TACA WG 2003 年度活動報告：制御ネットワークのIP化についての検討

岡部 宣夫[1] nov@tahi.org
井上 淳[2] inoue@isl.rdc.toshiba.co.jp
石山 政浩[3] masahiro@isl.rdc.toshiba.co.jp
坂根 昌一[4] sakane@kame.net
鎌田 健一[5] kamada@nanohz.org

2004.7.15

[1] 横河電機株式会社
[2] 株式会社東芝
[3] 株式会社東芝
[4] 横河電機株式会社
[5] 横河電機株式会社

第I部

taca

1.1 制御ネットワークの IP 化

1.1.1 制御ネットワークの特徴と課題

制御ネットワークは，IP (Internet Protocol) とは異なる用途と技術に基づいたネットワークである．IP はコンピュータに特化したネットワークとして普及したが，制御ネットワークはより身近に存在し，日々の産業と生活を支えている．例えば，ビル内の空調制御，石油や薬品工場における化学反応の制御，自動車内部の機器の制御などは，すべて制御ネットワーク技術である．制御ネットワークの主な特徴は以下である．

- 制御ネットワークは，IP とは異なる独自のネットワーク層を持っている．多くの技術に，OSI のモデルが採用されている．
- IP と比較すると，一般的にリアルタイム性や低コストに特化している．このため，ネットワークの機能としては IP よりも単純で軽く，8/16 ビット CPU の様な処理能力の限定されたシステムが用いられている．
- 適応する分野により，様々な規格が存在する．例えば，BA (Building Automation) には，LonWorks [13] [1] と BACnet [9] [2] という二つ標準が存在する．また，大規模なプラントの制御には，Fieldbus [10] が有名である．自動車内部の制御には，CAN (Control Area Network) [1] が用いられている．これ以外にも，様々な規格やベンダー固有のプロトコルが多々存在する．

インターネットの普及，省エネルギーやテロリズム対策などの社会的要請により，現在の制御ネットワークには以下の課題が存在する．

- インターネットが社会的インフラになったことで，制御ネットワークとインターネットとの接続性が求められている．しかし，制御ネットワークは独自のネットワーク層をもっているために，IP を単なるパイプとして利用する場合が多く，既に IP が持っている機能を利用できず，同様のものを再発明している．既に IP の持っている機能を制御ネットワークで利用できれば，制御ネットワークの利便性を向上できる．
- 省エネなどの社会的要求などから，従来より緻密な観測と制御が求められ，制御デバイス数や観測点が増大している．例えば，六本木ヒルズに置ける BA 制御デバイスの制御点数は 17 万点に達している [7]．しかし，大量のデバイスのセットアップの手間は削減されていないので，現場のエンジニアリングコストが問題化している．大量のデバイスが自律的にセットアップすることを，ネットワーク側で支援する仕組みが必要である．
- 多くの制御ネットワーク技術では，セキュリティ機能が備わっていない．今までは，制御ネットワークは外に対して閉ざされた系で実現されていたので，セキュリティは問題視されなかった．しかし，無線技術の普及や，IP ネットワークとの融合により，制御ネットワーク通信は外に露出するので，セキュリティは重要な要求になりつつある．

[1] LonWorks is registered trademarks of echlon corporation.
[2] BACnet is registered trademarks of ASHRAE.

1.1.2 IP 技術を適用する狙い

我々は，制御ネットワークに IP 技術を適用することで，以下の実現を目標としている．

- IP で使われている技術を導入することで，制御ネットワークをよりスマートにする．例えば，経路制御，優先制御，高可用性，スイッチ技術，などの技術の応用が考えられる．
- インターネットに接続することで，用途に応じて，広域に分散した制御ネットワーク環境を実現する．
- 大量のデバイスが自律的にセットアップすることを支援する枠組を実現する．
- 制御ネットワークにおける，通信のセキュリティを実現する．このセキュリティは，様々なデバイス性能に適用するためのスケーラビリティを有し，様々な運用環境に適合する柔軟性が求められる．

本章では今までの検討結果として，制御ネットワークに適したセキュリティ技術の考察を 1.2 章で，制御ネットワークの具体例として BA の考察 [24] を 1.3 章で，提案する制御ネットワークのアーキテクチャを 1.4 章で，提案するアーキテクチャで用いる鍵管理の運用上の考慮点 [25] [3] を 1.5 章で述べる．

1.2 セキュリティの検討

1.2.1 暗号の処理時間

制御ネットワークを構成するローエンドデバイスは 8/16 ビット CPU を使っているので，計算能力の限定されたれているため，計算負荷の軽いセキュリティシステムを導入する必要がある．本節では，8/16 ビット CPU 上で公開鍵暗号方式としての RSA 演算と Diffie-Hellman 演算 (以降 DH 演算と呼ぶ)，対称鍵暗号方式，ハッシュの実行時間を実測し，ローエンドデバイスとの相性を検討する．

まず，16 ビット CPU H8/3048 [12] 上で測定した，RSA 暗号の暗号化と復号化の実行時間と演算対象となるデータのビット長の関係を図 1.1 と図 1.2 に示す．この RSA コードは，GnuPG 1.0.7 [3] で使われているコードを移植したものである．PC 上の RSA コードも，GnuPG 1.0.7 を用いている．PKCS#1 [14] は，RSA 演算を行うデータにパディングをすることを規定しているが，本実験は，純粋な RSA 演算の負荷を調べるのが目的なので，データにパディングは行っていない．従って，暗号化の処理時間はデータ長に比例するが，復号化は鍵長毎に一定となる．

両図から，制御ネットワークのローエンドデバイスに用いられる CPU では，RSA の復号化に数分の処理を要することを示し，制御ネットワークに適したスケーラブルな暗号システムという目標に適していないことがわかる．

[3] [25] では，KINK の問題に対して「脆弱性」(vulnerability) という表現を使っているが意味が強すぎる．むしろ「オペレーション上の注意点」(operational consideration) という表現が相応しい．

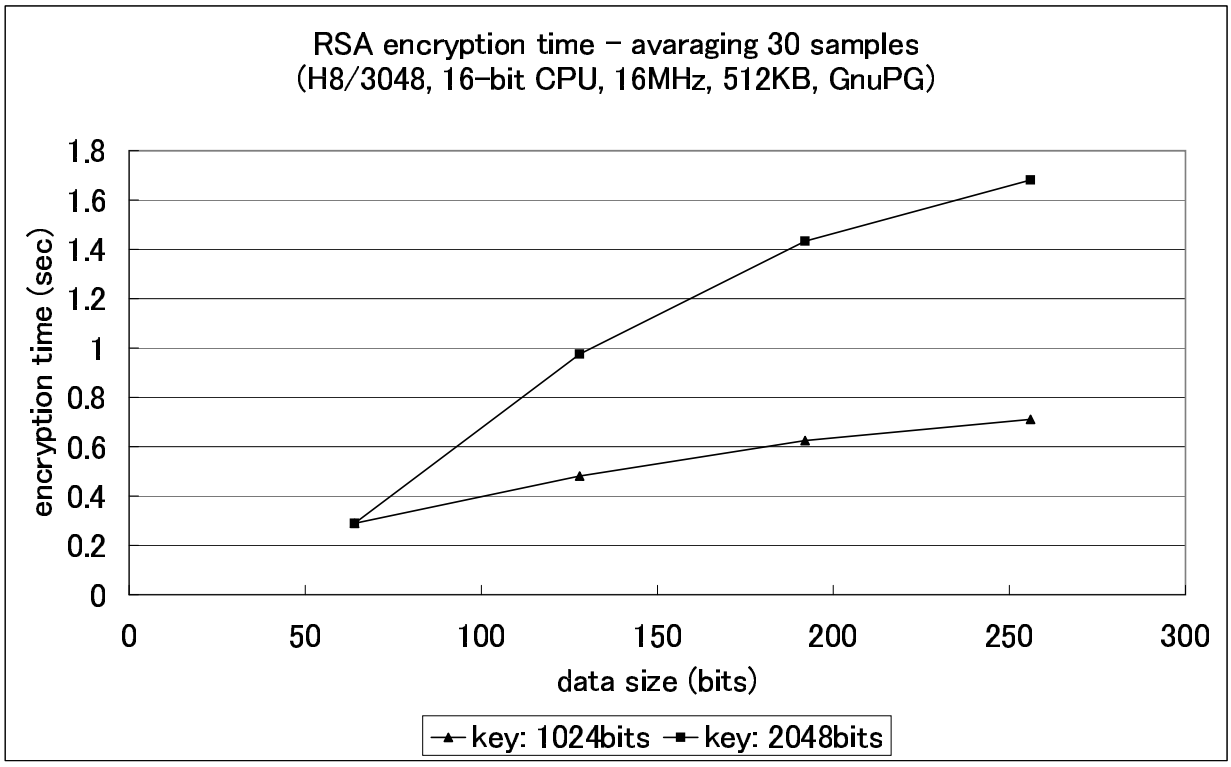

図 1.1: H8/3048 の RSA 暗号化の実行時間

次に，H8/3048 上で測定した，DH 演算の実行時間とその演算の元となる素数のビット長の関係を図 1.3 に示す．DH 演算の種類に関しては，IKE(RFC2409) [11] の利用を想定し，RFC2409 で実装必須と規定されている MODP について評価するが，ECP と EC2N は RFC2409 ではオプションと規定されているので評価の対象外とした．また，素数のビット長は，RFC2409 で規定された 768 ビットと 1024 ビットを中心に評価を行った．この DH 演算コードは，GnuPG 1.0.7 で使われている整数演算ライブラリ GnuMP [2] を利用し，独自に作成したものである．図 1.3 の演算時間は，DH の秘密値／公開値ペアの生成と共有値の生成に要する時間の合計であり，これらの値を生成するために必要な素数と原始根の生成時間は除いている．

本図から，制御ネットワークのローエンドデバイスに用いられる CPU では，DH 演算に数分の処理を要することを示し，制御ネットワークに適したスケーラブルな暗号システムという目標に適していないことがわかる．

また，8 ビット CPU DS80C390 [19] 上でアセンブラにより実装した，対称鍵暗号方式（AES-128 CBC と 3DES CBC）とハッシュ（MD5 と SHA1）の実行時間と演算の対象となるデータのバイト長の関係を図 1.4 と図 1.5 にまとめる．

両図から，制御ネットワークのローエンドデバイスに用いられる CPU では，1000 バイト程度のハッシュや暗号処理に数秒の処理を要することを示している．これは，IP パケット全体の暗号化と認証データの作成，または復号化と認証データの確認に数秒かかる事を意味し，制御ネットワークに適したスケーラブルな暗号システムという目標に対して，RSA 暗号や DH 演算より十分に処理が軽いことで知られている対称鍵暗号方式やハッシュでも，何らかの工夫が必要であることを示している．

