLファイル)
- 社内技術QAメール(メッセージファイル)
- ナレッジ共有サイト(HTMLファイル)
- ニュース記事(HTMLファイル)

Webサーバ基盤

- IIJ GIOインフラストラクチャー P2 Gen.2フレキシブルサービス仮想マシン
- IIJ GIOインフラストラクチャー P2 Gen.2境界ファイアウォール
- FastAPI(OSS)
- Uvicorn/Gunicorn(OSS)
- Nginx(OSS)

生成AI/Embeddingモデル

- OpenAI社製モデル

LLM開発フレームワーク

- LangChain(OSS版)

ベクトルDB

- ChromaDB(OSS版)

会話管理DB

- PostgreSQL(OSS版)

※生成AIとEmbedding以外は、IIJサービスのインフラやOSSを利用している。

ベクトルDBに格納する各ナレッジデータは、複数のプラットフォーム上で分散管理されているPDFやHTMLなどのファイルを対象としています(図-2)。これらのデータは、ファイル形式ごとにLangChainが提供する各種Loaderを利用し、テキスト情報として抽出しています。

当初は一般的な手法に従い、抽出した大量のテキストデータをText Splitterで適切な長さに分割(チャンク化)し、そのままベクトルDBに格納していました。しかしこの方法では、

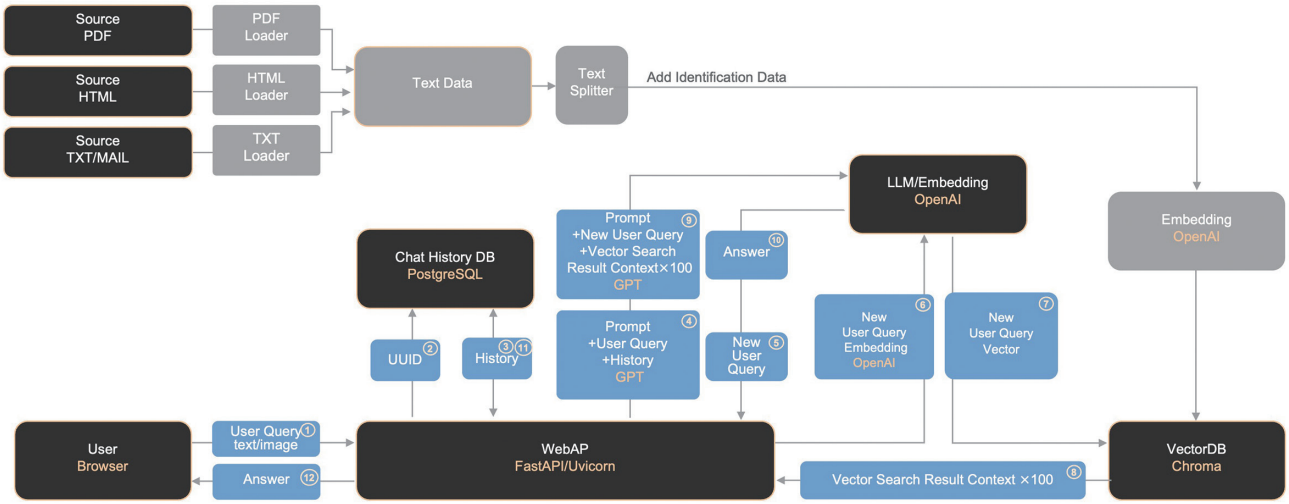


図-2 社内RAG動作フロー図

ユーザクエリに対して期待する回答が得られないケースが多く発生しました。

原因を確認した結果、各サービスドキュメントはPDFやHTMLといったファイル単位で意味が完結しているため、部分的に分割抽出された短文（例：200文字程度のチャンクデータ）だけでは「どのサービスの、何に関する記述なのか」という文脈情報が不足し、ベクトル検索時に意味的類似（semantic similarity）が十分に機能せず、関連情報として正しく取得できない状況が生じていました。

この課題を踏まえ、チャンク化の際には各ファイルに記載されていた「サービス名」「タイトル情報」などの既存メタデータから識別情報を生成し、各チャンクに付与することで、ベクトル検索のヒット率及び回答精度の向上につなげています。

