HTML5とRESTに基づく着脱容易なファイアウォールの実現・制御法の提案

早川 智一[1,a)]　小泉 修一[1,b)]　疋田 輝雄[1,c)]

概要：本論文では，悪意のあるソフトウェアによって引き起こされる情報流出の検出を支援するために，ネットワークへの着脱が容易で Web サービスによって制御されるファイアウォールの構築法を提案する．提案手法は，REST（Representational State Transfer）による汎用的な Web サービスを用いて OS に固有のファイアウォール機能を抽象化し，その Web サービスに容易にアクセスするための Web アプリケーションを HTML 5 で提供することで，利用者がファイアウォールの動作を容易に把握・変更できるように支援する．我々は，プロトタイプとしてハードウェアに Raspberry Pi を用いた予備的な評価を行い，提案手法が実際の計算機上で正しく動作することを確認した．

1. はじめに

インターネット利用者の数が増えるに連れ，悪意のあるソフトウェアによって引き起こされる情報流出への対策の重要性が増してきている．たとえば，特定のソフトウェアが利用者の個人情報を外部に送信している（いた）といった報道や，USB のファームウェアに悪用可能な脆弱性が存在する [7] といった報道がなされたことは記憶に新しい．

このような脅威への対策として，市場には多くのセキュリティソフトウェアが存在する．これらのソフトウェアをコンピュータにインストールすることで，ユーザ単位やアプリケーション単位での細かい粒度でのチェックが可能になる．

一方で，そのようなソフトウェアのみで情報流出を完全に検出・遮断することは困難である．なぜならば，悪意のあるソフトウェアに OS（Operating System）が既に感染している場合には，当該 OS 上のソフトウェアが正しく動作することは必ずしも期待できないからである．

したがって，ソフトウェアによる情報流出の検出・遮断をより確実なものにするためには，ソフトウェア以外の手段を併用することが望ましい．具体的には，適切な設定を行ったファイアウォールを利用者のコンピュータとインターネットとの間に設置することが考えられる．これを実現するための選択肢としては，高価な専用のファイアウォール製品を購入することや簡易ファイアウォール機能を備えた安価なブロードバンドルータを購入することが考えられるが，どちらの選択肢にも得失が存在する．まず，専用のファイアウォール製品はセキュリティソフトウェアと比較して高価である場合が多く，予算との兼ね合いで導入障壁が高くなる可能性がある．次に，安価なブロードバンドルータのファイアウォール機能での代用は，

(1) 多くのブロードバンドルータはこのような用途に適した見やすい TCP/IP[*1] ログを出力しないため，数少ない（であろう）怪しい TCP/IP 接続が数多くの正常な TCP/IP 接続の中に埋もれてしまうこと，
(2) 多くのブロードバンドルータはファイアウォールのルールの変更時に再起動を必要とするため，利用者は設定を変更するたびに何度も当該ルータを再起動する必要に迫られ多くの手間がかかってしまうこと，
(3) ブロードバンドルータを使うと，怪しい TCP/IP 接続の検出・遮断が当該ルータの設置箇所（LAN/WAN の境界）でしか行えず，同一ネットワーク内に複数のコンピュータを接続している場合に，コンピュータ単位での情報流出の有無をチェックできないこと

――などの理由から困難であると我々は考える．

我々は，この問題を解決するために，

(1) 安価で，
(2) 動作や挙動の把握・変更が容易にできて，
(3) ネットワークへの着脱が容易な

[1] 明治大学 理工学部
School of Science and Technology, Meiji University, Kawasaki, 214–8571, Japan
[a)] t_haya@cs.meiji.ac.jp
[b)] s-koizumi@cs.meiji.ac.jp
[c)] hikita@cs.meiji.ac.jp

[*1] TCP/IP 以外にもプロトコルは存在するが，本論文ではその普及率や重要性を鑑みて，特に TCP/IP に焦点を当てて説明する．

ファイアウォールの構築法を提案する．なお，提案手法は，既存のセキュリティソフトウェアと併用して安全性を向上させることを目的とし，単体での使用はここでは考慮しないものとする．