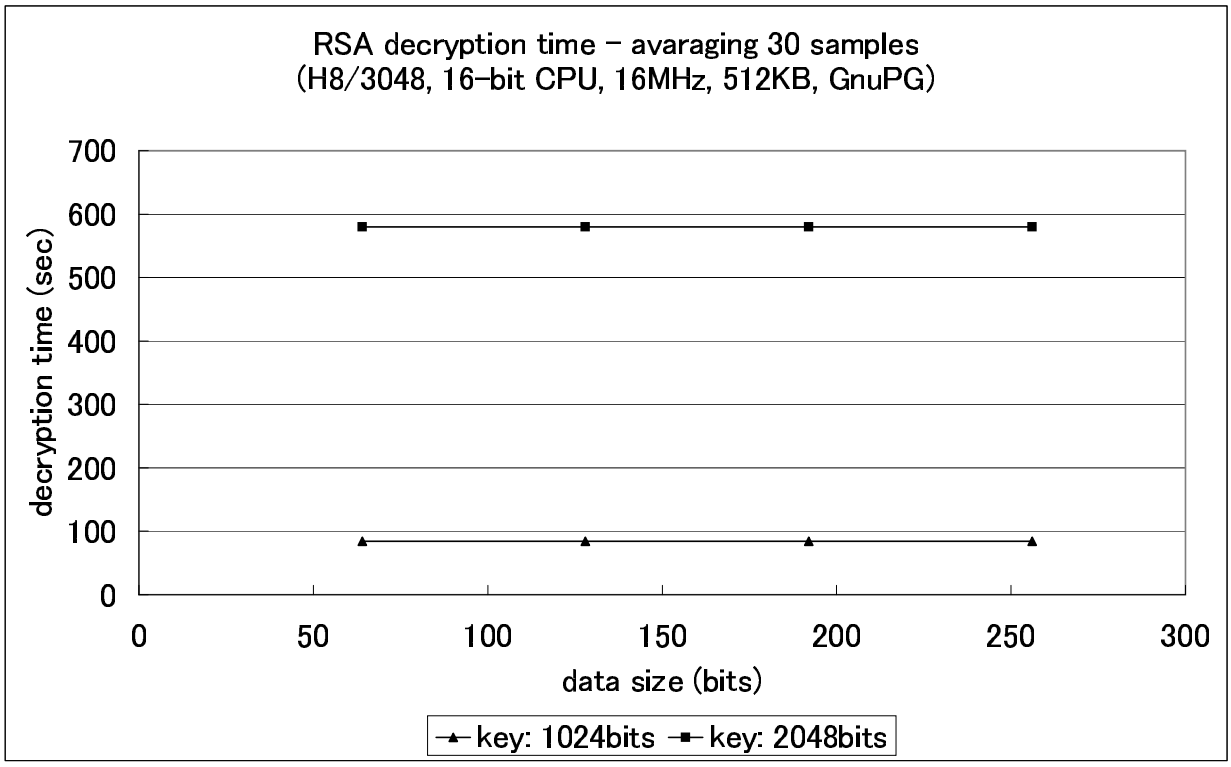

図 1.2: H8/3048 の RSA 復号化の実行時間

1.2.2 ハードウェア化容易性

1.2.1 節は，処理性能の制約されたシステムにとって，どのような暗号方式を選択しても，実用的な性能を得る事が困難であることを示している．これを解決するためには，暗号処理をハードウェア化することが考えられる．本節では，暗号処理のハードウェア化の容易性という観点で，RSA 暗号，DH 演算，対称鍵暗号方式，ハッシュを評価する．

対称鍵暗号方式とハッシュのハードウェア化には，ゲート数の少なさを優先した実装の場合，一アルゴリズム当たり 2000 〜 3000 ゲートが必要である．例えば，Dallas Semicondactor 社の CPU 設計者によれば，DS80C390 の場合にはチップ上に数千ゲートの空きがあるが，これはめずらしいことでは無く，CPU チップのダイサイズを変えること無く，対称鍵暗号方式とハッシュを実装することが可能であり，ハードウェア化は容易だと言える．

一方，RSA 暗号や DH 演算を高速化するためには，MODEXP (MODulo EXPonention) 演算 $g^x mod\ p$ のハードウェア化が必要である．ここで，g は原始根，p は素数，x は乱数を意味する．1024 ビット程度の素数の MODEXP 演算の場合を考えると，上述した対称鍵暗号方式やハッシュに必要なゲート数より数百倍多いゲート数が必要であり，ハードウェア化が容易であるとは言い難い．

以上の考察から，我々は対称鍵暗号方式とハッシュだけから成る暗号システムなら，暗号処理のハードウェア化を利用して，制御ネットワークに適したスケーラブルな暗号システムになる可能性が高いと判断する．

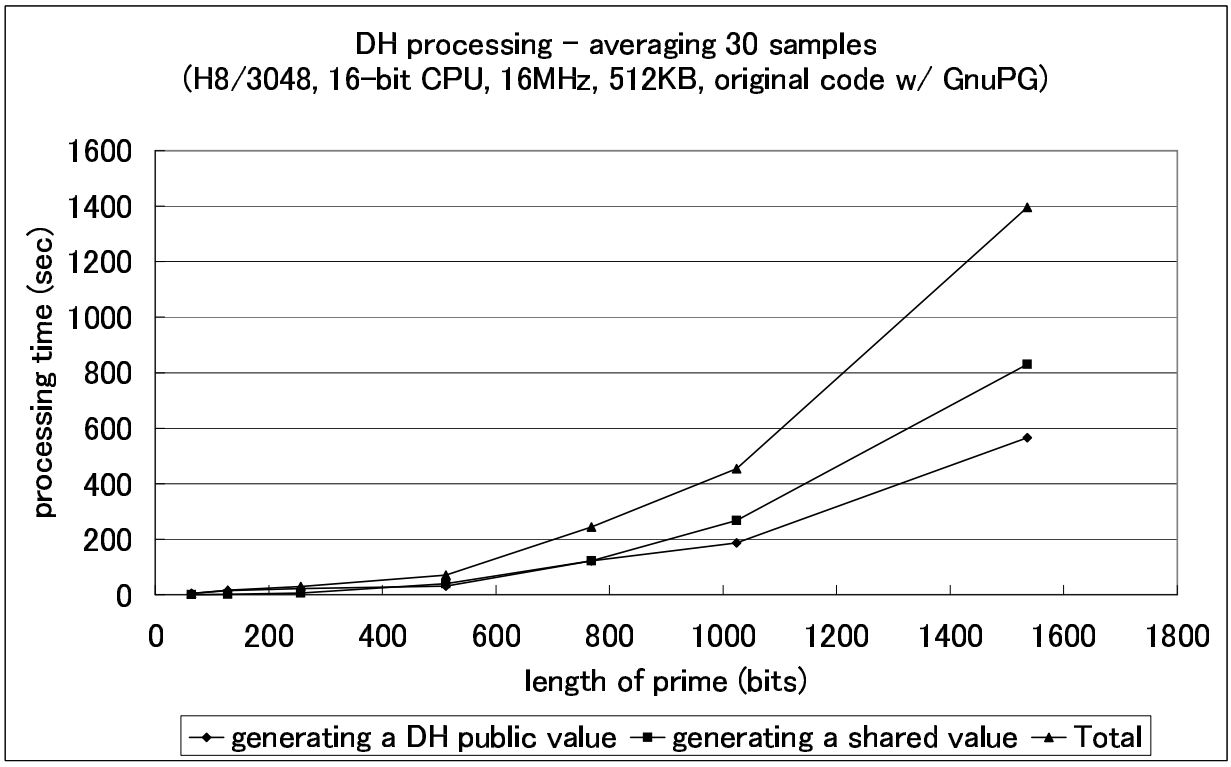

図 1.3: H8/3048 の DH 演算の実行時間

1.2.3 IPsec の処理時間

IPsec [16] は，アプリケーションに依存せずに通信のセキュリティを実現できるため，制御ネットワークの通信のセキュリティに有効な手段の一つと考える．本節では，処理性能の制約されたシステムで IPsec を実装した場合の性能に関して検討する．

ここでは，DS80C390 上でアセンブラにより実装した，IPsec ESP の Outbound 処理との実行時間と処理対象となるデータのバイト長の関係を図 1.6 にまとめる．この IPsec ESP コードで用いるハッシュと対称鍵暗号方式には，図 1.4 と図 1.5 で示すコードを利用している．IPsec ESP の Inboud 処理時間は，Outbound の場合とほぼ同じだったので説明を省略する．

対称鍵暗号方式の実行時間 (図 1.4 を参照) とハッシュの実行時間 (図 1.5 を参照) の和が，図 1.6 とほぼ等しいことから，IPsec 処理時間の大部分は暗号とハッシュに占められ，それ以外の処理は微々たるものであり，ハッシュと暗号処理の処理時間は，ハードウェア化で改善することが期待できる．

DS80C390 上でアセンブラにより IPsec ESP 機能を試作した．図 1.7 は，試作したコードのサイズを，機能別にまとめたものである．ハッシュのコードが IPsec ESP コード全体の 27 %，暗号のコードは全体の 67 % を占めていることから，ハッシュと暗号処理をハードウェア化することの効果は，コードサイズに顕著に出ることが期待される．実際の効果の評価については，今後の課題である．

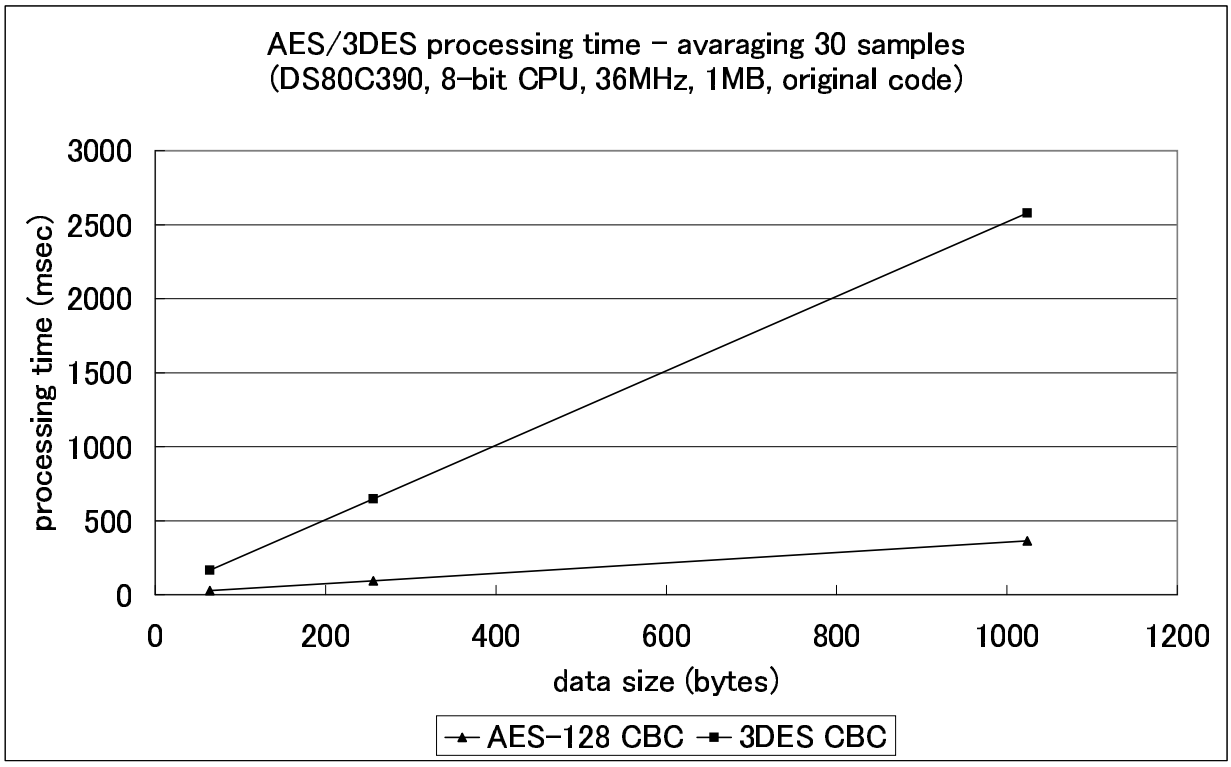

図 1.4: DS80C390 の暗号の処理時間

1.2.4 鍵交換プロトコルの選択

制御ネットワークを構成する機器には，組込み機器が多くもちいられ，そのユーザインターフェイスは限定されている．従って，IPsec を利用するためには，手動の鍵設定は難しく，鍵交換プロトコルが重要である．IETF IPSEC WG [4] では，IPsec の鍵管理方式として IKE を標準化しているが，DH 演算が必須なので，計算能力の限られている組込み機器に適していない（詳細は，1.2.1 節を参照）．IETF KINK WG [5] では，Kerberized Internet Negotiation of Keys (KINK) [18] と呼ばれる，IKE とは異なる IPsec 用の鍵管理方式を検討している．KINK は Kerberos [17] [4] を用い，対称鍵暗号だけを使っているので，IKE と比較して計算量が少なく，上述した組込み機器に適していることが期待できる (詳細は 1.2.1 節と 1.2.2 節を参照).

