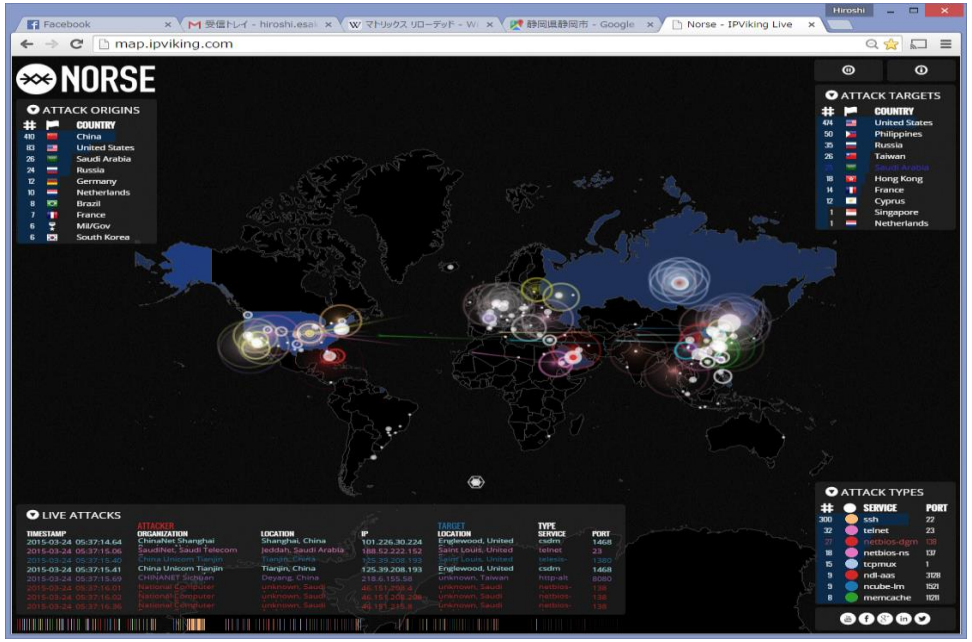
セキュリティー と プライバシー

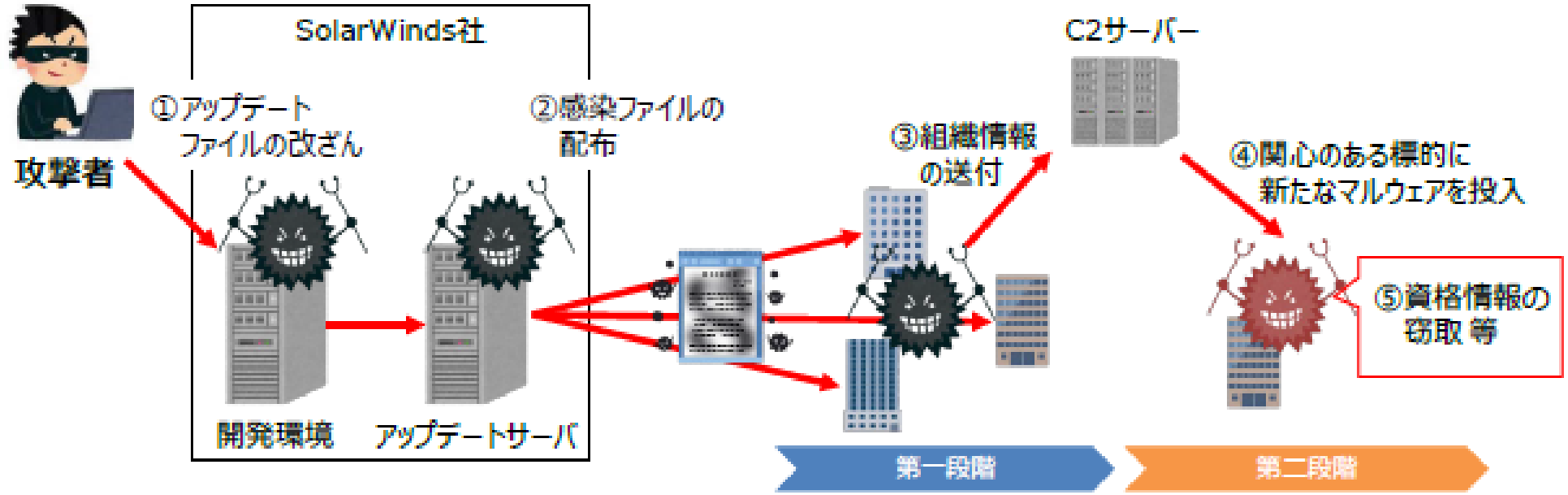


- https://cybermap.kaspersky.com/
- https://threatmap.fortiguard.com/
- https://threatmap.bitdefender.com/
- https://threatbutt.com/map/