提案手法は，ハードウェア（Linux 系 OS が動作するコンピュータであれば何でもよい）とソフトウェア（ファイアウォールの動作や挙動を把握・制御する Web サービス）とで構成される（2 節）．我々は，ファイアウォールの制御用 Web サービスを REST（Representational State Transfer）として設計し，そのフロントエンドとして HTML 5 で設計した Web アプリケーションを提供する．利用者は，Web ブラウザからこの Web アプリケーションにアクセスすることで，ファイアウォールの動作や挙動の把握・変更を容易に行うことができる．我々は，プロトタイプとして，安価に購入が可能なハードウェアである Raspberry Pi を選択し，これを用いてプロトタイプの実装と予備的な評価を行った（3 節・4 節）．

本論文の構成は次のとおりである．2 節と 3 節では設計と実装を概説する．4 節では予備的な評価の結果を報告する．5 節では関連研究を紹介する．6 節では今後の課題を述べる．

2. 提案手法の設計

2.1 前提

提案手法の前提は次のとおりである：

(1) 提案手法の想定利用者（以下，利用者）は，HTML 5 対応の Web ブラウザを操作できて，TCP/IP の概要を理解しているものとする．
(2) 利用者は，Ethernet で接続された IPv4 のネットワーク内で自分のコンピュータを利用しているものとする．
(3) 利用者のネットワークには，提案手法のために割り当て可能な 1 つ以上の余剰 IP アドレスが存在するものとする．

2.2 提案手法の概要

図 1 に，提案手法の概要を示す．

まず，提案手法の適用に必要なハードウェアは，Linux などが動作する一般的なコンピュータであり，2 つの Ethernet アダプタ（eth0 と eth1）が接続されていることを想定する．この eth0 と eth1 とを束ねて 1 つのブリッジとして動作させる（この際に，専用の IP アドレスが 1 つ必要になる）．そして，対象ネットワークのトラフィックがそのブリッジ間を経由するように設置して監視を行うことでファイアウォール機能を実現する．本提案手法は L2 ブリッジとして動作するので，ブリッジ用の IP アドレスさえ確保できれば，ネットワーク上の多くの場所に設置できるという特徴がある．

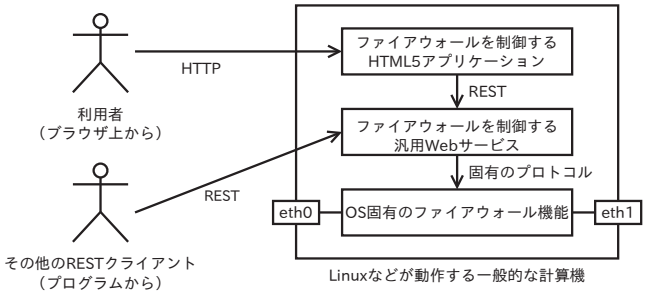

図 1 提案手法の概要

Fig. 1 Overview of our proposed method.

次に，OS 固有のファイアウォール機能を抽象化するためのレイヤーを，REST を用いた Web サービスとして提供する．これにより，要求仕様の変化などで OS の変更が必要になった場合でも，容易に OS を切り替えることができるようになる．

最後に，利用者の Web サービスへのアクセスを容易にするために，HTML 5 で記述した Web アプリケーションを提供する．これにより，利用者は REST のエンドポイントに直接アクセスすることなくファイアウォールを制御することが可能になる．なお，プログラムからファイアウォールを制御したい場合には，Web アプリケーションの存在が邪魔になる [*2]．この場合には，プログラムから Web サービスに直接アクセスさせることでファイアウォールの制御が容易になる．

2.2.1 制御用 Web サービスの概要

表 1 に，制御用 Web サービスのエンドポイントの抜粋を示す．ファイアウォールの動作モデルを REST のエンドポイントとして定義する場合には様々なモデルが考えられるが，我々は Linux の iptables のモデルを準用した．ここで，REST でエンドポイントを設計したことによる機能の拡充・改善の容易さが特徴として挙げられる．新たな機能が必要になった場合には，新たな REST エンドポイントを追加するだけで，既存の利用環境に影響を与えることなく実現することができる．

2.2.2 制御用 Web アプリケーションの概要

図 2 に，制御用 Web アプリケーションのスクリーンショットを示す（画面設計はあくまでも一例である）．

利用者は，制御用 Web アプリケーションにアクセスすることで，ファイアウォールを通過している現在の TCP/IP セッションの状態を把握することができる．制御用 Web アプリケーション上では，利用者の指定した情報によって各 TCP/IP 接続がグループ化され [*3]，デフォルトではファイアウォールへの最終到着時刻の降順で表示される（新しい接続ほど上に表示される）．

[*2] 一般的に，Web アプリケーションは人間用に作られており，プログラムにとっては必ずしも扱いやすいとは限らないため．
[*3] 設計に REST を用いているので，クエリパラメータなどを用いて，グループ化や範囲指定のための情報の指定が容易である．

表 1　制御用 Web サービスの REST エンドポイント（抜粋）
Table 1　REST endpoints for proposed controller Web service.

