情報セキュリティ技術動向調査

タスクグループ報告書

(2008 年上期)

2008 年 9 月

独立行政法人 情報処理推進機構
セキュリティセンター

目次

序 2008 年上期の技術動向 - 今日のセキュリティエンジニアリングの課題

宮川 寧夫

背景

広範な情報セキュリティ分野において、継続的に、かつ、質の高い技術情報を収集する能力を保つことは、IPA 内部においては困難な課題とされてきた。このような能力を確保するため、「情報セキュリティ技術動向調査 TG（タスクグループ）」[1]という注目すべき動向を見張る専門委員から成る会議体を設置した。情報セキュリティ技術動向調査 TG においては、情報セキュリティのエンジニアリングの分野において注目すべき動向を発表しあい、発表内容に基づいて討議する。その第 1 回目の会合を 2008 年 6 月 25 日に開催し、本報告書を取りまとめた。

はじめに

本報告書は、読者として IT 技術者を想定しているが、各要素技術を個別に紹介するものとはならない。ある要素技術を単独で使う局面についての動向を紹介するものでもない。技術者によるエンジニアリングとして一連の活動が行われる中で、一定の段階（もしくは工程）において注目すべき動向もしくは話題を各委員独自の視点から紹介・解説していただいただき、本書を取りまとめた。ここに言う「エンジニアリングにおける諸段階」としては、技術の標準化段階、実装段階・導入配備段階等、多様な段階が挙げられる。

今日、各段階の課題で様々な課題を抱えている技術者がおり、彼らが抱えている課題は、特定の条件もしくは状況のもとでのみ説明できるものとなっているので、広く一般の理解を得ることができるように説明するためには工夫を要する。そこで本報告書においては、各委員より紹介・説明していただいた技術については、それらが必要とされる条件、状況、背景あるいは文脈についても丁寧に説明していただくこととした。

概況

まず、わずかに今期をはみ出してはいるが、昨年末にはアイデンティティ管理に関する標準が公表された。そして、今期の標準化動向として注目すべき事項として、PKI 技術の基本文書が改訂されたこと、および、検疫ネットワークシステムに関する動向が挙げられている。イントラネット[2]の構成・技術の配備に関しての動向も報告されており、この中に今後、検疫ネットワーク関連の標準についての実装も加わっていくことになろう。

次に、ホストベースの情報セキュリティ関連の話題として、強制的なアクセス制御機能

---

[1] http://www.ipa.go.jp/security/outline/committee/isec_tech1.html

[2] 組織体内部のネットワーク

をも備えるようになっているOSのLinuxについては、コミュニティ内部の論争が収束し、実装の方針が定まった。また、仮想化技術は、もともとセキュリティ機能として開発された機能ではないものの、配備する局面において権限が異なるサービスと各々のゲストOSとして設定・構築する手法が注目されている。

実装局面における失敗としてのセキュリティ脆弱性に関しては、今期、報告された脆弱性の中には、別の今日的な観点からは良かれと考えられて行われた改変が返って脆弱性を生む結果となったものがあり、興味深い。大規模な震災が発生した今期、災害復旧計画についての関心は高まっており、備えるエンジニアリングも行われつつある。今期には、インターネットのルーティングにおいてもインシデントが発生した。

1 多様な文脈をもつセキュリティエンジニアリングの課題

宮川 寧夫

システム開発におけるセキュリティ要件エンジニアリング

システム開発の工程に基づいて考えると、技術の標準化段階、実装段階・導入配備段階等の各段階において、多様なエンジニアリング課題がある。近年、システム開発において、要件定義の段階からセキュリティ関連の要件を折り込む努力がなされつつある。

一般に、ソフトウェア／システムの開発プロセスには、下記のような工程がある。:

* 企画／要件定義
* 設計
* 実装
* テスト

情報セキュリティを確保するための技術は、後から追加するのではなく、当初の「企画／要件定義」の段階および「設計」の段階から折り込まれる必要がある。「企画／要件定義」の工程において行われる活動は、「要件エンジニアリング」と呼ばれ、この一環としてセキュリティの要件も抽出されることが期待される。この「要件エンジニアリング」に関して、2008 年 1 月 22 日に CMU の SQUARE の教材[3]が公開された。この中では要件エンジニアリングの工程も 9 工程に細分化して示されている。

1. Agree on definitions
2. Identify security goals
3. Develop artifacts to support security
4. requirements definition
5. Assess risks
6. Select elicitation technique(s)
7. Elicit security requirements
8. Categorize requirements
9. Prioritize requirements
10. Inspect requirements

[3] http://www.cert.org/sse/square/square-description.html

このように「企画／要件定義」の工程を細分化して工程を示すこともできるが、開発する案件あるいは文脈によって、実際に採用される技術の内容や組み合わせは多様なものとなる。今日、情報システムの構成は、多様なものがありうる状態になってきており、開発される機能の種類も増えている。

このように多様な情報システムの構成のもとで、適する技術を選択したいという動機がある環境のもとでは、技術動向を報告する側には、要素技術単位の説明に始終するのでは期待に応えることができない。様々な条件、状況あるいは文脈のもとで行われているエンジニアリングの課題を説明する必要がある。

文脈に拠らない技術動向の話題

文脈の応じた様々なエンジニアリングの課題について、各委員から紹介していただく前に、要素技術についての動向についてもふれておきたい。要素技術に関しては、特定の利用法を限定しない議論が可能である場合がある。

今期、暗号アルゴリズムレベルの話題として、公開鍵暗号技術 RSA-1024 やハッシュアルゴリズム SHA-1 については、計算能力の向上に伴って、近い将来、破られたり、衝突が発生したりする可能性があることが注目されており、内閣官房情報セキュリティセンター（NISC）によって「政府機関の情報システムにおいて使用されている暗号アルゴリズム SHA-1 及び RSA1024 に係る移行指針」（案）[4]が公開された。この暗号アルゴリズムについての検討の際にも、目的の視点として守秘用、ディジタル署名用あるいは本人認証用等の視点を入れる必要性についても唱えられている状況にある。

文脈に拠る技術動向の話題

多くの情報セキュリティ機能は、それらが実装される構成や利用される局面が、一応は想定されて開発されている[5]。各委員から、独自の文脈で技術動向を紹介していただくのに先だって、今日の情報システムの状況をまとめておく。

今日の情報システムは、多様な構成をとるに至っている。例えば、従前のクライアント／サーバ型の他、Web 技術に基づいて Software as a Service（以下、SaaS）の形態のシステムも構築されている。また、各サイトの技術者が意識しているか否かを問わず、インターネットの経路制御インフラストラクチャを構成するシステムも存在している。このようなシステム構成に応じて、情報セキュリティの観点からも多様なエンジニアリングの課題に取り組まれている。

---

[4] http://www.nisc.go.jp/active/general/pdf/crypto_pl_draft.pdf

[5] ただし、セキュリティ関連のライブラリ中には、多様な機能を盛り込んだ結果、単純には説明できない機能を備えたものも存在してはいる。

従前のクライアント／サーバ型のシステム構成

従前のクライアント／サーバ型のシステム構成においても、クライアントについては、PC のみならず、多様なものとなりつつある。:

* IC カードトークンや HSM（Hardware Security Module）を伴う構成がある
* シンクライアントが普及しつつある
* 携帯の SIM カード周辺の API が整備されつつある

また、サーバにおいては、仮想化技術が注目されており、OS の役割にも変化が見られる。権限を異にするサービスについては、それぞれ別々のゲスト OS の上で動作させるような配備・設定・構築法が注目されている。

SaaS におけるシステム構成

SaaS は、ガヴァナンスに影響を与える。SaaS のようなシステム構成においては、従前の情報セキュリティが話題として扱ってきた範囲を超えて、ネットワーク越しのサービスを信頼する仕組みも併せて考慮する必要性がある。

まず、ここにトラストあるいは信頼関係等の話題が台頭してきている。トラストモデルについては、アカデミックなの研究テーマとなっており、多様な評判システムの提案と実装が行われつつある。

また、ネットワーク越しに分散して存在しているサービスに対して、本人認証を連携させる必要性が台頭しつつあり、これは ID の管理法に影響を与えている。Web ページに ID を託する種類の ID 管理の規格として、2007 年 12 月 5 日に OpenID Authentication 2.0[6]等が公表された。

社会インフラを構成するシステムについて

X.509 証明書を利用する PKI（Public Key Infrastructure）において、基本文書ともいえる RFC 5280[7]が 5 月 9 日に発行された。この RFC 文書には、国際ストリングマッチングについての論点が盛り込まれていることが注目に値する。これは、証明書パスの検証法処理の一環で、文字列の比較をして一致を確認する処理が行われることに関するものである。

また､インターネットルーティングに関しては､中東においてインシデントが発生した。

---

[6] http://openid.net/specs/openid-authentication-2_0.html

[7] RFC 5280, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile”

情報セキュリティ分野においては、一定の攻撃を行う者を想定して、それに対する防護法に関する話題を扱うことがあるが、まず、攻撃側の動向から述べる。

攻撃側の文脈

今日、攻撃する標的を定めたウイルスが報告されている。典型的な攻撃のシナリオとして、電子メール等を利用して特定の Web サイトに誘導し、各種のウイルス等をダウンロードさせる。追加的にウイルスをダウンロードする機能も典型的である。このシナリオの一環として、近年では過去のウイルスの種類とも見なされることがあった「ファイル感染型ウイルス」が、ふたたび広く被害をもたらしている。

防護側の想定

上記のように標的を定めて攻撃するウイルスのシナリオに対しては、汎用のウイルス対策ソフトウェアでは対応できなくなる懸念がある。また、悪意ある誘導と、正規のリンク情報を区別できるようなユーザインタフェイスの重要性が再認識されつつある。（例：EV SSL 証明書[8]に対応したブラウザ等）

防護するイントラネットについての想定

検疫ネットワークを構成するために技術ついて標準化が進みつつある。このような技術は、不正な機器の接続等に対する対策として想定されている。

また、イントラネットにおいては、インベントリ管理（情報資産管理）のためのプロトコルについても経路の暗号化機能を備えるものを利用できるようにしたいという意向が台頭してきたことによって、ようやく SNMPv3 の規格が利用されつつある機運にある。

さらに、ネットワーク上で発生するイベントについて、記録する機能および可視化する機能への期待がある。

情報システム関連の災害復旧への関心

最近、国内外で大規模な地震災害が発生したこともあり、情報システムの災害対策についての関心も高まっているようである。

このように今日、情報セキュリティ関連の技術を挙げるときには、従来よりも広い範囲の事項が挙げられるようになってきた。情報セキュリティ技術動向調査 TG としては、は広い視野をもって期待に応える情報を提供していきたい。

以上

[8] http://www.cabforum.org/certificates.html

2 イントラネットセキュリティをめぐる技術動向

太田 耕平

1 イントラネットセキュリティが重視されるようになった背景

ネットワークの防御は、外からの攻撃が基本的な脅威と考えられてきた。しかし、ウイルス感染や、情報漏えい事故の多発などにより、外からの攻撃よりも内部での管理の欠如がより重大な脅威であるという認識が広まってきた。特に国内では個人情報保護法の施行以降、セキュリティ問題が身近なリスクであることの認識が広まり、イントラネットセキュリティへの意識が大幅に高まっている。

また、上記を踏まえて、法令、各種認証、セキュリティガイドラインでも内部の情報システムの管理体制を整備することが求められている。ISMS として知られる ISO/IEC 17799 認証や、個人情報保護の観点から JIS Q 15001 個人情報保護マネジメントシステム—要求事項への適合性を認定するプライバシーマーク制度は、マネジメントシステムの一部として内部ネットワーク管理の実施を既定しており、組織のガバナンスおよびコンプライアンスといった視点からもイントラネットの管理、セキュリティ体制の整備が急務となっている。

2 外部ネットワークから内部ネットワークへ - 何がネットワークに接続され、誰が、どのような利用をしているのか？

従来のセキュリティの考え方は、図 1 に示すように、インターネットとの接続点での制御によってイントラネットを防御するという考え方であったが、これはセキュリティ上の脅威は外部にあり、内部に脅威はないモデルである。しかし現在は情報漏えいなどの現実的な脅威は内部にあることが知られており、イントラネットに対する接続にこそ精密な接続管理が必要となっている。

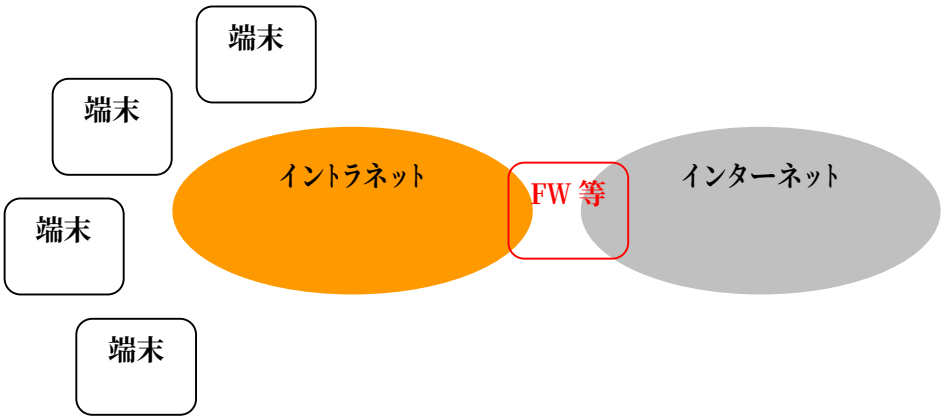


図 1: イントラネットセキュリティの重要性

イントラネットに接続される機器の中でもっとも多いのがユーザの利用する端末である。これらの端末は、持ち出し利用を含めて、外部を含め様々なサービスおよびネットワークに接続されるため、脅威にさらされる可能性が高いことから、最も大きなリスク要素となっている。しかし、端末接続の管理の重要性は十分に理解されているとはいえない。

