産業用機器への DDoS 攻撃防御方式 *

高橋 俊成 †

本論文は専ら公衆網を用いて通信する産業用機器に対する遠隔地からの分散型サービス停止攻撃(DDoS攻撃)を，〝end to end〟の高速認証により防御する 1 手法について述べる．

権限の無い機器からの通信(攻撃)を一般的な認証技術で遮断する場合，受信機器に極端に多くの通信セッションを要求し認証を強要する攻撃(Brute Force Authentication Attack - 本論文ではBFAAと略す)を受けた際には，そのセキュリティ・プロトコルの処理負荷に耐えられない．

一方IPS[*1]装置が用いるフローフィルタリングはヒューリスティックなルールに起因する検知漏れや誤検知による通信停止があり，完全性／可用性を要求するシステムでは解にならない．

本論文では，高負荷な既存の通信プロトコルをシンプルな事前認証方式で包み込む，新しいデバイス認証スキームDoSRAS[TM](DDoS-Resistant Authentication Scheme)を提案する．これは既存の標準的通信プロトコルを改造すること無く，高速なマルチ・レイヤ認証を TCP 層に軽量に埋め込む実装技術である．これにより，不正侵入防止能力とDDoS攻撃耐性を両立させた．本方式では，BFAAまでも含めた10万ppsオーダの激しいDDoS攻撃を原理的に防御可能である．

1. 背景と概要

遠隔地との IP 通信機能を持つ産業用機器が一般的になりつつある．従来は銀行 ATM／クレジット決済／チケット発行端末など，その機能上，通信を行うことが必須の機器が専用回線／独自プロトコルを用いて通信することが主であった．しかし今日では自動販売機の在庫管理／電力メータの集計／機器異常診断／建設機械の遠隔操作／ファームウェアの更新など，保守性を高める目的で積極的に利用される．さらには，回線費削減や汎用プロトコル使用による開発費（ソフト／ハード）低減のため，汎用IPネットワーク（インタネット）の利用も進んでいる．

しかしながら，PC(個人用計算機)など汎用OSの接続された公衆 IP 網では，さまざまな種類のサイバー・アタックが懸念される．パケットの盗聴や改竄は TLS(Transport Layer Security) などの汎用暗号ツールで回避でき，異常パケットの送信によるDoS攻撃や，ある種のDDoS攻撃はIPSなど高性能なフローフィルタリング装置で低減できるが，次に述べるBFAAスタイルのDDoS攻撃下でも正常にサービス継続できる有効な防御装置は無い．本論文の技術が対象とする典型的な BFAA 手法は次のような手順で実行されるものと想定する(図1は手順概略)．

想定する攻撃：
1. 認証機能を持つサーバを攻撃対象とする．
2. 攻撃者は，予めセキュリティの低い他人のPC等に多数の不正な通信ソフトを仕掛け遠隔操作可能とし，同時多発的攻撃のできる体制を用意する（一般にこれをBotnetと呼ぶ）．
3. 攻撃者はBotnetを用い，IPS装置で排除されない正常なパケットで攻撃対象のサーバに同時アクセスする．正当な鍵を用いない(認証を通らない)パケットでも，認証を通るパケットでも良い．
4. サーバは認証処理のスループットを超えるパケットを受信するため，受信キューが溢れドロップが発生し正当な機器からのパケットがほとんどサーバに届かなくなり通信機能が低下する．
5. 正当なパケットのドロップが一定割合以上になると，TCP の再送メカニズムではセッションが継続できなくなり，サーバは機能停止する．

図 1. 想定する DDoS 攻撃のモデル

* A DDoS-resistant authentication protocol for preauthorized electronic devices

† TAKAHASHI Toshinari,㈱東芝 研究開発センター , Computer Architecture & Security Systems Laboratory, Corporate Research and Development Center, TOSHIBA Corporation

*1) Intrusion Prevention System (DDoS攻撃対策の代表的手法)

このように本論文ではBFAAスタイルの攻撃を主たる脅威として説明するが，これに限らず一般に通信開始コストが高いことを利用した攻撃は同様の脅威となる．例えば分散SYN flood攻撃（TCPセッション開始要求を多数の拠点から送って放置することにより，IPSをパスしてサーバのTCPセッション管理キューをオーバーフローさせる）も想定し，あらゆる種類の攻撃下でサービス継続可能であることを目標とする．

この高い目標設定において，従来の防御技術は十分な成果を出さない．例えばSYN cookies[1]やSYN cache[2]，代理返答サーバを用意するhot spare方式[3]などはTCP接続時のセッション管理負荷を下げる働きをするが，いずれも処理負荷を軽減させる仕組みであり，Botnet等による高いレートの継続攻撃には無力である．また，一旦TCP接続を許してしまえばその先の(より高負荷な)TLS等の暗号／認証モジュールへの攻撃を防ぐ能力は無い．一方，ルータを多重化して負荷を下げる試みや，ルータ間で攻撃情報を交換するpush back[4]等のヒューリスティックな手法は，そもそも攻撃か否かを正確に峻別する具体的手法が存在しない．パケット数の矛盾を検出したりトラフィックを統計的に判定するなどの手法も，攻撃検知のヒントにしかならない．

一方，産業用機器の多くは，汎用PCとは異なり，事前に想定された機器以外と通信する必要が無いので，通信する機器間で事前の鍵交換を行っておけば高速な機器認証が可能である．これが十分に高速であれば，オーソライズされていない機器から殺到する攻撃パケットを時間内(on-the-fly)に排除できる．ここで言う産業用機器とは以下の特徴があるものとする．

産業用機器の特徴：
- **事前に想定された機器間でのみ通信する**
- **相互の機器認証を行う**
- **認証が不要な相手へのパブリックなサービス（公開webサーバ等）は行わない**
- **通信相手の機器とは事前に秘密鍵交換されている**
- **事前に決めたプロトコルでのみ通信する**
- **事前に決めた頻度内で通信する**
- **TCP/IP通信を行う**
- **機器には耐タンパ性があり秘密鍵漏洩の懸念は低く，コンピュータ・ウイルスも入りにくい**
- **通信はパブリックなIP網を想定する（機器のIPアドレスを知っている攻撃者は任意のパケットが送信できる）．また，通信経路の機器（ルータ等）は事業者の管理下に無く，自由に選定できない．**