4.4 ファクトチェックに関する取り組み

AIの生成回答には、ハルシネーション（誤情報の生成）が含まれる場合があります。そのため、AIの回答を業務利用する際にはファクトチェック（根拠確認）が不可欠となっています。AIがどの情報を引用して回答を生成したのかを把握するには、AI自身に引用情報を出力させることが重要であり、ユーザはその引用情報を確認することで、内容の正確性を担保する必要があります。

ただし、ファクトチェックの操作が煩雑であれば、ユーザの利便性は大きく低下します。そこで本ツールでは、最小限の操作でデータソースにアクセスできる仕組みを導入しました。具体的には、生成回答の最後に表示される引用リンクボタンをクリックするだけで、読み込んだナレッジデータ（PDF・HTMLなど）のソースを表示するように実装しています。

HTML以外のデータソースについては、事前にWebサーバ内に格納し、ブラウザ経由で静的に公開することで、PDFなどのソースにも即座にアクセス可能としました。この仕組みにより、AI回答の信頼性を確認するための導線を簡素化しています。

更に、各チャンクに付加している識別情報には、データソースファイルのFQDN（PDFの場合はページ番号を含む）も付与しています。そのため、システムプロンプトでは以下のような引用情報の出力形式を指定し、生成回答の直下に引用元を明示するようにしています。これにより、利用者はボタンをクリックするだけで、引用情報をテキスト表示し、Webサーバ上のソースファイルへも即時アクセスできます。

```
# 引用情報の出力形式例
<div hidden id="{{ id }}">
  <p class="cite_title">Cited Context 1</p>
  <a href="{{ ソースファイルの FQDN }}" target="_blank">{{ ソースファイルの FQDN }}</a>
  <p>{{ コンテキスト文章 }}</p>
</div>
<button class="open-button" onclick="refOpen('{{ id }}')"> 主な引用情報 </button>
```

4.5 業務効率効果とRAG基盤のマルチエージェント拡張

社内RAG基盤の利用を2023年夏に開始して以来、約2年間の運用を経て、月間約3万クエリが処理されるまでに利用が拡大しました。これにより、社内で月当たり約1,500時間相当の業務効率化が実現しつつあります。

一方で、運用を重ねる中で、RAG単体では社内情報に基づく回答に限界があることも明らかになりました。特に、最新の市場動向や競合比較、外部製品との仕様差分など、インターネット上の情報を参照しなければ適切な回答が難しいケースが増え、また、単なる仕様確認にとどまらず、営業担当やエンジニアに

よる市場リサーチやトレンド分析など、より高度な意思決定支援を行う必要性も出てきていました。

こうした課題を解消するため、社内RAG基盤を拡張し、複数の専門エージェントが連携して応答を生成するマルチエージェント構成を実装しました。従来のRAGは、IIJサービスのマニュアル、ナレッジ共有サイト、問い合わせメールなど社内データをベクトルデータベース化し、生成AIによって回答を生成する仕組みでしたが、社内データのみでは「IIJと他社のサービス仕様比較」や「最新技術トレンドに基づく提案検討」といった外部情報を必要とするクエリへの対応が困難でした。

この制約を解決するため、従来の社内RAG基盤をリファクタリングし、マルチエージェント型の基盤を構築しました。メインエージェントがルーティングロジックに基づいて、社内RAG基盤のエージェント版となる「IIJ Service Agent」や最新の外部情報を取得する「WebSearch Agent」を必要に応じて自動的に呼び出す仕組みです。

具体的な制御構成としては、LangGraphによる状態遷移グラフを用いてエージェント間の連携を統合的に管理しています。メインエージェントは質問文章を解析し「社内データで解決可能か」「外部情報が必要か」を判定した上で、該当するサブエージェントを呼び出します。サブエージェントの処理結果はメインエージェントに返却され、最終的に統合された応答文章としてユーザに出力されます。これにより、社内情報と外部の最新情報を組み合わせた包括的な回答生成が可能となりました。

更に、サブエージェントは並列実行にも対応しており、複数の情報領域の処理を同時に実行できるため、人力では実現できない速度で結果を出力することができます。加えて、LLMの推論過程を可視化することで処理の透明性も高めています。UI上では「どのエージェントが呼び出されたか」「どの情報源に基づいて回答が生成されたか」がストリーミングで逐次表示され、利用者はLLMの推論過程をリアルタイムで追跡できるようになっています。

