Vol.2019-CSEC-85 No.1
Vol.2019-IOT-45 No.1
2019/5/23

ロボットオペレーティングシステムのセキュリティ機能に関する考察

亘理 克好[1,a)]　大久保 隆夫[1,b)]

概要：ロボット産業は 2035 年には 9.7 兆円規模の産業へ成長すると予想されており，産業用ロボットだけでなく，サービス分野のような新しい分野の成長が期待されている．ロボットの増加は，新たなサイバー攻撃のターゲットとされる可能性があり，特に人の近くで動作するようなサービスロボットには深刻な問題になりかねない．様々なハードウェアによる様々な機能を持つロボットの開発にはミドルウェアがよく使用される．本稿では，ミドルウェアの中でも広く使用されている“ロボットオペレーティングシステム (ROS)”に着目する．2017 年 12 月にリリースされた次期バージョンの ROS2 はセキュリティに対応していると言われており，通信の暗号化などがサポートされている．ROS2 暗号化についてリアルタイム性を考慮するため様々な状況で性能測定を行った結果について示す．

キーワード：ロボットオペレーティングシステム，ROS，ROS2，ミドルウェア，セキュリティ

A Study on Security Function of Robot Operating System

Abstract: The robot industry is expected to grow to an industry of 9.7 trillion yen in 2035. Not only industrial robots but also new fields like service field are expected to grow. An increase in robots leads to the possibility of becoming a target of a new cyber attack and can become a serious problem especially for service robots that operate near people. Under such circumstances, OS or middleware for robots are drawing attention, which greatly improves the complexity in developing robots with various functions by various hardware and efficient development in a short period of time. In this paper, paying attention to “ Robot Operating System (ROS) ’which is widely used in the world among middleware, we present overview of ROS and problems of security along with prior research. In addition, we also shows the survey of ROS2 released in December 2017.

Keywords: Robot Operating System, ROS, ROS2, Middleware, Security

1. はじめに

近年，様々な分野でロボットの活躍が期待されている．**図 1** は国立研究開発法人新エネルギー・産業技術総合開発機構 (NEDO) が 2010 年に公表したロボット産業の将来市場予測 [1] である．ロボット産業は 2035 年には 9.7 兆円規模の産業へ成長すると予想されている．

様々な機能を持つロボットに対し，より容易に開発を可能にするロボット用のミドルウェアを使うことが主流になってきている．ロボットは個別機能を分散処理し機能を

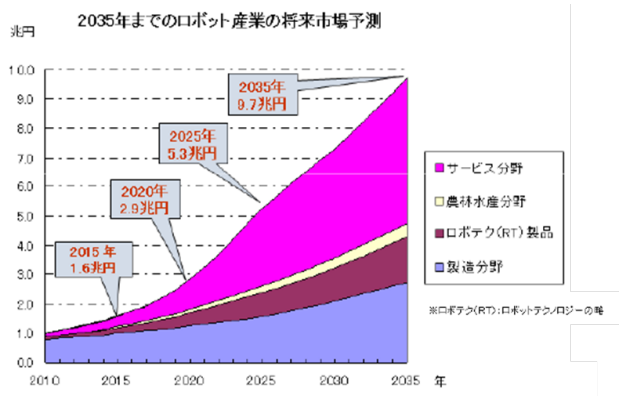

図 1　ロボット産業の将来市場予測：ロボット産業は 2035 年には 9.7 兆円規模の産業へ成長すると予想されている

[1] 情報セキュリティ大学院大学
221–0835 神奈川県横浜市神奈川区鶴屋町 2-14-1
[a)] mgs177501@iisec.ac.jp
[b)] okubo@iisec.ac.jp