実際の実装では，IPsec パケットの暗号化/復号化や鍵管理プロトコルが暗号ハードウェアの利用で競合することが予想されるため，実用的な性能を実現できることの確認は，今後の課題である．

[4] Kerberos is trademarks of the Massachusetts Institute of Technology (MIT).

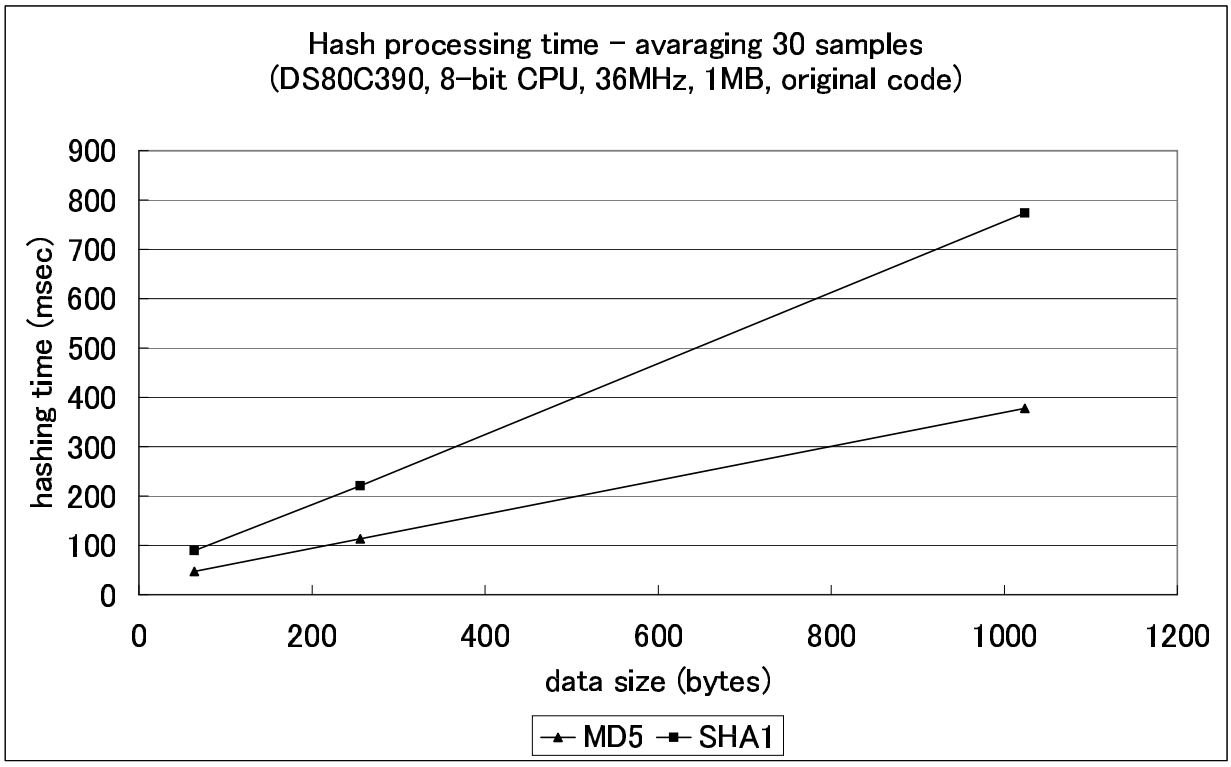

図 1.5: DS80C390 のハッシュの処理時間

1.3 ケーススタディ: Building Automation

1.3.1 BA システムの現状と課題

従来の BA システム

ビルの運用において，省エネルギー，快適性，利便性等を確保するには，ビルシステムに関わる空調・電気・防犯機器などの制御・監視をきめ細かく対応させながら，各機器の連動制御を行うことが必要である．しかし，従来システムでは空調なら空調，照明なら照明など設備毎にシステム化が行われ，システム間の連携制御は困難であった．一部で独自な解決法が提示されることもあったが，結局はクローズな手法であり，オープンな広がりを見せることはなかった．このため従来の BA システムでは，初期導入時に採用したメーカーへの依存度が極めて高く，シングルベンダー構成になりがちで，これにより機器更新および設備更新においても完全に特定メーカー主導になり，コスト構造の不透明さや，柔軟性，拡張性の欠如につながっていた．

オープンネットワーク化と相互接続

上記のような状況に対し，特に機器メーカー側では，相互運用可能で，よりきめ細かいビルの制御・運用が可能な世界標準のオープンなインターフェースを採用する動きが広まっている．

- LonWorks：米国のエシェロン社（Echelon Corporation）が提供するオープン・相互

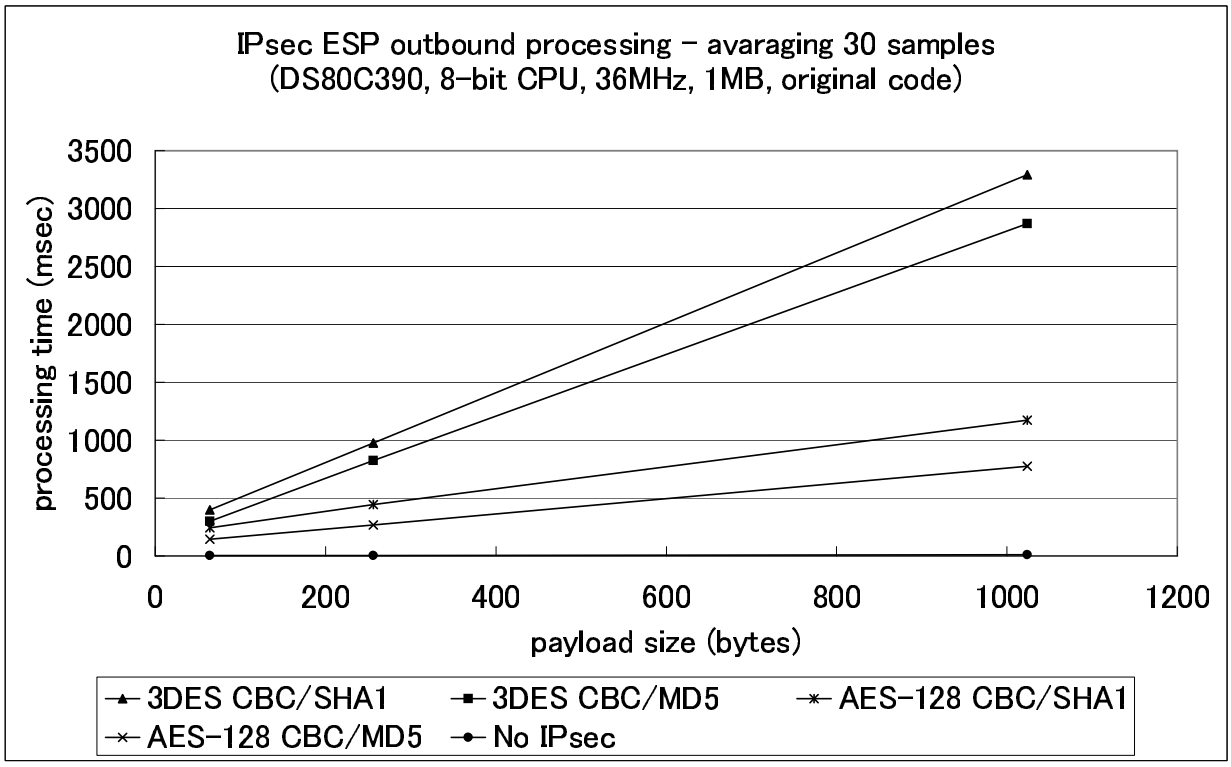

図 1.6: IPsec ESP Outbound 処理の実行時間

接続性を持つ開発環境である．エシェロン社では，1980 年代後半より，普遍的かつオープンな制御ネットワークの標準となる LonWorks プラットホームの開発を開始しており，広範囲にわたるメーカーが注目し，ここ数年で採用，実装されてきている．全世界の工業ネットワーク分野で 48%のトップシェアを持ち [8]，特にビルオートメーションでは世界のデファクトスタンダードになっている．Echelon 社は個々のデバイスを LonWorks 化し，ネットワーク対応するための Neuron チップ（数百円），ネットワークトランシーバー（数千円），開発キットを提供している．

- BACnet：ASHRAE(米国暖房冷凍空調工業会) が，メーカーの異なる空調機サブシステムを相互に接続するための標準化手法を作成する目的で作成したオープンスタンダードが BACnet(Building Automation and Control NETwork) である．BACnet の特徴の一つに高速イーサネットバックボーンの利用があり，システム全体のパフォーマンスを向上させることを意図している．日本では (社) 電気設備学会が「BAS 標準インターフェース」として規格化しており，殆どの BA ベンダーは設備サブシステム間通信ツールとして対応する方向にある．

  LonWorks と BACnet の特徴を表 1.1 に示す．

IP 技術によるさらなるオープン化

ビルシステムの大きな技術的変革にひとつとして位置付けられるのは，インターネットとの関わりである．オフィス用ビル，住居用ビルのいずれにおいても，コミュニケーショ

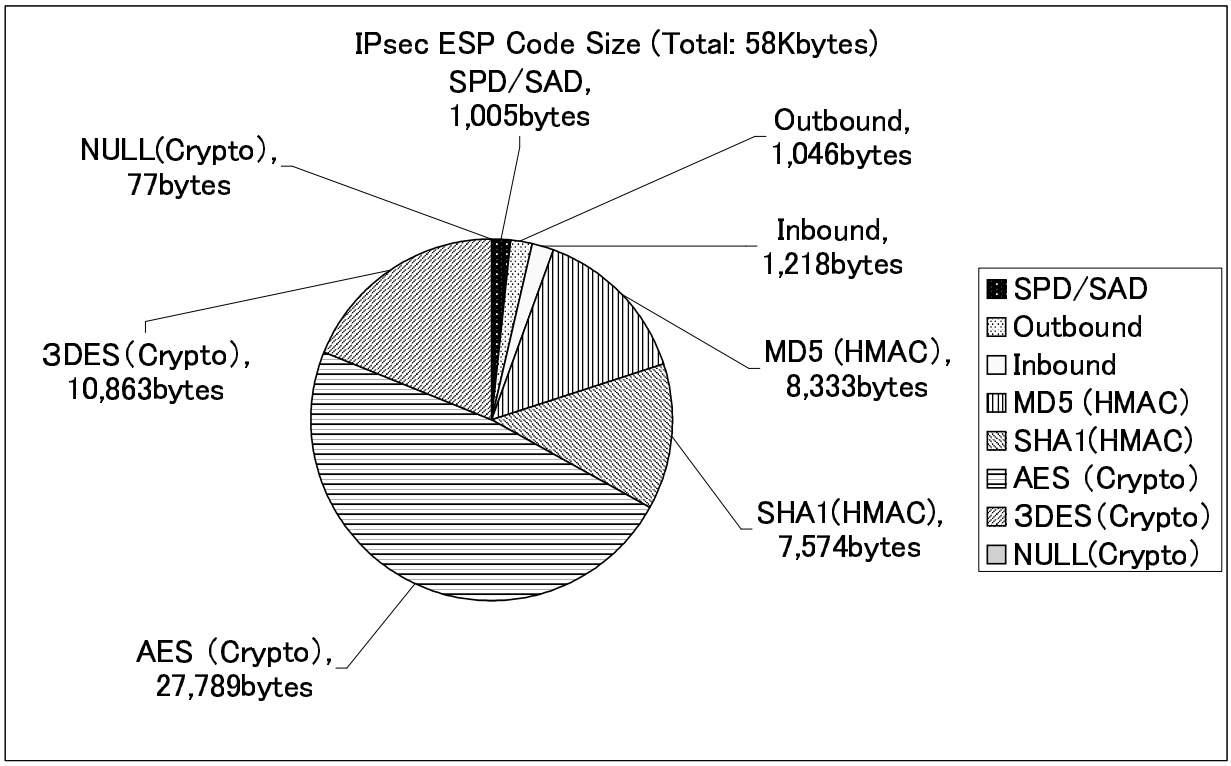

図 1.7: IPsec コードサイズの内訳

	BACnet	**LonWorks**
提唱元	ASHRAE (電気設備学会)	Echelon Corp.
観点	中央監視装置	ローカル制御
発展方向	ANSI/ISO 等の規格化	デファクトスタンダード
相互接続性	プロジェクト毎	LONMARK 協会