SolarWinds Orion Platformのアップデートを悪用した攻撃

- 2020年12月13日、SolarWinds社は同社のネットワーク監視ソフトウェア「Orion Platform」に、**正規のアップデートを通じてマルウェアが仕込まれた**ことを公表。
- 攻撃は2019年9月には始まっていたとみられ、2020年3月～6月のアップデートファイルが侵害されたことで、米政府機関等を含む**最大約18,000組織が影響**を受けたとされる。
- 初期段階のマルウェアは、セキュリティサービスの検知を回避しつつ被害組織の情報をC2サーバーへ送信。**攻撃者が関心のある標的に対しては第２段階のマルウェアが投入され、資格情報を窃取。**

◆攻撃イメージ



（出典）各種公開情報に基づき経済産業省作成

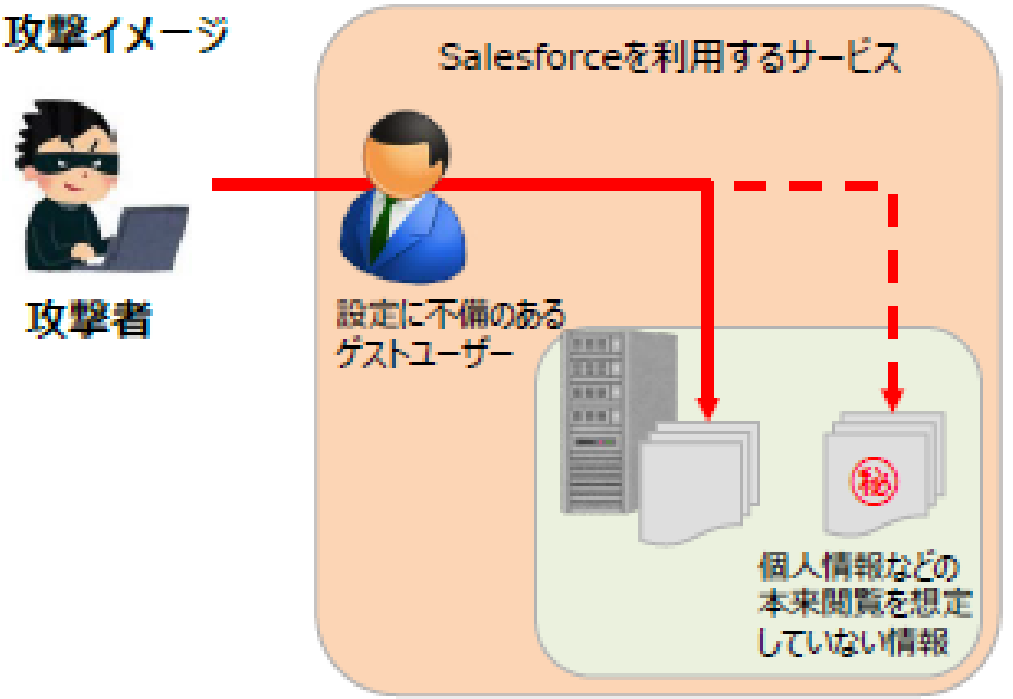
クラウドサービスの設定不備を原因とする不正アクセス

- 2020年12月25日、セールスフォース・ドットコムは、同社が提供するサービスにおける**ゲストユーザーに対する情報共有に関する設定が適切に行われていない場合、一部情報が第三者より閲覧できる**事象の発生を公表。また、**複数の国内事業者が本事象による不正アクセス及び個人情報漏えいの発生**を公表。
- 本サービスを組み込んだシステムがパッケージとして複数の顧客に提供され、同時に被害が発生したケースも。
- クラウドサービスを活用する際には、サービスの利用状況や各種設定の確認・見直しを行うなど、適切なセキュリティ対策を講ずることが重要。