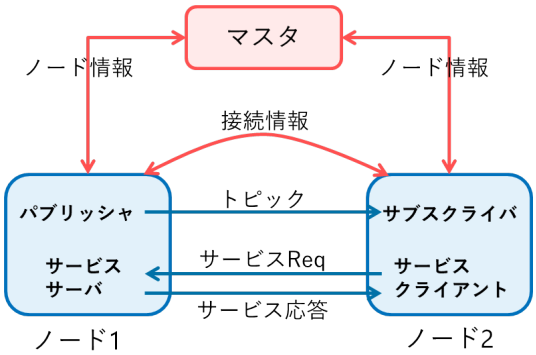

図 2 ROS の通信モデル：システムに一つ存在するマスタノードが各ノードの情報を持ち，各ノードはその情報をもとにお互いに通信を行う

実現するが，それに適したソフトウェア環境を使用することで，開発期間を短縮し，再利用可能なモジュールにより様々な機器を利用可能といった利点がある．ロボット用ミドルウェアには複数の種類があるが，その中でもオープンで利用者が多いのが“ロボットオペレーティングシステム（ROS）”である．ROS は Ubuntu Linux などの OS 上で動作するミドルウェアである．

ROS には，通信ライブラリ，分散システムの起動管理，パッケージ管理など有用なツールが多々用意され，ユーザコミュニティも活発に活動しており世界中から 5,000 を超える ROS パッケージと 80 種類以上のセンサがサポートされている．

2017 年 12 月には様々な機能拡張された後継バージョン ROS2 もリリースされている．

また，最近の ROS の適用例では，NASA による宇宙ロボット開発，Tier4 による自動運転車，SONY による AIBO など幅広い分野で利用されており，今後もこの傾向は続くと考えられる．

このような背景のもと，ロボット OS の共通基盤として広く使われている ROS に着目し，ロボットに必要とされるリアルタイム性が、ROS のセキュリティ対策によってどの程度影響を受けるのかを知ることが、ROS のセキュリティ向上にとって重要であると考えておりモチベーションとしている。そのため、本稿ではセキュリティ対策の一つとしての ROS の暗号通信時間の測定を行った。

2. ROS の動作モデル

2.1 ROS 通信の概要

ROS の動作モデル，通信モデルの概念図を**図 2** に示す．また，トピック通信フローの詳細を**図 3** に示す．

ROS では，実行可能な最小単位のプロセスをノードと呼びその集合で全体機能を実現する．

ノードの中でもマスタと呼ばれる特別なノードが DNS のネームサーバのように各ノード情報を保持する．このマスタはシステムに一つだけ存在する．

個別の機能を持つその他の各ノードはノード情報をマスタと交換し，他の実行中のノード情報をマスタから得て，その情報をもとにノード間の通信を行う．

ノード間の通信は用途に応じていくつかの種類が用意されているが，主要な方式はトピック通信とサービス通信である．トピック通信は，パブリッシャと呼ばれる送信者ノードからサブスクライバと呼ばれる受信者ノードへの一方向で連続的なデータを送信する通信方式である．センサデータなどの通信のために使われる．サービス通信はクライアントノードからのリクエストに応じてその都度サーバノードが応答を返す双方向の通信方式である．

2.2 セキュリティ問題

図 3 で示した ROS の通信フローでは，ノード間の通信は暗号化されておらず，またノードが通信を行う際のノードチェックのための認証は行われない．このため，マスタの起動ホストとポートが判明すれば，悪意者によって新たに偽のノードを作成しマスタに接続することが可能になる．

マスターに接続したノードは他のノード情報を取得することができ，その情報のみで他の正しいノードに接続することができるため，悪意ノードから偽のデータを流すことが可能となる．

デフォルトの ROS ではセキュリティについては十分に考慮されていないといえる．

2.3 ROS 確認テスト

セキュリティの問題を確認するため実際に ROS を使った動作確認を行った．

小さな車輪型のロボット GoPiGo3 を使用し，RaspberryPi 上の ROS で動作する簡易なロボットシステムを作成した．構成を**図 4** に示す．図において RaspberryPi 内の一つ一つの枠が ROS ノードである．