表 1.1: LonWorks と BACnet の特徴

ン，情報収集・共有のためにインターネットとの接続は，今や必須インフラとして求められている．

従って，ビル管理システムも統合情報インフラを利用することが期待されているのだが，前述の通り従来のビルシステムでは，(LonWorks，BACnet によるものも含め) 空調，照明，セキュリティ，防災など，様々なサブシステムが専用システムとして構成され，独立に運用管理されており，共通インフラを介して他のサブシステムと連携することは殆ど考慮されていなかった．このため，サブシステム相互に通信をするためには，ゲートウェイをそれぞれのサブシステムごとに配置し，どのサブシステムのどの機器と通信をするかをエンジニアリングすることが必要で，機器コストアップ，導入・メンテナンスコストアップの要因になっていた (図 1.8)．

また，それぞれのサブシステムが IP 化されたとしても，それらがプライベートアドレスを使用すると，サブシステム間のアドレス重複の問題を回避する非常に複雑なオペレー

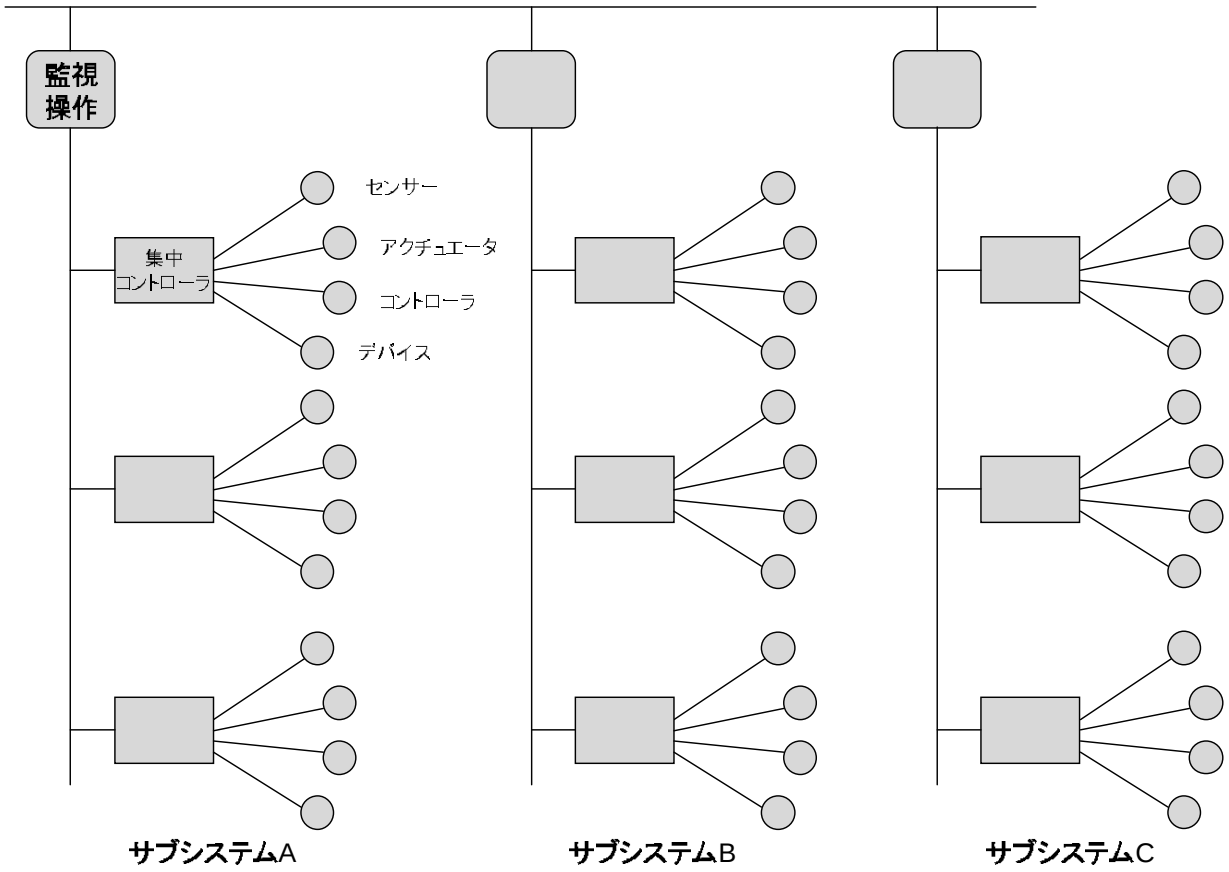

図 1.8: 旧来のシステム

ションが必要になる．一方で，ビル内に敷設されるセンサー，アクチュエータなどのノード数は膨大であり，IPv4 ではグローバルアドレス化は不可能である．

これを IPv6 により標準化することで，ゲートウェイを不要にするだけでなく，すべての機器のアドレスをグローバルユニークとすることができ，エンジニアリング工数削減，サブシステムごとの専用配線コスト低減や，フレキシブルなサブシステム間の連携など，今までの技術では実現が困難だった様々なアドバンテージを享受できる．

また，ビルシステムをインターネットにシームレス接続することで，ビル外部からのビル内設置機器のリモートメンテナンス，複数ビルの郡管理，他の IP アプリケーションとの連携など，従来はなかった付加価値の創造が可能で，運用・保守を含めたライフサイクルコスト低減やビル自身の資産価値保持に大きく貢献ができる．

1.3.2 IP 化 BA システムの要求事項

本節では，IP 化された BA システムに対する要求条件を示す．

アーキテクチャ

ビルシステムの将来を考えると，現状のシステムを包含した上で，相互接続性を保証されたマルチベンダーシステムの実現が必要になる．機能としては，今までの省エネ制御だけでなく，オフィスや店舗であれば，快適で仕事の効率が高まる環境の提供，住居であれば安全で快適な環境の提供をトータルにサポートできるアーキテクチャが求められる．また，ビルの建設・管理・運用・改装などを想定すると，現実的には，エンジニアリング，設置工事，変更作業，保守作業などの容易性が求められる．

このような将来のビルシステムに対する要求を満たすためには，各機器自身が，できる限り初期設置や変更時にも自身が自律的に動作するとともに，システムは，設置された機器を自動的に認識し，必要な制御パラメータ等を設定できるアーキテクチャを採用することが求められる．また今後の拡張性を考慮すると，単一のビルの中だけを意識するのではなく，インターネット上に広範囲で分散した機器も意識したアーキテクチャが求められ，例えば外部から多くのビルに設置された機器をインターネット経由でメンテナンスしたり，ビル内のセンサー情報を，インターネットを利用してモニタリングしたりすることも必要になる．

これらを実現するには，図1.8のような在来のプロプライタリなネットワークによる階層化されたものでは，サブシステム間の連携を行う通信が，ゲートウェイを用いた高価で複雑な作業となるためコストが大きく，またゲートウェイ1箇所の故障で，サブシステム間通信が停止するなどの信頼性の問題も起こる．これらの問題を回避するために，よりIPv6に親和性の高い図1.9のようなフラットで自律的な高度分散ピアツーピアシステムアーキテクチャを提案する．その特徴は，以下の通りである．

- 最大限のフレキシビリティと信頼性を獲得するために，ゲートウェイや中央集権のコントロールを持たず，各コントロールポイント自身にインテリジェンシーを持つ
- 階層化構造のマスター・スレーブ方式ではなく，最小限の通信，ボトルネックを作らないフラット構造のピアツーピア方式によるコミュニケーションを実現する
- システムへの追加・削除が容易に，フレキシブルにかつセキュアに実行できる
- 標準化されたプロトコル，アプリケーションインタフェースによるマルチベンダのオープンシステムとする

Secured Place & Play の実現

本研究の目指すビルシステムでは，センサーやコントローラ，設備機器 (これらを総称してノードと呼ぶ) を設置すると同時にシステムが自動的に認知し，システムからの制御パラメータ等の情報を受け取り，自律的に動作する Place&Play 機能を提供する．

一方，意図しない機器が Place&Play でネットワークに接続されることで，システム上のセキュリティ問題が生じないように，認証やメッセージの暗号化などのセキュリティ対応をコスト，運用容易性に関しての合理的なソリューションを提示することが必要である．また外部からのメインテナンスを想定する場合には，ベンダ技術者がビル内に設置されて

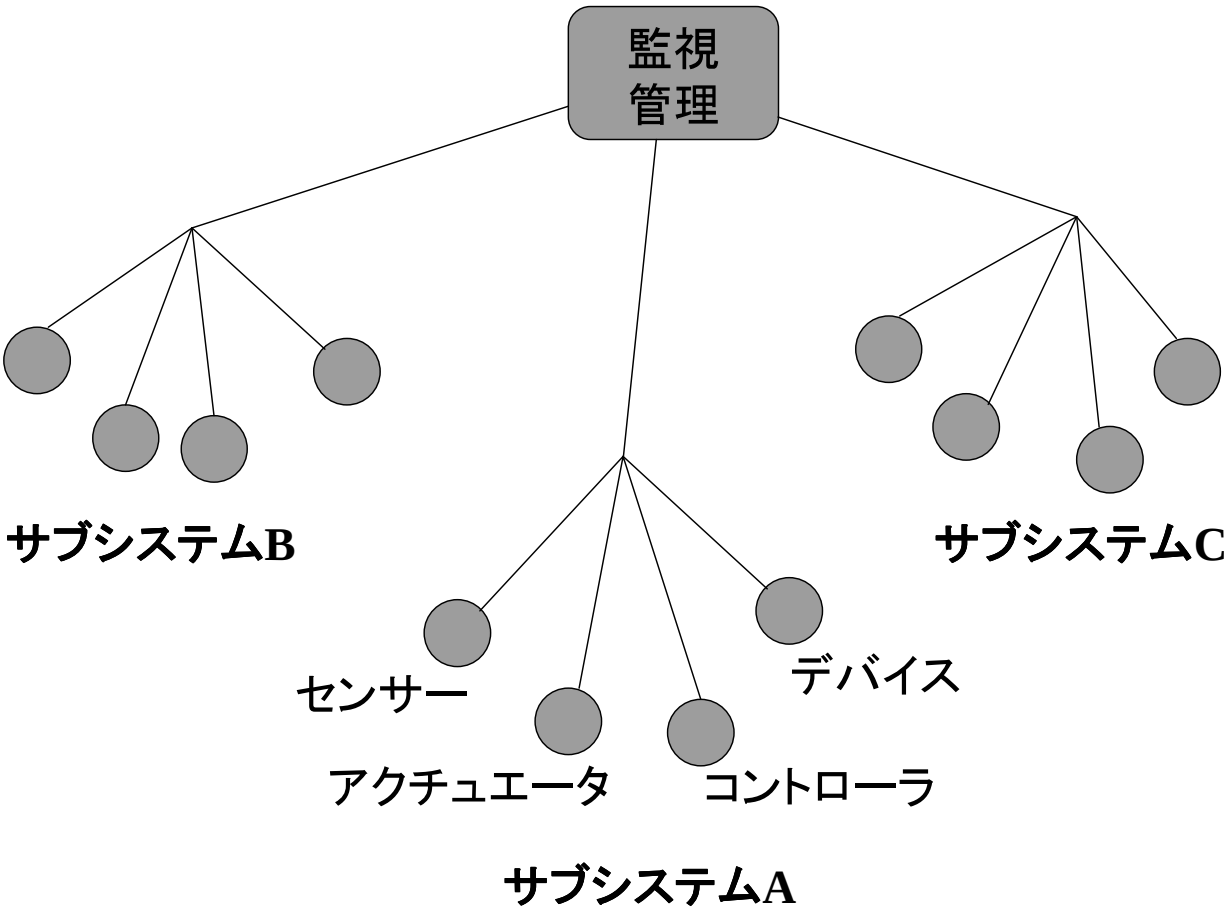

図 1.9: 提案システムアーキテクチャ

いるノードにセキュアな通信でリモートアクセスできるしくみも実現しなければならない．いずれの場合も，セキュリティの枠組，ポリシーの管理，ドメインの設定，運用などを検討することが必要になる．