結果として、従来の社内RAG基盤ではカバーしきれなかった領域まで回答範囲を拡張することができ、ユーザエクスペリエンスの大幅な改善につながりました。マルチエージェント化によって構築された社内エージェント基盤は、単独では解決困難だった情報不足の問題を克服し、AI活用の幅を大きく広げる重要なステップとなっています。

4.6 DeepResearchとRAGの融合

2024年末にGoogleが発表したDeepResearchは、インターネット上の膨大な情報源を自律的に探索・分析・統合し、研究者やアナリストのように多段階の調査を遂行できる仕組みとして注目を集めました。その後、2025年2月にはOpenAIからもDeepResearchがリリースされ、同年6月にResponses APIとして提供が開始されたことで、既存のエージェント環境に組み込める実用的な材料が整いました。

DeepResearchの最大の特徴は、「計画 → 検索 → 吟味 → 統合 → 生成」という一連のプロセスを自動的に繰り返す点にあります。Web検索を複数回行い、取得した情報を検証しながら再計画を立て直し、最終的に信頼性の高い要約レポートを返す。そのため処理完了には30分程度を要する場合もありますが、技術文献レビューや競合・市場調査、規制比較など、高い網羅性と根拠が求められる調査分野において強力な武器となります。

社内エージェント基盤では、LangChainとLangGraphにより既にマルチエージェント構成を実装していたため、DeepResearchをサブエージェントとして追加することは比較的容易でした。具体的には、@toolでDeepResearchエージェントを定義し、その内部でResponses APIを呼び出す関数を実装する形で統合しました。ユーザからのクエリを受け取ると、DeepResearchはWeb検索を複数回実行し、必要に応じて数段階の推論と統合を経て、レポート・データ表・出典リストを含む構造化された応答を返します。

加えて、ClarifyQuery（質問の明確化）エージェントをDeepResearch前段に配置する仕組みも導入しました。これは、曖昧なクエリがそのままDeepResearchに渡されることで、検索の再試行が増え、結果としてコストや処理時間が膨大になってしまうことを防ぐためのものです。このエージェントは、利用者の質問内容を確認し、必要に応じて追加の確認質問を行うことでクエリを明確化し、その上でDeepResearchを実行します。これにより、利用者の意図に沿った網羅的かつ効率的なリサーチが可能となっています。

また、DeepResearch APIが提供する推論パラメータ（summary、effort、verbosity）をユーザが調整できるようにし、調査の深さや出力量を用途に応じて選択できる仕組みを整えました。例えば、エグゼクティブ向けの短い要約を求める場合はeffort=low、verbosity=lowを選択し、仕様比較のような綿密な調査にはeffort=high、verbosity=highを指定することで、同じクエリでも出力粒度をクリック操作だけで柔軟にコントロールできるようにしました。

UI上では、処理の進捗やコストも可視化しています。AIの推論過程をストリーミング表示するほか、使用トークン数やモデルの消費コストを集計表示することで、利用者は回答生成に要したコストを把握できるようになり、更に開発者としてはトークン消費量やキャッシュトークンの使用状況をモニタリングできるので、ツールの最適化に向けた情報として活用しています。

こうしたDeepResearchの導入を進める中で新たに浮かび上がったのは、「社内データも含めた自律的なリサーチ機能の必要性」でした。通常のDeepResearchはインターネット上の公開情報を主な対象として設計されており、すべての社内ナレッジベースと直接連携する仕組みは備えていませんでした。また、社内ナレッジサイトについても、提供されているAPIを経由すればDeepResearchとの接続自体は技術的に可能と考えられましたが、社内調整や承認といった非技術的なハードルが高く、現実的ではありませんでした。

そのため初めに取られたアプローチが、IIJ Service Agent（社内RAG）とWebSearch Agentを組み合わせ、プロンプトで細かく処理ステップ手順を制御する方法でした。インターネット上の最新情報はWebSearch Agentが、社内情報はIIJ Service Agent（社内RAG）がそれぞれ探索し、最終的にその結果を統合してユーザに返すことが可能となりました。ただ、従来のLLM（GPT-4.1以前）では、推論ステップの制限やツール呼び出し制御の弱さにより、十分な網羅性と整合性を確保することが困難でした。