◆不正アクセスがあったと公表した事業者等

- キャッシュレス決済サービス事業者
- サービス事業者
- クレジットカード事業者
- 小売事業者
- 玩具メーカー
- ガス事業者
- 地方自治体
- 独立行政法人　　他

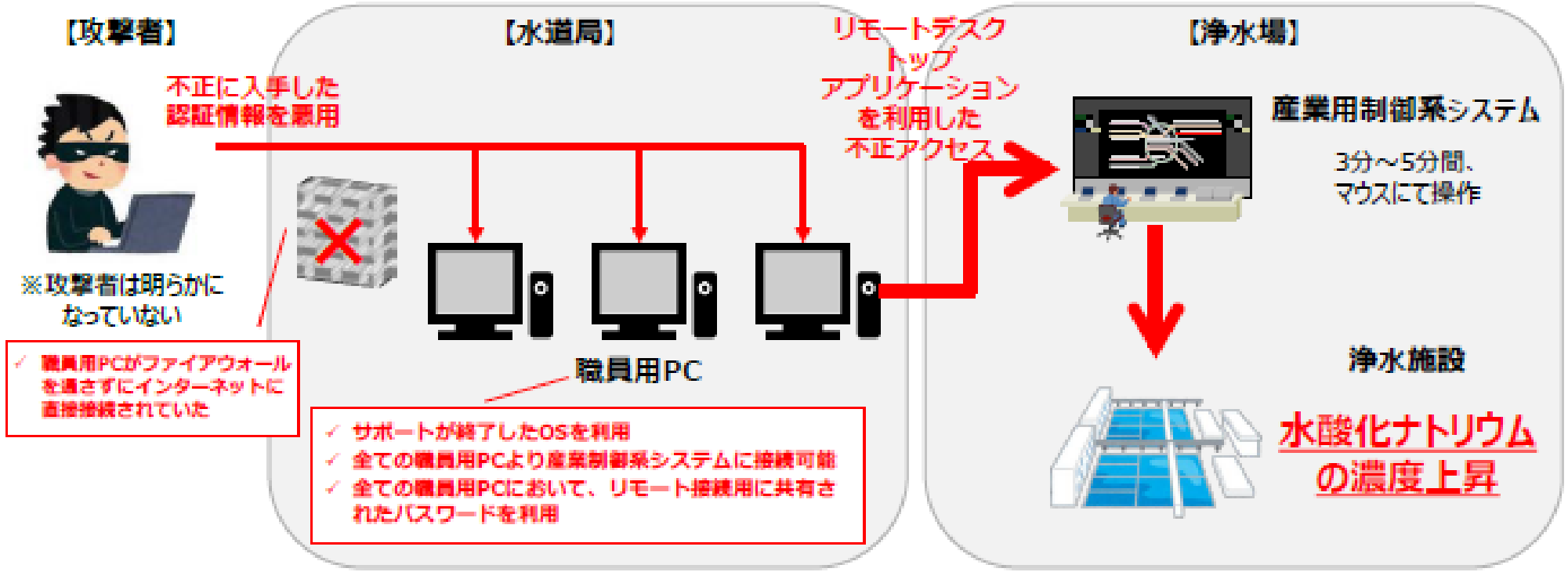
◆攻撃イメージ



（出典）各種公開情報に基づき経済産業省作成

水道システムへの不正アクセス事例

- 2021年2月、アメリカフロリダ州オールズマー市水道局は、**水道における産業用制御系システムを対象とした不正アクセス**によって、飲用水に含まれる水酸化ナトリウムの量が一時的に通常の約100倍に上昇したと発表した。なお、オペレーターが異常に気付き、即座に設定を戻したため、実際の被害はなかったとされる。
- 報道によると、**職員用PCよりリモートデスクトップアプリケーションを利用して、**産業用制御系システムへの不正アクセスが行われたとされている。

