5A-01

プロキシ環境における制御機器とクラウド間の通信性能評価

上野貴之† 山田竜也† 堀井圭祐† 追立慎吾†

三菱電機株式会社 情報技術総合研究所†

1. はじめに

リモート制御や運用効率化などを目的として，制御システムにおけるクラウド導入が注目を集めている．クラウド上で制御処理を行う場合，制御機器とクラウド間の通信遅延によりリアルタイム性が低下する懸念があり，システムが要求するリアルタイム性を達成できなくなるという課題がある．そこで，制御システムを模擬した環境において制御機器とクラウド間の通信性能を評価する必要がある．制御機器が外部ネットワークと接続する場合，一般的に外部ネットワーク上の不正アクセスから制御機器を保護するためにプロキシサーバが設置される．Amazon Web Service（以下，AWS）では，ローカルネットワーク内にプロキシサーバが設置された環境（以下，プロキシ環境）における機器とクラウド間の安全な接続を確立するために，AWS IoT Secure Tunneling [1]（以下，Secure Tunneling）というサービスを提供している．

本稿では，プロキシ環境において Secure Tunneling を用いてクラウドから制御処理を実現するアーキテクチャを検討した結果と，クラウドと制御機器間の通信性能を評価した結果を示す．

2. Secure Tunneling の概要

Secure Tunneling は，機器とクラウド間が接続するための専用トンネルを作成するサービスである．機器とクラウドのそれぞれがトンネルのエンドポイントへ接続することでトンネルを介した通信が可能となる．AWS では本エンドポイントに TCP 接続するためのエージェントソフトウェアとして，AWS IoT Secure Tunneling Local Proxy [2, 3]（以下，Local Proxy）を提供している．Local Proxy は，パケットをカプセル化することで，プロキシサーバが提供するセキュリティレベルを維持しつつ機器がプロキシをバイパスすることを可能にする．

3. システム構成

プロキシ環境において AWS クラウドから制御処理するために必要となるシステム構成を図 1 に示す．

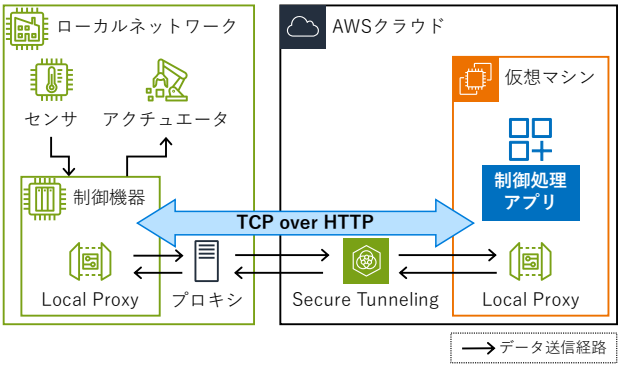

図 1 システム構成

図 1 において，制御処理アプリはクラウド内の仮想マシン上で実行される想定である．AWS では，仮想マシン構築サービスとして Amazon Elastic Compute Cloud [4]（以下，EC2）を提供している．仮想マシンは，センサから取得したデータを受信し，制御処理アプリの出力結果をアクチュエータへ送信する．仮想マシンとセンサ/アクチュエータ間の送受信時には，Secure Tunneling，プロキシ，制御機器を経由する．

Secure Tunneling のエンドポイントへ接続するために，制御機器および仮想マシンそれぞれで Local Proxy を実行する．Local Proxy は TCP しか転送しない仕様となっているため，制御機器と仮想マシン間の通信プロトコルは TCP に制限される．

4. 通信性能評価

3 章で示したシステムにおける制御機器とクラウド間の通信性能を評価するために，図 2 に示す環境を構築する．制御機器とクラウド間の通信性能が評価対象であるため，図 1 にあるようなセンサおよびアクチュエータは本評価では用意しない．本評価では通信性能として，図 2 における EC2 インスタンスから疑似制御機器へのパケット送信時のジッタを測定する．