そこで、開発検証の最中にリリースされた、エージェントタスク処理に最適化されたといわれるGPT-5をメインエージェントとして適用しました。GPT-5はツール呼び出し、命令遵守（instruction following）、ロングコンテキスト理解、ツール呼び出し間の推論保持といった能力が強化されており、これに

よりエージェントワークフローにおけるツールとの連携性が大幅に向上しました。結果として、各サブエージェントを複数回呼び出しながら網羅的に情報を収集し、社内データを含む30～80件程度の引用情報を基に最適な回答を生成できるようになりました。

これにより、従来は「情報不足」と判断されたクエリに対しても、社内データと外部情報を横断的に活用した高い網羅性と信頼性を備えた回答を提供できるようになり、大きな進展を得ることとなりました。また、開発者としても、既存アーキテクチャを大きく変更することなく最新のLLMを適用するだけで課題を解消できた点は重要であり、エージェント基盤の拡張性と継続的進化の可能性を実感するポイントでもありました。

4.7 提案書生成ツールの開発

社内RAG基盤の運用が安定した段階で、新たなWebアプリとして提案書生成ツール「Panorama」の開発が進められました。営業担当やエンジニアが要件に応じてPowerPoint形式の提案書を作成する業務は、社内テンプレートからスライドを選別・編集する手作業に多くの時間を要していました。こうした課題を解決するために開発されたのが本ツールであり、要件入力から提案書ドラフトの自動生成までを簡単な操作で行える仕組みを提供しています(図-3)。

提案書生成ツールは、ユーザがブラウザ上で要件や希望サービスを入力して実行すると、Webサーバが社内RAG基盤のAPIを呼び出し、入力データとシステムプロンプトを基にして最適