コスト制約とノード実装

現状の BA システムの製品競争力を考えると最重要課題がコストである．従って，たとえ IP 化しても大幅なコストアップを伴うのでは受け入れられない．このため，低コストな限定されたノード資源上でも実現できる技術を選択するための指標 (プロトコル，実装) を示すことが必要である．

IP 化の付加価値の創造

IP 化することで，他システムとの連携による付加価値を得ることができる．このためには共通インターフェースの策定や，データ・プロトコルの標準化・オープン化などを行って,「使える」アプリケーションの土俵を広げると共に，有用なアプリケーション連携の掘

り起こしを行っていくことが必要である．

従来技術からの継続性

1.3.1 で述べたように，既に BA 技術者の間で普及している技術群があり，それらに付随する方法論やエンジニアリング手法がある程度普及しているのであれば，再教育の負担を回避するためにも，その手法を踏襲するのが有効と思われる．また設計手法などだけではなく，一般に BA の世界で「常識」とされている事項については，IP 化されたシステムの場合についての必然性を吟味したうえで継承すべきか否かを判断すべきである．

1.4 IP 技術を用いた制御ネットワークアーキテクチャ

1.4.1 要求事項

本研究の最大の目標は，これまで産業応用やビルディング管理で使われていた制御ネットワークとインターネットを融合することである．制御ネットワークはインターネット以前から存在し，大部分がベンダ固有のプロトコルで制御目的に特化している．またそのデバイスは低コスト，低性能であり，インターネットとは大きく特徴が異なる．これらを 1.3.2 に示すようなゲートウェイを使わない end-to-end アーキテクチャで融合するためには，幅広いプラットホームで適用可能なスケーラブルなプロトコルが必要になる．

例えば，1.3.2 節で述べた Secured Place&Play 機能を持った制御ネットワークを実現するためには，個々のセンサーノードの認証処理や，外部からのアクセスを正しく受け付け，かつ通信内容を秘匿するようなセキュリティ機能が必須である．IP においては Security Header を含む IPsec が包含されているので，このレイヤでセキュリティサポートできるのが，アプリケーション非依存であり，最も容易なソリューションである．しかし実際には，センサーなどの比較的能力の低いチップ上の実装において IPsec 処理が合理的な時間で可能か否かについての評価が必要であった．これに関しては実際の処理性能評価などの結果から IPsec の暗号化や認証処理自体が問題なのではなく，初期相互認証を含む鍵交換プロセスの中で使用される公開鍵暗号系の処理が大きなネックになっていると考えられる (詳細は 1.2 章を参照)．

インターネット標準化団体である IETF では IPsec の標準鍵交換プロトコルとして IKE, IKEv2 [15] といった公開鍵暗号系を含むプロトコルが主流である．これらは高性能な PC などでは利用可能であるが，制御ネットワークのセンサーノードデバイスでサポートするのは非常に困難である．このため，本研究では低コスト (低性能) チップ上での実装を想定した IPsec の鍵管理手法として公開鍵暗号系を使わない Kerberos ベースの KINK と呼ばれる鍵交換方プロトコルを提案している [21]．

従って制御ネット側の KINK 方式とインターネット側の IKE 方式という複数の鍵管理プロトコルが並列することで高機能 PC から低機能のセンサーノードまでをカバーすることになる．これを実現するため，少なくともインターネット側には複数の鍵管理プロトコルに対応可能な鍵管理モジュールを備えることが必要で，本研究でも取り組みを開始している．

ここではセキュリティ処理における鍵管理プロトコルにフォーカスしたが，これ以外にもノード能力の幅を考慮する必要のある処理が存在する可能性があり，検討，対応が必要である．また共通ミドルウェアなどの導入によるスムーズなプロトコルの融合なども検討すべきである．

1.4.2 IP 化制御ネットワークシステムの課題

KINK/IPsec によるアクセス制御

KINK 方式によるアクセスコントロールは，Kerberos で相互認証したノード同士だけが通信を確立できるようにすることで実現する．図 1.9 においてノード群を示すサブシステムは，各々が自分の realm(Kerberos が扱う領域空間) を持つ．あるサブシステムにおいてアクセスできるデバイスは，該当する realm に登録され，そのデバイスに固有の Kerberos におけるエンティティ名 (principal 名) と秘密鍵を割り当てされる．これにより，例えばあるスイッチからそれに対応する機器ノードに対して，適切な principal 名を用いた相互認証が可能となり，結果として機器ノードへの通信のアクセスコントロールが達成される．

Kerberos は認証するエンティティに関する情報を管理サーバに集中させるために，一般に大量のエンティティを管理する場合の拡張性に欠けている．制御ネットワークの場合，前述のようなサブシステム単位の realm 名を設定し，それ毎に鍵配布センター (Key Distribution Center, KDC) を配置するのが適正なエンティティ数に対応するのかどうかを評価する必要がある．

また，通信を開始する時にあらかじめ通信する相手の principal 名を知っている事は一般に仮定できない．しかし，principal 名をユーザが直接扱うのは不便なので，システムが与えた名前 (もしくはそれに対しユーザが自由に定義した名前) を Kerberos の principal 名にマップする仕組みが必要である．このプライベートな名前管理サーバについても，適切なスケールがどの程度であるかを評価することが必要である．

また運用にあたっては，realm 名を誰が初期設定し，運用にあたっての制御権限を誰に委譲するのか，という問題も実際には発生する．これは例えば，ビル施工時にはオーナーが権限を保持するある部屋の空調制御権を，テナント決定時点に委譲する，というケースである．この場合，例えば以下の問題を解決せねばならない．

1. セキュリティの全体の枠組を維持しながら，権限を委譲できる枠組．Kerberos ベースでは，KDC の管理者は，KDC に関わる全ての操作の権限を持っている．もし，単一 KDC の場合，テナント主に権限を委譲すると，彼は KDC に関する全ての権限を得てしまうので，問題である．他方，複数の KDC と複数の realm 名からなる Kerberos システムは非常に複雑になり，運用上の問題となる可能性を秘めている．

2. 委譲された側の動作・制御が限度を越えた場合に，それを適切に検出し，制御に介入できねばならない．また Logging, Auditing などを簡便に行えるスキームを提示し，不正な操作を即時に発見できるようにすることが必要である．

3. 共有スペースのゲスト問題このようなテナントへのサービスを発展させると，テンポラリーな利用者に，同様の権限委譲をするサービスが考えられる．例えば，会議室

の空調の制御権を一時的にその会議室の利用者へ与えるようなことである．

いずれの場合も，複数の管理主体 (例：ビル管理会社，テナント，外部委託されたセキュリティ管理会社など) が１つのデバイス群を管理するために起こる複雑さであるので，個々のケースに対し，行いたい操作と，禁止したい操作をきちんと整理し，整合するような枠組の提示が必要である．

ネットワーク制御プロトコルの選定

IP 化した制御ネットワークにおけるノード同士の通信は，従来の制御ネットワークにおけるプロトコルに相当するが， 現状の制御プロトコルは各ベンダ独自であり，インターネット側でも直接該当するプロトコルは標準化がなされていない．これをどのようにオープンプロトコルとして統一していくかは今後の大きな課題である．

例えば IP ネットワーク管理に使われる SNMP/MIB の枠組や，最近になって議論が始まっている IETF NETCONF WG [6] などが考えられるが，これまでに各ベンダが開発した既存技術資産をいかに活用し，オープンプロトコルへのスムーズな移行を実現するか議論が必要である．

インターネットシステムの共存の課題

インターネットアクセスサービスは今や必須インフラとして求められており，統合インフラの観点からも，制御ネットワークが IP 化すると，両者が物理的な回線を共有することが考えられる．その場合に発生し得る事態を分析し，それに対する対策 (例えば，両者の帯域保証や，どちらを優先するか，等) を検討する必要がある．特に緊急性の高い制御をリアルタイムに即時応答で実現できるのかを評価，検討することが必要である．

1.5 KINK に関するオペレーション上の注意点

1.5.1 KINK の弱点の概要

IPsec は IP 層で実現するセキュリティであり，IP パケット単位の秘密性と完全性と提供する．IPsec を動かすためには，通信する両端が IPsec Security Association(以後 SA) と呼ばれる，秘密情報を交換し設定しなければならない．SA 情報には，両端の IP アドレス，暗号アルゴリズム，共有鍵，有効期限などが含まれる．

KINK は，Kerberos をベースにした，SA の設定ための鍵交換プロトコルである．Kerberos は，principal 名と realm 名から構成される Kerberos 識別子を用いて，通信するノードの相互認証と通信を守るためのセッション鍵を提供する．Kerberos とノードが共有する秘密情報は，KDC で集中管理する．KINK においては，principal 名にノードの FQDN を用い，交換する SA 情報に必要な相手の IP アドレスは，principal 名から求める．本論文では，FQDN で表現される principal 名から IP アドレスを求める手段として DNS[20] を仮定する．

DNS は一つの IP アドレスに対して複数の異なるドメイン名を割り当てることが可能であり，日常的に利用されている．KINK を用いて SA 情報の交換を行う環境下で，この DNS

の柔軟性を利用し，自分のノードの IP アドレスを他人のノードの A/AAAA レコード [22] として登録されると，自分の SA を摩り替えられてしまう危険がある．

1.5.2 IPsec の鍵管理と KINK の特徴

IPsec 鍵管理プロトコルである KINK は，次の特徴を有する．a) KINK は，ホストベースの IPsec のみをサポートするため，SA の識別は自分と相手の IP アドレスの組となる．つまり，あるノード上で複数のユーザやアプリケーションが，同一の相手ノードと通信していても，対応する SA は一つだけとなる．b) IPsec を確立するノード同士は，Kerberos 識別子に対して相互認証を行う．認証の手段には，Kerberos のチケットを用いる．c) KINK は，Kerberos 化したアプリケーションの一種として，SA を交換する．d) Kerberos をベースにしているため，公開鍵暗号系が不要であり，計算能力の限定されたノードでも実現が容易である．e) Kerberos 識別子は principal 名と realm 名から構成されている．KINK においては，principal 名にノードの FQDN を使うことが規定されているため，識別子は FQDN@realm となる．例えば，realm MY に属し FQDN が X.FOO.ORG のノード X の Kerberos 識別子は，X.FOO.ORG@MY となる．

本論文で言及する KINK のオペレーション上の注意点は，上述した特徴の a と b と c から生じる．第一に，KINK がホストベースの IPsec だけをサポートしているため，あるユーザ/アプリケーションが既に設定した SA を，別のユーザ/アプリケーションが置き換える余地が存在する．第二に，KINK は Kerberos 識別子に対して相互認証するが，設定される SA にその識別子の情報を伴わないので両者の一貫性を確認できなくなる．第三に，KINK が Kerberos 識別子から，対応する IP アドレスを求める方法は，Kerberos や KINK の仕様の範囲外であること．Kerberos 識別子と IP アドレスの対応の正しさは，採用する名前解決の仕組みに依存する．第四に，一旦 SA が設定されると，IPsec が自立的に機能するので，以後 SA の IP アドレスと Kerberos 識別子との関係はチェックされない．

つまり，名前解決の仕組みに不正な名前と IP アドレスの関係を登録できる場合には，既に存在する SA を異なるものに摩り替えることが可能になる．この詳細については，1.5.5 章で述べる．

1.5.3 KINK の動作の概略

本節では，KINK のオペレーション上の注意点を説明するために，まず KINK の動作の具体例を述べる

本例では，IPsec を確立するノード X と Y，両ノードの A レコードを管理する DNS サーバ DNS-FOO，両ノードの鍵を管理する Kerberos サーバ KDC がネットワークを構成しているものとする (表 1.2 を参照)．