メソッド	URI	備考
GET	/policies	全チェイン [a] の許可・拒否ポリシ [b] の取得
GET	/policies/{chain}	チェイン毎の許可・拒否ポリシの取得
PUT	/policies/{chain}/{policy}	チェイン毎の許可・拒否ポリシの設定
GET	/logs	全プロトコルのログの取得
GET	/logs/{protocol}	プロトコル毎のログの取得
GET	/rules	許可・拒否ルールの全取得
GET	/rules/{protocol}	プロトコル毎の許可・拒否ルールの取得
POST	/rules/{protocol}	プロトコル毎の許可・拒否ルールの追加
PUT	/rules/{protocol}/{id}	プロトコル毎の特定の許可・拒否ルールの変更
DELETE	/rules/{protocol}/{id}	プロトコル毎の特定の許可・拒否ルールの削除

[a]　パケットを検査するルールのリストのこと（iptables の用語を準用）.
[b]　該当ルールがない場合に，接続を許可するか拒否するかの設定のこと（iptables の用語を準用）.

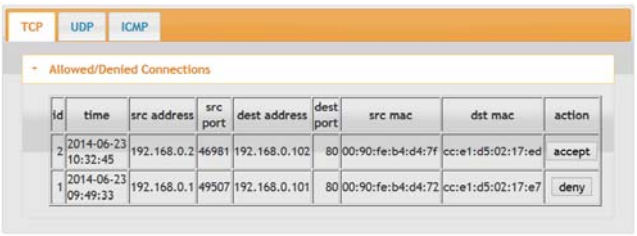

図 2　制御用 Web アプリケーションのスクリーンショット
Fig. 2　Screenshot of proposed controller Web application.

利用者は，それぞれの接続について，「accept」または「deny」ボタンを押下して接続の許可／拒否を決定することができる．ボタンを押下すると，「accept」ボタンは「deny」に，「deny」ボタンは「accept」に変化し，再押下することで許可／拒否を取り消す（反転させる）こともできる．ボタン押下時の設定変更は，システムを再起動することなく直ちに適用される．これにより，利用者は各々の接続について，必要であれば試行錯誤的に許可／拒否を繰り返して検証することが可能になる．このことは，（セキュリティソフトウェアでは検出されなかった）怪しい TCP/IP 接続を利用者が発見した場合に，当該接続を試しに遮断して様子を見ることができることを意味し，セキュリティ向上の一助になると我々は考える．

3.　プロトタイプの実装

3.1　ハードウェア実装

表 2 に，提案システムの実装に我々が用いたハードウェアを示す．我々は，提案手法を安価に実装するためのハードウェアとして，Raspberry Pi のモデル B を選択した（他のハードウェアでも動作する）．この選定理由としては，

(1) 本論文の執筆時点で 5,000 円前後で安価に購入できる；
(2) 電源をマイクロ USB で供給できるため，利用者のコンピュータの周囲で電源の確保が容易になる

——などが挙げられる．

表 2　プロトタイプの実装に用いたハードウェア
Table 2　Used hardware for prototype implementation.

ハードウェア	備考
Raspberry Pi	モデル B
Ethernet アダプタ	USB 3.0 接続・Gigabit 対応（AX88179）

我々は，提案手法に必要な Ethernet アダプタ 2 つを確保するために，USB 3.0 接続の Gigabit 対応の Ethernet アダプタ 2 つを接続し（ただし，Raspberry Pi は USB 2.0 までしか対応していない），これらをブリッジとして使用した．

3.2　ソフトウェア実装

3.2.1　OS の選定

我々は，Raspberry Pi の OS として，公式サイト [6] からダウンロードできる 6 種類の OS（Raspbian・Pidora・OpenELEC・RaspBMC・RISC OS・Arch Linux）の中から Raspbian を選択した（他の Linux でも動作する）．