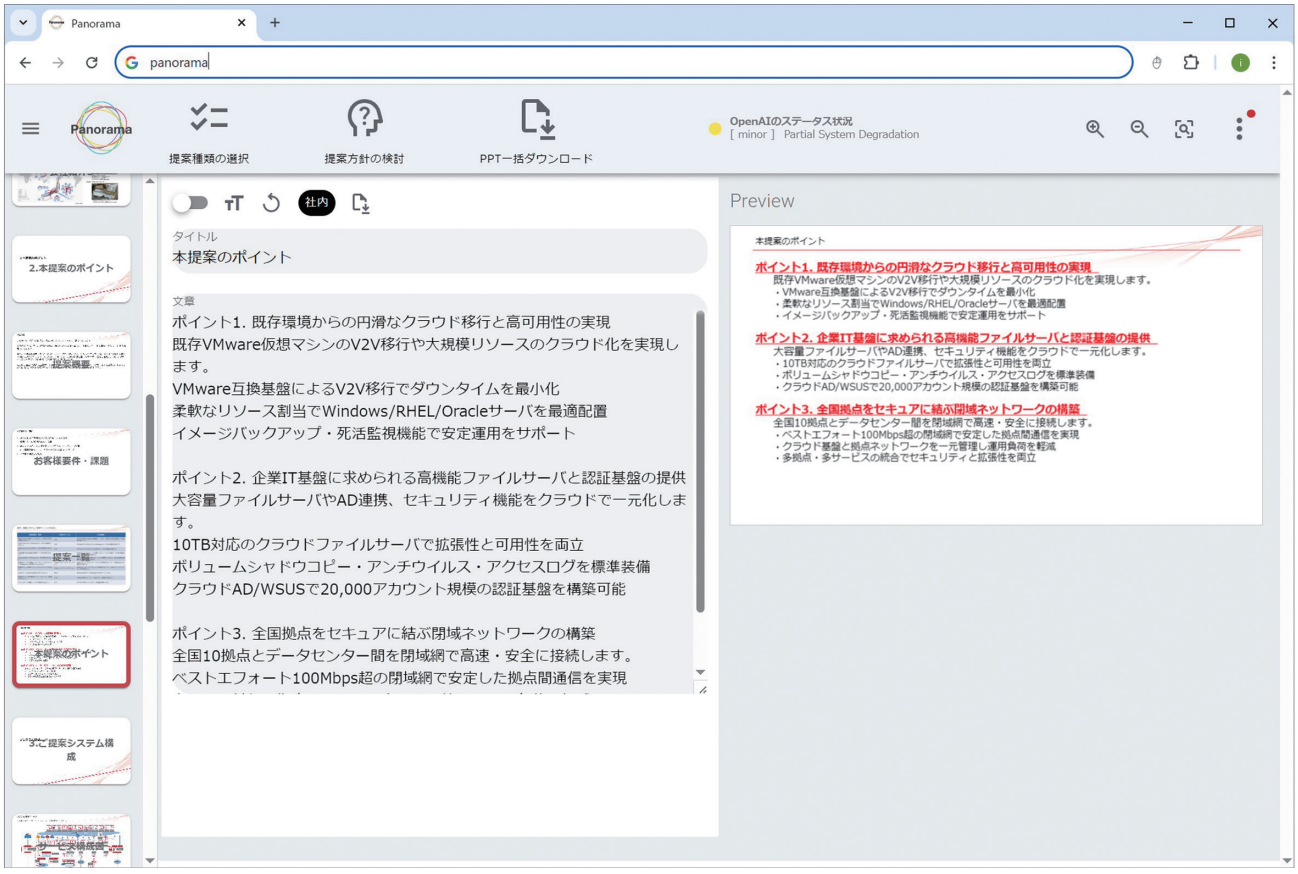
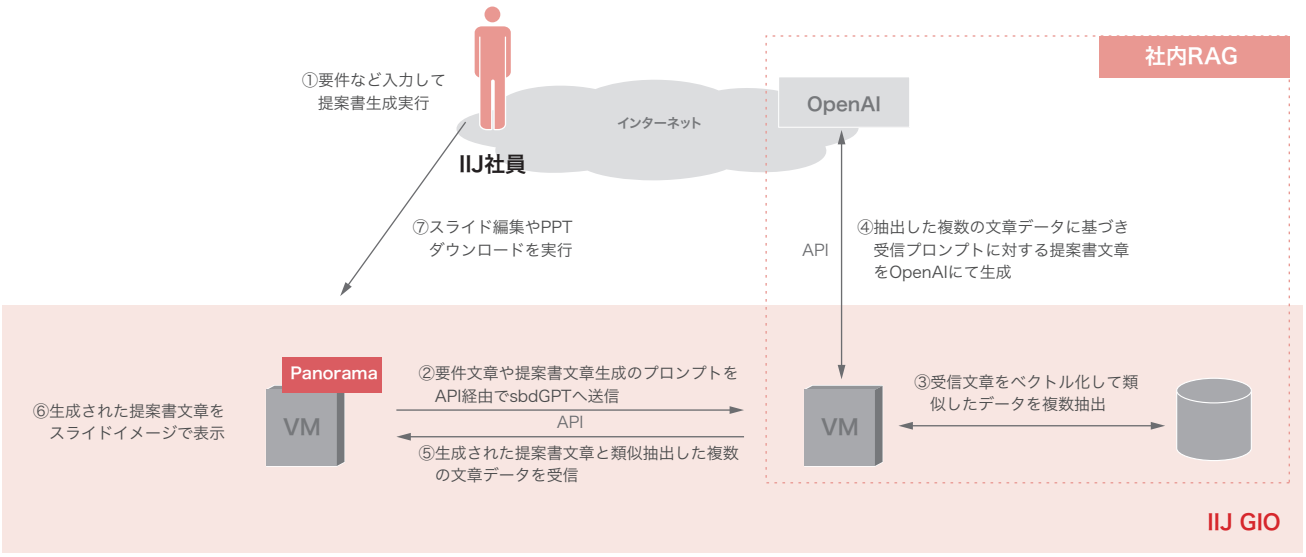


図-3 提案書ツール利用者画面

なサービス提案文面を自動生成させる仕組みを備えています（図-4）。生成結果はスライド単位の構成情報としてJSON形式で処理され、ブラウザ上の編集画面に反映されると共に、プレビュー用のHTML5 Canvas上にリアルタイムで描画されます。描画編集にはfabric.jsを用いており、ユーザはブラウザ上で構成図レイアウトや提案文面を確認・修正しながら、最終的にPowerPointファイルとしてダウンロードできます。