ロボットに接続したカメラからの画像をスマートフォンで確認しながら，ロボットを操作することを想定している．スマートフォン上の WebUI で操作した前後左右の指示情報を websocket ノードを通して送り，cmd_vel トピックでドライブノードに送信する．トピックを受信したドライブノードはその情報を元に実際に車輪を動かす．

このようなシステムにおいて悪意者ノードとして，ドライブノードに cmd_vel トピックを送信する新たなノードを起動し，ROS システムに参加する．cmd_vel はスマートフォンからの正しいトピックと同じ名前であり，参加後スマーフォンの操作とは関連のない別の指令値を悪意者ノードから送信する．結果として，スマートフォンの操作とは違う動作が実行されてしまうことが確認できた．

3. ROS のセキュリティに関する先行研究

このような問題に対し検討している先行研究がいくつか

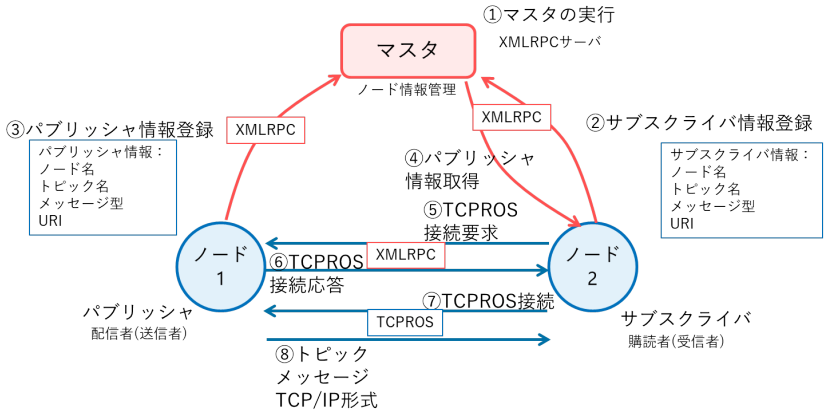

図 3 ROS トピック通信の接続フロー：マスタへの登録と他ノードの情報取得，ノード間の通信は暗号化されず，ノードの参加時にノードに対する認証はされていない

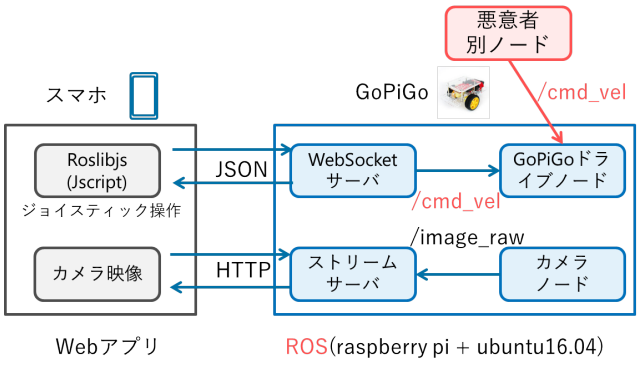

図 4 ROS 動作確認用簡易システム：カメラの画像をスマーフォンで確認しながら ROS で動作するロボットを操作する．悪意者ノードは正しいトピックと同じ名前の偽データを送信する

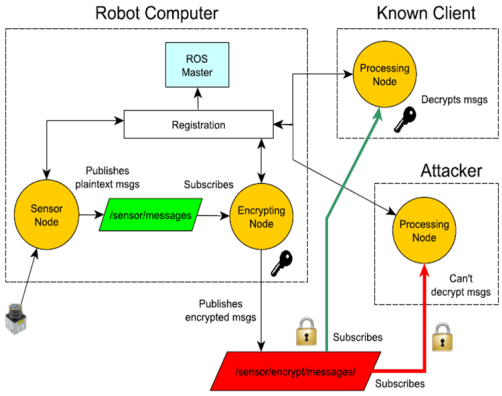

図 5 暗号化ノードを追加しトピックを中継することで通信を堅牢化する（出典 [3]）

ある．Bernhard Dieber らは通信の暗号化とファジングテストツールを提案 [2] し，Francisco Javier らは通信を暗号化する中継ノードを設置することを提案 [3] している．さらに，ノードを認証する別のノードを用意し，ノードのログインリクエストに対し，認証と権限を応答するようにするもの [4]，認証サーバを設置し，信頼できるノードのリストを持ち，サブスクライバがそれを受け取りそれを元に接続するもの [5]，通信監視ノードを追加し，事前定義されたモデルのランタイム検証を使用して予想される動作に適合しないアクションを検出するもの [6]，TLS を用い認証付き暗号で通信を行い，アクセス制御が可能であるもの [7]，など様々な提案がされている．