本論文では，この想定された前提を有利に生かした耐DDoS攻撃通信プロトコルを提案する．本提案は，DDoS攻撃手法の中でも最も防御が困難である，BFAAを排除することを主たる目的としており，専ら産業用機器への適用に向いた方式である．

以下本論文では，BFAA排除の基本方針について２章で述べ，それをDoSRASで実装するにあたっての課題を３章で述べ，４章で安全性に関する考察，５章で適用要件，６章でプロトタイプ実装と評価，７章でまとめを述べる．

2.BFAA 排除の基本方針と実現課題

１章で述べた，Botnetを用いたBFAAなど広域分散タイプのDDoS攻撃では，送信パケットにDoS攻撃に特徴的な性質が無いためフローフィルタリング等で攻撃検知できるケースは限られ，正当な通信を守れるほどに検知率を上げることはできない．

その一方で，産業用機器に限定した場合においては，機器認証を安全かつ高速に行えば，BFAAを完全に排除できる可能性が高い．すなわち，直近のルータ（スイッチ）から流れ込むパケットの全てに対して（ドロップ無しに），on-the-fly認証処理を行えることが必要かつ十分な条件である，と考えるのが基本方針である．

本方針の要求を満たす高速認証を実現するにあたり，以下の障害（課題）が想定される．

- 広く用いられている認証モジュール (TLS など) は，パケットの転送処理に比べると桁違いに演算処理が多く，高速化困難である．
- 共有鍵ベースの高速認証アルゴリズム単体でも速度が十分とは言えない．
- 要求を満たす新たな高速認証アルゴリズムを開発したとしても，その高速性を実装レベルでも残し，かつ既存プロトコルを妨害しないことは難しい．
- TLSなどでは，認証モジュールに処理が渡る前にTCPセッションを確立するため，TCPレイヤに対する攻撃 (SYN flood攻撃など) には無力である．

これらの障害に対応するため，DoSRASではTCPセッション確立前に行う軽量認証を追加（既存の認証処理に加えて）する方針を採用した（図2）．これによりTCP管理へのDDoS攻撃に加え，その先にある暗号・認証モジュールや，最

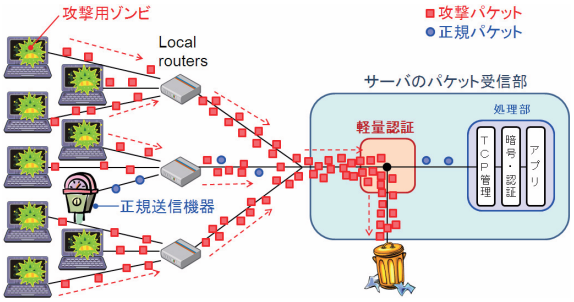


図 2. DDoS 攻撃防御のイメージ

終的なアプリケーションをDDoS攻撃から守る．

なお，事前の軽量認証を追加したことにより，結果的にDoSRASがDDoS攻撃対策だけでなく，上位認証モジュールへの未知脆弱性攻撃への簡易な対策を兼ねる効果も得られた．

3. DoSRAS のメカニズム

DoSRAS は一つの要素技術ではなく，目標の高速認証を実現するのに向いた複数の既存技術を選択し組み合わせ，新しいアイデアを付加した，新たな認証プロトコルおよびその実装法である．本章では各技術を説明する．

3.1. 認証子生成のアルゴリズム

メッセージ認証にあたっては以下の方法で生成したメッセージ認証子(MAC[*2])をTCP/SYNパケット(TCP接続開始要求パケット －以下SYNパケットと略す)に設定する．

3.1.1. 鍵付きハッシュによる認証

事前にオーソライズされた送信デバイス（クライアント）を受信デバイス（サーバ）が高速に認証するには，事前に両者で共有した秘密鍵を用いてハッシュ関数によりMACを生成する方法(例えばHMAC[5])があり，DoSRASでもこの種の鍵付きハッシュを用いることを想定している．ただし，MAC生成／認証はend-to-endの通信機器間の合意事項なので，独自開発の軽量ハッシュ・アルゴリズム等，任意の認証方式適用が許される．現実装における認証の高速化手法については3.3節で述べる．

ハッシュの対象とする入力データは該当IPパケットの全フィールド値ではなく，転送途上で値が変化するフィールドは原則として演算対象から外す（例外として，サーバ／クライアント間での共有が可能な情報は含めても良い)．

*2) Message Authentication Code - 秘密鍵に基くメッセージ認証子(ネットワーク機器のMACアドレスとは無関係)
*3) 認証に成功したパケットを取得し再利用する攻撃方法

3.1.2. 片道認証

Client　Puzzles を使って負荷の高い認証モジュールをDoS攻撃から守るアイデアは古くから知られているが，チャレンジ&レスポンスを使いパケットの往復で達成されるプロトコルは，そのセッション管理部分がDDoS攻撃のターゲットとなるため実用的な解にならない．DoSRASでは，サーバが受信パケットのMACを検証するのみの認証を行い(片道認証)，セッション管理の開始前に攻撃を排除する．

片道認証を選択したことにより，MACが再利用可能な場合，盗聴した正規パケットをBotnetに配布される危険があるため，MACが随時変化するようMAC値算出に時刻情報を用いリプレイ・アタック[*3]を防ぐ．詳細は3.4節で述べる．

3.2. MAC の格納場所

一般的な認証プロトコルではMACをパケットの何処に格納するかは通常重要な意味を持たないが，DoSRASの耐DDoS性能および上位プロトコルの透過性の意味で留意すべき点が多い．