図 2 にイントラネットで管理されるべき接続管理サイクルを示す。端末はネットワークへの接続、利用、離脱のサイクルを繰り返すが、離脱から接続までの間は接続されるネットワーク側からみればまったくの管理外であり、不正や事故などを含めリスクを管理することができない。そのため「検疫」の概念が登場し「接続」時の条件をより詳細に管理することが求められつつある。しかし、それでもサイクルのうち、ネットワークへの入り口にあたる部分の管理が始まったにすぎず、いったん接続が許可された後は十分な管理が行えるとはいえない。本稿では、イントラネットでの端末接続およびネットワーク利用管理の現状と危険性、および今後の動向を述べる。

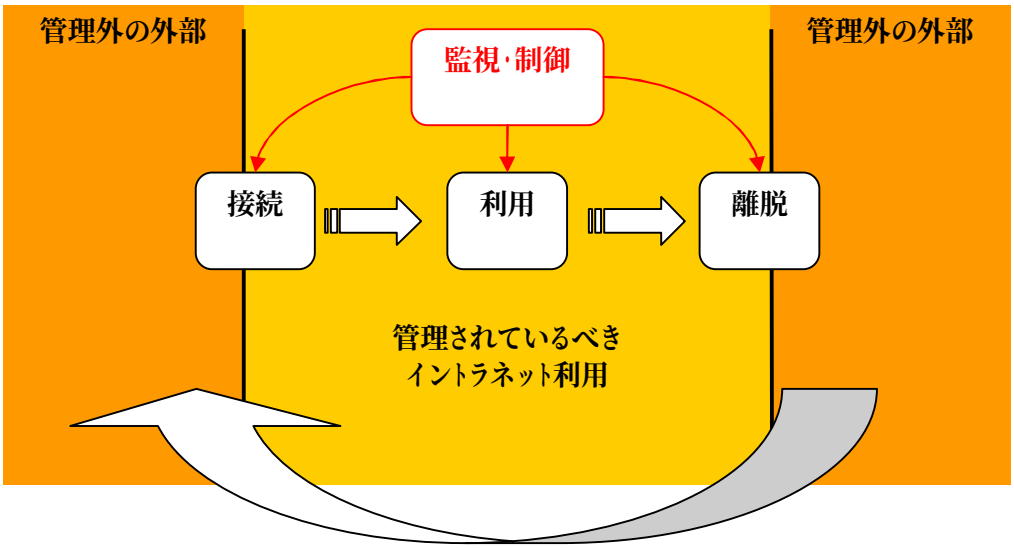


図 2: 接続管理サイクル

2.1. ネットワーク接続からネットワーク接続制御へ

「接続管理」は、ネットワーク管理の基本であるが、現時点で万人が使える端末レベルの決定的なソリューションは存在しない。したがって各種のベンダーによって様々な方式が提供されている状況である。無線接続の普及などによって、接続されている端末が動的に変化することが一般的になっているため、接続管理にはリアルタイム性が要求されているが、現状では 802.1X などの認証プロトコルなどを利用していない場合は、個別の接続を管理する一般的な方法はなく、接続管理は容易ではない。

また、その制御性にはさらに問題がある。接続には DHCP をはじめ標準化されたプロトコルが広く普及しているが、特定のネットワーク接続を強制的に「遮断」あるいは「隔離」

する手段は従来の TCP/IP の中で提供されていない。また認証機能のある 802.1X 接続や、VPN 接続でさえも、いったん正規の手続きで接続されてしまえば、それを制御することはできず、管理サイクルの他の局面では十分な制御性がない。

したがって、接続管理の現状は、接続そのものの受け入れ態勢については不十分ながらも多くの技術があり、なんらかの対策を講じることができるが、「遮断」あるいは「隔離」を実現する一般的なの仕組みは整備されておらず、最初に問題がなかった場合は、その後も問題がない、という楽天的なもので、接続の手続きだけではなく接続後の管理と制御が重要な課題となっている。

しかし、特に強制排除を含めた接続制御には課題が多く、下記の要件を備える必要がある。

1. OS、端末種別、認証の有無などの接続方式に依存しない
2. 管理者によって端末接続を個別に制御できる
3. 管理者によって任意に遮断できる

現在、上記の要件を満たすことができる一般に利用可能な接続制御として、下記の方式が知られている。

- レイヤー2 プロトコルによる強制的制御
- 802.1X などの接続認証とスイッチポート制御

2.1.1. レイヤー2 プロトコルによる強制的制御

現時点で、基本的に IP ネットワーク全般に適用可能であり、OS、端末種別等に依存せず、かつ端末を個別に、また任意に遮断できる技術として、レイヤー2 でのアドレス解決を妨害する手段が知られている。

この方式は、IP 通信の前提となるアドレス解決（ARP の動作）を阻止するため、既存の IP 接続すべてに適用できる他、任意のタイミングでこれを実施し、通信を遮断することができる強力な手法である。この方式を利用してネットワーク接続を制御する製品も複数登場しており、現時点で特別な設備の更新を必要としない現実的な技術であるといえる。

しかし一方で、正規の通信プロトコルとして認められた方法ではないため、遮断する端末数が極端に増加した場合にはネットワークへの負荷にもなりえる。

2.1.2. 認証接続とスイッチポートの制御

「認証スイッチ」は、ポート制御と認証が連動するソリューションであり、認証結果と連動してスイッチのポートを制御することで接続を制御することができるスイッチである。非常に強力であるが対応するスイッチ、対応するクライアント、RADIUS などの認証インフラ等が必要となることから、大規模ネットワークで導入するには慎重な設計と投資が必要となるため、現時点での普及は限定的である。

また、要件 2 を満たすために運用上の注意が必要となるが、現実的で確実な制御として、スイッチポート単独での制御が挙げられる。これは認証スイッチの認証部分を簡略化し、制御の部分のみを利用するものであるが、認証機能に対応したスイッチとは異なり、すでに広く普及したインテリジェントスイッチに標準的に備えられた機能のみで実現でき、端末側に特別な機能を要求しないため、既存の端末を含めて、すぐに利用できる技術である。利用には SNMP に対応したスイッチで、以下の MIB がサポートされている必要があるが、これらの MIB は、それぞれ 1992、1993、2000 年に RFC が発行されており、十分普及していることから、幅広いベンダーの製品でサポートされているため、現時点で利用できる現実的な技術である。

1. RFC1213-MIB
2. BRIDGE-MIB (RFC 1493)
3. IF-MIB (RFC 2863)

接続管理の重要性は技術的にも認知されており、IETF において標準化が進められているため、将来的には、標準化された制御方法が利用される方向に進んでいるが、普及には今しばらく時間を要する。

2.2. 端末接続制御からネットワーク制御へ

検疫システムに代表される既存の接続管理技術の多くは、接続時に一定の条件を満たしていることを保障するものであり、安全性を担保するものではない。言い換えれば既知の主要な脅威に対して一定の予防対策をとっていることを確認するものであり、未知のウイルスへの感染、好ましくないサイトの利用、さらには意図的な不正などの現実に問題となる多くの脅威は接続時の検査で明らかにすることはできない。

これらの利用上の脅威に対応するのは、運用管理技術であり、不正なアプリケーション利用、禁止サイトへのアクセスを監視、制御することが求められている。端末接続制御とネットワーク管理が連携することによって、様々な脅威に対してその発信元の特定と当該端末の強制遮断や隔離を実現することができる。

ネットワーク管理は古くからある概念であり、その重要性は総論的には認められているが、企業利益に対する貢献度合いが見えにくいことなどから体系だった運用がなされておらず、運用現場のツールという位置づけにとどまっていた。しかし、今日では、上述のように端末接続制御と連携することによって、図 1 に示す接続管理サイクルの中で重要な役割を果たすことが期待されている。

図 3 に接続管理と連携したネットワーク管理システムの概要を示す。トラフィック監視などによって不正を検知するのみならず、リアルタイムに管理された端末接続管理によって、当該端末を特定し、さらには必要に応じて強制遮断などの実効的なアクションを伴った運用管理を実現することが可能であり、接続管理サイクル全般にわたる管理を実現することができる。

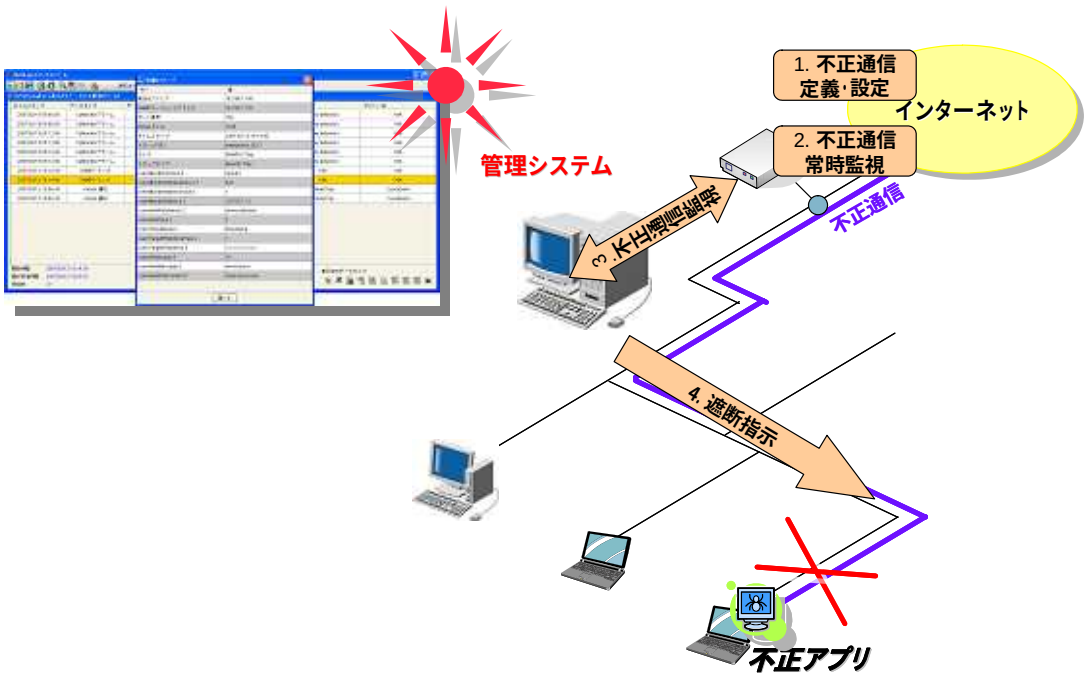


図 3 接続管理と連携したネットワーク管理

3.セキュアなネットワーク管理プロトコル SNMPv3 へ

ここまで見てきたように、イントラネットのセキュリティ確保にはこれまで以上に「ネットワーク管理」の考え方が重要になる。インターネット標準管理プロトコルとしては SNMP がその地位を確立しており、多くの機器、OS で実装され、広く普及している。しかし、そのセキュリティは、バージョン 3 によって、セキュリティ機能が導入されるまで、長らくの間脆弱なものであった。

SNMPv3 は従来からある SNMPv1 および v2c の UDP によるシンプルなプロトコルという運用形態を維持しつつ暗号化、認証、アクセス制御の機能を実現している。またリプレイ攻撃に備えた遠隔鍵更新の機能を備えるなど巧妙な技術となっている。SNMPv3 は 2002 年に RFC が発行されているように標準化は完了しているが、設定の複雑さと、それ以前の SNMPv1 および v2c が広く普及し、成功したものであるがゆえに、依然として導入が進んでいなかった。

ネットワーク管理情報は、機器のアドレス、ネットワーク構成、サービス構成など多くの重要な情報を含んだものであり、慎重に保護されるべき情報となっている。しかし多くのネットワークでは十分に保護されたものではないため、内部ネットワークに接続しさえすれば比較的容易にそれらの管理情報にアクセスすることができる。**図 1** に示した接続管理の考え方のように、内部であっても適切なアクセス制御が必要であり、ネットワーク管理にもさらに一歩進んだセキュリティ対策が必要となっている。

近年では、上記のようなリスクの認識が進むとともに、多くのネットワーク機器の SNMPv3 対応が進み、実用的な段階となっているので、今後は積極的に配備されるようになろう。

3 まとめと今後の見通し

現時点で接続管理サイクルの中で最も注目され技術開発および標準化が進んでいるのは、検疫システムとして知られる接続認証の部分であるが、接続管理の必要性は急速に認知が進んでおり、実質的なセキュリティの観点からも早期の普及が期待されている。

一方で、ネットワーク管理システムは、大きな転換点を迎えているといえる。「接続管理」（特に、積極的な接続制御）との連動が進むことによって、従来の管理システムの役割の重要性が増しており、具体的なアクションを起こすことが可能な管理システムとしてイントラネット管理の重要な要素となることが期待されている。