これらのうち暗号化ノードの追加と監視ノードの追加について概要を説明する．

3.1 堅牢化 ROS のパフォーマンス [3]

この研究では ROS を堅牢化した場合のパフォーマンスについて実際に提案手法を構築し，パフォーマンスを測定している（**図 5**）．3DES 暗号化アルゴリズムを使用した提案システムを構築しテストした結果，共通鍵を使うこの方法は重大な遅れが生じることが判明した．

このような暗号化を考える上では，リアルタイムで動作する分配アーキテクチャを検討すべきとしている．今後としては ROS2 で使われている Real-Time Publish Subscribe(RTPS) を同様にテストし，さらに DDS ベースの通信について長所と短所を評価していくとしている．

実際にこの研究で提案され，テストに使用されたシステムを図 5 に示す．ロボットコンピュータにはセンサーノードと暗号化ノードの 2 つの ROS ノードがあり，暗号化ノードはセンサーノードから受信したデータを暗号化し，Known Client に送る．キーはそれぞれが持っているので，Known Client は復号することができる．攻撃者はキーを知らないので，復号することができずデータを読むことができない．

測定結果としては，サイズの大きいデータに対しては実行時間のうち 98 %もの時間が暗号化に使用されていることがわかった．暗号化を容易に導入することはパフォーマンスに影響を与えることが示された．

3.2 ROSRV:ロボットのための実行時検証 [6]

この研究では，事前に定義されたモデルのランタイム検

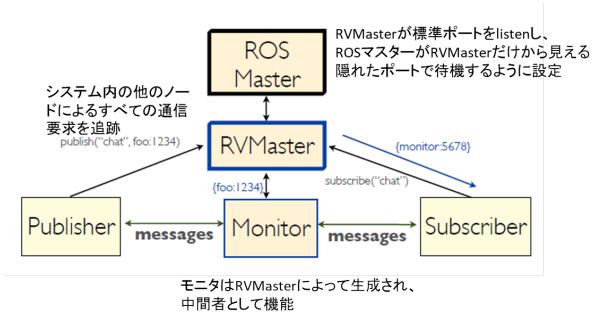

図 6 ROSRV アーキテクチャ：RVMaster が全ての接続要求を追跡，Monitor ノードを生成しノード間通信をチェックする（出典 [6]）

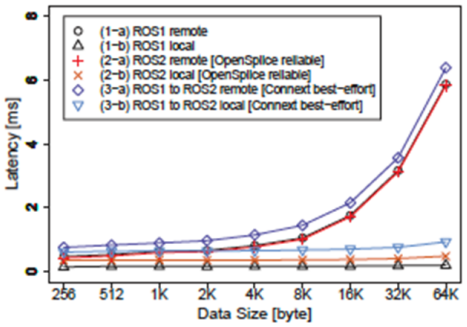

図 7 ROS2 における DDS の違いによる性能評価結果 ([8] より抜粋)

証を使用し，予想されるアプリケーションの動作に適合しないアクションを検出して不正動作を防止する手法を提案している（**図 6**）．

適合アクションは事前に定義しておき，システムの動作は自動で生成されるモニターノードによってチェックされる．この方法では，ROS 自体もアプリケーションコードもどちらも変更しなくてよいという利点がある．

弱点としては，IP アドレスに基づいてノードの認証を実行するため，ROS ノードを実行するホストにアクセスする攻撃者はこれを回避することができてしまうことである．また，全ての通信をチェックするため，スケーラビリティのための複数マスターノードの実現が必要になる．

