「セキュアプロトコルを設計する」

（IETF セキュリティチュートリアル）

仮訳

2003/01/17

Homepage: http://jis.mit.edu/

Jeffrey I. Schiller and Steve Bellovin

内容

問題点

- インターネットは、セキュリティを念頭において設計されていない。
- 初期においては「善良な者」のみが接続されていた。
- 今日の状況は異なる。
- 単一のマネジメントインフラが存在するわけではない。
  - ICANN はインターネットを管理していない。
- ソリューションは、「エンド to エンド」において機能する必要がある。

攻撃

- 受動的：盗聴（典型的には、パスワードを。クレジットカード番号である可能性あり。）
- 能動的：悪意を持ってネットワークにデータを注入する。
  - 接続を遅くするために「tcpnice」。
  - サービス妨害。
    最近、数多くの注目を集めている。
- ネットワークにおけるもの：　プロトコル機能を利用して、パスワードを盗聴。
- ネットワーク以外のもの　：　バッファオーバーラン；実装の欠陥を利用。

セキュリティは難しい

- 回避不能
  - 持っているときには知らない。失って初めて気付く！
  - 伝統的なプログラミング、デバッグ作業、テストのサイクルとは異なる心得が要求される。
- エスカレート（権限上昇）するプロセスが存在
  - 攻撃者による脆弱性の攻略
  - 開発者による対策の開発
  - 攻撃者は、異なる脆弱性、もしくは新しい脆弱性の攻略を試みる
  - 繰り返し

認証について

- ネットワークセキュリティとは、まず、どこからパケットが来たかについて知ること。
- 認証とインテグリティが鍵。
- アクセスコントロールや監査のような他のセキュリティサービスも重要ではあるが、「回線上」においては、さほどではない。

パケット

<table>
<tr><td colspan="2">Version</td><td colspan="2">IHL</td><td colspan="3">Type of Service</td><td colspan="3">Total Length</td></tr>
<tr><td colspan="5">Identification</td><td>Flags</td><td colspan="2">Fragment Offset</td></tr>
<tr><td colspan="3">Time to Live</td><td colspan="3">Protocol</td><td colspan="3">Header Checksum</td></tr>
<tr><td colspan="8">Source Address</td></tr>
<tr><td colspan="8">Destination Address</td></tr>
<tr><td colspan="7">Options</td><td>Padding</td></tr>
<tr><td colspan="5">Source Port</td><td colspan="3">Destination Port</td></tr>
<tr><td colspan="8">Sequence Number</td></tr>
<tr><td colspan="8">Acknowledge Number</td></tr>
<tr><td rowspan="2">Deta
Offset</td><td rowspan="2">Reserved</td><td>U
R
G</td><td>A
C
K</td><td>P
S
H</td><td>R
S
T</td><td>S
Y
N</td><td>F
I
N</td><td colspan="2" rowspan="2">Window</td></tr>
<tr><td colspan="6">Checksum</td></tr>
<tr><td colspan="7">Options</td><td>Padding</td></tr>
<tr><td colspan="8">Data</td></tr>
</table>

どのようにパケットを認証するか

- パケットの残りの部分についての暗号技術によるチェックサムを含むフィールドを提供する。
- 鍵付きハッシュ（HMAC-MD5 または HMAC-SHA1）が、この「ハッシュ」処理に使

われる。
- ハッシュ値は、両端によって共有されるパケットの中身と鍵に依存するが、決してネットワーク上を流れることはない。
- これは、パケットの中身の守秘性を提供するものではない
- 守秘性は、暗号化技術を要する。

暗号とネットワーク

- 伝統的暗号：鍵を使ってデータを変形させる。鍵には、暗号を復号し、当初の内容を得るために使われるものがある。
- あなたが独自に設計するものではない。入手可能で許容されている暗号を使う。
  - DES は、25 年間、米国における標準であった。
    DES は、今でも「良い」が、現代的な利用のためには鍵長が短すぎる。
  - AES：新しい Advanced Encryption Standard
    長い鍵を持ち、30 年間は、強さを保つことが期待される。

暗号の利用

- 最も卑近な暗号は、ブロック暗号。（例外：Web ブラウザによって使われる RC4。）
- ブロック暗号は、データの「ブロック」上で動く。
- より長いブロック、もしくはデータストリーム上において動くためには、サイファーチェイニングというテクニックを使う。
- それらについては、ここでは掘り下げない。

セキュリティの 4 つの部位

- 暗号技術
  - 注意深くない場合、性能問題を起こす可能性がある。
- 鍵配布
  - 難題であり、共有された秘密鍵を配布してきた。

- 対プレイバック保護
  - どのように、様々なパケットがリプレイされることから防ぐか。
- API
  - すべてのシステムが、スタック全体を実装する必要はない。コミュニケーションファシリティを規定する必要がある。
  - このことが、単に「IPsec を使え」というのが難しい理由である。