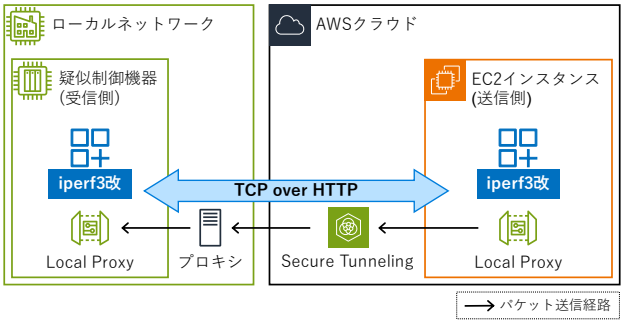

図 2 通信性能評価環境

制御処理を模擬するために，測定ツールには iperf-3.1.3 [5, 6]をベースとしてパケット周期送信機能を追加した自作ソフトウェア（以下，iperf3 改）を使用する．iperf3 改は，一定サイズのパケットを一定周期で送信したときのジッタ平均値を測定するためのツールであり，周期送信の精度は 1 ミリ秒である．本評価では表 1 のとおり，パケットサイズを 2 パターン，送信周期を 3 パターン用意し，合計で 6 通りの組合せでジッタの平均値を測定する．

表 1 パケットサイズと送信周期の組合せ

項目	組合せ		
パケットサイズ	1 KB	10 KB	
	×		
送信周期	10 msec	100 msec	1 sec

本評価で用いるジッタ平均値の定義は次のとおりである．

$$\text{ジッタ平均値[msec]} = \frac{\sum|\text{パケット受信間隔[msec]} - \text{パケット送信周期[msec]}|}{\text{送信パケット数} - 1}$$

上記定義におけるパケット受信間隔とは，疑似制御機器におけるパケット受信時刻の今回値と前回値の差分である．

その他の測定条件は次のとおり．

- 測定時間は 5 秒とする．
- ネゴシエーションによるオーバーヘッドの影響をできる限り小さくするために，測定開始 1 秒前からパケットを送信しておく．
- ACK 送信による影響をできる限り小さくするために，ウィンドウサイズは充分大きな値とする．
- プロキシとしては，ローカルネットワークに設置されているプロキシを用いる．

Evaluation of communication performance between cloud and device behind proxy

† Information Technology R&D Center, Mitsubishi Electric Corporation
Takayuki Ueno, Tatsuya Yamada, Keisuke Horii, Shingo Oidate

また，今後 NAT 環境での測定の比較にも使えるようにするため，疑似制御機器側から EC2 インスタンスへ接続する[1]．

評価に用いる疑似制御機器および EC2 インスタンスの動作環境を表 2，表 3 に示す．

表 2　疑似制御機器の動作環境

H/W	CPU	Intel® Core™ i5-8500 CPU @ 3.00GHz × 6 core
	メモリ	32 GB
S/W	OS ディストリビューション	Ubuntu 22.04 LTS
	Linux カーネル	5.15.0-91-generic
	Local Proxy	3.0.2

表 3　EC2 インスタンスの動作環境

H/W	CPU	Intel® Xeon® CPU E5-2676 v3 @ 2.40GHz × 2 core インスタンスタイプ[2]：t2.large [7]
	メモリ	8 GB
S/W	OS ディストリビューション	Ubuntu 22.04 LTS
	Linux カーネル	6.2.0-1015-aws
	Local Proxy	3.0.2

5.　結果および考察

4 章で説明した 6 通りの組合せに対して，それぞれ 5 回測定した結果を図 3，図 4 に示す．図 4 は，1 KB/1 sec および 1 KB/100 msec の結果を拡大した図となっている．なお，今回の測定では TCP によるパケット再送は発生しなかった．