3.2.1. MAC の格納レイヤと耐 DDoS 性能

自分の管轄外の機器で構成された公衆網でtrust chainを組むのは現実的で無いので，通信するサーバ／クライアント間のend-to-endの認証が望ましい．認証は低レイヤで完結した方が他レイヤへの攻撃を防ぎやすくなるので，IP層を用いるのが最適である．それより下層では送信元アドレス情報を持たないので実装が難しい．

通信にTCP/IPを使う前提ではIP層もTCP層もパケットとしては（通常）同一なので，格納に便利なSYNパケットへのMAC格納とした．

なお，DDoS耐性の高いTCP準拠プロトコルを新たに設計することは可能だが，透過性等の面でデメリットが多く検討対象としていない．

3.2.2. TCP レイヤ認証と MAC ビット数の制約

BFAAを防御するには，上位の（本来の）認証モジュールが起動する前に，高速な認証を行う必要があるが，それだけでなく，TCPセッション管理を狙ったDDoS攻撃(SYN flood攻撃など)も防ぎたいので，サーバ側でのTCPセッション確立前に高速な認証を行うのが望ましい．

最も単純な実装は，TCPパケットにMAC書き込み用フィールド(認証ヘッダ)を用意することである．例えばTCPヘッダのオプション部に新たな属性(kind)を定義したり，SYNパケットの

data部を利用したりすることが考えられるが，この方法は採らなかった．パケットサイズを修正する転送は処理速度の点で望ましくない上，新しい実装は既存の機器群おいて解析困難な通信不良を起こすリスクが高いからである．例えば家庭用ルータの一部には，SYNパケットのdata部を転送しないものや，TCPオプション部を無視するものがあることが判明している．

このため，TCPヘッダの限られた冗長フィールドを利用してMACを書き込むこととした（方法は3.2.3節で述べる）．しかし，それでは安全なMACを格納するための十分な領域（ビット数）を確保することができない．

この問題を防ぐ，TCPレイヤ認証の過去の研究にTAP(TCP layer Application Protector)[6,7]がある．TAPでは，TCPセッション確立時に正規のSYNパケットと並行して認証のみを目的としたダミーのSYNパケットを複数送信することで，見かけ上通常の複数セッションを張りつつ十分な長さのMACの送信を可能にした(図3a)．

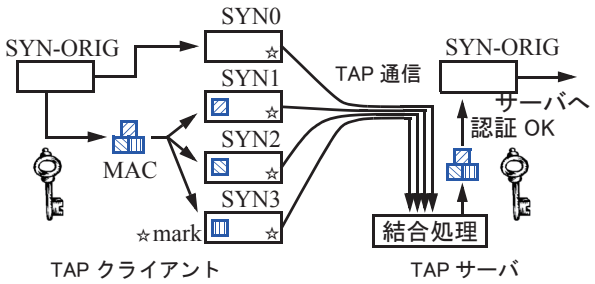


図 3a. TAP（従来例）における MAC 分割送信

これにより未知脆弱性攻撃対策とすると同時に，主流なDoS攻撃手法である[8]SYN-flood攻撃への耐性を460倍向上させた．しかしこの方法ではサーバが複数のSYNパケットを束ねて管理するため，一般的なBFAAにまで対抗する性能を出すには製造コストの面で限界があった*4．

これに対しDoSRASでは，使うMACの長さを限定したまま1個のSYNパケットのみで簡易な認証を行う（図3b）．MACが十分な長さを持てないため，DoSRAS自体は暗号理論的な安全性確保を担保せず，DDoS攻撃防御機能に特化する．認証の安全性はDoSRAS認証を通過した後の上位認証モジュール(TLS等)が担保する．この実装法により，MAC検証時間以外には，認証メカニズムを追加することによる受信（サーバ）側オーバヘッドの無い設計とした．

*4) TAPは本来，高性能な主要サーバのTLS等への脆弱性攻撃を防止するために追加する認証機能であり，DDoS 攻撃防御を主目的として設計されたものではない．

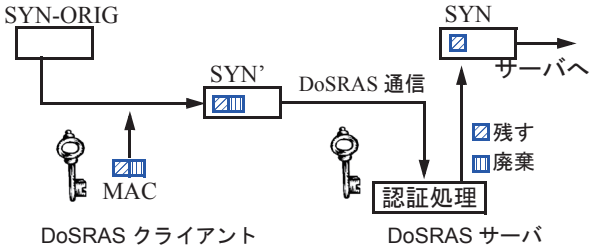


図 3b. DoSRAS における単純 MAC 送信

またDoSRASでは，SYNパケットに付加したMACの一部をあえて削除せずに本来のTCPサーバに渡すため，以降サーバでTCPセッション管理のための余分な変換処理を行わなくともインタネット上に正しいTCPパケットのみを流すことができる．これもDDoS攻撃耐性を高める上での１つの利点である．このTCPセッション管理詳細については６章で述べる．

3.2.3. MAC 格納可能な TCP ヘッダフィールド

DoSRASでは，MACを埋め込む場所としてTCPヘッダ内のSequence numberフィールドと，Acknowledgement numberフィールドの各32ビット部を利用する(図4)．これらフィールドはTCPセッション管理に用いる連続番号であるため本来自由に書き換えることはできないが，Sequence numberフィールド値はクライアントが決める自由な値であるので，他のTCPセッションとの混同が生じないよう調整さえすれば，ほぼ乱数フィールドとして利用できる．また，SYNパケットのAcknowledgement numberフィールドは使用されない（通常0を格納する）ので，乱数フィールドとして利用できる．つまり前者には約31ビット，後者には32ビットのスペースがある．１つのSYNパケットで見れば，後者の転送が保証されない場合で約31ビット，保証される場合で約63ビットのMACを埋め込むことができる．現実装ではこの約63ビットをフルに利用しているが，インタネット上の複数のプロバイダ間での接続実験を行った範囲では問題となるケースは存在しなかった．