10

鍵の配布

- コミュニケーションの両端宛てに共有鍵を配布する必要がある。
- 無防備に鍵をネットワーク越しに送ることはできない。
- 鍵配布プロトコルの例
  - IKE（RFC2409）
    「IKE 後継」が開発中。
  - Kerberos（RFC1510）

11

- 2 種類の鍵
  - セッション鍵
    両端間において交渉され、短期間有効。
  - 長期間のもの
    典型的には、人もしくはシステムを識別する。

12

リプレイ攻撃

- 暗号技術的チェックサムによってパケットを保護するとき、リプレイ攻撃にさらされている。
  - 攻撃者は、様々なトランザクションを記録し、後で再生する。
- 最も多い問題は、ピア間において延長期間について同一の共有鍵が使われるときの問題である。
- リプレイ攻撃を防ぐことは可能であるが、痛みを伴う。それは、状態の保持を要求する

からであり、潜在的に、一方または両方の再起動が必要となる。

13

API

- 規定された API なしには、上位層のアプリケーションが下位層においてどのサービスが利用可能であるかを判断することは困難である。
- 例：ブラウザから Web サーバー宛てのコネクションは、下位層のソフトウェア宛てに IPsec による暗号化されたアソシエーションが存在するかと尋ねることができることを望み、そのとき、おそらく自身の暗号化を行わない可能性がある。
- 我々は IETF においては API を扱わない。
  - 個人的意見：我々は、このことに挑戦する必要がある。セキュリティモデルの設計は、まさに、アプリケーションの「ヴィジョン」と設計について理解することを要求する。

14

公開鍵による暗号化

- 1 つの鍵を暗号化と復号に使う代わりに 2 つの鍵が使われる。
- ひとつは暗号化のために使われる。
  - しばしば「公開鍵」と呼ばれる。
- 他方は復号のために使われる。
  - 「私有鍵」と呼ばれる
- 公開鍵の知識は、私有鍵を明かすものではない。
- 1976 年以降の比較的新しいテクノロジーである。
- 鍵配布問題をシンプルにする。

15

インターネットセキュリティの各層

- 下位層：パケットは、暗号化、かつ／または、パケットの HMAC を持ったフィールドを運ぶことを行う。
  - 下位層保護は、すべてのパケットに必要とされる。

- 下位層保護は、高速であることが必要とされる。

16

鍵管理

- 管理：鍵配布、アルゴリズム交渉等
- セッション初期化において行われる可能性がある。
  - 複数のセッションについてセッション鍵交換用の高速メカニズムによって行われる可能性がある。
  - 例：IKE「クイックモード」および TLS のセッション拒否機能。

17

セキュリティは、うまく行うことができる

- 下位層パケットフォーマットの正しいエンジニアリングは、適切な鍵管理と組み合わされて、プロトコル性能についての影響を最小化することにつながる。
- ここにおけるキーワードは、「エンジニアリング」
  - 陳腐な場合、性能に影響を与える可能性がある。

18

トランスポート対オブジェクトセキュリティ

- データのフローを保護するのか、それとも、データ自体を保護するのか？
- セキュアな Web セッション：回線上のデータのフローを保護する。
- セキュアな E メール：メール自体を保護する。なぜならば、何度も転送され、しばしばスプールディレクトリに留まるからである。

18

IETF セキュリティの案内

- IPSEC：IP 層におけるセキュリティ。
- TLS：ネットワークストリーム（TCP）層以上におけるセキュリティ。

- PKIX：X.509 と公開鍵暗号システムを利用した鍵管理。
- S/MIME と PGP/MIME：セキュアメッセージング。
  - S/MIME：PKIX 鍵を使うセキュアメール。
  - PGP/MIME：「PGP」プログラムフォーマットを使うセキュアメール。
  - 主たるアプリケーションは、E メールであるが、他の用途においても使うことができる。

IPSEC

- 各 IP パケットに保護（暗号化およびインテグリティ）を提供する。
- 標準は、（下位層の）パケット処理を含む。
  - AH：
  - ESP：
- 上位層の鍵管理
  - ISACMP/IKE
- 主にファイアウォール間において VPN 端点を形成するために使われる。
- ホストにおいても使うことができるが、まだ、十分なホスト実装がない。
- トランスミッション（転送）セキュリティを提供する。

TLS

- SSL（Secure Sockets Layer）の発展。
- データストリームの暗号化、認証およびインテグリティを提供する。
- トランスミッション（転送）セキュリティを提供する。
- Web ブラウザとサーバーによって使われる。
  - おそらくインターネット上において最も使われているセキュリティ技術。
- プロトコルは、多くの暗号アルゴリズムとインテグリティアルゴリズムを提供する。
  - よって、与えられたタスクに最も適切なものを選択することができる。