今後、接続管理は、本稿で述べたレイヤー2 でのアドレス解決を検知、阻止する管理方式、認証技術を背景とするネットワーク機器での接続管理方式、あるいは既存のスイッチの管理機能を利用する方式などによって導入が進み、ネットワーク管理は、既存の管理システムが他の多くのネットワーク要素と連携して積極的な役割を果たすことによって新たな重要性が認知されることが期待されている。

イントラネットの管理は、「接続管理」と「運用管理」を統合して、積極的なネットワーク制御技術としての地位を確立しつつあり、今後は、障害対策、災害復旧などを含めた総合的なリスク管理システムへと進化することが予想される。

以上

3 検疫ネットワーク技術の標準化動向

馬場 達也

はじめに

2008 年上期（1 月～6 月）は、ネットワークセキュリティ分野では、SQL インジェクションワームや、PDF ウイルス、USB ウイルスなど、新たなマルウェアの出現があったが、これらのマルウェアの対策技術の一つである、検疫ネットワーク技術の標準化に進展があったため、今回はその標準化動向について報告する。

検疫ネットワークが必要となる背景

現在、ワームやスパイウェアなどのマルウェアと呼ばれる不正プログラムの被害が深刻となっている。このマルウェアへの対策としては、ネットワークベースの対策とホストベースの対策に分けることができる。ネットワークベースの対策としては、インターネットからのマルウェアの侵入を防ぐために、ファイアウォールや IPS（Intrusion Protection System：侵入防止システム）を導入したり、メールサーバや Web プロキシサーバに、ウイルス対策ゲートウェイ製品やスパイウェア対策ゲートウェイ製品などのマルウェア対策製品を導入することが実践されている。このようなネットワークベースの対策はインターネットからの侵入に対して効果があるものの、外出先などでマルウェアに感染したノート PC を企業ネットワークに持ち込むことによって感染が拡大するケースに対しては無力である。このような場合は、エンドポイントとなる各クライアントでのホストベースの対策が有効である。

ホストベースの対策としては、ウイルス対策ソフトやスパイウェア対策ソフトなどのマルウェア対策ソフトの導入や、パーソナルファイアウォールの導入、マルウェアが感染する際に悪用するセキュリティホールを塞ぐためのセキュリティパッチの適用が有効である。しかし、クライアントで対策を行うには、そのクライアントを使用しているユーザが適切な対策を行う必要があるが、それを徹底させることが難しいという問題がある。このため、検疫ネットワークシステムという仕組みが提案されてきた。

検疫ネットワークシステムの定義

検疫ネットワークシステムは、PC をネットワークに接続する際に、接続 PC のセキュリティ対策状態を検査し、その状態が、ネットワークの基準に合格する場合にのみ、ネットワークへの接続を許可する。基準を満たせなかった場合は、検疫ネットワークに隔離され、基準を満たすために行うべきアクションが指示される。検査されるセキュリティ対策状態には次のようなものがある。

・マルウェア対策ソフトでリアルタイム保護が有効になっているかどうか

・マルウェア対策ソフトの定義ファイルが最新になっているかどうか
・適切な OS のパッチが適用されているか
・不適切なソフトウェア（P2P ソフトなど）がインストールされていないかどうか
・利用すべきソフトウェア（パーソナルファイアウォールなど）が動作しているか

検疫ネットワークシステムの導入の目的としては、①マルウェアに感染した PC をネットワークに接続させない、②ネットワークに接続される PC の対策を徹底させる、の 2 つが考えられるが、①の目的は「検疫」という意味に合致しているものの、実際はネットワーク接続時にマルウェアに感染しているかどうかをネットワーク側がチェックすることは難しい。このため、現段階では、①の目的ではなく、②の目的で導入する必要がある。また、ネットワーク接続時に、ユーザ認証を行うだけのシステムや、MAC アドレスをチェックするだけのシステムは、マルウェア対策という目的に合致しないため、検疫ネットワークシステムには含まれない。

検疫ネットワークシステムの方式

以上の仕組みを実現するため、検疫ネットワークシステムでは、クライアント上の検疫ソフトウェア、検査を実施する検疫サーバ、ネットワークへの接続を制御するレイヤ 2 スイッチや無線 LAN アクセスポイント、VPN ゲートウェイなどの様々なコンポーネントが協調して動作する必要がある。

検疫ネットワークの方式としては様々なものが提案されているが、アクセス制御を実施するポイントで分類すると、「認証スイッチ方式」「ゲートウェイ方式」「DHCP 方式」「パーソナルファイアウォール方式」の 4 種類に大別される。

「認証スイッチ方式」は、PC を有線 LAN または無線 LAN でレイヤ 2 スイッチや無線 LAN アクセスポイントに接続する際に、IEEE 802.1X や Web 認証の仕組みと連携して、セキュリティ対策状態をチェックする。ネットワークのエンドで制御できるので、基準を満たせなければ、全くネットワークに参加できない方式であり、効果が高い方式と言える。しかし、すべてのエンド機器が IEEE 802.1X や Web 認証に対応していなければならないため、コストがかかるという問題がある。

「ゲートウェイ方式」は、ネットワーク上の中間ノードであるルータやレイヤ 3 スイッチ、ファイアウォール、VPN ゲートウェイなどをトラフィックが通過する際に検査を行う方式である。認証スイッチ方式と比較して、対応機器が少なく済むため、導入のコストを低く抑えることが可能である。ただし、ゲートウェイを通過しない通信はチェックを受けなくても可能となってしまうという問題がある。

「DHCP 方式」は、DHCP の仕組みを利用した方式であり、ネットワークへの接続時には、限られた範囲しかルーティングされない一時的なアドレスを割り当て、検査に合格した場合にのみ、通常のアドレスを割り当てるという仕組みである。導入が比較的容易な方式ではあるが、DHCP を利用していない環境では導入できないという問題や、静的にアド

レスを設定されてしまった場合には効果がないという問題がある。

「パーソナルファイアウォール方式」は、クライアント上にインストールされたパーソナルファイアウォールでアクセス制御を行う方式である。ネットワーク上には、チェックを行うパーソナルファイアウォールのサーバがあればよいというメリットがあるが、パーソナルファイアウォールがインストールされていないクライアントは、検査を受けなくてもアクセスが可能となってしまうという問題がある。

検疫ネットワークシステムの標準化動向

検疫ネットワークシステムはさまざまなベンダから製品がリリースされているが、現在、有力とされているのが、以下の 3 方式である。

・Cisco Systems の NAC（Network Admission Control）
・Microsoft の NAP（Network Access Protection）
・TCG（Trusted Computing Group）の TNC（Trusted Network Connect）

このうち、Cisco NAC と Microsoft NAP は相互接続可能にするために、NAP クライアントとして動作する Microsoft Windows Vista および Windows XP SP3 に対して、Cisco NAC の技術である”EAP-FAST”と”EAP over UDP”を提供しており、Windows Vista および Windows XP SP3 は特別なクライアントソフトウェアを導入する必要なしに、Cisco NAC クライアントとして動作可能となっている。また、Microsoft は、NAP プロトコルを TCG に提供し、TNC の仕様（IF-TNCCS-SOH）としてリリースした。これにより、TCG TNC に準拠したシステムは、Microsoft NAP の機能も持つようになり、Microsoft NAP と TCG TNC は相互接続できるようになる。このように、相互接続のための努力は行われているものの、実際は、それぞれがお互いの仕様を実装しているだけであり、方式が一本化されているわけではない。

このような状況の中、IETF（Internet Engineering Task Force）では、2006 年 7 月に NEA （Network Endpoint Assessment）BOF を実施し、検疫ネットワークシステムのプロトコルの標準化に乗り出した。NEA BOF は 2006 年 11 月に正式に WG として活動を開始し、前述の Cisco Systems や Microsoft、TCG TNC 陣営の Juniper Networks、Symantec などのベンダが参加し、プロトコルに対する要求仕様をまとめてきた。そして、この要求仕様が固まり、2008 年 6 月に RFC 5209 として発行された。これに伴い、NEA WG は具体的なプロトコルの検討に入り、2008 年 2 月に TCG TNC 陣営から、PA（Posture Attribute）プロトコルおよび PB（Posture Broker）プロトコルの仕様が提示され、2008 年 4 月にこれらを WG の正式ドキュメントとして標準化を進めることに合意した。

IETF NEA WG では、「NEA 参照モデル」にて次の 3 つのプロトコルを定義している。

・PA（Posture Attribute）プロトコル
・PB（Posture Broker）プロトコル
・PT（Posture Transport）プロトコル

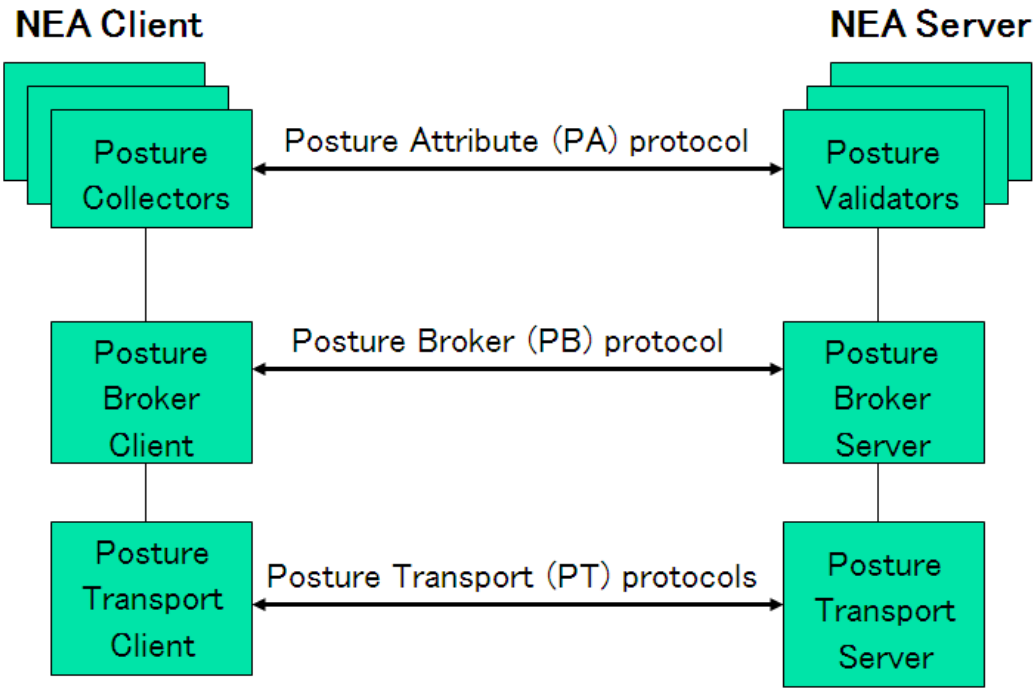


図 1: IETF NEA 参照モデル

PA プロトコルは、Posture 情報と呼ばれる、セキュリティ対策状態を運んだり、アセスメント結果や改善手順（Remediation）を通知するためのプロトコルである。PB プロトコルは PA プロトコルで運ばれる属性メッセージを集約して運ぶプロトコルであり、総合判断結果（Global Assessment Decision）を通知する役割を持つ。PT プロトコルは、セキュリティを確保しながら PB プロトコルを運ぶ役割を持つトランスポートプロトコルであるが、EAP（Extensible Authentication Protocol）および EAP を運ぶための IEEE 802.1X（EAPOL: EAP over LAN）、IKEv2、RADIUS などの既存のプロトコルを使用することとなっており、標準化対象には含まれていない（認証スイッチ方式では IEEE 802.1X、VPN ゲートウェイ方式では IPsec の IKEv2 が利用可能）。

PA プロトコルおよび PB プロトコルは、TCG TNC 仕様をベースに標準化が進もうとしている。PA プロトコルは、TNC の IF-M 1.0 プロトコルとほぼ同一の内容となっており、PB プロトコルは、TNC の IF-TNCCS 2.0 プロトコルとほぼ同一の内容となっている。実のところ、TNC の IF-M 1.0 および IF-TNCCS 2.0 は、IETF NEA WG が策定したプロトコル要求仕様に従って規定されており、PA プロトコルおよび PB プロトコルが RFC 化された場合には、TNC の IF-M および IF-TNCCS 仕様と相互接続性が確保されることになる（TNC IF-M 1.0 および IF-TNCCS 2.0 は現在、TCG においてパブリックレビュー中というステータスになっている）。

まとめと今後の展開

複数の方式が乱立していた状態であった検疫ネットワークシステムであるが、IETF NEA WG において、TCG TNC 仕様をベースに具体的なプロトコルの標準化が進み始めた。現在は、各ベンダがお互いのプロトコルを実装しあうことで相互接続性を確保しようとしているが、実装のための手間や、相互接続の条件の複雑さなどによる利用者側の混乱など、様々な問題が発生すると考えられる。

IETF NEA WG では、2009 年を目途に検疫ネットワークプロトコルを RFC 化すると表明している。これを機に、各ベンダの検疫ネットワークシステム製品も IETF 標準に準拠すると予想され、検疫ネットワークシステムが普及するきっかけとなると考えられる。