4. ROS2

4.1 ROS2 概要

ROS2 は 2017 年 12 月に正式リリースされた ROS であり，デフォルトでセキュリティをサポートしており，環境変数の設定によりセキュリティの設定が可能である．ただし，これまでの ROS(以後 ROS1) との互換性は持っておらず，ROS1 と ROS2 の間の通信をサポートするブリッジが用意されている．また，通信方式が大きく変わっており，DDS (Data Distribution Service) を使用する．DDS は OMG で規格化されている通信ミドルウェアであり，複数の実装があるが特に指定しない場合はデフォルトでは FastRTPS が使われる．他にプラットフォームに Windows が含まれていること，DDS の使用に伴い ROS マスターが使われなくなっていることが大きな変化点である．

4.2 ROS2 に関する先行研究

ROS2 の先行研究としては，組み込み機器の観点からパフォーマンスを測定したものがある [8]．これは用途からリアルタイム性にフォーカスしており，DDS の複数の実装の比較を行ったものである．ローカル PC 内に閉じたノードだけでなくリモート PC 間や ROS1 と ROS2 の混在の場合についても考慮したテスト条件になっている．結果の一部を**図 7** に示す．DDS の実装により性能が異なることが明らかになり，それらを踏まえて実際にどの DDS 実装を使用するかを検討する必要があると結論づけている．

ただし，セキュリティについては特に触れておらず，また評価している ROS2 がアルファ版となっていることや，現在では実装が中断されている DDS についても検討されている．

また，DDS 単体に関しては，Pardo らは暗号化に対するオーバヘッドを計測しており [9]，1 %から 40 %のオーバヘッドがあるとの結果を示している．

DDS に限らずパブリッシュ/サブスクライブシステム全体のセキュリティに関しては，Christian らは 1998 年から 2014 年までの学術分野における文献の調査と分析を行なっている [10]．学術文献でのセキュリティ対策と製品レベルのセキュリティ対策にはまだギャップがあるとし，暗号化などのセキュリティ対策はオーバーヘッドになることから，どのようなセキュリティ対策が最良であるかを決定するための適切なセキュリティ評価手段を持つことが大切であるとしている．さらに，最近では，Belguith らは悪意のあるサブスクライバを無効化する手法を提案している [11]．パブリッシュ/サブスクライブシステムの機能に影響を与えず，新しい鍵の再生成や再配布を行うことなく実現できる手法である．

4.3 ROS2 想定システム

このような ROS2 に対し，ROS2 ノードで動作するテスト用の車輪ロボットとコントロール PC を使うシステムを想定し構築した．構成を**図 8** に示す．

このシステムでは一般的な移動型ロボットが移動に関して行う機能を想定している．ロボットはコントロール PC からの指令により移動を行う．移動中にロボットは LIDAR からのデータにより周りの障害物を検知し，地図を作成する．地図の作成後は，目的地のみの指示により自動で経路移動を行うことができる．

ノード構成は**図 9** のようになっており，ROS2 ノードは RemotePC 内の 4 つの円で示されている．悪意者からの攻撃としては，移動指示をなりすまし偽の指示を送信するこ