これらのノードとサーバのネットワーク構成を図 1.10 に示す．

X が Y と KINK を用いて SA 情報を交換し，SA を設定する手順の概略は以下となる．

1. X は，KDC から Y と SA 情報を交換するためのチケットとセッション鍵を取得する．

Player	Description
X	A node belonging to realm MY
	IP address: IPx
	Kerberos ID: X.FOO.ORG@MY
	Secret key: Kx
Y	A node belonging to realm MY
	IP address: IPy
	Kerberos ID: Y.FOO.ORG@MY
	Secret key: Ky
KDC	Kerberos KDC for realm MY
	IP address: IPkdc
	Managing secret keys: X.FOO.ORG@MY:Kx, Y.FOO.ORG@MY:Ky
DNS-FOO	DNS system for zone FOO.ORG
	Managing A-records: X.FOO.ORG:IPx, Y.FOO.ORG:IPy

表 1.2: 構成要素

2. X は，取得したチケットとセッション鍵を用いて，SA 情報を交換する．この部分の交換を KINK と呼ぶ．

手順 1 の詳細を以下と図 1.11 に示す．

1.1. 何らかの手段により，X には，KINK を用いて Y と IPsec を確立する要求が発生し，Kerberos による交換を開始する．

1.2. X は，KDC に対して，任意のチケットを発行するためのチケット TGT(Ticket Granting Ticket) の発行を KDC に要求する．

1.3. KDC は，Kerberos 識別子 X.FOO.ORG@MY に応じた TGT としての TGTx とセッション鍵 Sx を返す．

1.4. TGTx は，秘密鍵 Kx を知っている KDC だけが生成できるので，X は KDC が正しいことを確認できる．

1.5. X は，Y と SA を交換するために必要なチケット TICKETxy を KDC に要求する．チケットの取得には，事前に取得した TGTx と Kerberos 識別子と X の現在時刻 Tx を Sx で暗号化した認証データ AUTHx を KDC に提示する必要がある．TGTx が有効である間は，手順 1.2〜1.4 は省略できる．

1.6. KDC は，X と Y が安全に通信するために必要な情報として，TICKETxy とセッション鍵 Sxy を返す．この情報は，Sx で暗号化されているので，X だけが解読できる．

1.7. X は，Y と KINK 交換を開始する．

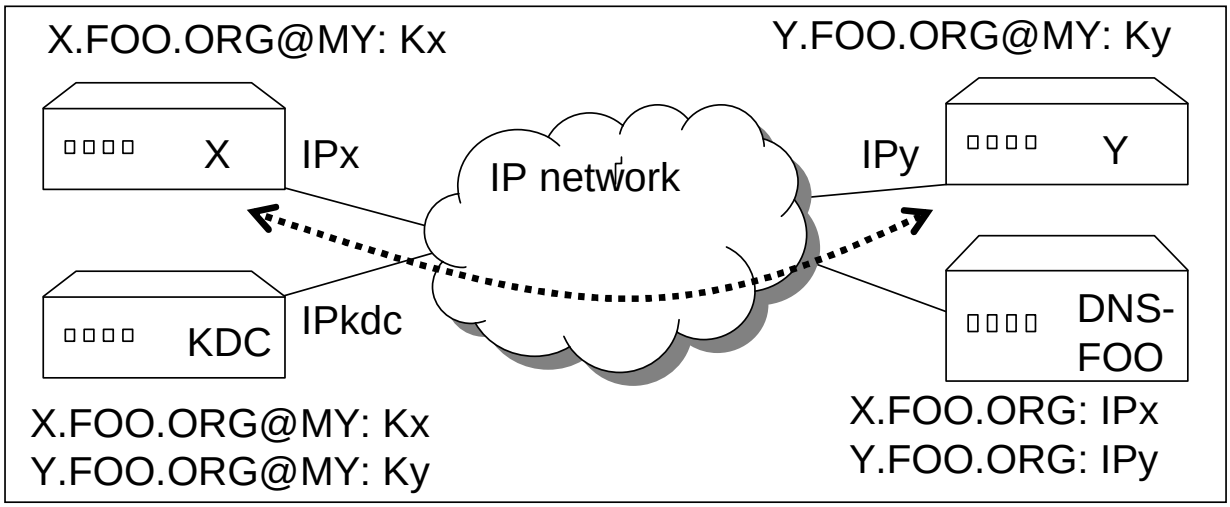

図 1.10: ネットワーク構成

手順 2 の詳細を以下と図 1.12 に示す.

2.1. X は，Y の principal 名を FQDN とみなして名前解決を行い，IP アドレス IPy を得る．X は，疑似乱数関数 prf_x とセッション鍵 Sxy から IPsec で使う鍵 prf_x(Sxy) を生成し，SA[IPx←IPy, prf_x(Sxy)] を生成し，自らの SA として設定する．

2.2. X は，生成した SA[IPx←IPy, prf_x(Sxy)] と TICKETxy と Kerberos 識別子と X の現在時刻 Tx からなる認証データ AUTHxy を Y へ送る．この情報は，Sxy で暗号化されているので，X と Y だけが解読できる．

2.3. Y は，TICKETxy と AUTHxy を解読することで，パケットの送信元が X.FOO.ORG @MY であることを認証する．Y は，疑似乱数関数 prf_y とセッション鍵 Sxy から IPsec で使う鍵 prf_y(Sxy) を生成し，SA[IPx→IPy, prf_y(Sxy)] を生成し，受信した SA[IPx←IPy, prf_x(Sxy)] と生成した SA[IPx→IPy, prf_y(Sxy)] を自らの SA として設定する．

2.4. Y は，生成した SA[IPx→IPy, prf_y(Sxy)] と新たな認証データ AUTHyx を返す．この情報は，Sxy で暗号化されているので，X と Y だけが解読できる．

2.5. X は，AUTHyx を解読することで，パケットの送信元が Y.FOO.ORG@MY であることを認証する．X は，受信した SA[IPx→IPy, prf_y(Sxy)] を自らの SA として設定する．

1.5.4 問題の発生する環境

ユーザ Alice は，X/Y 間で通信を行っている．この通信は，KINK によって確立した SA で守られている．X と Y はドメイン FOO.ORG に属し，realm MY に属している．

ユーザ Bob は，Alice の通信の盗聴を試みるが，パケットは SA で守られているので，解読ができない．そこで，SA を自分の解読できるものに置き換えることを計画する．まず，Bob

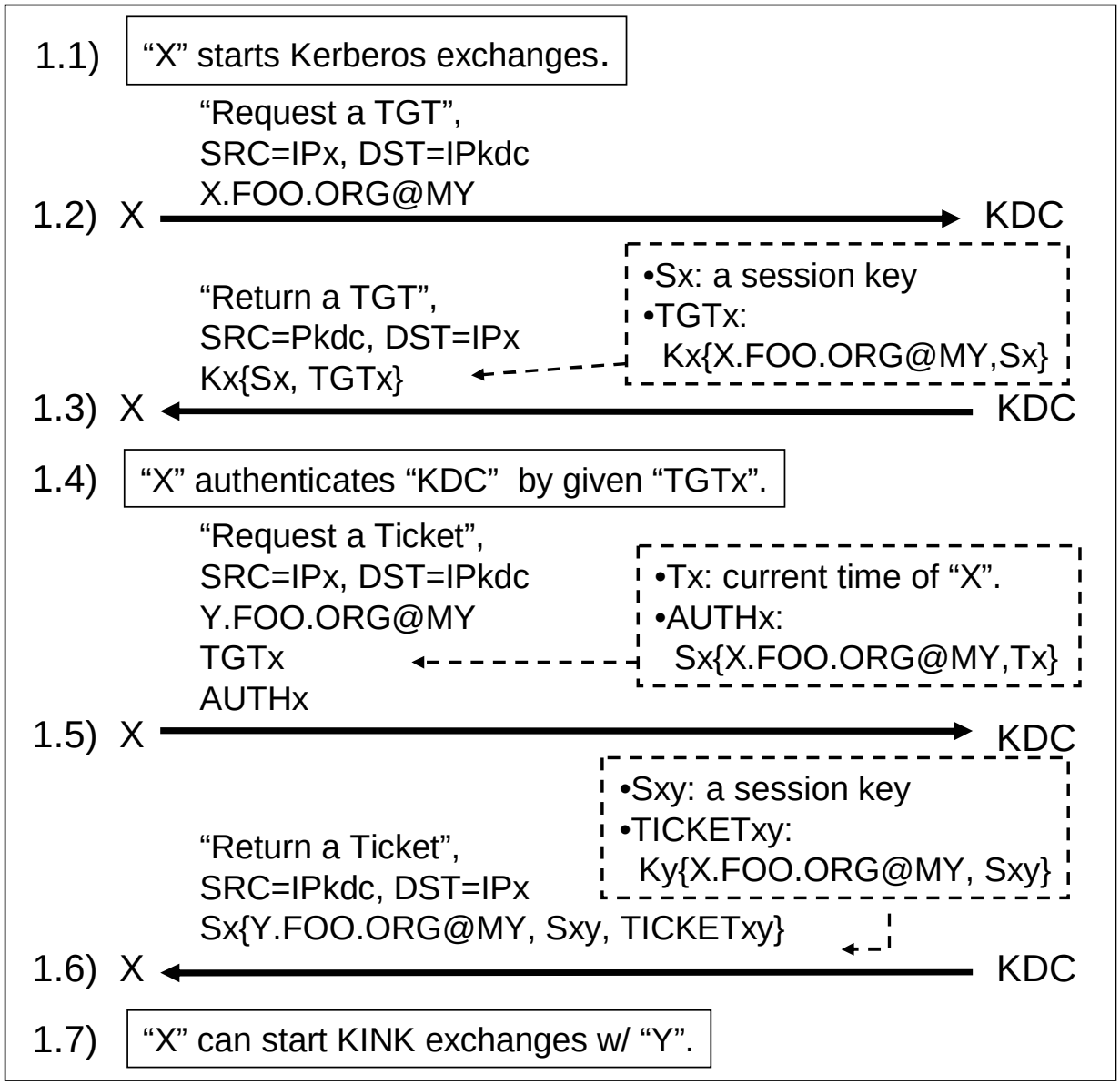

図 1.11: Y と通信するためのチケットの取得

は，何らかの手段により，X と Y の経路上の中間に Z を用意する．Bob は Z の特権ユーザのアカウントを持っている．次に，Bob は，自分の管理下にあるドメイン名 BAR.ORG と，それを管理する DNS サーバ DNS-BAR を用意する．そして，Z の FQDN として Z.BAR.ORG を登録し，Z の Kerberos 識別子として Z.BAR.ORG@MY を申請し，それが認められる．