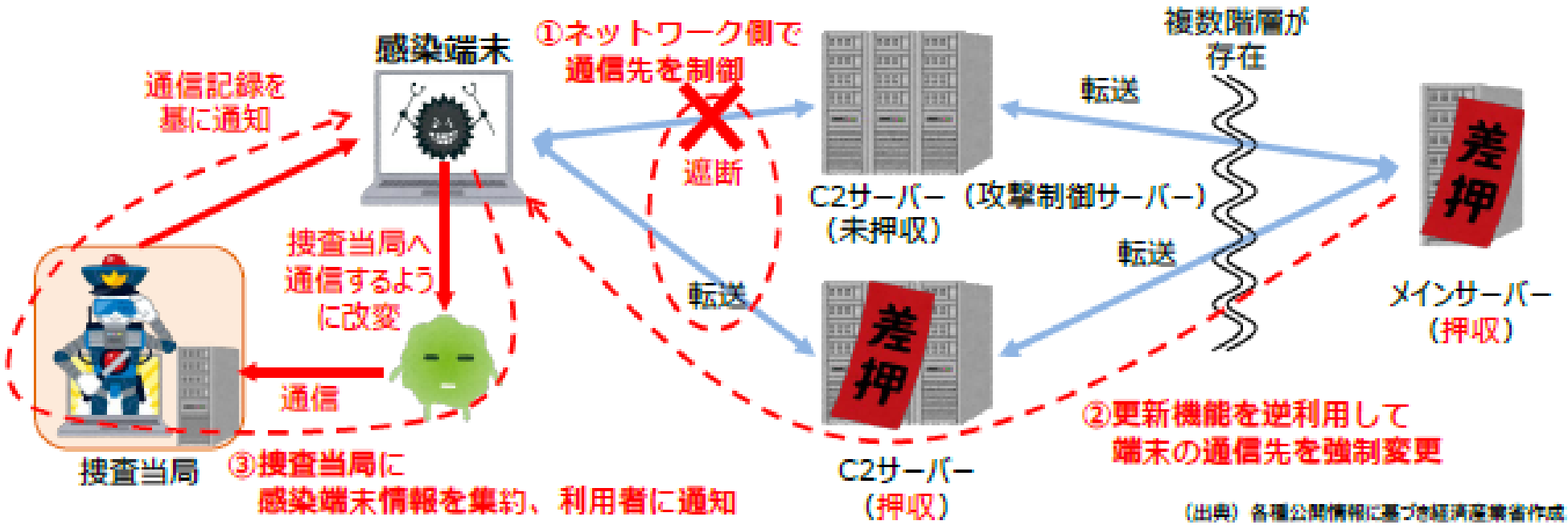


（出典）各種公開情報に基づき経済産業省作成

Emotet テイクダウン作戦（Operation Ladybird）

- 2021年1月27日、Europol（欧州刑事警察機構）は、世界的に猛威を奮ったマルウェア「**Emotet**」の**国際的なテイクダウン作戦**（サーバー等攻撃インフラの接収）が成功裏に実施されたと公表。
- 日本でも、海外の捜査当局からの情報提供に基づき、**インターネットサービスプロバイダからEmotetに感染している機器の利用者に対する注意喚起**を行うことを、総務省、警察庁、（一社）ICT-ISACの連名により2月19日に公表。
- 感染端末では、Emotet感染を原因とする認証情報の窃取・別のマルウェアへの二次感染が疑われるため、調査と対処が必要。

◆Emotet攻撃インフラとテイクダウン（赤字）のイメージ

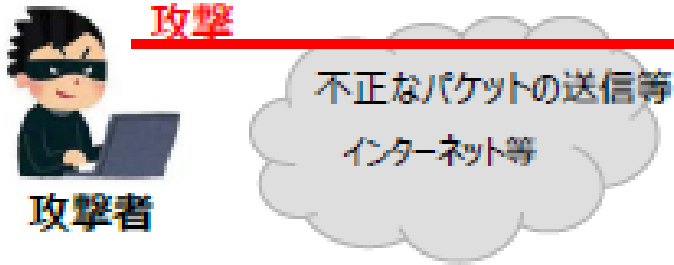


（出典）各種公開情報に基づき経済産業省作成

プロトコルスタックの脆弱性：“Ripple20”