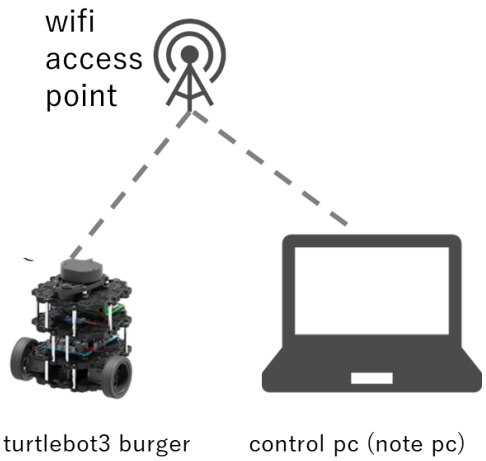

図 8 リモート PC(control pc) と車輪ロボット (turtlebot3) を使った ROS2 想定システム

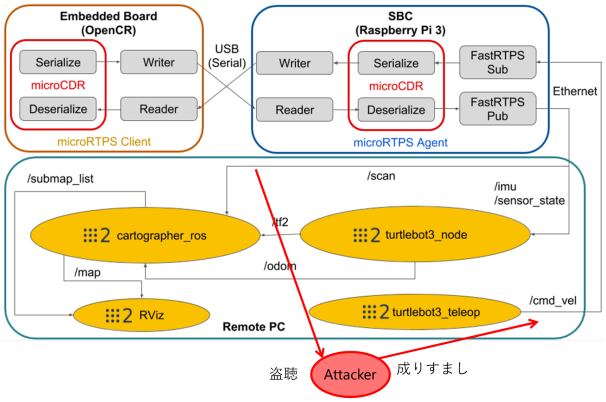

図 9 想定システムでのノード構成と悪意者による脅威
(RemotePC 内の 4 つの円が ROS2 ノード)
(ノード構成引用 turtlebot3 emanual)

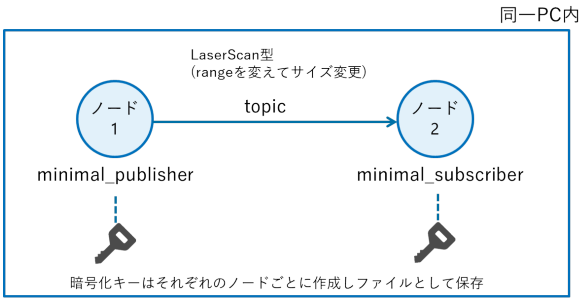

図 10 ROS2 における通信テストノード

と，LIDAR のデータを盗聴し他の地図を作成することを想定している．

4.4 ROS2 暗号通信時間測定

本稿では，このうち盗聴を想定した ROS2 の暗号機能を使用した場合の通信時間の測定を実施した．

図 10 のようにデータを送信するノード 1 とデータを受信するノード 2 を C++にて作成した．ノード 1 は 100ms ごとにデータを送信する．通信時間の測定はノード 1 の

```
std_msgs/Header header
float32 angle_min
float32 angle_max
float32 angle_increment
float32 time_increment
float32 scan_time
float32 range_min
float32 range_max
float32[] ranges
float32[] intensities
```

図 11 テストに使用した LaserScan 型

表 1 テストに使用した複数 OS の PC スペック

Type	LG gram 13Z980-GA56J	Macbook pro 2018
CPU	Intel Core i5-8250U 1.6GHz	Intel Core i7 2.7GHz
Mem	8GB	16GB
Graphics	Intel UHD 620	Intel Iris 655
Disk	SSD256GB	SSD1TB
OS	ubuntu 18.04LTS	macOS 10.14.4
ROS	ROS2 crystal	ROS2 crystal
DDS	FastRTPS	FastRTPS

送信時刻およびノード 2 の受信時刻の差分をとり 100 通信分の平均を結果とした．送信データは前述した LIDAR のデータ通信（/scan）で使われる LaserScan 型を使用し，サイズを変えて実施した．LaserScan 型は ROS2 であらかじめ定義されている構造体であり，内容を**図 11** に示す．今回は 1 回の通信ごとにトピックが受信されるのを確認するために LaserScan 型にある header 内に通信ごとに通し番号を代入した．また，通信データ量を変更するために，LaserScan 型の ranges の配列サイズ（以降データサイズと呼ぶ）を変更してテストした．配列サイズは 100 から 100000 まで 10 倍ごとに変更した．暗号化については ROS2 のデフォルトである TLS を使用し，暗号に使うキーは publisher と subcriber のノード毎に作成し同一 PC 内のディレクトリにファイルとして保存した．