ここで，Z は必ずしも複数のネットワークインターフェイスをもったルータである必要はなく，両端でやりとりされるパケットの経路上の中間に位置していれば良い．

本章で想定するノード等とその役割に関しては，表 1.2 からの差分を表 1.3 にまとめ，本章で想定するネットワーク構成は，図 1.13 に示す．

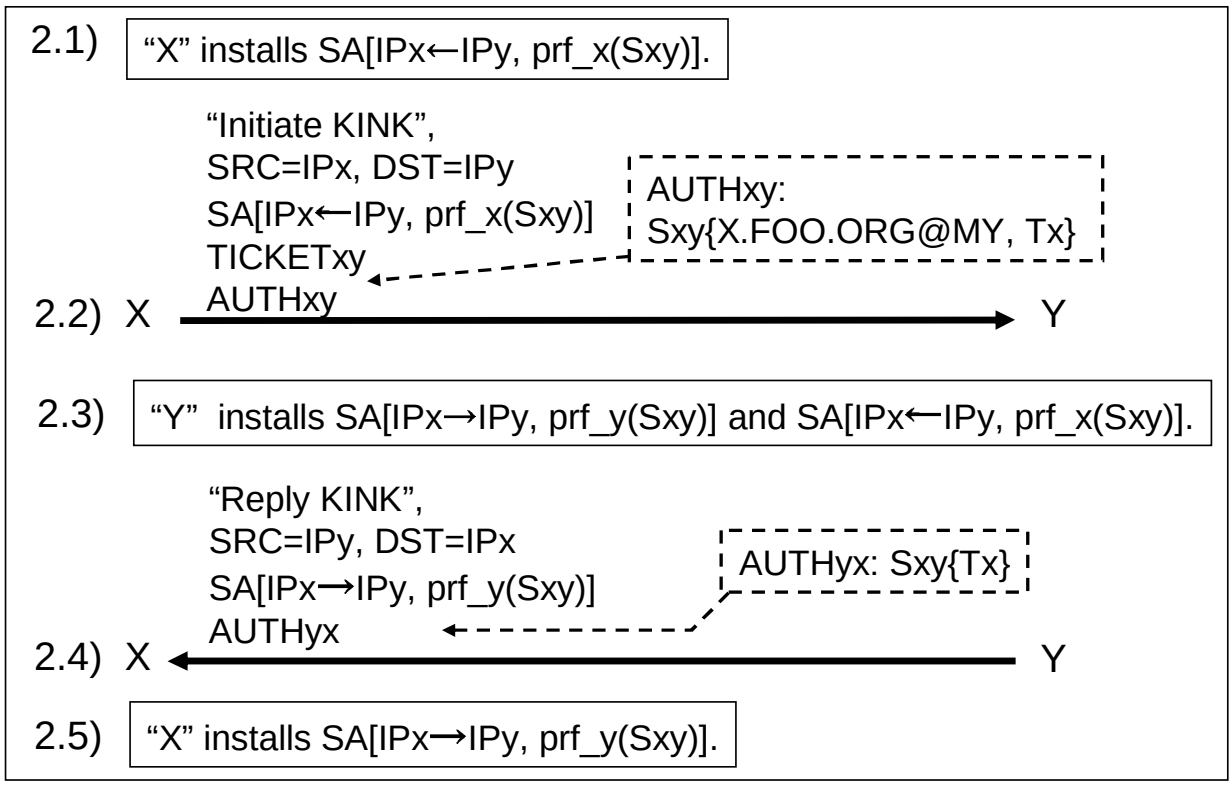

図 1.12: KINK の交換

1.5.5 IPsecSA の摩り替え方

本節では，名前解決を悪用して SA を置き換える方法に述べる．まずは，その手順の概略を説明する．

1. 1.5.3 節の手順 1 と同じ．

2. 1.5.3 節の手順 2 と同じ．Alice は，この上で通信を行うのため，Z でパケット観測しても，その内容は解読されない．

3. Bob は，Z.BAR.ORG の A レコードに，Y の IP アドレスの値を設定する．即ち，Z.BAR.ORG の名前解決を行うと，IPy が返ってくる．そのうえで，Bob は X に対して，Z と IPsec を確立させる何らかのきっかけを与える．X は，Z と KINK による鍵交換をするためのチケットを KDC から入手する．Z は KDC に Z.BAR.ORG@MY として登録されているので，Kerberos 識別子の観点からは，正しいチケットが発行される．

4. X が KINK の交換を開始する．しかし，交換する SA 情報は，Z の principal 名に対応する IP アドレスに基づいた内容となる．即ち，KINK パケットは，Z.BAR.ORG が解読できる内容になっているが，そのパケットで搬送される SA 情報は IPx/IPy 間の内容になっている．Z は，X と Y の中間に位置しているので，Y になりすまして，KINK 交換を完了させる．この結果，X に設定されていた SA が置き換えられてしまう．以後，X から Y へ投げられる IPsec パケットは Z により解読できてしまう．

Player	Description
Alice	A victim user
	Having an account on X and Y
Bob	An attacker to eavesdrop Allice's communication
	Having an account on X, and a root account on Z
	Having a right to update Z's A-record on DNS-BAR
Z	An intermediate node between X and Y
	IP address: IPz and IPz'
	Kerberos ID: Z.BAR.ORG@MY
	Secret key: Kz
KDC	Kerberos KDC for realm MY
	IP address: IPkdc
	Managing secret keys: X.FOO.ORG@MY:Kx, Y.FOO.ORG@MY:Ky, Z.BAR.ORG@MY:Kz
DNS-BAR	DNS system for zone BAR.ORG
	Managing A-record: Z.BAR.ORG:IPz→IPy (updated by Bob later)

表 1.3: 想定する構成要素

以下では，より詳細に手順を説明する．ここで，手順 1 と 2 は 1.5.3 節と同じなので，詳細な説明を省略する．手順 3 を以下と図 1.14 に示す．

3.1. Bob は，Z.BAR.ORG の A レコードを IPy に変更する．これは，Y の IP アドレスを不正に登録したことになる．X 上の Bob は，何らかの手段により，Z に対する KINK 交換を開始させる．

3.2. X は，Z と SA を交換するために必要なチケット TICKETxz を KDC に要求する．

3.3. KDC は，X と Z が安全に通信するために必要な情報として，TICKETxz とセッション鍵 Sxz を返す．この情報は，Sx で暗号化されているので，X だけが解読できる．

3.4. X は，Z と KINK の交換を開始する．

手順 4 を以下と図 1.15 に示す．

4.1. X は，Z の principal 名を FQDN とみなし，IP アドレス IPy を得る．ここで，IPy は Y のアドレスであることに注意すること．X は，疑似乱数関数 prf_x とセッション鍵 Sxz から IPsec で使う鍵 prf_x(Sxz) を生成し，SA'[IPx←IPy, prf_x(Sxz)]) を生成し，自らの SA として設定する．これにより，手順 1 で設定された受信側の SA が上書きされてしまう．

4.2. X は，生成した SA'[IPx←IPy, prf_x(Sxz)] と TICKETxz と Kerberos 識別子と X の現在時刻 Tx からなる認証データ AUTHxz を IPy へ送る．この情報は Sxz で暗号化されているので，X と Z だけが解読できる．

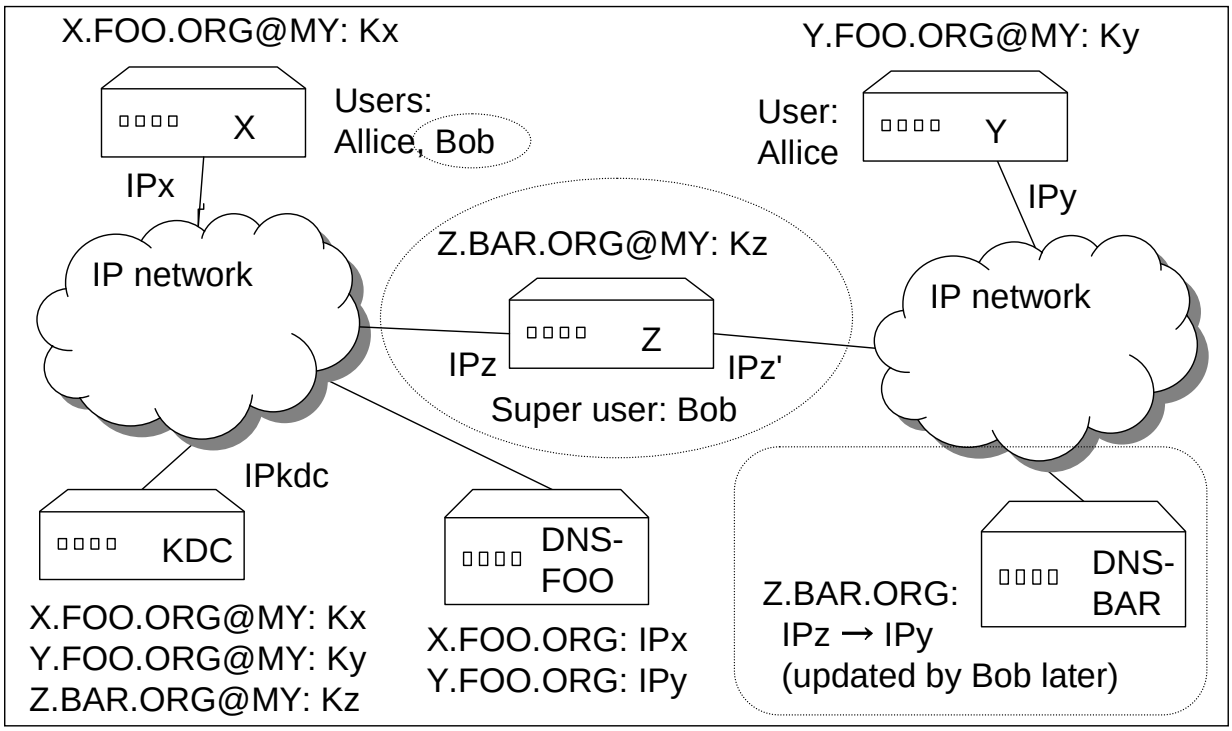

図 1.13: 想定するネットワーク構成

4.3. Z は，IPy 宛の KINK パケットを横取りし，Y のふりをして，KINK 交換を行う．Z は，疑似乱数関数 prf_z とセッション鍵 Sxz から IPsec で使う鍵 prf_z(Sxz) を生成し，SA'[IPx→IPy, prf_z(Sxz)] を生成し，受信した SA'[IPx←IPy, prf_x(Sxz)] と生成した SA'[IPx→IPy, prf_z(Sxz)] を自らの SA として設定する．

4.4. Z は，生成した SA'[IPx→IPy, prf_z(Sxz)] と新たな認証データ AUTHzx を返す．この情報は，Sxz で暗号化されているので，X と Z だけが解読できる．

4.5. X は，AUTHzx を解読することで，パケットの送信元が Z.BAR.ORG@MY であることを認証する．X は，受信した SA'[IPx→IPy, prf_z(Sxz)] を自らの SA として設定する．この結果，X に設定されていた SA が置き換えられてしまう．以後，X から Y へ投げられる IPsec パケットは Z により解読が可能になる．

1.5.6 KINK の運用に関する考察

1.5.5 章の方法は，X の SA だけを摩り替えるので，Z は X が出す XY 向きのパケットを解読して Y へ平文で投げられるが，Y の出す YX 向きのパケットを解読することはできない．同様に，X も Y が出す YX 向きのパケットを解読できない．つまり，通信の双方向性が失われるので，この方法は，継続的な通信の盗聴よりは，通信妨害に利用されると考えられる．

KINK のオペレーション上の注意点の原因は，1.5.2 章で述べた通り，Kerberos 識別子に対して不正な IP アドレスを登録できてしまうことである．つまり，この問題の本質は