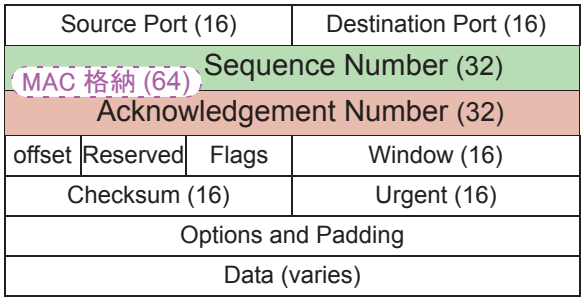


図 4. TCP ヘッダの構造と MAC 格納

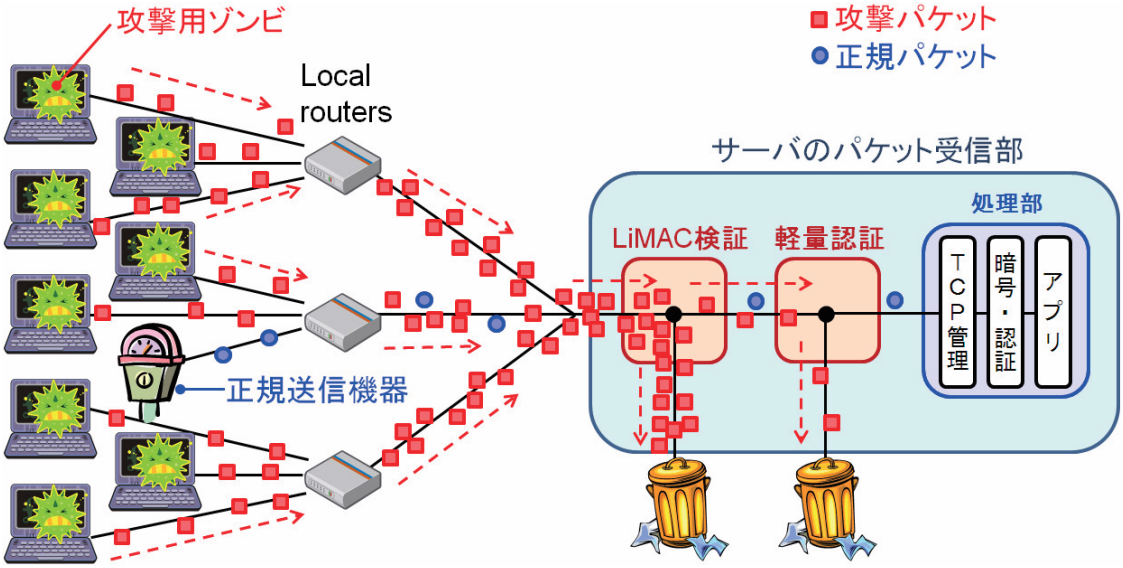


図 5. マルチ・レイヤ認証による高速化

TCP サーバは TCP クライアントがいかなるルールでSequence numberを発行したかを知る必要は無いので，SYNパケットにMACの情報が含まれていることを知らずに，TCP セッション管理を行うことができる．単にMACの検証ルーチンを加え，不正なパケットを破棄する処理を行えば足りる．

以上の方法により DoSRAS では，TCP規格に準拠したまま，BFAA防御に必要となる64ビット(情報量的には約63ビット)のMACをSYNパケットに埋め込む．MACのビット数と安全性については4.1節で論じる．

3.3. マルチ・レイヤ認証

２章(図2)で概説した通り，Botnetを利用したBFAA を防ぐにはサーバのパケット最大受信頻度に比べ無視できる程度に高速なパケット認証(MAC 検証)を行えることが必要十分な条件であるが，知られている HMAC などの鍵付きハッシュ方式でこの速度を達成するのは難しい．

MAC の検証をより高速化する手法として，DoSRAS ではマルチ・レイヤでの認証を採用した．すなわち，64ビットのMACを一度に検証するのではなく，そのごく一部分（典型的には8ビット）を一段目で検証し，一段目を通ったものについてのみ残りのビットを二段目で検証するという２ステップで認証を完了する（図5)．

この１段目用検証コードのことを，本論文では LiMAC(Light Message Authentication Code) と呼ぶ．LiMACはパケットを完全に認証するMACではないが，攻撃パケットを間引くヒントとして使われる．また，１段目（間引き）の超軽量な検証処理を〝LiMAC検証〟，２段目の処理を単に〝軽量認証〟，と呼ぶ．

大雑把に言えば一段目の LiMAC 検証で攻撃パケット数を200分の1程度(8ビット使用の場合)に減らし，二段目の軽量認証で0(またはほぼ0)にする．手順が二重化した分，正規パケットの認証処理時間は長くなるが，LiMAC 検証処理時間が二段目の軽量認証に比べ十分短い前提では，BFAA を受けた際の不正パケット廃棄に要する認証処理は，１回の軽量認証のみで廃棄する場合に比べ200倍高速になる．

LiMAC ビット数を現実装の８ビットよりも増やせば１段目での不正パケット廃棄率は上がるが，LiMAC検証処理時間は長くなるため，LiMAC に何ビット割り当てるのが最適であるかは LiMAC検証アルゴリズム(処理速度)に依存する．

MACの使い分けを例示したのが図6である．この例では64ビットのMAC用フィールドのうち8ビットを LiMAC に割り当てるとともに，予備の8ビットを持ち，残り全て(48ビット)を二段目の軽量認証用 MAC に用いる．予備フィールドは，同一IP addressで複数のクライアントが共存する

図 6. MAC 用フィールドの使用例

場合の識別などシステム毎に異なる要求に柔軟に対応することを意図しているが，一般的には無くても良い（安全性を重視する場合は軽量認証用MACに割り当てるべきである）．

なお，第１段と第２段のアルゴリズムは独立でも良いし，第２段処理途上で第１段用 LiMAC 検証ができるアルゴリズムでも良い．現実装は前者である(アルゴリズムは６章で述べる)．

3.4. 時刻管理

3.1節で述べたとおり，DoSRASではチャレンジ・レスポンスの無い片方向認証を行っているため，リプレイ･アタックを防止するには，MAC/LiMAC の生成にあたり時刻情報を入力に用いることがほぼ必須な要件となる．通信カウンタを用いるなどの他の方法では，DDoS攻撃下での性能が保証できないからである．