- 2020年6月、JSOF社は、**Treck社※1が開発したTCP/IPプロトコルスタック※2「Treck TCP/IP Stack」に複数の脆弱性**があることを発表（発表年や当スタックが20年以上前から存在していること等に由来し、19の脆弱性の総称をRipple20と命名）。遠隔の第三者によって、**任意のコード実行、情報の窃取、サービス運用妨害（DoS）等の攻撃を受ける可能性**があり、最新バージョンへの更新やパッチの適用、IPパケットのフィルタリング等の対策を呼び掛けている。
- Treck TCP/IP Stackは多数の企業が製品に採用しており、数億台かそれ以上の機器が影響を受けるとされ、家庭向けデバイス、ネットワーク機器、医療機器、産業制御機器／システム、重要インフラ分野などの幅広い領域への影響が懸念される。

◆攻撃イメージ／影響範囲の例

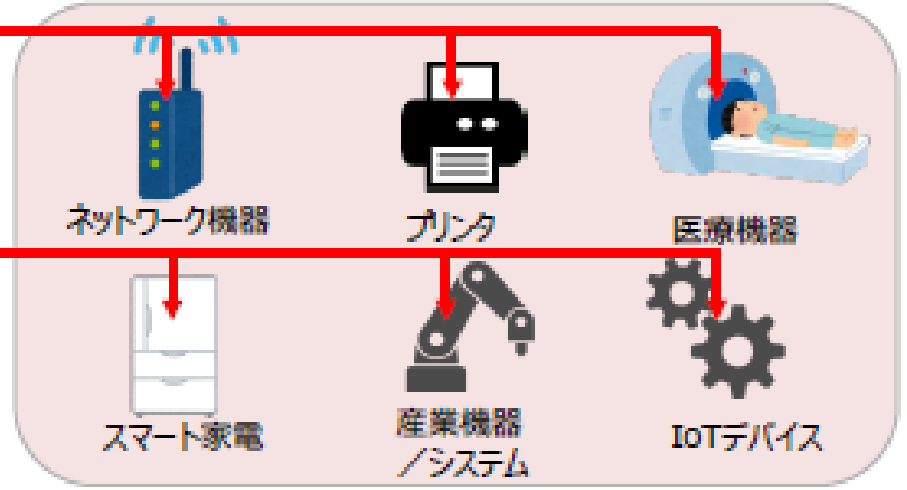


想定被害：任意のコード実行、情報漏えい、DoS

- ✓ Treck TCP/IP StackはHP社、Schneider Electric社、Intel社、Rockwell Automation社、Caterpillar社、Baxter社等の製品が採用。
- ✓ 同様の脆弱性が、関連する他のTCP/IPスタックにも存在することが報告されている。