また、現在は標準化対象としていない PT プロトコルではあるが、エンドのレイヤ 2 スイッチや無線 LAN アクセスポイント、VPN ゲートウェイ以外のファイアウォールやルータなどでアクセス制御を行う場合には、標準となるプロトコルが存在しないのが現状である。Cisco NAC では、中間のルータで検疫を実施するために、IEEE 802.1X で使用されている EAPOL（EAP over LAN）を、レイヤ 3 ネットワークで運べるように拡張した EAP over UDP を独自に開発して利用しているが、このようなプロトコルの標準化が必要になる可能性もある。

以上

4 インシデント対応・災害復旧

武田圭史

インシデント対応･災害復旧に関する技術動向としては大変広範な領域が対象となるが、本稿では特に 2008 年上半期に発生したインシデントで多く用いられた攻撃手法について分析する。昨年あたりから日本国内においても不正な行為によって利益を得ることを目的とした攻撃の増加が確認されておりその攻撃手法の態様も若干変化している。ここではその変化について述べるとともに、用いられる複数の攻撃手法の違いについて着目する。

概要

2007 年までのここ数年の､企業官公庁等組織における情報セキュリティインシデントの動向としては職員の自宅での暴露型ウイルス感染による匿名ファイル共有ネットワークへの情報漏えい、パソコンおよび USB メモリなどの外部記憶媒体の紛失・盗難、電子メールの宛先に互いに第三者のアドレスを列挙することによる電子メールアドレスの漏えいなど､ユーザのミスや管理の不備に起因する事故が多く報告されてきた。2007 年の後半あたりからは、主に海外のアドレスから悪意を持った攻撃者が意図的にシステムの脆弱性を攻略しサーバに格納されている情報を盗んだりウェブサイトのコンテンツが改ざんされるといった事件が増加している。特に 2008 年上半期の大きな特徴としては、SQL データベースを利用したウェブアプリケーションを運用するウェブサイトの大規模な改ざんや、情報の盗難事件が多く発生したことがあげられる。本稿では 2008 年上半期のインシデントにおいて特に印象的な要素となった､SQL インジェクションによる大規模ウェブサイトの改ざん、ウェブ改ざんを通じたクライアントへのマルウェアの感染、クライアントへの中規模なマルウェアの感染を狙ったターゲテッドアタックの攻撃態様について解説する。

SQL インジェクション

SQL インジェクションは脆弱性のあるウェブサイトに対して､データベースへの問い合わせ言語である SQL の命令要素を含むリクエストを送信し、データベースマネジメントシステムの機能を悪用することによって攻撃者が顧客情報やウェブで処理を行う各種コンテンツに対して任意の処理を行わせる攻撃である。

SQL インジェクションの攻撃手法自体は決して新しいものではないが、2005 年頃から比較的多くのユーザを抱える企業などのサービスがこの攻撃を受け、顧客情報などを流出させる事件が断続的に発生していた。攻撃は断続的に行われているが事故が多く発生する時期は集中していることから比較的少人数の攻撃者が大規模な範囲に対して攻撃を行っていることが推察される。また近年 SQL インジェクションの脆弱性の発見を容易にするツールなどが高度化しており、こういったツールを使用して広い範囲を走査してウェブアプ

リケーションの脆弱性が発見されていることも攻撃の増加の一因となっていると考えられる。

2008 年上半期では特に 3 月以降に SQL インジェクションの攻撃事例が IPA や JPCERT/CC に対して報告されている他、日本のみならず世界各国のセキュリティ企業が大規模な SQL インジェクション攻撃に対する警告を発している。この 2008 年上半期に発生した大規模な SQL インジェクション攻撃で被害を受けたサイトの多くは、ウェブページに HTML の iframe タグが埋め込まれ動的な DNS によって指定される攻撃者が利用するウェブサイトからマルウェアをダウンロードするようなものに書き換えられ、このサイトにアクセスした利用者が脆弱性のあるブラウザなどを利用していた場合にアカウント情報を搾取したりスパム送信に利用可能なマルウェアに感染するというものだった。

SQL インジェクション自体はウェブアプリケーションの開発者がこの脆弱性のリスクを理解し、プログラム処理において文字列を動的に連結して SQL 文を作成するのではなく、あらかじめ定義された命令文にユーザ入力をパラメータとして適用するプリペアードステートメントを利用する、ユーザ入力文字を適切にエスケープするなどの方法を用いることで回避することが可能である。またウェブアプリケーションに対するソースコード監査、ツール及び手動による脆弱性検査により脆弱性の存在を確認することも有効な対策といえるだろう。情報処理推進機構からは「ウェブサイトの脆弱性検出ツール iLogScanner」が無償で公開されており、こういったツールを使ってウェブサーバのログを解析することで、SQL インジェクションによる攻撃が発生した場合に、ある程度の確率で事後的に発見することができる。

（関連情報）

SQL インジェクション攻撃に関する注意喚起　(情報処理推進機構, 2008/5/15)
http://www.ipa.go.jp/security/vuln/documents/2008/200805_SQLinjection.html

ウェブサイトの脆弱性検出ツール iLogScanner　(情報処理推進機構, 2008/4/18)
http://www.ipa.go.jp/security/vuln/iLogScanner/

「安全なウェブサイトの作り方　改訂第 3 版」を公開　(情報処理推進機構, 2008/3/6)
http://www.ipa.go.jp/security/vuln/websecurity.html

ウェブサイトの改ざんによるマルウェアの配布

前項で紹介したものを含め 2008 年上半期の傾向としては比較的規模の大きな企業等著名なサイトが SQL インジェクション等の攻撃によってウェブページを改ざんされ、このサイトにアクセスをしたサイトの利用者がマルウェアに感染するといったタイプの攻撃が行われることが多くなっている。従来の ウェブアプリケーションやウェブサーバに対する

攻撃の多くは、サーバやデータベースに格納される情報を搾取したり愉快犯的にウェブページの内容を書き換えるといったタイプのものが多かったが、アクセスの多いサイトがマルウェアの感染を広げる媒体として悪用される形となっており新しい傾向といえるだろう。確認されている攻撃の多くはウェブページの内容を書き換え中国にある無料の動的 DNS サービスを利用して任意の IP アドレスから不正な Java スクリプトをユーザにダウンロードさせ、クライアントの環境の脆弱性を攻略してマルウェアに感染させるというタイプが多い。こういった動きは、従来自己増殖型のウイルスなどによって不正な処理を行うプログラムを配布していた攻撃者が、ウイルス対策ソフトの普及によって従来の感染を広げる方法の効率が悪くなったことや、攻撃の動機がより利益を追求するようになってきたことなどもあり、より短時間に多くのユーザに感染を広げることを狙いとした手法として多くのユーザを集めるサイトが狙われるようになっている。

スパム型トロイ攻撃（SPAM Trojan Attack）

前項でも述べたように攻撃者の関心が攻撃によって利益を得ることに向いてきたことから、短時間で効率のよい方法が攻撃に用いられることが増えておりその一つに、スパム型トロイ攻撃がある。これはセキュリティパッチや自己解凍型の圧縮ファイルなど一見有用なプログラムのように見えるプログラムに不正な処理を行うコードを埋め込んだプログラムを迷惑メールの容量で大量の受信者に対して一斉送信するというものである。何らかの有用なプログラムやデータに見せかけてユーザに不正プログラムを実行させるような攻撃ファイルやデータをトロイの木馬と呼ぶことからこのような名称となっている。使用される不正プログラムは都度新しいものが使用されるために、ウイルス対策ソフトの検知パターンにこれらは含まれず、従来の単純なパターンマッチでは検出できない。以前よりスパム型トロイ攻撃は行われてきたが、実行形式のファイルが送付される場合が多く、またメッセージの本文が英語で書かれている場合が多かったりしたために容易に発見することができ、日本国内では実際の被害に結びつくことはそれほど多くはなかった。今年に入ってからの傾向として、本文に巧妙な日本語でメッセージが書かれているものや添付ファイルが実行形式ではなく、ワードやエクセル、PDF ファイルなど一般にメール添付などでやりとりされるデータ形式のファイルであり、ユーザを安心をさせてこれらを開かせるようなものも多く出現している。これらのファイルはパッチが公開されていない脆弱性を攻略するようなものも多く、攻撃者の知識やスキルが高まっていると思われる。

標的型トロイ攻撃（Targeted Trojan Attack）

標的型トロイ攻撃は、特定の企業や組織、人物、あるいは企業の経営者などのように特定の属性を持つ人々を対象に、そういった受信者が感心を持つ事柄をメッセージに含め、巧みに添付するファイルを開かせるようなタイプの攻撃である。攻撃にあたっては攻撃対象に対する知識や理解が深ければ深いほど、よりそれらしい文面となり攻撃の成功が高く

なる。ターゲテッドアタックについてはセンセーショナルに報道されることが多いが、攻撃を受けたという事例が確認されることは多いがそれによって実際にどの程度の深刻な被害が発生しているかという点については不明である。ターゲテッドアタックに関しては 2008 年上半期日本国内において政府関係の人物を騙る日本語の巧妙なメールが出回るなどの事態が確認されている。

スピア型フィッシング攻撃（Spear Phishing Attack）

一般的なフィッシング攻撃は、クレジットカードのサービス更新や各種アカウントの更新を呼びかけるようなメールを大量のユーザに対して送信し、これらのメールから虚偽のサイトに誘導し、不用心なユーザがこの偽サイトに ID パスワードなどのアカウント情報や、クレジットカード情報を入力することを待つという攻撃である。「スピア型フィッシング攻撃」は、より対象を限定し、特定のユーザグループに対してフィッシング攻撃の仕組みを適用するものである。日本国内では前項の標的型トロイ攻撃を差して「スピア攻撃」などという用語が用いられることがあるが、ここであげたスピア型フッシング攻撃と混同を招くため避けた方がよいだろう。当初の「スピア型フィッシング攻撃」は、上記のようにいわゆるフィッシングサイトに誘導し情報をユーザ自身に入力させるものを意味していたが、最近では巧みな電子メールによって被害者を攻撃者の支配下にあるサイトにアクセスさせ、ここからトロイの木馬など不正なプログラムをダウンロードさせるような手口を意味するものとして使用されることも多い。米国など海外では、こういった形での「スピア型フィッシング攻撃」による被害が多く報道されている。

まとめ

本稿では、2008 年上半期のインシデントに関する動向として、日本国内においても不正な攻撃者が効率よく利益を得ることを目的として、不正なプログラムを短時間に効率よく配布し感染させるための手段として主に SQL インジェクションを用いてアクセスの多いウェブサイトを改ざんをするケースと、様々な形で電子メールを使用するケースをとりあげた。これらの脆弱性に対する対応は短時間で普及させることは容易ではなく下半期においても引き続き被害の発生が見込まれる。また、ここで紹介した手法が組み合わされて用いられることも多くなっており、早急にこれらの攻撃被害の実態を継続的に把握し、有効な対策技術について研究開発を推進することが必要である。

以上

5 Linux 用の新しいセキュア OS：SMACK

面 和毅

2008 年 4 月 17 日に、Linux の新しいカーネル 2.6.25 が登場した。2.6.25 からは、SELinux の他に、新たなセキュア OS として SMACK（Simplified Mandatory Access Control Kernel）が追加された。これは、昨年から続いていた、Linux におけるセキュア OS 実装の論議に対して、カーネルメンテナー側が出したひとつの回答となっている。

Linux におけるセキュア OS 実装の議論(2007 年末)

Linux 上では種々のセキュア OS が 2.4 カーネルの時代から実装されて来た。2.6 カーネルで Linux のカーネル内にセキュリティ実装を盛り込むことが決まったが、SELinux や AppArmor(当時は Subdomain と呼ばれていた)など、どれか一つのセキュリティ実装に縛られることを嫌った Linus が、LSM(Linux Security Module)と呼ばれる共通のプラグインを提案し、全てのセキュア OS は LSM 対応させた実装を行うようになった。この LSM に対応して、2003 年にいち早くカーネルツリーにマージされた物が SELinux である。

その後、2006 年まで SELinux 以外のセキュア OS は提案されず、2006 年に LSM を廃止するべきではないかと言う議論が LKML(Linux Kernel Mailing List)で行われた。この提案がきっかけとなり、AppArmor が LKML に提出された。また、その他のセキュア OS 実装(SMACK、TOMOYO Linux など)もほぼ同時に提出された。

以降、SELinux と AppArmor によるセキュア OS の大議論が展開されて行く。双方とも異なるセキュリティ理論の実装となっているが、特にベースとなる identifier が、SELinux では個々にラベルを貼って行く形となっているが AppArmor ではファイルへのパス名を identifier としているという点から、両者の議論は単なる理論のディベートから、互いの中傷合戦にまで発展した。また、SMACK に関しては、「SELinux のポリシの書き方で SMACK と同様の動作をさせられるので、SMACK は不要なのではないか」という意見まで出され、再度「現状では SELinux しか LSM を使っていないので、LSM を廃止できるのではないか」という意見が出された。

この議論に終止符をつけたのが、LKML に出された Linus のメールである。このメールでは、「You security people are insane. I'm tired of this "only my version is correct" crap.」と言われており、LSM を残し、AppArmor や SMACK のマージをしてこの種類の議論の終結にしたいとの発言がなされた。