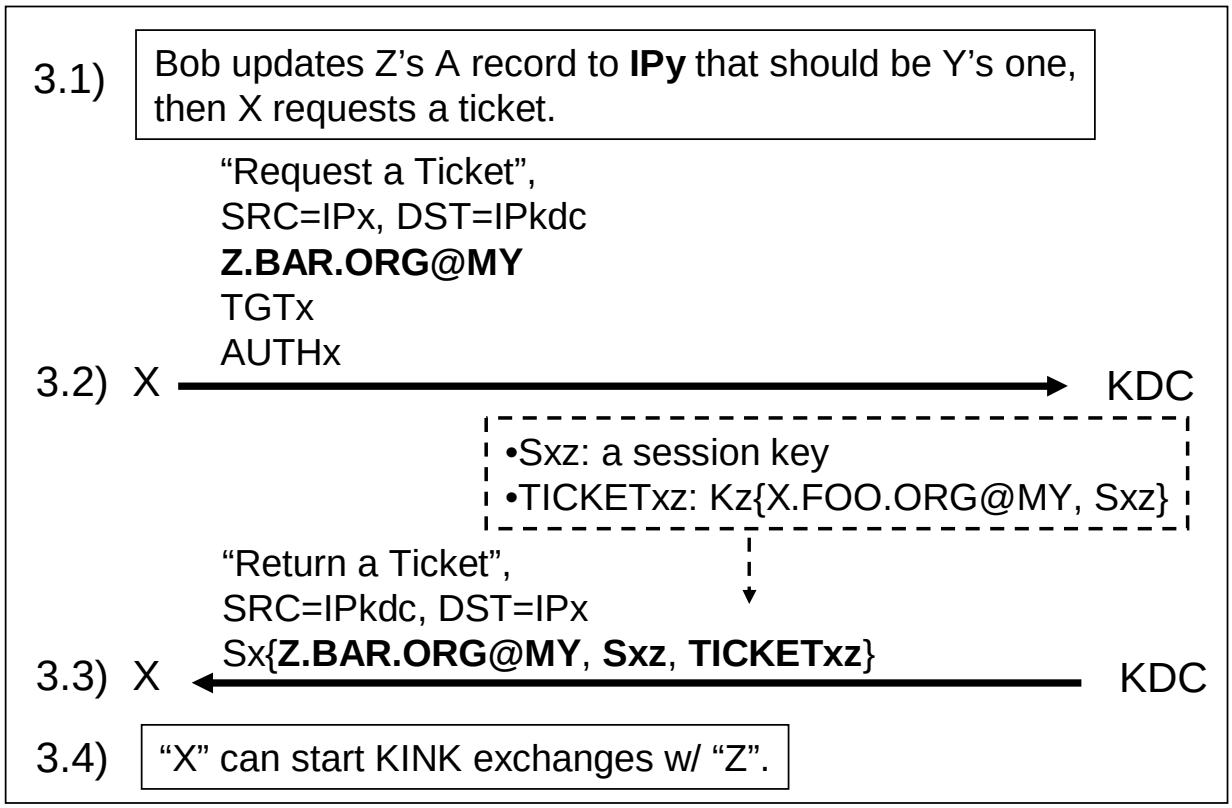

図 1.14: Z と通信するためのチケットの取得

Kerberos と KINK と DNS に跨った，システム全体の一貫性にある．

DNS では，既に存在するノードの IP アドレスを，異なるノードの A レコードとして登録することは可能である．一方，DNS システム側でこのような矛盾を検出することは容易ではない．一つのアドレスに複数の異なる名前を割り当てることは，通常に行われていることであり，それらの名前の一部が不正に登録されたかの判断は，運用に求められるからである．また，特定の利用環境では悪意がなくても，同様の事態が起き得る．例えば，アドレスを DHCP で割り当て，そのアドレスを動的な DNS 更新 [23] で自らの A レコードや AAAA レコードとして登録する環境下で，このアドレスが別のノードに再び割り当てられたにも拘らず，動的に登録された古い A レコードや AAAA レコードが残ってしまった場合である．つまり，対応付けの正しさの絶対的な保証はしない．

KINK では，名前と IP アドレスと実体の対応の一貫性が重要なので，DNS を使うのは本質的に好ましくなく，この一貫性を満たすための別の名前解決の仕組みが必要である．もし DNS を使うなら，以下の点に考慮すべきである．

まず，管理権限の及ばない，あるいは管理ポリシーの異なるドメインに属する名前を，Kerberos の principal 名として認めないこと．次に，前述した一貫性を保証する運用が望ましい．DHCP と動的な DNS 登録を使っている環境では，この一貫性を運用で保証することは不可能なので，それを保証するなんらかの仕組みが必要となる．

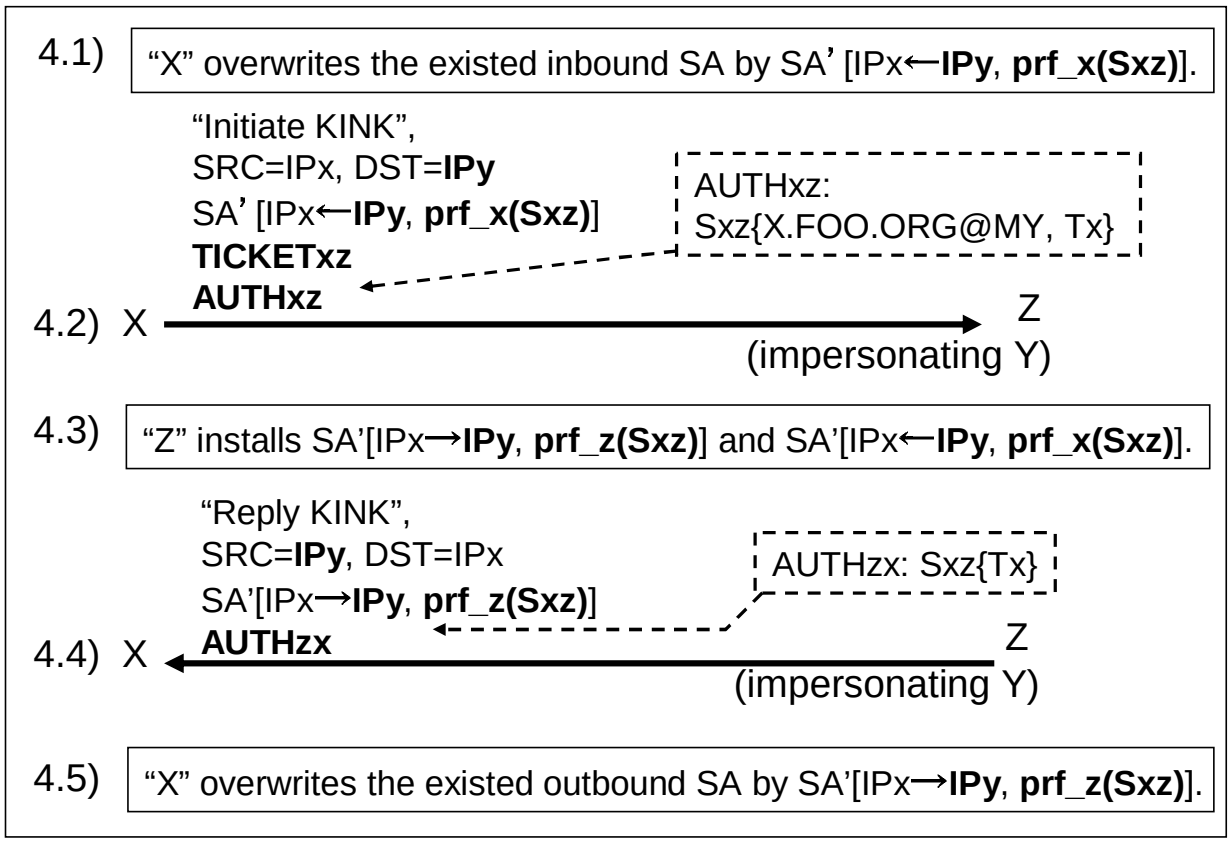

図 1.15: Z による X/Y 間の SA の摩り替え

1.6 まとめと今後の課題

本研究は，a) 制御ネットワークの IP 化とセキュリティの必要性，b) その実現手段として Secured Place & Play のアイデア，c) 処理能力の限定されたシステムでも Secured Plug & Play を実現するために，暗号処理のハードウェア化と IPsec と KINK と Kerberos が重要であること，d) KINK の運用に注意すべき点が存在すること，を示した．

今後の課題としては，以下が存在する．

- 暗号処理のハードウェア化：暗号処理をハードウェア化することで，処理能力の限定されたシステムでも妥当な性能でセキュリティを実現できる可能性を示した．しかし，実際の実装では，IPsec パケットの暗号化/復号化や鍵管理プロトコルが暗号ハードウェアの利用で競合することが予想されるため，実用にあった性能を得られることの評価が必要である．

- Secured Place & Play：本研究では，Secured Place & Play を制御ネットワークを IP 化するうえで重要な機能と位置付けている．これの有効性を示すために，本機能の詳細な仕様と評価システムのによる評価が必要である．

- Kerberos の運用とアクセス制御：本研究では，Kerberos に基づいたセキュリティシステムを提案している．Kerberos では，realm という領域空間が管理単位となるが，realm 間の相互運用性は柔軟性に欠けている．一方，実際の運用では複数の領域空間

に跨った管理や，他の管理システムへの権限の委譲などが必要になる．実用に合った運用を実現するための Kerberos の運用やアクセス制御の検討が必要である．

- 制御プロトコル：現状の制御ネットワークの要求を満たす，IP 上のプロトコルが存在するかは不明確である．SNMP が候補と考えられるが，その有効性は未確認である．現状の制御ネットワークのプロトコルの分析と IP 化に関する検討が必要である．
- 監視システム：従来の制御ネットワークを監視するシステムにとって，ネットワーク網自体の監視は重要な要素ではなかった．これは，従来の制御ネットワークの規模が小さく単純だったためである．しかし，制御ネットワークを IP 化すると，ネットワーク網の監視は重要な監視対象なる．従来の監視機能とネットワーク網の監視を一体化するこの検討が必要である．
- 通信の高信頼性とリアルタイム性：制御ネットワークの場合，通信のリアルタイム性と高信頼性が必ず問われるので，IP で確立している帯域制御や L2 技術等を有効に利用したリアルタイム性と高信頼技術の検討が必要である．

関連図書

[1] *Control Area Network*. ISO 11898.

[2] *GnuMP: A free library for arbitrary precision arithmetic*. http://www.swox.com/gmp/.

[3] *GnuPG: a RFC2440 (OpenPGP) compliant application*. http://www.gnupg.org/.

[4] *IETF IP Security Protocol WG (ipsec)*. http://www.ietf.org/html.charters/ipsec-charter.html.

[5] *IETF Kerberized Internet Negotiation of Keys WG (kink)*. http://www.ietf.org/html.charters/kink-charter.html.

[6] *IETF Network Configuration WG (netconf)*. http://www.ietf.org/html.charters/netconf-charter.html.

[7] エシュロンジャパンのプレス発表. http://www.echelon.co.jp/AC/press030909.html.

[8] ARC Advisory Group 社の調査, 1998. http://www.arcweb.com/.

[9] ANSI/ASHRAE Standard 135-1995. *BACnet–A Data Communication Protocol for Building Automation and Control Networks*. http://www.bacnet.org/.

[10] Fieldbus Foundation. http://www.fieldbus.org/.

[11] D. Harkins and D. Carrel. *The Internet Key Exchange (IKE)*, November 1998. RFC 2409.

[12] Hitachi, Ltd. *16-bit micro-controller H80/3048*. http://www.hitachisemiconductor.com/sic/jsp/japan/jpn/PRODUCTS/MPUMCU/8_16BIT/H8300H/3048/.

[13] IEA-709.1. *CONTROL NETWORK PROTOCOL SPECIFICATION*. http://www.lonmark.org/.

[14] B. Kaliski. *PKCS #1: RSA Encryption Version 1.5*, March 1998. RFC 2313.

[15] Charlie Kaufman. *Internet Key Exchange (IKEv2) Protocol*, February 2003. Internet-draft.

[16] S. Kent and R. Atkinson. *Security Architecture for the Internet Protocol*, November 1998. RFC 2401.

[17] J. Kohl and C. Neuman. *The Kerberos Network Authentication Service (V5)*, September 1993. RFC 1510.

[18] M. Thomas, J. Vilhuber. *Kerberized Internet Negotiation of Keys (KINK)*, January 2003. Internet-draft.

[19] Maxim/Dallas Semiconductor. *8-bit micro-controller DS80C390*. http://pdfserv.maxim-ic.com/arpdf/DS80C390.pdf.

[20] P.V. Mockapetris. *Domain names - implementation and specification*, November 1987. RFC 1035.

[21] S.Sakane, N.Okabe, K.Kamada, and H. Esaki. Applying Kerberos to the Communication Environment for Information Appliances. In *Proceedings of SAINT2003 workshop*. IEEE, 2003.

[22] S. Thomson and C. Huitema. *DNS Extensions to support IP version 6*, December 1995. RFC 1886.

[23] P. Vixie, Ed., S. Thomson, Y. Rekhter, and J. Bound. *Dynamic Updates in the Domain Name System (DNS UPDATE)*, April 1997. RFC 2136.

[24] 井上淳, 石山政浩, 岡部宣夫. Building Automation システムの IP 化における要求事項、課題の考察. 情報処理学会 マルチメディア・分散・協調とモバイル (DICOMO2003) シンポジウム, June 2003.

[25] 岡部宣夫, 石山政浩, 坂根昌一, 鎌田健一, 井上淳. KINK の脆弱性: 名前解決を悪用した IPsec SA の摩り替え. コンピュータセキュリティシンポジウム 2003 (CSS2003), October 2003.