時刻情報や秘密鍵をMAC生成関数の入力として用いることは，認証の安全性を確保する一般的な手法である．ただし，サーバ／クライアント間で時刻が完全に同期していれば良いが，一般には正確な同期は通信コストが掛かるため実装上の工夫で回避する必要がある．例えば，RSA SecurID[10]のトークンでは，内蔵時計と工場出荷時に生成されたシード(乱数列)を基に通常1分毎に〝Tokencode〟を生成するとされている[11]．安価な時計でも，接続時に随時時刻修正を行う措置で1分程度の精度は維持できる．1分以内であればリプレイ・アタックが可能であるものの，フィッシング詐欺の少なかったインタネット創成期にはオンライン・バンキング等に向いた簡易ソリューションとして広く利用された．

なお，SecurIDではワンタイム・パスワードを実現する一つの手段として時刻を用いるが，パケット往復のある認証プロトコルにおいて時刻を用いることは必須ではない．これに対しDoSRASでは，(チャレンジ・レスポンスなどのできない)片道プロトコルで，ステート・レスな機器認証を実現する手段として，時刻情報をMAC生成に利用することは本質的である．

DoSRAS ではクライアント毎にパラメータを可変にすることで，機器毎の時計精度と要求される安全性に応じた時刻同期機能を持つ．各クライアントは，パラメータで指定された精度で丸めた大まかな時刻でMACを算出し，別のパラメータで送信の詳細時刻を調整する．サーバは，通信時間を考慮し，サーバへのパケット到着時刻を元にクライアントからのパケット送信時刻を推定し，クライアントがMAC算出に用いた時刻を推定する．

これらの演算は全て単純な整数演算なので認証処理時間への影響は無視できる．この点がDDoS攻撃防御のために重要な特性である．なお時刻取り出しはOSのシステム・コールを介さずリアルタイムに行えることがより望ましい．

多くのシステムではこの方法で時計誤差を1秒未満に維持できるため，攻撃者がパケット盗聴に成功した場合においても，その情報を基に全世界の Botnet に攻撃準備を指令することは困難であり，防御の目的は達する．

また，厳密な精度パラメータの下では想定外の通信遅延等による時刻の不一致により正当な送信パケットの廃棄が確率的に起こりうるが，通常のTCP再送機構によって通信に成功する．

なお，MAC 生成に用いた時刻情報をパケットに埋め込むことで時計の狂いの補正を容易にする方法もあるが，DoSRAS の TCP パケットに余ったフィールドは無いため，時刻情報をパケットに埋め込まずに暗に使用する．

3.5. 既存プロトコルとの整合と実装上の特徴

既に述べた通り，DoSRASではBFAA排除能力を持つ新しい認証プロトコルを設計するというアプローチは採らず，既存の高負荷なセキュア通信プロトコルを，考案したシンプルな事前認証方式で包み込んだ．その理由は，

- TCP 層以上にある，既に普及している複雑化したプロトコル群を改造せずに済ませる
- TCPプロトコルのセッション管理機能の良い性質をそのまま生かす
- シンプルな事前認証部分を独立させることにより，プログラミングの誤りによるセキュリティ・ホール発生を最小限にする
- 通信部へのハードウェア実装を容易にする

等である．

また，MAC を書き込むための新たなフィールドを作らず，TCP ヘッダの既存フィールド内にMAC情報を埋め込むことにより

- ネットワークの通信オーバヘッドを全く上げない
- 保護対象サーバの処理オーバヘッドは，純粋な MAC 検証演算，鍵処理を除き一切無い．
- ルータ等ネットワーク機器のTCP処理部を混

乱させない (フル実装されていない家庭向け機器等でも通信不能とならない)

といった特徴が出るよう工夫した．この結果，TCP 上位レイヤの実装（OpenSSL など）は DoSRAS の存在を全く意識する必要が無いので，デメリット無くBFAA耐性の高いシステムが構築できる．

4. 安全性に関する考察

4.1. 短い MAC の安全性

3.2 節で述べた通り，DoSRAS の実装法では，十分に大きなサイズのMACをパケットに埋め込むことができない（最大でも 64 ビット）．そのため，暗号理論上十分に安全とされる情報量を 1 パケットで送信できず，MAC 値を恣意的に一致させる攻撃法が存在する懸念が残る[*5]．

しかし，この制約はBFAA防御の目的に限れば問題にならない．なぜなら，万一僅かな数の攻撃パケットが不正にMAC検証をパスすることに成功しても，それは上位の暗号プロトコルレイヤまたは個別のアプリケーションにより再度検証され排除されるため，認証の安全性は低下しないからである．

また，認証を不正にパスできる確率は十分に小さいため，MAC衝突性を利用したDDoS攻撃でサーバ機能が低下させられることも無い．

ただし，以上は耐衝突性が保証されたMAC生成関数を用いた場合の話である．DoSRAS のパケット認証は，安全性を軽視してでも演算速度を上げる「適度に手抜きな」MAC 生成が期待されるが，生成関数が貧弱な場合 (例えば任意の MAC を出力させる時刻を推定する攻撃法が発見された場合)，DDoS 攻撃防御機能自体が無効化される．そのため，MAC 生成関数は完全なハッシュ関数の性質を持つ必要は無いものの，全く安全性を検証しなくて良いわけではない．特に，LiMAC 検証部分については新規の高速演算法を前提とするので安全性評価が別途必要となる．

また，別の問題として，ハッシュ関数出力ビット数を極端に短くすることで高速化を狙っているが，一般に単純に出力ビット数を減らしても高速化に直結しない．なぜなら，SHA-1[9] などの汎用ハッシュ関数アルゴリズムでは，演算のメイン・ループである撹拌処理は出力ビット数とは独立なので出力段処理のビット数は重要でない．撹拌処理バッファのビット数や段数を減らせば高速化可能だが，その場合に(出力が短いビット数とはいえ) 出力値の衝突による安全性低下が起こるか否かについて当該分野における公知の研究は無い．