ここまでが、2007 年末までのセキュア OS に関する大議論の要約である。

SMACK の説明

SMACK は SELinux と同じく identifier としてラベルを用いたセキュリティの実装になっている。SMACK では、プロセス、ファイル、SVIPC などにラベルを付けて行く。ファイルに関しては、そのファイルを生成したプロセスと同じラベルが付けられて行く。
これらのラベルに対して、SMACK では 23 文字までの名前を付けることができる。また、次の 4 つの特別なラベルがある。

"*" - "star"
"_" - "floor"
"^" - "hat"
"?" - "huh"

アクセスルールとしては、下記のルールが強制される。

(1) "*"ラベルが付けられているタスクからの要求に対しては、全て DENY となる。
(2) "^"のラベルが付けられているタスクからの読み込み／実行要求は、許可される。
(3) "_"のラベルが付けられているオブジェクトにたいする読み込み／実行要求は、許可される。
(4) "*"のラベルが付けられているオブジェクトに対しては、全ての操作が許可される。
(5) タスクとオブジェクトのラベルが一致する場合には、全ての操作が許可される。
(6) ロードされたポリシにより明示的に定義されている操作は、全て許可される。
(7) その他のアクセスは全て拒否される。

/smack/load に subject, object, access に関して明示的にルールファイルを書き込んで行く。この SMACK の面白い所は、BLP(Bell&LaPadula)モデルや Biba モデルなどをルールファイルによって簡単に設定できる所にある。例えば、

C(confidential) Unclass rx
S(secret) C rx
S Unclass rx
TS(TopSecret) S rx
TS C rx
TS Unclass rx

とすれば、TopSecret のプロセスが Confidential のファイルに Read/Execute することができる。

SMACK のマージと今後の Linux におけるセキュア OS 実装

一方、AppArmor は再度 2008 年 6 月 3 日に LKML にパッチを提出したが、今度も大議論を巻起こしている。それは、彼らのアクセス制御に必要となるため、セキュリティ部分以外に VFS レイヤなどにもパッチをマージさせようとしていることに起因している。

ここでも SMACK という前例を見ると、SMACK ではセキュリティ部分(カーネルソース security/以下)以外を変更するようなパッチを出していないため、他の開発者達を刺激しなかったことが LKML でスムーズに受け入れられた原因となっているようだ。

このように、SMACK を前例としてそのカーネルへのマージまでのプロセスを見ることで、Linux への新しいセキュア OS の「良い」提案方法が理解できる。

セキュア OS も、単一のものよりも複数実装があった方が、それぞれでブラッシュアップしていく効果が見込まれるため、今後も SMACK に続いて AppArmor や他のセキュア OS が実装されていくことがユーザとしての観点からは望まれる。

以上

6 仮想化ソフトウェアのセキュリティ

面 和毅

仮想化というキーワードは昨年から IT 業界でホットな流行語となっており、様々な製品が市場に出て来ている。

1 仮想化製品のセキュリティアドバイザリ

仮想化製品の 2008 年上半期でのセキュリティアドバイザリをふりかえる。下記の 3 つの仮想か製品について調査を行った。

(1) VMWare
(2) Xen
(3) VirtualPC/HyperV

(1) VMWare

米 VMware、管理スイートに災害復旧機能を追加 (2008 年 5 月 14 日)

i) VMSA-2008-0005.1(2008 年 3 月 21 日)
VMWare 製品群に見付かった、数種類のセキュリティ脆弱性の修正となっている。

1. ホスト OS が Windows の際に、ゲスト OS との間で共有フォルダ(HGFS)を使った際に、ゲスト OS からホスト OS のファイルシステムにアクセスして実行可能なファイルを作成できる可能性があるという脆弱性が発見されたため、修正が行われた。

2. 悪意のあるユーザが"authd"処理で名前付きパイプを用いてコントロールし、LocalSystem 権限を取得して悪用できるという脆弱性が発見されたため、修正が行われた。

3. さまざまなセキュリティ上の脆弱性を取り除くために、libpng ライブラリがバージョン 1.2.22 に更新された。

4. 古いバージョンの OpenSSL ライブラリ中に発見された脆弱性を取り除くために、OpenSSL ライブラリが更新された。

5. Windows2000 がホスト OS となっている場合に、いくつかの脆弱性のあるプロセスを通じて不正な権限昇格が行われる脆弱性が発見されたため、修正が行われた。

6. ホスト OS 上で動作している DHCP サーバに影響する DoS 脆弱性が修正された。

7. "config.ini" ファイルのパーミッションの問題。権限の無いユーザが、セキュリティ回避または、VMX 起動または、フルパス操作を行うことが可能なため、修正が行われた。

8. Virtual Machine Communication Interface(VMCI)で、メモリの破損が発生することが確認されたため、修正が行われた。

ii) VMSA-2008-0006 Updated libxml2 service console package (2008 年 3 月 28 日)
libxml2 パッケージに DoS 脆弱性があったため修正を行った。

iii) VMSA-2008-0009.1 (2008 年 6 月 4 日)
VMware Host Guest File System (HGFS) 共有フォルダ機能に buffer overflow する欠陥があり、guest から任意のコードを vmx プロセス権限で実行できる脆弱性があったため、修正を行った。

(2) Xen
CVE-2008-1619 (2008 年 4 月 2 日)
IA64 アーキテクチャ上の Xen 5.1 内で ssm_i エミュレーションを動作させた際に、ドメイン 0(dom0)に DoS 攻撃を行って Panic を起こさせることができるという脆弱性が発見された。

(3) Virtual PC/Hyper V
該当は無かった。

これらの脆弱性を見てみると、大別して 2 つの傾向があるように感じられる。

- 使用しているライブラリにセキュリティ上の問題があったことによる脆弱性
- 他の脆弱性を利用して、ゲストからホスト OS に影響を与えることができる脆弱性

2 仮想化製品とセキュリティ技術

米 VMware 社が 2008 年 3 月 27 日に、仮想環境で実行されているアプリケーションを保護する新セキュリティ技術「VMware VMsafe API」を発表した。

(1) VMSafe 前

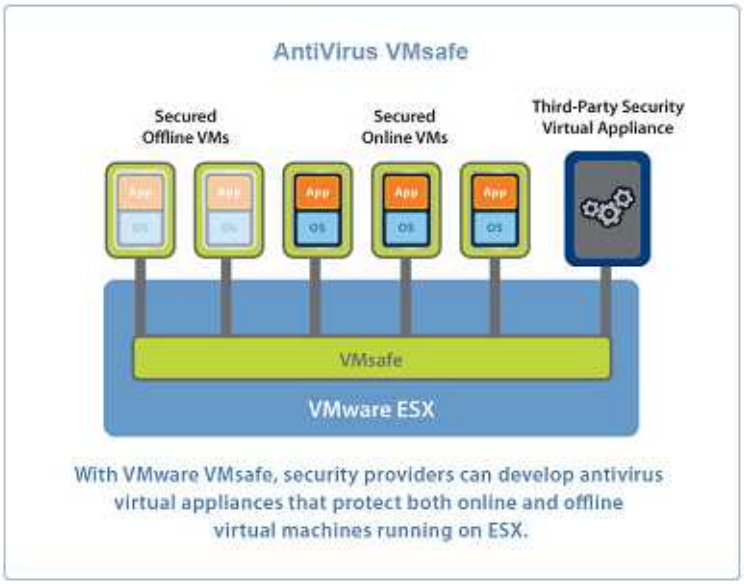
これまでのセキュリティ製品では、各ゲスト OS 内にエージェントをインストールして、各ゲスト OS を通常の物理システムと同様に見立ててセキュリティの対応を行って来た。

また、セキュリティに特化した仮想アプライアンスを使用し、仮想スイッチを使って他のゲスト OS に接続してトラフィックなどを監視していた。しかし、VMotion のように、仮想マシンを動的に別のホスト OS 上に移動した際には、ライブ移行と再起動のプロセスが複雑になってしまっていた。

(2) VMSafe 後

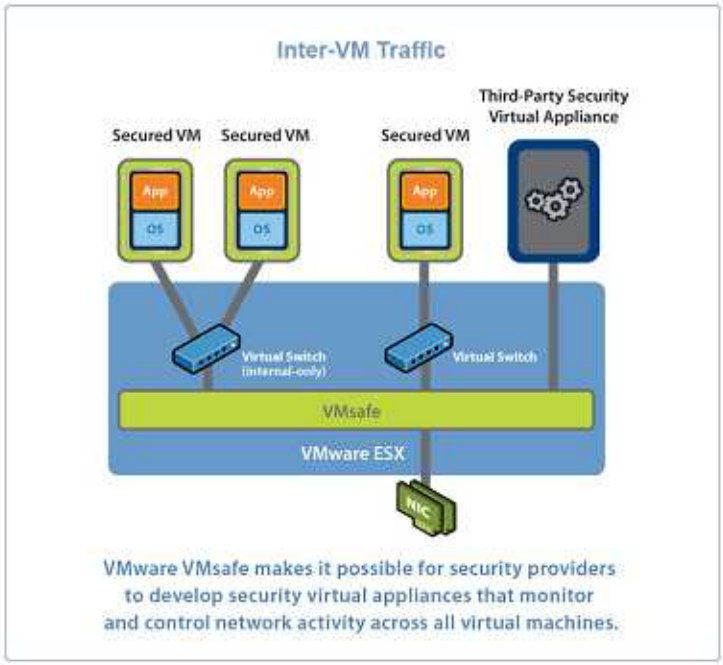
VMsafe API を用いると、ウイルス対策のようなセキュリティ製品は依然として各ゲスト OS 内にインストールする必要があるが、ハイパーバイザーを介して他の仮想マシン内部で起こっていること(CPU ステータス、メモリ、OS プロセス、トラフィックなど)を監視することができる。

これにより、ハイパーバイザーレベルでセキュリティポリシーを強制できる専用のセキュリティアプライアンスを作成すれば Antivirus をオンラインにしていないゲスト OS の内部も監視することができるため、オンライン／オフライン両者を保護することができる。



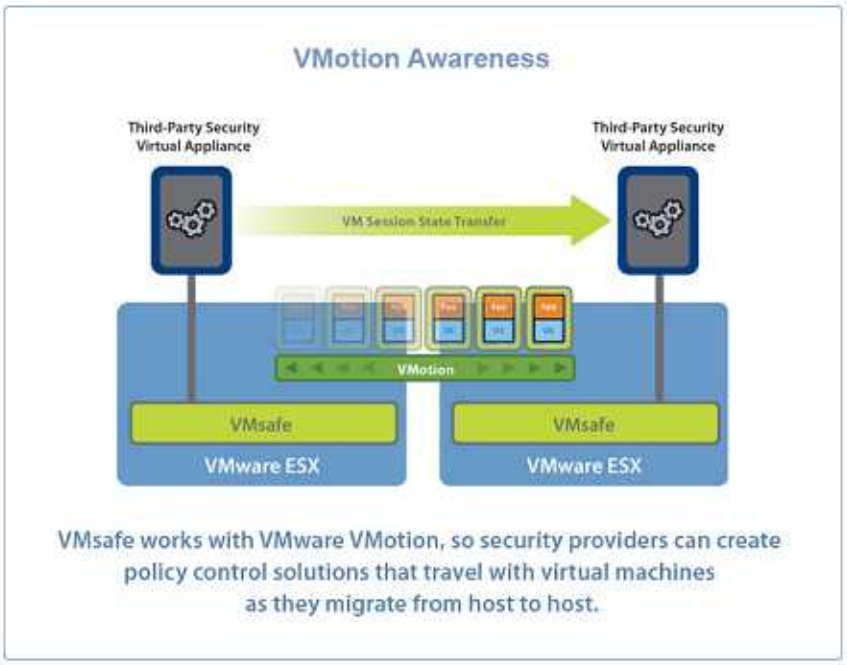
図：vmsafe_antivirus

また、ハイパーバイザーレベルでアクセスできるため、全ての仮想マシンの仮想ネットワークを監視したり切替えたりすることができる。



図：vmsafe_kernel

その他にも、各ホスト OS に仮想アプライアンスをあらかじめインストールしておけば、VMotion を使ってゲスト OS を他のホスト OS に切替えた際にも、動的に処理を引き継ぐことが可能になる。



図：vmsafe_vmotion

この VMsafe API **を利用して、**CheckPoint **や** McAfee**、**Symantec **といった** 20 **以上のセキュリティ企業が現在セキュリティ製品の開発を行っている。**

以上

7 PKI 関連の動向（RFC 5280）

木村 泰司

2008 年 5 月、IETF PKIX WG より RFC 5280[9]が出された。これはインターネットにおける PKI の、基本的な仕様を定義した RFC で、多くの RFC が準拠しており、与える影響が大きいドキュメントである。PKI 関連の出来事の中で特に注目すべきトピックとして紹介する。

概要

RFC 5280 はインターネットで使われる X.509v3 証明書と、X.509v2 CRL を定めたもので、RFC 3280 の後継の位置づけにある[10]。RFC 5280 の大きな特徴としていえるのは、国際化された名称[11]への対応であろう。この他に、名称（DN[12]）の UTF8String 形式への限定が解除されるなどした。

本稿では、本 TG の開始直後に出た RFC 5280 の特徴を挙げた上で、インターネット PKI の現状について考察する。

国際化された名称への対応

国際化ドメイン名[13]に関する議論では、しばしば、見た目が同じでも異なる文字であるとか、逆に、見た目が異なっていても同じと見なされる文字の取り扱いが問題となる。すなわち、文字の取り扱いルールの明確化が必要となる。