また，ROS2 は複数の OS をサポートしているため，OS による違いの確認として，Linux と MacOS のそれぞれの上でテストを実施した．

使用した PC と ROS2 のスペックを**表 1** に示す．

このスペックの PC にてそれぞれの PC 内で 2 ノード間のトピック通信を行い，データサイズと暗号設定を変えた場合の測定結果を**表 2** に示す．Linux の場合のグラフを**図 12** に，MacOS の場合のグラフを**図 13** に示す．

5. 考察

平文の場合，通信時間は Linux，MacOS ともにデータサイズが 10K 程度までは 0.5ms 程度であるが，それを超えると 1ms 以上まで増加している．さらにデータサイズが増え 1M になると MacOS では通信時間が 4ms 程度であるが，Linux では 12ms 程度必要となる．

表 2 OS の違いによる同一 PC 内ノード間の通信時間測定結果

ranges		Linux 通信時間 [ms] 暗号化 (e)	平文 (p)	e/p 比	MacOS 通信時間 [ms] 暗号化 (e)	平文 (p)	e/p 比
100	平均	1.02	0.63	1.61	0.69	0.48	1.45
	標準偏差	0.32	0.16		0.11	0.09	
1000	平均	1.22	0.78	1.57	0.72	0.52	1.39
	標準偏差	0.16	0.07		0.11	0.09	
10000	平均	1.51	0.88	1.72	0.93	0.58	1.61
	標準偏差	0.16	0.13		0.16	0.12	
100000	平均	4.08	1.98	2.06	1.93	0.96	2.01
	標準偏差	1.58	1.62		0.31	0.16	
1000000	平均	27.48	12.82	2.14	7.39	4.07	1.82
	標準偏差	8.23	4.24		1.08	0.52	

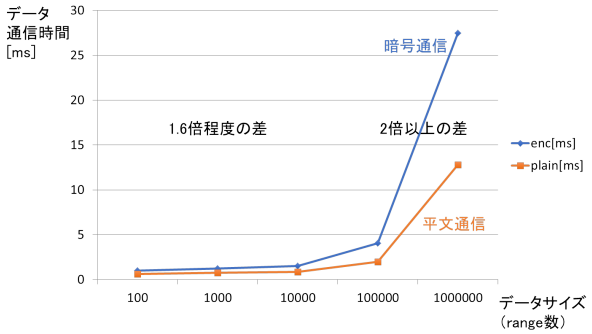

図 12 ROS2 における同一 PC 内 2 ノード間の平文および暗号通信時間測定結果 (Linux)

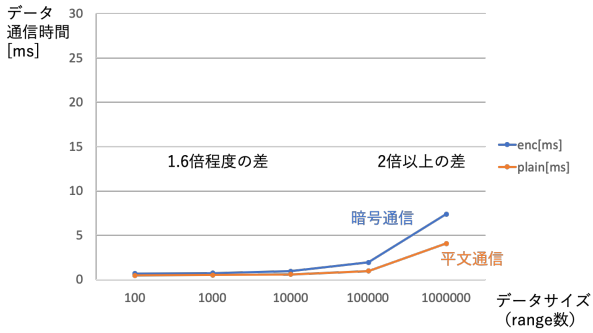

図 13 ROS2 における同一 PC 内 2 ノード間の平文および暗号通信時間測定結果

また，暗号化した場合，通信時間はデータサイズが 10K 程度までの比較的小さいうちは，平文の約 1.6 倍の 1ms 程度であるが，データサイズが 100K 以上に増えてくると 2 倍程度通信時間が必要となり，Linux では最大 27ms 必要となっている．

リアルタイム性が必要とされるロボットの場合，単純にシステム内のノード間通信を暗号化することは，通信時間の増加が動作に影響を与える場合があると考えられる．実際のロボットシステムではノード数は今回実験した 2 ノードより多く存在することが一般的であり，その場合はさらに通信時間が増加すると考えられる．ただし，この時間の遅れが具体的にリアルタイム性にどう影響するのかまでは試せていない．