3.2.2　OS 固有のファイアウォール機能の選定

表 3 に，ファイアウォール機能の実装に我々が用いたソフトウェアを示す．我々は，ファイアウォールの機能を，iptables と ulogd を用いて実装した．ここで，iptables は Linux のファイアウォール機能を制御するためのコマンドであり，ulogd は iptables が出力するメッセージを記録するためのデーモン（常駐ソフトウェア）である．処理の流れは次のとおりである：

(1) 利用者のコンピュータが TCP/IP 接続を開始すると，それに対応するパケットがファイアウォールに到着する．
(2) 当該パケット情報が，iptables に事前設定されたルールに従って，ログとして ulogd に転送される．
(3) ulogd は，iptables から転送されたログを SQLite デー

表 3　プロトタイプの実装に用いたソフトウェア
Table 3　Used software for prototype implementation.

ソフトウェア	バージョン	備考
ulogd	1.24	iptables 用のロギングデーモン
iptables	1.4.14	ファイアウォール
kernel	3.12.22+	Raspbian のカーネル
Raspbian	January 2014	Debian ベースの OS

タベースに記録する [*4].

(4) 制御用 Web サービス（後述）は，SQLite データベースから ulogd のログを読み出して処理する．

3.2.3　制御用 Web サービスの実装

表 4 に，制御用 Web サービスの実装に我々が用いたソフトウェアを示す．制御用 Web サービスは，PHP のマイクロフレームワークである Slim [4] を用いて REST として実装した．この Web サービスが，ファイアウォールの制御に必要なエンドポイントを提供する（機能の追加が必要であれば，エンドポイントを増やすことで容易に対応できる）．利用者は，後述の Web アプリケーションを用いて，これらのエンドポイントに対して各 HTTP リクエスト（GET・POST・PUT・DELETE）を発行することで，ファイアウォールの動作を把握・制御することができる．

3.2.4　制御用 Web アプリケーションの実装

表 5 に，制御用 Web アプリケーションの実装に我々が用いたソフトウェアを示す．制御用 Web アプリケーションは，制御用 Web サービスを利用者が容易に利用するためのフロントエンドであり，HTML 5 [1] と jQuery [9,10] を使って実装した．ここで，HTML 5 を採用した理由は，制御用 Web サービスからの更新通知を受け取るために Server-Sent Events [2] を使うためである．これにより，明示的なポーリング（問い合わせ）を行うことなく，制御用 Web サービスから変更通知を受け取ることが可能になる．

3.2.5　Web サービスと Web アプリケーションの連携

制御用 Web サービスと制御用 Web アプリケーションとの間の動作の流れは次のとおりである：

(1) 制御用 Web サービスは，ulogd によって書き込まれる SQLite データベースを監視する．
(2) 制御用 Web アプリケーションは，HTML5 の Server-Sent Events を用いて，新しい TCP/IP 接続の監視を制御用 Web サービスに依頼する．
(3) SQLite データベースにエントリが追加される（新しい TCP/IP 接続が開始される）と，制御用 Web サービスがそれを検出し，Server-Sent Events を用いて制御用 Web アプリケーションに更新を通知する．
(4) 更新通知を受け取った制御用 Web アプリケーションは，ブラウザ上の情報を最新の状態に更新する．

[*4] ulogd は，SQLite データベースにログを出力することができる．

表 4　制御用 Web サービスの実装に用いたソフトウェア
Table 4　Used software for proposed controller Web service.

ソフトウェア	バージョン	備考
Slim	2.4.2	PHP マイクロフレームワーク
PHP	5.4.4	スクリプト言語
Apache HTTPD	2.2.22	Web サーバ
SQLite	3.7.13	単一ファイルのデータベース

表 5　制御用 Web アプリケーションの実装に用いたソフトウェア
Table 5　Used software for proposed controller Web application.

ソフトウェア	バージョン	備考
jQuery UI	1.11.0	jQuery 用の UI ライブラリ
jQuery	2.1.1	JavaScript ライブラリ
HTML 5	5	Server-Sent Events のために使用

4.　提案手法の予備的評価

我々は，提案手法に対する詳細評価を予定しているが，まずは予備的な評価としてプロトタイプのスループットを様々な条件下で計測した．具体的には，

(1) 3 種類の OS（Raspbian ver. June 2014，Pidora ver. 20，Arch Linux ver. June 2014），
(2) 2 種類の USB ハブ（USB 2.0 と USB 3.0），
(3) 2 種類の Ethernet アダプタ（USB 2.0 対応の 100 Mbps と USB 3.0 対応の 1 Gbps）