高速性と安全性のバランスについての暗号理論的評価は未検証であり，本論文は，現在設計中の高速で安全性検証の済んだ LiMAC 検証が実現した場合においては，BFAA など DDoS 攻撃対策が可能となることを示すものである．

4.2. ペイロードを認証しないことによる問題

DoSRASはSYNパケットの認証を行い，後続のパケットは直接には認証しない．そのため，パケット改竄のできる場所にいる攻撃者は，既に確立している TCP セッションを乗っ取ることができる．DoSRASはサーバの存在するローカル・ネットに攻撃者が侵入する問題を防ぐものではなく，あくまで Botnet 等を利用した全世界的大規模DDoS攻撃を防ぐためのものである．TCPのセッション・ハイジャックは，DDoS攻撃の手法としては効果的でなく，問題にならない．盗聴／改竄の対策は，既存のさまざまな暗号技術で防ぐべき問題であり，DoSRASは関知しない．

4.3. IP パケット全フィールドを認証しないことによる問題

DoSRAS は end-to-end でパケット認証するため，通信途上で変化するフィールドはMAC生成関数の入力に用いない(例えばIPヘッダのTTL値は用いない)ので，パケット改竄のできる場所にいる攻撃者は当該フィールドの値を改竄することができるが，それを基に Botnet に攻撃準備を指令してもリプレイ・アタックとして検知できるので大きな脅威にはならない．また，完全に壊れたパケットを用いたDoS攻撃(注:DDoSではない)を防ぐのは OS カーネルの実装の問題であり，DoSRASの機能とは無関係である．

NAT(Network Address Translation) ルータを経由した通信の場合，Source IP Addressが修正されるため，単純にMAC生成の入力からそれを外すと安全性が下がる懸念があり，NAT と DoSRAS の整合はあまり良くない．これを回避するには，例えばクライアントは自分の NAT ルータの外側 IP addressを知り，それを基にMACを生成するなどの措置が必要となり，何も措置しなかった場合の安全性はシステム依存である．

*5) 現代暗号では80bit安全性(攻撃に必要な計算量が2^80のアルゴリズム)でも不足とされる時代に突入している．

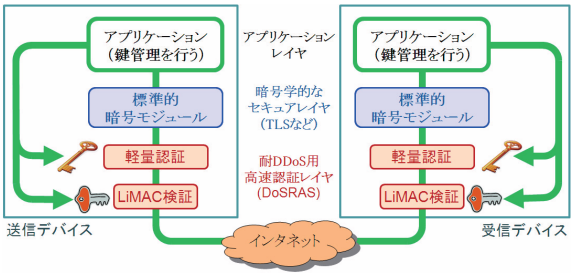


図 7. 鍵更新

4.4. 鍵更新

DoSRAS では，事前に秘密鍵を共有しているものとし，鍵の事前共有や更新の方法については関知しない．要注意点として，DoSRAS 自体の認証機能は DDoS 攻撃を防ぐ能力しか持たないので，DoSRAS の安全性に基づき DoSRAS 自体の鍵交換を行ってはならない．

典型的な鍵更新方法としては，DoSRAS で接続しさらに上位レイヤの認証（TLS 等のセキュア・レイヤや，アプリケーション個別の認証）が完了した後に，アプリケーションが交換する方法が考えられる(図7)．

他には，通信する機器とは別の鍵配布サーバを用意する方法，TLS に DoSRAS 鍵更新機能を組み込む方法，などが考えられる．

5. DoSRAS プロトコル適用に向いたシステム

DoSRAS のプロトコルは，TCP で動く汎用的なものであるが，実際に DoSRAS を適用すべきシステムは限定される．その理由は以下のとおりである．

- DoSRAS は事前に共有された秘密鍵を持つ送信機器からの通信を正当とし，攻撃は全て不当な機器から来ると仮定している．このため，予め正当／不当が区別できないシステム，例えばユーザ認証を必要としない web サーバには適用できない．また，正当な機器が（ウイルス等により）攻撃の踏み台となる可能性を考えると，クライアント機器がセキュリティ機能を持たない汎用 OS で作られた場合，十分な DDoS 攻撃耐性を持てない．
- DoSRAS では MAC の埋め込みに Sequence number フィールドの冗長性を利用しているため，サーバへの通信頻度が DoSRAS の管理下になければならない．管理外のプログラムが DoSRAS サーバへ多数の通信を行うことを許すと，TCP セッション管理を維持できる当該フィールド値の選択自由度が下がる．
- 通信するサーバとは別に専用の鍵配布サーバを用意するなどしない限り，初めての通信の際，事前に秘密鍵を共有している必要があるため，クライアント機器出荷時に固有の秘密鍵を持たせておくことが現実的である．
- 極めて少数のクライアントとのみ通信するサーバの場合，固定 IP アドレスによるフィルタリングや VPN 機能などの組み合わせで DDoS 攻撃に対抗できる可能性がある．

以上の性質を考慮すると，例えばスマート・グリッドで使われる多数の電力量メータと小数の電力量集計サーバの通信のように，専ら 1 章で列挙した産業用機器で使うのに適している．

6. プロトタイプ実装と性能評価

6.1. MAC 生成アルゴリズム

攻撃下での DoSRAS 理論性能は 3.3 節で述べた LiMAC 検証（1段目の超軽量検証）の処理時間でほぼ決定され，パケット転送速度に比べ十分に短いのが理想的である．近年では 10 万 pps から 100 万 pps オーダ（10 μ 〜1 μ 秒/packet）もの攻撃も観測されており[12]，これが全て 1 台の TCP サーバに集中するケースまで想定すれば 1 μ 秒を大きく下回る速度が要求される場合もありうる．