技術的には、Vue.js＋fabric.jsによるフロントエンドと、FastAPI＋python-pptxによるバックエンドで構成されています。フロントエンド側では、スライドや構成図の編集内容をリアルタイムに反映し、すべての編集データをJSON形式で一元管理します。これらのデータはFastAPIを介して送信され、バックエンドのPPTXファイルの生成処理に利用されます。バックエンド側では、JSONデータを基にPythonでテンプレートスライドに対応するXML構成ファイルを操作・更新し、その編集結果を基にpython-pptxライブラリで最終的な提案書ファイルを生成します。

これらの仕組みにより、AIが生成した提案文章や自動プロットした構成レイアウト図をスライドに反映し、編集作業を最小限に抑えつつ、一定した品質の提案書を短時間で作成することができます。更に、社内RAG基盤との連携により、提案文章にはIIJサービスの仕様や提案理由、提案ポイントなどの社内ナレッ



提案書生成ツール

P2 Gen.2 FSR 仮想サーバ
WebAP 環境：FastAPI/Uvicorn/Nginx
PPT ライブラリ：Python-pptx
グラフィック描画ライブラリ：Fabric.js/canvas

社内RAG基盤

P2 Gen.2 FSR 仮想サーバ
WebAP 環境：FastAPI/Uvicorn/Nginx
LLM ライブラリ：Langchain
VectorDB：ChromaDB

図-4 提案書ツールシステム構成概要図

ジが自動的に反映されています。編集画面には参照元情報が明示され、クリック操作でデータソースを即時確認できるため、提案内容のファクトチェックも容易です。

今後はテンプレートスライドを順次追加し、PDF形式でのエクスポートや構成図編集機能の強化など、機能拡張を予定しています。

4.8 今後の展望

最新の言語モデル及びエージェント技術の進化により、複雑な処理の自律実行が現実のものとなり、業務自動化や効率化を支える基盤としてエージェントの重要性が急速に増しています。当面は、利用ニーズを見極めながら特定業務に特化したツールを順次開発・登録し、本AIエージェント基盤の機能を継続的に拡充していく予定です。また、AIが参照・引用するデータソースの作成元へのインセンティブと、引用頻度の向上が相互に循環すべく仕組みの検討を進めます。

中長期的には、ローカルLLMを活用して自社インフラ内で完結する構成へ移行し、IIJの多様なサービスと連携したパッケージとして、お客様に提供できる環境の実現を目指していきたいと考えています。

執筆者:

海老根 和徳（えびね かずのり）
IIJネットワークサービス事業本部 システム開発本部 AIプラットフォーム推進室
2005年IIJ入社。公共・民間向けシステム構築に従事し、要件定義から設計・構築・運用まで幅広く担当。クラウドサービスの提案・導入や技術体制整備を通じて売上拡大に貢献。副部長・室長としてAI活用による業務効率化や部門横断施策を牽引した後、現在はAI基盤の開発・企画を推進している。

December 2025 Vol.68

IIJ
Internet Initiative Japan

株式会社インターネットイニシアティブ(IIJ)について

IIJは、1992年、インターネットの研究開発活動に関わっていた技術者が中心となり、日本でインターネットを本格的に普及させようという構想を持って設立されました。
現在は、国内最大級のインターネットバックボーンを運用し、インターネットの基盤を担うと共に、官公庁や金融機関をはじめとしたハイエンドのビジネスユーザに、インターネット接続やシステムインテグレーション、アウトソーシングサービスなど、高品質なシステム環境をトータルに提供しています。
また、サービス開発やインターネットバックボーンの運用を通して蓄積した知見を積極的に発信し、社会基盤としてのインターネットの発展に尽力しています。

本冊子の情報は2025年10月時点のものです。

IIJ-MKTG019-0068

株式会社インターネットイニシアティブ
〒102−0071 東京都千代田区富士見2-10-2　飯田橋グラン・ブルーム
E-mail：info@iij.ad.jp　URL：https://www.iij.ad.jp