を用いて，ソフトウェアに iperf [8] を用いて計測を行った．

計測手順は次のとおりである：

(1) 3 台のノード（テストクライアント・プロトタイプ・テストサーバ）を，Gigabit Ethernet で同一セグメントに接続する．
(2) iperf を用いて各区間のスループットを計測する．
(3) 計測を 10 回繰り返して，その平均値を評価値とする．

まず，我々は，Raspberry Pi に接続した各 Ethernet アダプタのスループットを計測した．**表 6** に計測結果を示す．表の見方は，たとえば，「100 M（USB 2.0）」となっているものは，100 Mbps の Ethernet アダプタを USB 2.0 対応の USB ハブ経由で接続した場合のスループットを示している．表より，多少の差はあるものの，約 50 Mbps 前後のスループットが出ていることが分かる．ここで，接続方式の違いによる差が出ないのは，Raspberry Pi の性能にスループットが律速されているためと我々は推測する [*5]．

次に，プロトタイプとテストサーバとの間のスループットを計測した．**表 7** に計測結果を示す．表より，多少の差はあるものの，約 40 Mbps 前後のスループットが出ていることが分かる．スループットが表 6 の平均値（約 50 Mbps）から低下した理由は，ブリッジ動作をさせていることによ

[*5] 対照実験として，これらの Ethernet アダプタを高速なコンピュータに接続した場合には，100 Mbps 対応のアダプタで 94 Mbps，1 Gbps 対応のアダプタで 296 Mbps のスループットが得られた．

るオーバヘッドによるものと我々は推察する．

最後に，プロトタイプを経由した場合の，テストクライアントとテストサーバとの間のスループットを計測した．**表 8** に計測結果を示す．この結果から，表 7 に近い約 40 Mbps のスループットが出ていることが分かる．

これらの評価結果より，プロトタイプのスループットは，Gigabit Ethernet が一般的となった昨今のネットワーク内で常設して利用するには不十分と言える．一方で，提案手法の用途を考慮すれば，このスループットは許容できると我々は考える．なぜならば，提案手法は，ソフトウェアのみによる利用者のセキュリティをさらに向上させるための付加的な手段であり，Web ブラウザから容易に操作できることの方がスループットよりも重要と考えられるためである．今後は，このプロトタイプを用いて実際のネットワーク上での評価を行い，提案手法の有用性を明らかにしていく予定である．

5. 関連研究

Liu ら [3] は，内部犯によるセンシティブな情報の流出を検出するためのフレームワークを提案している．彼らと我々の手法は，L2 ブリッジとして実装する点に類似性がある．一方で，両者は使用方法において異なる：彼らのシステムは内部犯によるセンシティブな情報の流出の検出に焦点を当てており，提案システムを対象ネットワークの境界に設置する必要があるが，我々のシステムは悪意のあるソフトウェアによる情報の流出の検出の強化に焦点を当てており，提案システムを対象ネットワークの境界以外の場所にも設置することができる．

Maeta ら [5] は，Linux 上でセンシティブな情報の流れを追跡するシステムを提案している．このシステムの追跡対象は，ファイル操作と IPC（inter-process communication）である．彼らと我々の手法は次の点で異なる：彼らの手法は Linux のシステムコールをフックして実現しているために OS に Linux を必要とするが，我々の提案手法は特定の OS には依存しない（依存しにくい）[*6]．

大野ら [11] は，電子工作愛好者向けのセキュリティゲートウェイとして，Raspberry Gate と Raspberry Guardian を提案している．彼らと我々の手法は，ハードウェアに Raspberry Pi を用いて利用者のセキュリティ保全を図る点に類似性がある．一方で，2 つの手法は想定利用者や利用シナリオが異なる．大野らは，提案手法を設置するだけで利用者がセキュリティを担保できるようにしているが，我々は，怪しい TCP/IP 接続を利用者が容易に検出・遮断できるようにすることが目的である．

[*6] 現時点では Raspberry Pi 用の OS は Linux が大半のために，プロトタイプにも Linux（iptables）を採用しているが，仮に OS を FreeBSD に変更したとしても，iptables の代わりに pf などを用いて同じ機能を提供できる．