RFC 5280 では、国際化の対応の為に 7 章が設けられ、先行して議論されていた RFC4518[14]が適宜参照される形で組み込まれた。DN、IDN、IRI[15]、メールアドレスの各々について、名称の比較ルールの定義が行われている。以下に、特にルールが複雑な DN と IRI の扱いについて述べる。

- DN
  RFC 3280 では、UTF8String 形式のみが DN で使われ、比較の際にはバイナリー値として扱われるとされていた。一方、RFC 5280 では UTF8String 形式ない

---

9 RFC5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile

10 そのため Son of RFC3280 や rfc3280bis などと呼ばれてきた。

11 Internationalized Names

12 Distinguished Name

13 Internationalized Domain Names

14 RFC4518: Lightweight Directory Access Protocol (LDAP): Internationalized String Preparation

15 Internationalized Resource Identifiers

し PrintableString 形式でよくなり、オプションとして TeletexString 形式、BMPString 形式、UniversalString 形式を使ってもよいことになった。ただし比較は RFC 4518[16]で定められた 6 つのステップを持つアルゴリズムが適用される。

- IRI
  IRI は Unicode 文字列の URI である。直接、証明書や CRL に記載されることはないが、RFC3987[17]で定義されたマップに従って ASCII に変換されたものが subjectAltName 拡張などに入っている可能性がある。しかしこれらは GeneralName 形式であり、IA5String である。つまり Unicode が ASCII 変換された文字列で入っていることになる。この比較は複雑で、始めに RFC3987 で定義された ASCII 変換が行われ、スキーマ部やホスト部を正規化（normalize）し、パーセント符号化部分を正規化し、比較する。URI の比較の際には大文字と小文字を区別する。

このふたつの他にも、フィールドごとにルールがあり、期待される処理は複雑である。証明書検証の安全性に大きく影響すると考えられるため仕方がないという側面はあるが、証明書検証を行うプログラムは、多くの比較ルールをサポートする必要が出てきている。

UTF8 対応に関わる日本勢の活動

前節で述べた UTF8 を使う国際化の対応には、当時 IPA で行なわれた調査結果からのインプットがあった。

IPA では「PKI における UTF8String 問題に関する調査」を JNSA（NPO 日本ネットワークセキュリティ協会）に委託して実施していた。この調査結果は、当時 RFC 5280 の著者に選任されたばかりの David Cooper 氏にインプットされている。

発行者および主体者の名称に対するエンコーディング

RFC 3280 では、発行者および主体者の名称を格納する際のエンコーディングとして、2003 年 12 月以降は UTF8String 形式にしなければならない、という記述があった。RFC 5280 では、この制限が廃止され、複数のエンコード方式から選択できるようになった[18]。

その他の課題への対応

RFC 5280 では、RFC 3280 では証明書の処理の上で起こる不具合を避けるための改善

16 RFC4518: Lightweight Directory Access Protocol (LDAP): Internationalized String Preparation
17 RFC3987: Internationalized Resource Identifiers (IRIs)
18 RFC4630 で記述されたものが組み込まれた。

も行われた。CRL に AIA[19]拡張を入れられるようにしたことや、security consideration の節への記述の追加などが行なわれた。

考察

本節で述べた変更点によって、複雑であった RFC 3280 が更に複雑さを増したと言える。しかし、インターネットで使われる PKI には、この議論とは別の、早急に解決すべき課題が存在している。例えば、ハッシュアルゴリズムの代替可能性確保の課題などである。

今後は、複雑化の方向性よりも、現状に存在する問題解決を図る議論が必要であると考えられる。

以上

[19] Authority Information Access

8 インターネット経路制御のセキュリティ動向

木村 泰司

本節で取り上げるインターネット経路制御のセキュリティは、運用技術の側面が強く、これまでの情報セキュリティ対策という枠の中では捉えにくいものである。しかし、その影響範囲は大きく、また特定の事業者の努力によって回避可能になるものではない。本節では、本分野において近年特に顕在化しつつある課題として紹介したい。

概要

インターネット経路制御は、国際的なインターネットを維持するための一種の生命線である。現代の国際的なインターネット経路制御においては、BGP4 と呼ばれる経路制御プロトコルが用いられており、日本における ISP も例外ではない。

2008 年の前半は、インターネット経路制御に関する大きなふたつの事件が起こった。これらの事件は、インターネット経路制御における混乱がインターネットの維持に大きく影響することを身近に感じさせるものであった。また、あと 3 年程に迫っている IPv4 アドレスの在庫枯渇などの、インターネット経路制御に大きな変化をもたらす事象がおこりつつある。

本節では、インターネット経路制御の安全性に関わる事件や事象を紹介し、ネットワーク運用における安全対策について考えてみたい。

2008 年の初頭に起きたインターネット経路制御に関わる事件

ここではふたつの事件を紹介する。エジプト近海で起きた海底ケーブルの障害とパキスタンで起きた経路ハイジャックである。

■エジプト近海のケーブル障害の事件 [1][2]
期間：2008 年 1 月 30 日 ～2 月 10 日
内容：エジプト近海の 3 箇所のケーブルの障害のため広い地域で不通となる。
起こった事象：

- エジプト、スーダン、クウェートなどで少なくとも 40%の到達性が失われる。
- 14%の AS パスの消失、AS パス長が平均 9%増加（迂回）
- BGP の迂回経路が多発し、一部のピアにトラフィック集中

■YouTube に対する経路ハイジャック事件 [3]
期間：2008 年 2 月 24 日 18:47～21:03(UTC)
内容：YouTube の AS が経路広告している IP アドレスを Pakistan Telecom が経路ハイジャック。more specific route を流すことで、特定の Web サーバに対する到達性を、一定期間失わせた。
起こった事象：

- 本来と異なる Origin AS による経路ハイジャック
- /25 のような通常は経路フィルターでフィルターされてしまう経路広告で対抗するも、回避は不可能
- 経路ハイジャックの停止に伴い、復旧

日本のインターネットのトポロジーを考えると、このふたつの事件の要因は、日本においても大きな問題になりうる。日本国内で、国際線のトランジット[20]のサービスを提供している AS は多くない。このような、いわば到達性依存度が高い AS が海底ケーブルの障害の影響などを受けると、大規模なネットワーク障害に発展する可能性がある。

社内および国内の特定のサーバと通信できればよいと言われることがあるが、実際にはそれらの通信も影響を受けてしまう。例えば、DNS が利用できなければ、エンドユーザの観点では実質的に国内の通信も不可能である。分散配置されたコンテンツ（検索サービスやウィルスデータベースなど）にもアクセスできなくなる可能性が高い。このように、インターネット経路制御の問題はその影響範囲が大きい。

「共存時代」のインターネット経路制御

本節では、インターネット経路制御に影響のあるふたつの事象を紹介する。

ひとつは IPv4 アドレスの在庫枯渇である。APNIC の Geoff Huston 氏の予測[4]によると、2011 年 4 月に IPv4 アドレスの IANA プールが枯渇する[21]（**図 1**）。枯渇期になると、IPv4 と IPv6 のネットワークが混在すると共に、既存の IPv4 の接続性を保つための”キャリアグレード NAT”と呼ばれる大規模な NAPT[22]が構築されることが考えられる。これまでのインターネットにおける End-to-End の透過性に、変化がおきると考えられる。

---

[20] トランジット ― 他の AS に対して当該 AS を超えたインターネット経路制御を直接的に行えるようにすること

[21] JPNIC では 2006 年 3 月、国内の状況や影響などをまとめた「IPv4 アドレス枯渇に向けた提言」を公開した。
http://www.nic.ad.jp/ja/research/ipv4exhaustion/ipv4exh-report.pdf

[22] NAT ― Network Address and Port Translation

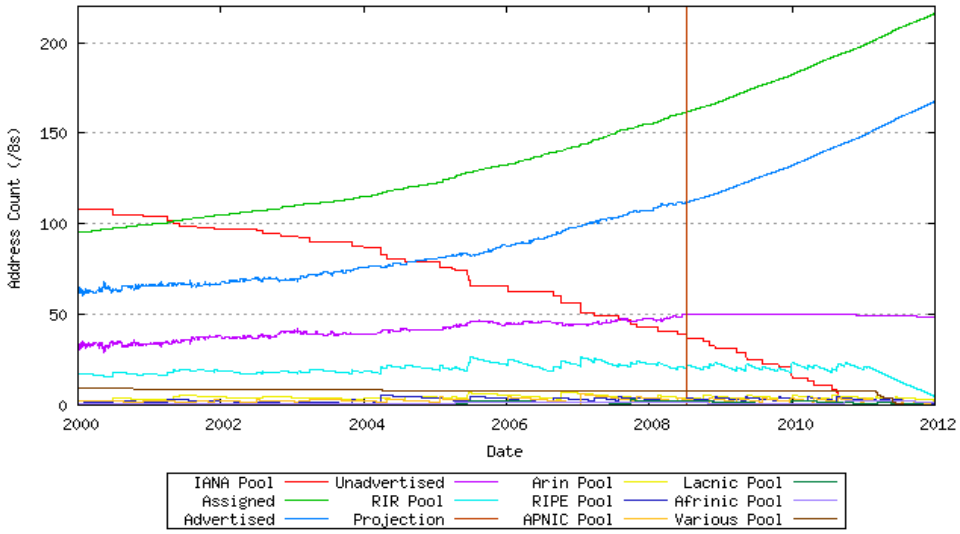


図 1: Geoff Huston 氏による予測 （赤い線が IANA プール）

もうひとつは、2 オクテット AS 番号の枯渇である。同じく Geoff Huston 氏の予測[5]によると、2011 年 4 月に IANA のプールが枯渇する。すでに RIR や NIR（JPNIC）では、4 オクテット AS 番号の配布が始まっている。

4 オクテット AS 番号の影響はインターネット経路制御に現れる。2 オクテット AS 番号にしか対応していないルータは、4 オクテット AS 番号の AS の判別に曖昧さを残してしまう可能性がある。今後本格的に広まることで、AS パス[23]の見え方にどのような影響が出るのかは未だわかっていない。これに乗じた、経路ハイジャックの危険性についても考慮すべきである。

結論

本節では、インターネット経路制御に関わるふたつの事件を紹介し、局所的な不具合が広範囲に及ぶことを述べた。また IP アドレスや AS 番号の「共存時代」を迎え、インターネットの透過性に変化が起ころうとしていることについても述べた。

インターネット経路制御のリスクは、これまで大きく取り上げられることは少なかったが、今後、その障害や不正行為のリスクを明らかにし、対策を検討していく必要がある。

以上

[23] AS パス ― ふたつの AS の間で、経路情報のメッセージデータが経由されている AS の順列

参考文献

[1] Mediterranean Fibre Cable Cut - a RIPE NCC Analysis
http://www.ripe.net/projects/reports/2008cable-cut/index.html

[2] BGP Analysis of the Middle East Fiber Cut
http://www.apnic.net/meetings/25/program/apops1/earl-fiber-cut-analysis.pdf

[3] YouTube Hijacking: A RIPE NCC RIS case study
http://www.ripe.net/news/study-youtube-hijacking.html

[4] IPv4 Address Report
http://ipv4.potaroo.net/

[5] The 16-bit AS Number Report
http://www.potaroo.net/tools/asns/

9 アイデンティティ管理技術

工藤 達雄

1 はじめに

最近のアイデンティティ管理技術の動向についてまず言えるのは、サービス間でアイデンティティを相互連携させる上でコアとなる仕様の標準化が、昨年 12 月の OpenID 2.0 の最終版の確定をもって、ほぼひと段落したということである。今後は、それらコアとなる仕様を基盤とした応用技術の進展が期待される。

本報告では、まずコアとなる仕様に関して大きなトピックである OpenID 2.0 について概観し、後に今年上半期の技術動向を紹介する。

2 OpenID 2.0

OpenID とは、ユーザが自身の ID を自由に選択し、それをさまざまな Web サービスへのログインに利用できる、非集中型のアイデンティティ・フレームワークのことである。

2007 年 12 月にリリースされた、いわゆる「OpenID 2.0」とは、「OpenIDAuthentication 2.0」[1] と「OpenID Attribute Exchange（AX）1.0」[2] の両仕様から構成される。

OpenID Authentication 2.0 仕様では新たに、ユーザ識別子としての XRI（Extensive Resource Identifier）のサポートや、XRDS（Extensive Resource Descriptor Sequence）文書によるサービス (エンドポイント等) の発見機構が盛りこまれた。これにより、利用者の利便性とサービス拡張の自由度が向上している。

加えて導入されたコンセプトである「ディレクテッド・アイデンティティ」によって、利用者は、OpenID プロバイダ (アイデンティティ情報を管理するプロバイダ) が提示した複数のデジタル・アイデンティティ候補の中から、リライング・パーティ (アイデンティティ情報を利用するプロバイダ) ごとにどれを用いるかを選択することができ、プライバシーを保つことが可能となっている。

一方の OpenID AX 1.0 は、OpenID Authentication 仕様をコアとする拡張仕様のひとつである。本仕様では、OpenID に対応した Web サイト間でやり取りするユーザの属性情報に関して、その交換方法を定めている。AX 1.0 に準拠することで、OpenID プロバ