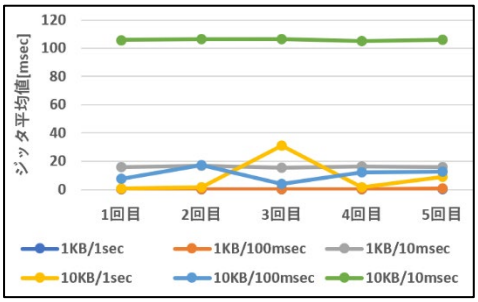

図 3　全組合せの測定結果

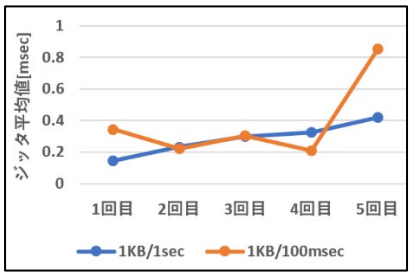

図 4　1 KB/1 sec および 1 KB/100 msec の測定結果

図 3，図 4 より，次のことが確認できる．

- 1 KB/1 sec および 10 KB/1 sec の測定結果の比較より，パケットサイズが大きくなることで測定結果のばらつきが大きくなる傾向があることがわかる．1 KB/100 msec および 10 KB/100 msec の測定結果を比較しても同様の傾向がある．10 KB のパケットは単一のイーサネットフレームに収まらないため，1 KB の場合と比較して受信側の負荷が大きくなった結果，動作が不安定になったと考える．
- 1 KB/1 sec, 1 KB/100 msec および 1 KB/10 msec の測定結果の比較より，パケットサイズが 1 KB の場合，送信周期を 10 msec まで短くしても測定結果のばらつきは小さいままであることがわかる．
- 送信周期が 10 msec である 2 つの組合せを除く，4 つの組合せにおいてジッタ平均値が送信周期を大きく下回っている．そのため，上記 4 つの組合せによる制御処理は，ユースケースによっては実用可能である．
- 10 KB/10 msec の測定結果は，より長い送信周期である 10 KB/100 msec の測定結果よりもばらつきが小さくなっている．本事象は，Local Proxy またはプロキシにおいてパケットが詰まっていることが原因であると考える．プロキシのスループットは 1Mbps を上回っているため，Local Proxy の転送速度上限が 1Mbps 程度であると予想する．

6.　おわりに

今回，プロキシ環境において Secure Tunneling を用いてクラウドから制御処理を実現するアーキテクチャを検討し，クラウドと制御機器間の通信性能としてジッタ平均値を測定した．通信性能測定は，パケットサイズおよび送信周期の異なる複数の組合せで実施した．測定結果より，送信周期が 10 msec である 2 つの組合せを除く，4 つの組合せではジッタ平均値が送信周期を大きく下回り，ユースケースによっては実用可能であることがわかった．

制御機器とクラウド間の通信がインターネットを介する以上，通信性能の限界は存在するため，今回のような通信性能評価を通してクラウド上で行うべき制御処理とローカルで行うべき制御処理を選別していくことが重要であると考える．

今回は，Local Proxy が TCP でしか転送できないという仕様により，制御機器および仮想マシン間の時刻同期が困難であったため，レイテンシの測定を実施しなかった．今後は，プロキシ環境におけるレイテンシについても評価方式を検討し，評価する予定である．

参考文献

[1] “AWS IoT secure tunneling - AWS IoT Core (amazon.com)”, https://docs.aws.amazon.com/iot/latest/developerguide/secure-tunneling.html [アクセス日: 2023 年 12 月 25 日]

[2] “Local proxy - AWS IoT Core (amazon.com)”, https://docs.aws.amazon.com/iot/latest/developerguide/local-proxy.html [アクセス日: 2023 年 12 月 25 日]

[3] “GitHub - aws-samples/aws-iot-securetunneling-localproxy: AWS Iot Secure Tunneling local proxy reference C++ implementation”, https://github.com/aws-samples/aws-iot-securetunneling-localproxy [アクセス日: 2023 年 12 月 25 日]

[4] “Cloud Compute Capacity - Amazon EC2 - AWS”, https://aws.amazon.com/ec2 [アクセス日: 2023 年 12 月 25 日]

[5] “iPerf - Download iPerf3 and original iPerf pre-compiled binaries”, https://iperf.fr/iperf-download.php [アクセス日: 2023 年 12 月 25 日]

[6] “GitHub - esnet/iperf: iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool”, https://github.com/esnet/iperf [アクセス日: 2023 年 12 月 25 日]

[7] “Amazon EC2 T2 Instances – Amazon Web Services (AWS)”, https://aws.amazon.com/ec2/instance-types/t2 [アクセス日: 2023 年 12 月 25 日]

1 本処理は元々iperf-3.1.3 に備わるリバースモード機能によって実現する．

2 EC2 のハードウェア構成はインスタンスタイプによって決まる．今回は汎用向けのインスタンスタイプを選択した．