今回の実装では，LiMAC 検証の暫定コードとして SHA-1 をベースに開発した 8 ビット出力関数を用いた．具体的には，時刻情報と秘密鍵を含む 128 ビットの入力に対し，SHA-1 と同様の入力拡大を行い，高速化のため撹拌処理を 4 段に減らした出力から 8 ビット切り出すものであり，特に 8 ビット出力に合わせた新アルゴリズムではない．後述する安価なデスクトップ PC による評価環境で LiMAC 算出に 0.27 μ 秒掛かる．これは OpenSSL の SHA-1（同じく 1.9 μ 秒/HMAC 処理時間を除く）と比べ約 7 倍高速である．

なお，この比較値は DoSRAS の耐 DDoS 性能ではない．例えば，市販の SSL ハードウェア・アクセラレータ IPCOM EX2500LB のカタログ値は最大 1 万 tps（transaction/sec）であるが，これは DDoS 攻撃下の性能という意味で 100 μ s/packet に相当し，DoSRAS の適用によりスループットは理論上 2 桁向上する．

一方，2 段目の軽量認証は処理性能は問題にならないため，単純に OpenSSL の SHA-256 と自前の HMAC(同じく計 4.3 μ 秒)を用いて実験した．

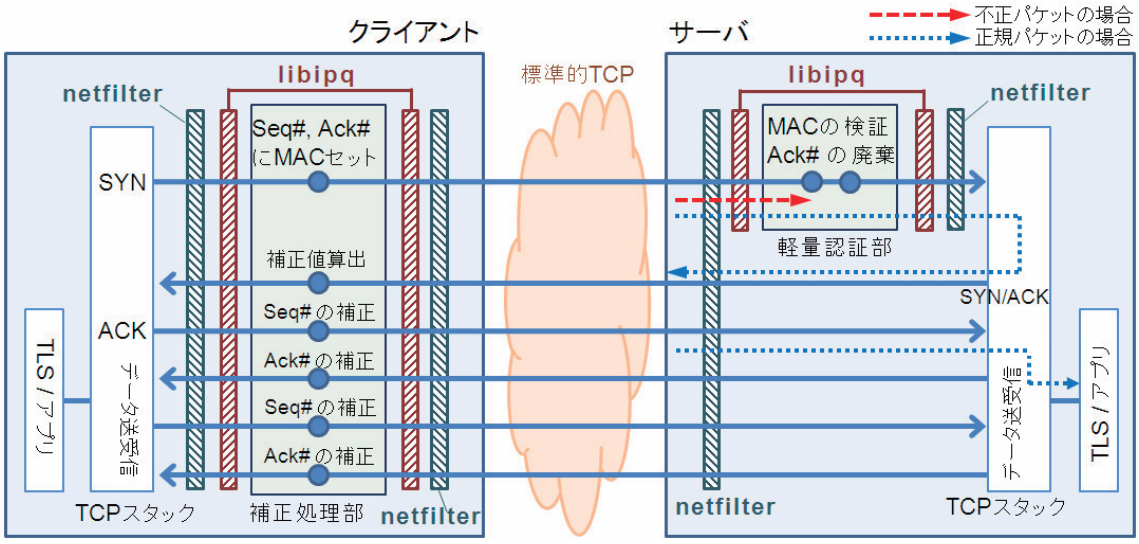


図 8. netfilter による実装例

6.2. TCP スタックへの実装法

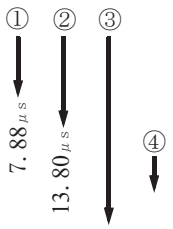
DoSRASのサーバおよびクライアントをLinux PCに実装し，接続性を確認した．実装にあたっては，OS カーネルを書き換えずに済ますためnetfilter[13] を用いた．netfilter でフックしたパケットをlibipqライブラリに渡すことで，カーネル空間の通信パケットを一旦ユーザ空間のキューにコピーした状態で，データを修正したり破棄したりすることができる．例えばサーバは以下のような指定で受信 SYN パケットのみをフックしlibipqのキューに投げる．

```
% iptables -I INPUT 1 -p tcp --syn --dport
  443 -j QUEUE -s (client) -d (server)
```

netfilter による実装ではTCP スタック外でパケットを書き換えるため，クライアントOSの管理するフィールド値と，通信路上の（サーバOSの管理する）フィールド値が別のものとなる．DoSRAS クライアントはパケット送受信毎にフィールド値の補正を行う（図8）. DoSRASサーバ側はMACの検証を行うのみで補正処理は不要な点，および不正パケットはTCP スタック処理に渡らない点が耐DDoS攻撃性能上重要である．

表 1. DoSRAS 認証＆レスポンス時間

	処理内容	時間
1.	netfilter+libipq 格納	7.61μs
2.	LiMAC検証（1段目）	0.27μs
3.	簡易認証（2段目）	5.92μs
4.	libipq 排出+netfilter	6.70μs
5.	TCPスタック	8.15μs
6.	netfilter送出処理	3.94μs



不正パケット処理 - ①（まれに②）
正規パケット処理 - ③
基本的 TCP の場合 - ④

6.3. 認証処理速度測定と攻撃耐性

以下本論文の性能評価には全てデスクトップPC（EQUIUM3520 Core2duo 3GHz Turbolinux Server11 1Gbps）に実装したDoSRASサーバおよびクライアント各 1 台と別の攻撃専用機 2 台を用いた．

まず攻撃等の負荷が無い状態で，サーバが 1 回のDoSRAS接続を処理する時間(SYNパケットを受信し SYN/ACK パケットを送信する迄の時間)の各処理ステップ毎の平均値を表1に示す．

耐DDoS性能の観点では，LiMAC検証で廃棄できる場合(表のステップ1および2の合計)と2段目（ステップ 3）まで行く場合の 255:1 の重み付き平均値7.90μs(12.6万pps相当)がシングルプロセッサでの不正パケット廃棄処理時間となる．

本実験ではステップ 1 のnetfilter処理に主な時間を取られているが，処理をOSカーネルに組み込む等により省略可能であるため，原理的にはステップ2とステップ2+3の重み付き平均である0.29μs(345万pps相当)まで高速化できる．