イダが管理しているユーザの個人情報を、リライング・パーティが取得 / 更新できるようになる。

OpenID Authentication は適用分野が「Web アプリケーションへのログイン」に限定されており、比較的実装しやすいことから、Web アプリケーションを OpenID 対応にする (リライング・パーティ化する) ためのライブラリが多数開発されている。またリライング・パーティの数は急増しており、有力 OpenID プロバイダの一社である JanRain が確認したところによれば、2008 年 7 月 1 日の段階で 18,000 サイト余りとなっている [3]。

3 異種プロトコルの相互運用

コアとなる各仕様がほぼ確定したことは先に述べたが、一方それらの異なる仕様同士の相互運用は、まだ十分に検討されているとは言いがたい。この「アイデンティティ・プロトコル間の相互運用」の推進を目的に昨年設立されたのが「コンコーディア・プロジェクト」である。本プロジェクトには各種仕様の策定メンバーやその仕様を実装するベンダー、そして仕様を実システムに活用するユーザ企業が参加しており、さまざまな導入シナリオをベースに議論を続けている。

4 月に開催された RSA Conference 2008 において、コンコーディア・プロジェクトは、以下の相互運用試験を実施した [4]。

- 「インフォメーション・カード」(後述) によるユーザ認証と、SAML(Security Assertion Markup Language) / WS-Federation によるシングル・サインオンの連携: ユーザはアイデンティティ・プロバイダにおいて インフォメーション・カードによって認証される。後にその認証に成功したという結果をもって、ユーザは SAML 2.0 もしくは WS-Federation プロトコルに対応したリライング・パーティにシングル・サインオンする。

- SAML と WS-Federation との間でのシングル・サインオンの連鎖: ユーザは SAML 2.0 アイデンティティ・プロバイダにログインし、後に WS-Federation リライング・パーティへシングル・サインオンする。また逆にユーザは WS-Federation アイデンティティ・プロバイダにログインし、後に SAML 2.0 サービス・プロバイダ (アイデンティティ情報を利用するプロバイダ) にシングル・サインオンする。

4 アイデンティティ情報の交換におけるガバナンスとアシュアランス

サービス間でのアイデンティティ属性交換におけるガバナンスとアシュアランスを確立するための標準化を、現在複数の組織が進めている。

リバティ・アライアンスは 6 月に、「Identity Governance Framework (IGF)」仕様のドラフトを公開した [5]。IGF は、サービス間で交換される属性の利用や保管を制御するための方法の確立を目的としている。今回公開された仕様は以下のふたつである。

- Client Attribute Requirements Markup Language (CARML) は、アイデンティティ情報の利用者側が、提供側に対し、必要な属性や利用目的などの情報を伝達するための形式を定義する。CARML 自体は情報交換のプロトコル (例えば LDAP, WS-Trust, ID-WSF など) には依存しない。またアクセス・ポリシーや認証方法などの定義はせずに、外部の仕様を取り込むことにしている。

- Privacy Constraints は、収集目的、伝搬、格納、表示に関する基礎的なプライバシー条件を定義するための仕様である。Privacy Constraints によって定義される要素は、WS-Policy と併せて、CARML 内でのポリシー定義に用いられる。

また認証ポリシー／アシュアランスに関しては、OpenID とリバティ・アライアンスの両組織において、仕様策定の動きがある。

- OpenID は 6 月に Provider Authentication Policy Extension (PAPE) ワーキング・グループ (WG) を結成した[6]。本 WG では、OpenID プロバイダの認証ポリシーをリライング・パーティと交換し、ユーザの本人確認における認証レベルを担保するための仕様である PAPE の策定を目指す。

- リバティ・アライアンスは 6 月に Identity Assurance Framework(IAF)仕様のリリースを一般公開した [7]。IAF はアイデンティティ情報が確かなものかどうかの保証レベル (assurance level) の定義と、アイデンティティ・プロバイダが各レベルを達成するための手順 (運用プロセス)を規定し、そして組織に対する各アイデンティティ保証レベルの認定を行なっていく予定となっている。

5 リピュテーション（評判）

サービス間でのアイデンティティ情報の交換には、単に技術的な接続性だけではなく、交換する相手の信頼可能性も考慮すべき要素のひとつである。例えばアイデンティティ情報を利用するプロバイダでは、アイデンティティ情報を管理するプロバイダが表明したユーザの認証結果や属性情報が、どの程度信頼可能かを認識する必要がある。また逆にアイデンティティ情報を管理するプロバイダにおいては、アイデンティティ情報を利用するプロバイダの信頼可能性に応じて、提供する情報の内容を制限したり、あるいは提供しない

と判断することになる。

先行して策定された SAML や Liberty Alliance ID-FF (Identity Federation Framework) に代表されるアイデンティティ連携仕様では、相手の信頼可能性を認識するために、事前にサービス同士が信頼関係を確立する。これにより、価値の高いアイデンティティ情報の交換 (企業間での情報共有や、医療情報の提供など) へ適用しやすい。しかしこの前提が、一般向けの Web サービスのような、サービス同士が連携相手を事前に特定しづらい分野への、導入の障害となっている。

一方 OpenID のように事前の信頼関係を必ずしも必要としない仕様は、これとは逆に「ブログへのコメントの認証」のような、比較的重要ではないアイデンティティ情報を交換する用途に向いている。だが相手の信頼可能性を事前に測ることができないため、例えばクレジットカード番号を提供する場合、あるいは特別なサービスへのログイン処理を他のサイトに委ねる場合などには、SAML / ID-FF と同様、連携する相手を制限せざるを得ない。

このように、アイデンティティ情報の交換における信頼可能性の把握には、信頼できる相手を静的に限定することが一般的に行なわれている。これに対し先般、リピュテーション (評判) 情報を使って動的に相手の信頼可能性を推測し、その結果をアイデンティティ情報を交換するかしないか、あるいはどの程度の情報を交換するか、の決定に役立てるための試みが始まっている。

OASIS は 4 月に、Open Reputation Management Systems (ORMS) 技術委員会 [8] を設置した。ORMS は評判データの表現形式の共通化と、評判スコアの標準的な定義を目的としており、評判スコアの計算方法やアルゴリズムの定義は検討対象に含まない。

ORMS は今後、ユースケース文書や要件等をまとめ、2009 年 3 月に評判データおよび評判スコアのための XML スキーマなどを策定する予定である。

6 まとめ

2008 年上半期 (正確には昨年末から)、アイデンティティ管理技術においては、OpenID 2.0 の確定や、異種アイデンティティ連携仕様間の相互運用に関して一定の成果があった。さらに、より重要なアイデンティティ情報をセキュアに交換する上での基礎となるガバナンス / アシュアランス、そしてリピュテーションについて、標準化の動きが活発化している。

また本稿では紙数の関係から割愛したが、アイデンティティ情報を利用目的に応じた「カード」に見立てて、ユーザがサービス利用時にどのようなアイデンティティ情報を提示するか (典型的には「どの ID でログインするか」) を、カードを選択する感覚で制御できるようにする「インフォメーション・カード」についても、2008 年 6 月に Information Card Foundation が設立されたこともあり、今後実装間の相互運用や実サー

ビスでの試行が進むものと思われる。

2008 年下半期も引き続きこれらの動向に注目しつつ、エンタープライズ分野でのアイデンティティ管理技術の動きにも着目していきたい。

以上

参考文献

[1] Final: OpenID Authentication 2.0 - Final
http://openid.net/specs/openid-authentication-2_0.html

[2] Final: OpenID Attribute Exchange 1.0 - Final
http://openid.net/specs/openid-attribute-exchange-1_0.html

[3] JanRain ≫ Blog Archive ≫ Relying party Stats as of July 1st 2008
http://janrain.com/blog/2008/07/08/relying-party-stats-as-of-july-1st-2008/

[4] RSA IOP Scenarios - Project Concordia
http://projectconcordia.org/index.php/RSA_IOP_Scenarios

[5] Liberty Alliance Announces First Release of Identity Governance Framework Components
http://www.projectliberty.org/liberty/news_events/press_releases/liberty_alliance_announces_first_release_of_identity_governance_framework_components

[6] Notice of vote on the proposal to create the PAPE working group
http://openid.net/pipermail/specs/2008-June/002334.html

[7] Liberty Alliance Releases Identity Assurance Framework
http://www.projectliberty.org/liberty/news_events/press_releases/liberty_alliance_releases_identity_assurance_framework

[8] OASIS Open Reputation Management Systems (ORMS) TC
http://www.oasis-open.org/committees/orms/

10 C コンパイラの最適化の問題

田中 哲

最近のセキュリティ問題として、整数オーバーフローの問題が注目されている。そのため、プログラミング言語に関連するセキュリティの話題として「JVNVU#162289 ある種の範囲チェックを破棄する C コンパイラの最適化の問題」[1] についてその背景と関連する話題をあわせて解説する。

1 ポインタの足し算

JVNVU#162289 [1] の問題は、以下のような話である。

```
char *buf;
int len;
```

このような型宣言が行われているとき、下記の「長さチェック」が働かない可能性がある、というものである。

```
len = 1<<30;
[...]
if(buf+len < buf)   /* 長さチェック */
   [...オーバーフローに対処するコード...]
```

これだけでは、どのような状況であればこのようなコードが書かれるのか明らかでないが、ARR38-C [3] には Plan 9 の sprint() 関数におけるテストを変形したものとして下記のコードが載っている。

```
char *buf;
size_t len = 1 << 30;

/* Check for overflow */
if (buf + len < buf) {
```

```
    len = -(uintptr)buf-1;
}
```

このコードの意図は、「buf + len < buf の検査により、buf + len が大きすぎてアドレス空間の終端を越えてしまうならば、アドレス空間の終端を越えないように len を小さくする」というものである。

JVNVU#162289 は、直接的には、この buf + len < buf の検査が C コンパイラの最適化によって常に偽になると判断される可能性があることを指摘している。その理由は、この buf + len の足し算がオーバーフローしたとき、オーバーフローしたキャリービットを除いた値が結果として得られるという保証がないためである。

この「保証がない」というのは C の規格がその保証を要求していないためであり、C コンパイラは、そのような場合にどのような振舞をするかを自由に選べる。最適化は一般に高速なオブジェクトコードを目指すわけであるから、if 文自体を削除しても C の規格には反しないとなれば、そうすることは高速かつ小さなオブジェクトコードに結びつくため、妥当な選択である。

そのような C コンパイラの選択を封じるための方法として、JVNVU#162289 は、下記のように uintptr_t にキャストしてから比較するコードを紹介している[24]。

```
#include <stdint.h>
[...]
if((uintptr_t)buf+len < (uintptr_t)buf)
  [...]
```

このように変更すれば、足し算はポインタの足し算ではなく、uintptr_t の足し算になる。uintptr_t は符号無し整数型のひとつであるから、足し算においてオーバーフローが起きた場合はキャリービットを除いた値が結果となることが保証されている。このため、C コンパイラはこの if 文を除去することができなくなる。

しかし、この修正はアドレス空間が連続的であり、ポインタを uintptr_t にキャストしたときにアドレスがそのまま整数になることを前提としている。残念ながら、この前提は

[24] 注： JVNVU#162289 では <stdint.h> のアングルブラケットがエスケープされていないため、firefox ではその部分が表示されなかった。また条件部の小なり記号もエスケープされていない。きちんとエスケープすべきである。

C の規格では保証されない。uintptr_t について保証されているのは、void へのポインタを uintptr_t に変換し、それをさらに void へのポインタに変換したとき、元のポインタと等しくなるということだけである。例えば、64bit ワードマシンにおいてバイトを指すポインタを実現する手段として、ポインタ内の上位ビットにワード内のオフセットを格納する環境もある [9], [10] が、そのような環境でポインタを uintptr_t に変換した結果がその 64bit ワードそのものになるとすると、uintptr_t の足し算でオーバーフローが起きることはアドレス空間の終端を越えることには対応しない。

この例の本質的な問題は、アドレス空間の終端という概念を C 言語で扱おうとした点である。そもそも C 言語では、アドレス空間がひとつの連続的な空間であることは保証しておらず、アドレス空間の終端という概念がない。実際、ハーバードアーキテクチャなど、アドレス空間の終端という概念が自明でない環境は存在するため、その概念がない事自体は不適切なことではない。しかしそのことは、アドレス空間の終端を扱おうとすれば、必然的に C 言語で保証されていない仮定を導入せざるを得ないことに結び付く。

したがって、正しい考え方は、本当にアドレス空間の終端について判断しなければならないのかどうかを考え直すことである。カーネルのようにアドレス空間自体も処理の対象となる場合にはそのような判断が必要になることもありうるが、通常のアプリケーションではアドレス空間終端でなく、バッファの終端を越えるかどうかの判断で十分なはずである。

ここで、Plan 9 の sprint を調べると、現在では以下のようなコードになっている。[5]

```
int
sprint(char *buf, char *fmt, ...)
{
    ...
    n = vsnprint(buf, 65536, fmt, args);      /* big number, but sprint is deprecated
anyway */
    ...
}
```

この関数は C 言語の sprintf() と同様な機能を持つ。このコードでは、JVNVU#162289 で指摘された問題は見つからないが、65536 という定数が使用されており、sprint は deprecated というコメントがある。

sprint() は sprintf() と同じく、バッファの大きさを引数で受け取らないという問題がある。このためバッファ終端は sprint() 内部では不明である。推測としては、バッファ終