次に OS の違いについては，PC のスペックが異なることから単純な比較はできないが，どちらの OS の場合もデータサイズの増加に対しては同じような傾向を示している．OS によらずデータサイズが 10K 程度までは平文の場合より暗号化通信が 1.6 倍，10K を超えると暗号化通信に必要な通信時間が平文の場合と比べ 2 倍程度必要となる．

これらから，データサイズが 100K を超えるような場合は特に通信時間をよく検討した上で暗号化通信を使用することが重要であると言える．また，ノードの増加につれてノード間の通信も増えていくと考えられることから，セキュリティのためにすべてを暗号化するのではなく、それぞれのロボットシステムに合わせて適切に選択することが大切といえる．

6. おわりに

ROS のセキュリティの問題点から，セキュリティを強化する先行研究を調査し，実際に悪意者を想定した攻撃を車輪型簡易ロボットシステムで行うことで確認した．また，セキュリティが組み込まれていると言われている ROS2 について，動作概要を調査し，想定システムについて実際の簡易ロボットにて動作確認を行った．セキュリティ機能について ROS2 ではノード間の暗号化がサポートされているが，ロボットシステムでは通信時間が動作に影響を与える場合もあり，暗号化と平文での通信時間について計測した．結果としてデータサイズの増加により暗号化は約 2 倍の通信時間が必要となることがわかった．今後は ROS2 における暗号通信についてリモートでの場合や OS 違いなどの追加テストを行い，もう一つの機能である権限設定について調査する予定である．

参考文献

[1] NEDO: ロボットの将来市場予測を公表，https://www.nedo.go.jp/news/press/AA5_0095A.html.
[2] D, B., B, B., T, S., K, S., R, S. and S, P.: Secu-

rity for the Robot Operating System, *Robotics and Autonomous Systems*, Vol. 98, pp. 192–203 (online), DOI: 10.1016/j.robot.2017.09.017 (2017).

[3] F.J.R.Lera, J.Balsa, F.Casado, C.Fernandez, F.M.Rico, V.Matellan: Cybersecurity in Autonomous Systems: Evaluating the performance of hardening ROS, *Workshop on physical Agents*, p. 47 (2016).

[4] R. Dczi, F. Kis, B. St, V. Pser, G. Kronreif, E. Jsvai, M. Kozlovszky: Increasing ROS 1.x communication security for medical surgery robot, *IEEE International Conference on Systems, Man, and Cybernetics(SMC)*, pp. 004444–004449 (2016).

[5] B. Dieber, S. Kacianka, S. Rass, P. Schartner: Application-level security for ROS-based applications, *Intelligent Robots and Systems (IROS)*, pp. 4477–4482 (2016).

[6] J. Huang, C. Erdogan, Y. Zhang, B. Moore, Q. Luo, A. Sundaresan, G. Rosu: ROSRV: Runtime Verication for Robots, *Springer International Publishing*, pp. 247–254 (2014).

[7]] R. White, H. I. Christensen, M. Quigley: SROS: Securing ROS over the wire, in the graph, and through the kernel, *HUMANOIDS 16 Workshop Towards Humanoid Robots OS* (2016).

[8] Maruyama, Y., Kato, S. and Azumi, T.: Exploring the Performance of ROS2, *Proceedings of the 13th International Conference on Embedded Software*, EMSOFT '16, New York, NY, USA, ACM, pp. 5:1–5:10 (online), DOI: 10.1145/2968478.2968502 (2016).

[9] Gerardo Pardo, R. W.: DDS Security concepts for SROS, `https://ruffsl.github.io/IROS2018\_SROS2\_Tutorial/content/slides/Background.pdf`. (Accessed on 04/14/2019).

[10] Esposito, C. and Ciampi, M.: On Security in Publish/-Subscribe Services: A Survey, Vol. 17, No. 2, pp. 966–997.

[11] Belguith, S., Cui, S., Asghar, M. R. and Russello, G.: Secure Publish and Subscribe Systems with Efficient Revocation, *Proceedings of the 33rd Annual ACM Symposium on Applied Computing - SAC '18*, ACM Press, pp. 388–394.