https://www.jsof-tech.com/ripple20/

Treck TCP/IP Stackの採用製品は、下図以外にも多岐に渡る

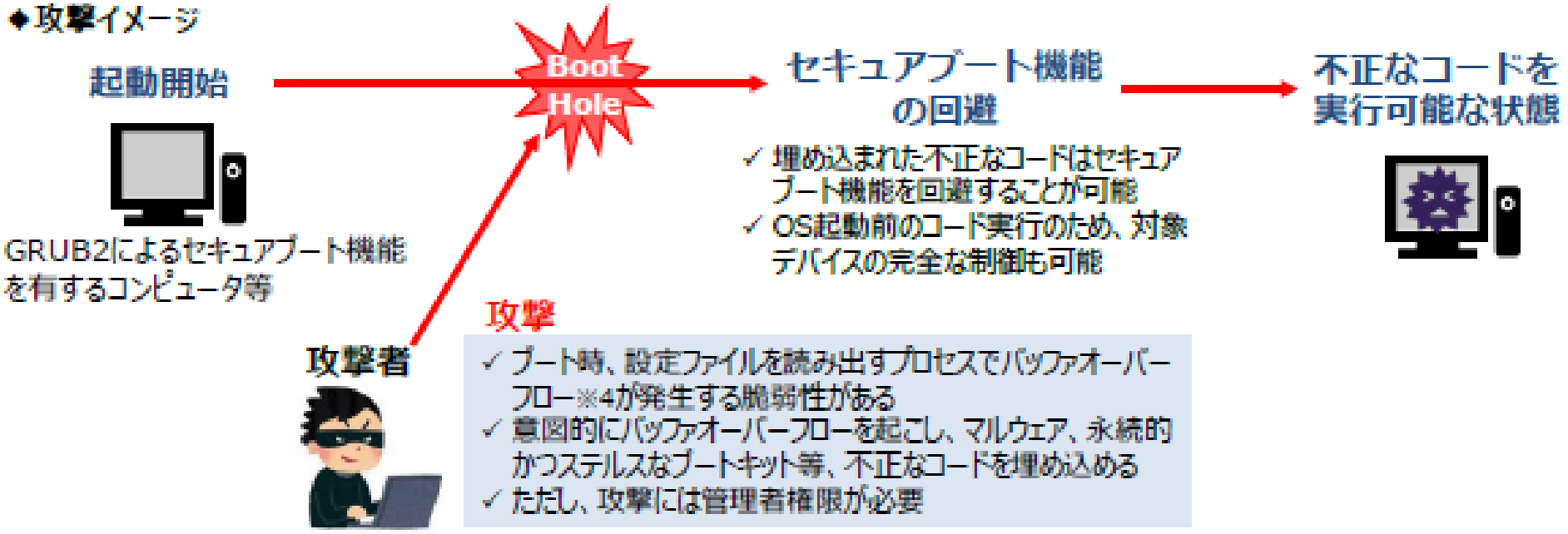


※1 組み込み機器向けのインターネットプロトコルスタックを設計・開発する米国の企業
※2 階層構造で構成されるインターネットプロトコル群

GRUB2ブートローダーの脆弱性：“BootHole“

- 2020年7月、Eclypsium社※1は、Linux等で用いられるブートローダー※2**「GRUB2」の脆弱性**（BootHoleと命名）を報告した。OSが起動する前段階において不正なプログラム実行を防ぐ**「セキュアブート機能」※3を回避**できることが確認されている。この脆弱性の悪用により、対象のデバイスが**完全に制御される可能性**がある。
- Red Hatなどの主要Linuxディストリビュータ等は、この問題に関するセキュリティ情報を公開し、対応を表明している。

◆攻撃イメージ



※1 企業向けファームウェア/ハードウェア分野における米国のセキュリティ企業
※2 コンピュータの起動直後に自動的に実行されるコンピュータプログラム
※3 OS起動前に実行されるプログラムの署名を確認することでデバイスを保護する機能
※4 データの一時記憶領域に想定以上の長さのデータが入力されてしまう現象

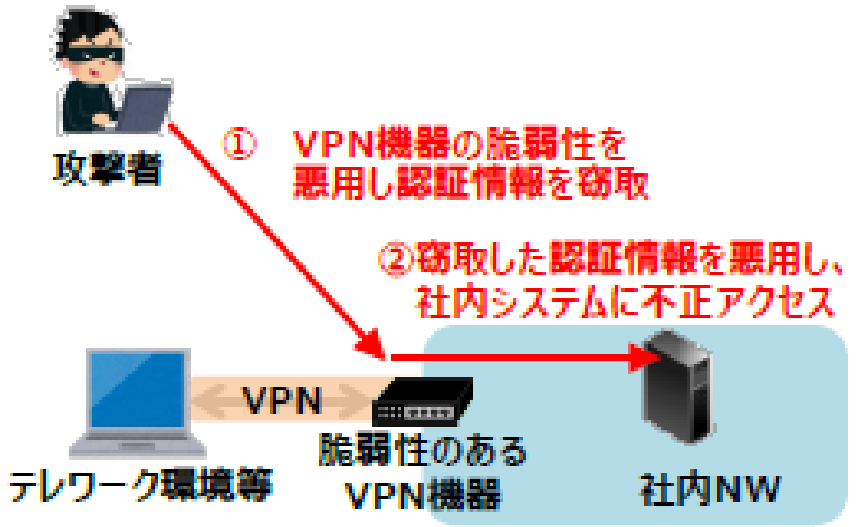
https://eclypsium.com/2020/07/29/theres-a-hole-in-the-boot/