端が不明で判断できなかったために、アドレス空間終端についての判断を行ったという可能性が考えられる。

しかし、正しいやりかたは、sprintf() に対して snprintf() があるように、バッファの大きさを引数で渡すことである。実際、Plan 9 にはバッファの大きさの引数を加えた snprint() [6]があり、sprint() は deprecated であるため、この正しい方向に向かっているといえる。

なお、ARR38-C にはバッファの大きさを検査する場合のコードについての注意も述べられている。下記のコードには問題がある。

```
int process_array(char *buf, size_t n) {
  return buf + n < buf + 100;
}
```

このコードは buf が 100 バイト固定長のバッファを指すとして、buf + n がバッファ内部を指すかどうかを判断する。しかし、n が 100 よりも大きければ、buf + n はバッファの外を指すことになる。C の規格では、そのようなバッファの外を指しているポインタに関してはポインタの比較は定義されない。例えば、buf + n がオーバーフローしたときにキャリービットを無視した結果になるという素朴な環境を考えると、アドレス空間内における buf の位置と n の大きさによって足し算がオーバーフローしたときには、buf + n は buf + 100 よりも小さいことになり、process_array の意図する判断に失敗してしまう。したがって、これは下記のように実装すべきである。

```
int process_array(char *buf, size_t n) {
  return n < 100;
}
```

なお、コンパイラによっては、buf + n がオーバーフローした場合の挙動は未定義であることを利用し、buf + n < buf + 100 を n < 100 と変形して 2 回の足し算を削減する可能性がある。これが可能なのは buf + n < buf + 100 と n < 100 は定義されている範囲内においては同じ結果が得られるためである。その場合は、結果的に正しい実装と同じ結果になるが、これはコンパイラの最適化に依存する。最適化によって正しい結果が得られたり、得られなかったりする実装は、不適切である。

一般に、C 言語におけるポインタで保証されるのは、確保されたメモリ領域の内部で済む演算だけである。（正確には、配列の最後の要素を越えた直後の場所も扱える。）その範囲を外れたポインタは、それが指す対象が存在しないため、基本的にポインタとして不適切であり、本当に必要なのかどうか熟慮すべきである。

2 符号つき整数の足し算

ポインタの足し算と同様に、符号つき整数の足し算もオーバーフローの場合の結果は規定されていない。（なお、符号無し整数の場合は規定されているため、前述の uintptr_t の用法は問題ない。）
これは、歴史的には、負数の表現に 1 の補数を使用する実装が存在したことに由来する。C の規格には、INT_MIN の絶対値が (32768 ではなく) 32767 以上となる負の値でなければならないという規定など、1 の補数に対する配慮が他にも見られる。

現在では、負数の表現に 1 の補数を使用する実装はまずない。しかし、このオーバーフローの結果が規定されていないことはコンパイラの最適化に利用される。JVNVU#162289 から参照されている MSC15-A には符号つき整数と最適化に関する問題の例が述べられている。下記のコードには問題がある。

```
#include <assert.h>

int foo(int a) {
  assert(a + 100 > a);
  printf("%d %d¥n", a + 100, a);
  return a;
}

int main(void) {
  foo(100);
  foo(INT_MAX);
}
```

このコードには a + 100 > a という条件判断があるが、ここで、a は (signed) int なので、その演算におけるオーバーフローの計算結果は規定されていない。規定されている範囲内、すなわちオーバーフローが起きない範囲内では、a + 100 > a は常に真である。そうすると最適化により (assert は引数が真ならなにもしないので) assert 全体を削除することができる。

しかし、コードの意図は a + 100 がオーバーフローしないことの検査であり、assert が削除されるのは意図とは異なる。意図どおりの動作を保証するには、下記のように書かなければならない。

```
#include <assert.h>

int foo(int a) {
    assert(a < (INT_MAX - 100));
    printf("%d %d¥n", a + 100, a);
    return a;
}

int main(void) {
    foo(100);
    foo(INT_MAX);
}
```

このコードでは、100 を加えてもオーバーフローしないことを a < (INT_MAX - 100) として判断しており、この判断ではオーバーフローは起きない。

また、ループ回数を求められると最適化に都合がいい場合がある。[7] 下記のコードを題材に考える。

```
int i, n;

for (i = 0; i <= n; i++) {
    ...
}
```

ここで n が非負であれば、i が 0 からちょうど n まで変化しながら回るから、ループ回数は n+1 回に見える。しかし、残念なことに n が INT_MAX である場合にはそうではなく、このループは終わらない。そのため、最適化においてループ回数は n+1 回という仮定は使えない。

しかし、符号つき整数の足し算がオーバーフローする場合の挙動が規定されていないこ

とを考えると、i++ がオーバーフローを起こす場合の挙動は最適化において変化しても許される。n が INT_MAX の場合にループが終わらないというのは i++ がオーバーフローしてしまうことが原因である。したがって、その場合は除去して考えることができ、そうすればループ回数は n+1 回と推論できる。

このように、符号つき整数のオーバーフローの挙動が規定されていないことは、最適化に役に立つ性質である。つまり、コンパイラの開発者には、この性質を利用する動機がある。

整数のオーバーフローの検査は、演算によってはそれほど自明でない。[8] には四則演算のオーバーフローの検査を行う方法が記載されている。

まとめ

プログラムを作成する際には、コンパイラの最適化の進歩によってプログラムの挙動が変化しないよう、可能なかぎり C 言語の規格の範囲内で記述することが肝要である。少なくともポインタや符号つき整数の演算については実際にコンパイラが規格の範囲外の挙動を最適化のために任意に変更することがあり、注意が必要である。

以上

参考文献

[1] JVNVU#162289: ある種の範囲チェックを破棄する C コンパイラの最適化の問題,
http://jvn.jp/cert/JVNVU162289/

[2] Vulnerability Note VU#162289: C compilers may silently discard some wraparound checks
http://www.kb.cert.org/vuls/id/162289

[3] ARR38-C. Do not add or subtract an integer to a pointer if the resulting value does not refer to a valid array element
https://www.securecoding.cert.org/confluence/display/seccode/ARR38-C.+Do+not+add+or+subtract+an+integer+to+a+pointer+if+the+resulting+value+does+not+refer+to+a+valid+array+element;jsessionid=0140EC1A0DA3804694FBFC9C151B9404

[4] MSC15-A. Do not depend on undefined behavior
https://www.securecoding.cert.org/confluence/display/seccode/MSC15-A.+Do+not+de

pend+on+undefined+behavior;jsessionid=37984BD27B6857683F5B5E23993B14EB

[5] Plan 9: sprint.c
http://plan9.bell-labs.com/sources/plan9/sys/src/libc/fmt/sprint.c
参照日時: 2008-06-15

[6] Plan 9: snprint.c
http://plan9.bell-labs.com/sources/plan9/sys/src/libc/fmt/snprint.c
参照日時: 2008-06-15

[7] GCC optimizes integer overflow: bug or feature?
http://gcc.gnu.org/ml/gcc/2006-12/msg00459.html

[8] **ヘンリー・S・ウォーレン、ジュニア,『ハッカーのたのしみ』, エスアイビー・アクセス発行**

[9] Cray C/C++ Reference Manual: 9.1.2.2. Types
http://docs.cray.com/books/004-2179-001/html-004-2179-001/rvc5mrwh.html#ZFIXEDDICM1VZB

[10] C FAQ Q5.17: Seriously, have any actual machines really used nonzero null pointers, or different representations for pointers to different types?
http://c-faq.com/null/machexamp.html

11 Debian の openssl

田中 哲

セキュリティ問題を予防するためにツールを使ってコードを検査することがあるが、それに関連して 2008 年 5 月 13 日に公開された「DSA-1571-1 openssl -- 予測可能な乱数の生成」[1] という Debian セキュリティ勧告がある。

この問題は、Debian において変更された openssl ライブラリでは乱数が予測可能となり、その結果として、生成される暗号鍵が予測可能となってしまった、というものである。これにより、多くの鍵を捨てて新しく作りなおすという多大な手間がかかる対処が必要になった。ここでは、なぜそのような変更が行われたのかという原因について述べる。

この変更が行われたのは 2006 年のことであり、openssl に対する「Valgrind[25]が発生させる警告を取り除いてほしい」という要望がきっかけであった。[2]

Valgrind には、さまざまな機能があるが、デフォルトの機能で、もっともよく使用されるものはメモリの使いかたを検査する memcheck というものである。memcheck はデフォルトなので、単に Valgrind といった場合は memcheck を指していることが多い。memcheck 以外にはキャッシュシミュレータの cachegrind やヒーププロファイラの massif などがあり、オプションで指定できる。

memcheck は、例えば初期化されていないメモリの値に基づいて条件判断をすることに警告を発生させる。memcheck はこれを実現するためにプログラムが扱うデータが初期化されているかどうかのフラグを管理している。そして、初期化されていない値、もしくはそのような値から導出された値で条件判断が行われると、警告を発生する。なお、初期化されていない値をメモリから読んだり書いたりするだけでは警告されない。これは、フィールド間に空きがある構造体を memcpy するなど、害が無いケースが多いためである。

Debian に対する希望は、openssl を利用したアプリケーションで発生する警告を除去するため、openssl に変更を加えるというものである。一般に、初期化されていない値を使用することは不適切なので、この希望自体は妥当なものである。実際、openssl を利用するアプリケーションの検査を Valgrind で行った場合に、openssl に起因する警告が大量に出ればアプリケーション自体の検査が難しくなる。そのため、openssl 自体に問題があれば、その問題を解決するのは方向性として正しい。

この希望を受けて、Debian が行った変更は、openssl 内の crypto/rand/md_rand.c というファイルで、下記の「MD_Update(&m,buf,j);」を含む 2 行を除去することであった。

```
ssleay_rand_add:
```

[25] http://valgrind.org/

```
                    ....
                    MD_Update(&m,buf,j);
                    ....

ssleay_rand_bytes:
                    ...
 #ifndef PURIFY
                    MD_Update(&m,buf,j); /* purify complains */
 #endif
                    ...
```

最初の行は ssleay_rand_add 関数にあり、もうひとつの行は ssleay_rand_bytes 関数にある。セキュリティ問題となったのは前者を除去したことであった。

ssleay_rand_add は乱数の種となるデータを外部から追加する関数であり、ssleay_rand_bytes はその種から乱数を発生させて呼出元に返す関数である。どちらも与えられたバッファをハッシュして乱数の種に追加する。

前者は乱数の種に追加することが目的の関数であるから、与えられたバッファのデータを使うのは必須である。これを除去したために、乱数の種にデータが追加されなくなり、/dev/urandom などから読んだデータが乱数の種に追加されることなく捨てられることとなった。その結果として、乱数が予測可能というセキュリティ問題につながった。

これに対し、後者の除去はセキュリティ問題ではない。後者の ssleay_rand_bytes では、呼出元が与えたバッファに発生した乱数を埋めて返すのであるが、同時に、埋める前の内容を乱数の種に追加する。これは openssl のドキュメント[3] に明記された仕様である。

ここで呼出元の興味が発生される乱数にしかなければ、ここには初期化されていないバッファが与えられることがあり、その場合、Valgrind は乱数の種が初期化されていない値に依存していると解釈することになる。そして、アプリケーションが生成された乱数を用いて条件判断などを行えば、memcheck で警告が発生する。しかし、この件に限定していえば、初期化されていない値に依存していても、結果として生成されるものは乱数であるから問題は起きない。

この警告の問題については openssl の開発者も認知しており、それが PURIFY マクロによる条件コンパイルに表れている。Purify は Valgrind の memcheck に類似した機能を持つ検査ツールであり、警告を避けるために ssleay_rand_bytes 内の「MD_Update(&m,buf,j);」 を除去する機構が用意されている。

このような条件コンパイルが用意されていることからも分かるように、後者は除去してもセキュリティ上問題とはならない。後者の「MD_Update(&m,buf,j);」は、乱数の種に

データをいくらか追加するが、それはおまけのようなものであり必須ではない。このため、セキュリティ勧告後のパッケージでも除去されたままである。

なお、Debian の開発者はこれらの除去を行うにあたり、事前に openssl-dev メーリングリストで相談を行った。[4] その議論では、除去してもいいのではないかという反応もあれば、PURIFY マクロの定義で Valgrind の警告を抑制できているといった反応もあった。

まとめ

ソフトウェアの変更は、対象のコードを正しく理解して行う必要がある。あいまいな理解のもとに、検査ツールが警告を出したという理由だけで変更を行うのはかならずしも望ましい結果をもたらさない。また、-DPURIFY を用いる等によって、コードの変更無しに要求される効果が得られるのであれば変更すべきでない。変更を行わなければ、少なくとも新しい問題を発生させることはない。

以上

参考文献

[1] Debian セキュリティ勧告 DSA-1571-1 openssl -- 予測可能な乱数の生成
http://www.debian.org/security/2008/dsa-1571

[2] Debian BTS #363516 - valgrind-clean the RNG - Debian Bug report logs
http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=363516

[3] OpenSSL: Documents, RAND_bytes(3)
http://www.openssl.org/docs/crypto/RAND_bytes.html

[4] 'Random number generator, uninitialised data and valgrind.' thread - MARC
http://marc.info/?t=114651088900003&r=1&w=2