6. おわりに

本論文では，ネットワークへの着脱が容易な，Web サービスによって制御されるファイアウォールの構築法を提案した．我々は，ネットワーク上の様々な場所に設置でき，試行錯誤的に TCP/IP 接続の検出・遮断が行える本システムは，悪意のあるソフトウェアによって引き起こされる情報流出への対策の一助として有用であると考える．一方で，現時点では提案手法の有用性が十分に示せておらず，より直接的な有用性の評価が急務である．今後は，より具体的なユースケースやテストシナリオを練り込むとともに，Web サービスの汎用性や Web アプリケーションの利便性を向上させ，最終的な評価につなげることを予定している．

参考文献

[1] Berjon, R., Faulkner, S., Leithead, T., Navara, E. D., O'Connor, E., Pfeiffer, S. and Hickson, I.: HTML5, W3C (online), available from ⟨http://www.w3.org/TR/html5/⟩ (accessed 2014-08-11).

[2] Hickson, I.: Server-Sent Events, W3C (online), available from ⟨http://www.w3.org/TR/eventsource/⟩ (accessed 2014-08-11).

[3] Liu, Y., Corbett, C., Chiang, K., Archibald, R., Mukherjee, B. and Ghosal, D.: SIDD: A Framework for Detecting Sensitive Data Exfiltration by an Insider Attack, *Proc. of 42nd Hawaii International Conference on System Sciences (HICSS)*, Hawaii, IEEE, pp. 1–10 (2009).

[4] Lockhart, J.: Slim Framework, New Media Campaigns (online), available from ⟨http://www.slimframework.com/⟩ (accessed 2014-08-11).

[5] Maeta, A., Takahashi, K., Kawamura, T. and Sugahara, K.: Implementation of Logging for Information Tracking on Network, *Proc. of International Conference on IT Convergence and Security (ICITCS)*, Macao, IEEE, pp. 1–4 (2013).

[6] Raspberry Pi Foundation: Raspberry Pi, Raspberry Pi Foundation (online), available from ⟨http://www.raspberrypi.org/⟩ (accessed 2014-08-08).

[7] Sophos Ltd.: USB デバイスの脆弱性 -「BadUSB」，Sophos Ltd.（オンライン），入手先 ⟨http://www.sophos.com/ja-jp/press-office/press-releases/2014/08/ns-badusb.aspx⟩（参照 2014-08-10）.

[8] The Iperf team: Iperf, The Iperf team (online), available from ⟨https://iperf.fr/⟩ (accessed 2014-08-22).

[9] The jQuery Foundation: jQuery, The jQuery Foundation (online), available from ⟨http://jquery.com/⟩ (accessed 2014-08-11).

[10] The jQuery Foundation: jQuery UI, The jQuery Foundation (online), available from ⟨http://jqueryui.com/⟩ (accessed 2014-08-11).

[11] 大野浩之，北口善明，鈴木裕信：電子工作愛好者向けセキュリティゲートウェイの構築第三報：構成要素の検討と性能評価，研究報告マルチメディア通信と分散処理（DPS），Vol. 2014-DPS-160, No. 4, 情報処理学会，pp. 1–6 (2014).

表 6　各 Ethernet アダプタのスループット（Mbps）
Table 6　Throughput for each Ethernet adapter (Mbps).

OS	100 M（内蔵）	100 M（USB 2.0）	1 G（USB 2.0）	100 M（USB 3.0）	1 G（USB 3.0）
Raspbian	53.29	53.09	53.09	52.93	50.87
Arch	56.18	52.17	51.87	52.37	51.97
Pidora	43.73	42.81	43.36	43.24	43.39

表 7　プロトタイプとテストサーバ間のスループット（Mbps）
Table 7　Throughput between prototype and test server (Mbps).

OS	100 M（USB 2.0）	1 G（USB 2.0）	100 M（USB 3.0）	1 G（USB 3.0）
Raspbian	46.6	46.64	45.2	45.77
Arch	38.6	33.73	36.61	36.92
Pidora	37.89	39.37	38.03	38.88

表 8　プロトタイプ経由時のテストクライアント・サーバ間のスループット（Mbps）
Table 8　Throughput between test client and server through prototype (Mbps).

OS	100 M（USB 2.0）	1 G（USB 2.0）	100 M（USB 3.0）	1 G（USB 3.0）
Raspbian	47.24	48.59	47.67	48.24
Arch	41.92	44.41	41.93	44.23
Pidora	36.98	36.64	36.31	36.59