PKIX

- X.509 に基づいた鍵管理。
- 主たる要素は、エンティティ（ユーザ／プロセス／ホスト）を特定の公開鍵と結びつける証明書。
- 標準は、証明書フォーマットを規定し、認証のためと証明書をエンティティに発行するために必要なプロトコルを規定する。
- 証明書は、TLS と IPSEC の両方によって使われる可能性がある。
- 証明書は、セキュアなオブジェクトである。

セキュアな E メール

- MIME に基づくメッセージにデジタル署名と暗号化を提供する。
- IETF においては、2 つのワーキンググループがこの問題について取り組んでいる。
  - S/MIME：PKIX 鍵を使うセキュアメール。
  - PGP/MIME：「PGP」プログラムフォーマットを使うセキュアメール。

他のグループとテクノロジー

- SSH（Secure Shell）
  - 遠隔コマンド実行のプロセスと関連する機能をもつ TLS のようなプロトコルの変形。

- 侵入検知
  - 侵害が起きた場合、どのように判定するか。
  - パターンに基づくメカニズム、シグニチャに基づくメカニズム
  - いまだに革新的分野
  - IETF は、IDS の様々な要素の相互運用を確保することに関心を持つ。

他のグループ

- DNS セキュリティ
  - DNS 自体をセキュアにする。
  - 他のプロトコルに鍵管理を提供することができる。
- 現在、これを採用することについて多くの関心がある。
  - （そうとも！）

26

完璧な世界

- 各ユーザと各ホストは、証明書を持つ。
- すべてのトランザクションに IPSEC もしくは TLS が使われる。
- アプリケーション層プロトコルは、常に適切なセキュリティテクノロジーを利用する。
- **パスワードが平文でネットワーク上を流れない。**

27

我々が住む世界

- IPSEC が利用可能であることに依拠できない。
  - SSL が採用された当初の理由のひとつは、アプリケーション（ブラウザ）において採用することができたからであった。IPSEC は、OS カーネルに採用される必要がある。
- みんなが証明書を持っているわけではない。
  - テクノロジーは、まだ、大部分の人には理解し難い。
  - ポリシーや実務について論争する法律家の関与がエンジニアたちをおびえさせている！
  - PKI は、階層的になりがちであり、階層は、不愉快な人を増やす。

28

我々が聞きたくないこと

- 我々のプロトコルはここにあるので、セキュリティを望む場合、IPSEC を使いなさい。

- このことは、どのように IPSEC が他のプロトコルと相互作用するかについての問いに答えていない。
- 例えば、IPSEC は、蓄積交換型コミュニケーション（メッセージング）についての問題を解決しない。
- どのようにプロトコルが IPSEC によって提供された認証された名前と相互作用するか。

命名の難題

- 認証を行うということは、識別することである。
- 我々は、インターネットにおいて、エンティティをどのように命名するか？
  - X.509 は、X.5xx スタイルの名前を使う。
  - DNS 名？
  - E メールアドレス？
- 我々は、共通命名システムについて何らかの合意を必要としている。
  - 我々は、これをまだ持っていない。
- プライバシーについては？
  - 我々は、ネット上のすべてのトランザクションが、個人について追跡可能であることを要求できない。

認可

- 一度、誰かを認証したら、彼らはどのようなことをする権利を持つのか？
- 2 者間クライアント／サーバープロトコルは、比較的容易。サーバーは、誰がクライアントであるかを知っている。
- 複数ホップ、複数主体のものは、難しい。
  - A が B を知っていて、
  - B が C を知っている場合、
  - A は、C を信頼するか？A は、誰が C であるかを知っているか？

  注意：同一の組織によって動かしているプロキシは、しばしば、この問題についての簡単な例外である。

31

認可テクニック

- ID に基づくもの
  - サーバーは、リソースと認可された主体のリストを持つ。
- 能力に基づくもの
  - クライアントは、リソースにアクセスすることができるようにするために付与する何らかのクレデンシャルを持つ。（例えば、証明書。）

32

複雑性

- まさに無駄なことですが、複雑なメカニズムは、設計が難しく、実装が難しく、正しく使うことが難しい。
- シンプルさは、単なるアーキテクチャに関する原則ではなく、セキュリティ上、必要不可欠なものである。
- ハッカーとバグが多いプログラムの間の暗号化されたチャネルは、ハッカーを招き入れ、IDS をブロックしてしまう。
- （我々のセキュリティプロトコルは、複雑すぎるか？）

33

そこで、あなたはセキュアプロトコルを必要とする

- 質問は？
  - TCP 上でうごかしますか？それとも UDP 上？
  - 大部分の相互作用が 2 者間？それとも、もっと複雑？
- TLS を使うことができますか？
  - TCP の上であれば、これがあなたの答えであろう。
- おそらく、あなたのもの自体の認証フィールドを追加する必要がある。
  - しかし、あなたは、IPSEC を使うことができる。少なくとも、鍵管理のために IKE を使うことができる。

34

質問は？