以上の測定値から算出した性能をまとめたのが表 2 である．netfilter を用いた現実装でも，通常知られている攻撃の 10 万 pps 程度はクリアする(現実には，より高速なプロセッサ，または通信ハードウェア実装を行うので，本数値より余裕がある)．しかし，6.1節で触れた最悪シナリオ

表 2. 処理時間から算出した攻撃耐性

LiMAC 検証	netfilter	処理時間	攻撃耐性
あり	あり（現状）	7.90μs	12.6 万 pps
なし		13.53μs	7.4 万 pps
あり	なし（理論限界）	0.29μs	345 万 pps
なし		5.92μs	16.9 万 pps

100万ppsまでもクリアするには，LiMAC検証の時間が支配的となるまで通信パケット処理速度のチューニングが必要である．このレベルの性能にまで達すると，LiMAC検証を使ったマルチ・レイヤ認証の効果が顕著となる．

6.4. 耐 BFAA 性能の実機測定

耐BFAA性能評価として，Apacheサーバへのhttpsリクエストを正当な通信に見立て，攻撃機から多数のhttpsリクエストを連続/並列送信している下で，正当な通信が通る率をDoSRAS無しの場合，DoSRAS有りの場合，DoSRASからマルチ・レイヤ認証を省いたもの(LiMAC検証を行わず簡易認証のみ行う)の3通りを計測した(図9)．5000pps程度で攻撃側装置の性能が追い付かなくなったため，それ以上は測定できていない．

なお不正機器からのSYN要求はDoSRASに排除されTCP確立しないため，本測定はBFAA耐性評価と同時にSYN flood耐性評価の意味も持つ．

DoSRAS無しの場合，100pps程度の緩い攻撃が継続しただけで接続不能となる．これは受信処理のスループットを超えた段階でTCPのlisten queueが溢れ新たなSYNパケットが受信できなくなるためと考えられる．これに対しDoSRASを適用した場合，2000pps程度まで接続率100%を維持する．5000pps以上は攻撃側装置の性能限界に達し測定不能だった．2000ppsはDoSRASの処理限界ではなくスイッチの処理能力を超えサーバ到達以前にパケットのドロップがおきているものと推察され，理論性能付近まで測定できる高速な攻撃環境が必要である．また同様に，マルチ・レイヤ認証の効果も測定限界に近く，netfilterの処理限界も含めた十分な評価結果は得られなかった．以上限定付きながら，ネットワーク機器性能に達する激しいDDoS攻撃に対しても問題無く防御できることが実証された．さらに，前節の測定結果から得た理論値に肉薄する性能を実証することが今後の課題である．

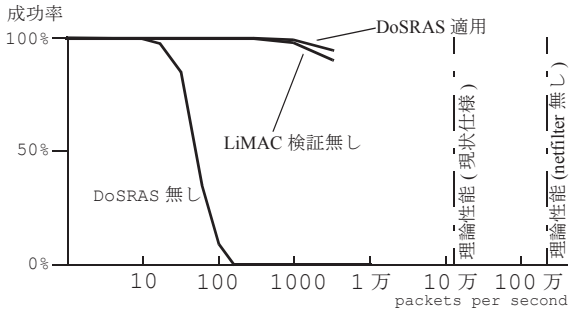


図 9. 耐 BFAA 性能(攻撃頻度と接続成功率)

7. まとめ

既存のプロトコルを修正することなく，DDoS攻撃に高い防御能力を備えることを可能とする新しい認証スキームDoSRASを提案し，その効果を示し，完全な有効性を証明する目途が立った．DoSRASは近年社会問題ともなっている，スマートグリッドなどの社会インフラ・システムへのBotnetを使ったDDoS攻撃を防御する手段として効果的である．

今後は，LiMAC検証のアルゴリズム改良(速度，安全性)と，パケットフィルタに依らない高速実装をサーバに行い，クライアント台数の多い実システムにおいてもコンセプト通りのDDoS攻撃耐性が出ることを実証してゆく．

謝辞

本研究の一部は新エネルギー・産業技術総合開発機構(NEDO)から受託したプロジェクト「米国ニューメキシコ州における日米スマートグリッド実証」の全体総括研究「サイバーセキュリティ及び情報通信技術の研究」に関するものである．

参考文献

[1] A. Zuquete, Improving the functionality of SYN cookies, Proc. of 6th IFIP TC6/TC11 joint working conf. on communications and multimedia security, pp.57-77, 2002/9
[2] J. Lemon, Resisting SYN flood DoS attacks with a SYN cache, Proc. of USENIX BSDCon 2002, pp.89-98, 2002/2
[3] T. Darmohray, et al., "Hot Spares" for DoS attacks, The magazine of USENIX & SAGE, vol.25, no.4, p.3, 2000/7
[4] S. Floyd, et al., Pushback messages for controlling aggregates in the network, internet-draft, 2001/7
[5] H. Krawczyk, et al., Keyed-Hashing for Message Authentication (RFC2104), 1997/2
[6] 高橋俊成 他，隠伏サーバを実現するTCP認証方式，第47回プログラミングシンポジウム，2006/1
[7] 梅澤健太郎 他，サーバを攻撃から守る接続許可プロトコルの提案と評価，電子情報通信学会論文誌，D Vol. J90-D, No.3, pp. 862-873, 2007/3
[8] D. Moore, et al., Inferring internet Denial-of-Service attacks, Proceedings of the 10th USENIX Security Symposium, pp. 9-22, 2001/8.
[9] Secure Hash Standard, NIST Federal Information Proceessing Standard (FIPS) 180-1, 1993/4
[10] RSA SecurID, http://www.rsa.com/node.aspx?id=1156
[11] S Krychiw, SecurID: A Secure Two-Factor Authentication, SNAS Institute certification paper, 2001/2
[12] M. Saito, et al., Infrastructure Security, Internet Infrastructure Review (IIR), Vol.15, pp.4-26, Internet Initiative Japan Inc., 2012/5
[13] netfilter/iptables project home page, http://www.netfilter.org/