8

VPN機器の認証情報流出

- **VPN機器の脆弱性**が相次いで報告され、そうした脆弱性を**悪用するコードが公開**されるなど深刻な状況が発生。**攻撃者はこうした脆弱性を通じて直接的に社内ネットワークへ侵入**し、攻撃を展開。
- 2020年8月、Pulse Secure製VPN機器の脆弱性が悪用され、**国内外900以上の事業者からVPNの認証情報が流出**。2020年11月、Fortinet製品の**VPN機能の脆弱性の影響を受ける約5万台の機器に関する情報が公開**。**認証情報等が悪用されることで容易に侵入されるおそれ**。
- **どちらのケースも既に悪用されている可能性**があるため、**機器のアップデートや多要素認証の導入といった事前対策**に加え、事後的措置として**侵害有無の確認や、パスワード変更等の対応**が必要。

VPN機器に対する不正アクセス



Pulse Secure製VPN機器の脆弱性

2019年4月	脆弱性情報公開
2019年8月	脆弱性の悪用を狙ったとみられるスキャンを確認
2019年9月	脆弱性を悪用したとみられる攻撃を確認
2020年8月	国内外900社（国内は38社）の認証情報が公開

Fortinet製FortiOSの脆弱性

2019年5月	脆弱性情報公開
2019年8月頃	脆弱性の詳細情報公開、悪用やスキャン開始
2020年11月	脆弱性の影響を受ける約5万台の機器情報が公開 IPアドレス、ユーザーアカウント名、平文パスワード等

https://www.jpcert.or.jp/newsflash/2020112701.html
https://www.jpcert.or.jp/at/2019/at190033.html

ビル分野のセキュリティ事故事例：病院

- 警備員による病院のHVACシステムのハッキング
  （内部犯行による空調システムへのハッキング）

日時	2009年4月～6月
攻撃対象	米国テキサス州ダラス W.B. Carrell Memorial Clinic
侵入経路	病院のHVACシステム（暖房換気空調システム）、患者情報を扱うコンピュータ等の不正アクセス
被害	**システムへの侵入、システム画面のオンライン上での公開、未遂だがDDoS攻撃の計画あり**

TimeLine	経緯・概要
（背景）	同病院の夜勤の契約警備員（当時25）は、オンライン上で“Ghost Exodus”という名前で活動し、ハッカーグループ“Electronik Tribulation Army” のリーダも務めていた。
攻撃 *2009.4-6*	警備員は同病院のHVACシステムや顧客情報のコンピュータに侵入し、HVACシステムのHMI画面のスクリーンショットをオンラインで公開。公開された画面では、手術室のポンプや冷却装置を含め、病院の様々な機能のメニューが確認できる。さらに、病院内のPCにマルウエアをインストールする（DDoS攻撃のため、PCをボットネット化したものとみられる）様子なども動画に撮って公開している。
―	一方、病院の職員はアラーム設定が停止されたことで、HVACシステムのアラームがプログラム通りに機能せず、不思議に思っていたが、内部から発覚することはなかった。
発覚・逮捕 *2009.6*	SCADAセキュリティの専門家がハッカーの知り合いからの情報を得て調査し、FBI及びテキサス州検察局に報告したことで発覚し、2009年6月26日警備員は逮捕された。（連邦刑務所への9年の禁固刑を受ける。）
攻撃計画 （未遂） *2009.7*	逮捕により未遂に終わったものの、警備員は、乗っ取られた病院のシステムを使って2009年7月4日（独立記念日）に大規模なDDoS攻撃を仕掛ける計画を立てており、インターネット上で協力してくれるハッカー仲間を募っていた。また、既に攻撃予定日の前日に辞職する旨を所属する警備会社に伝えていた。

出典：DOJプレスリリース (https://www.justice.gov/archive/usao/txn/PressRel09/mcgraw_cyber_compl_arrest_pr.html)

(写真出典) http://www.gsr-andrade.com/#!/page/118644/healthcare

その他ビル分野のセキュリティ事故事例

● 海外を中心に多くの実際の事件や脆弱性の発見事例が見られる。

時期	内容
2011年11月	コロンビア大の研究者がオフィス等に導入されているLaserJetプリンターに脆弱性があり、ハッカーからのアップデート指示により過剰な運転状態となって、最終的には発火することを証明した。対象は何百万台にもおよぶ。
2012年4月	MITの学生が同大学グリーン棟の照明システムをハッキングし、ビルの窓照明を巨大なテトリスゲームにしてしまった。
2013年8月	フロリダ州マイアミのターナー・ギルフォード・ナイト矯正センターの警備システムが何者かにハックされ、収容房の扉のロックをリモート解除し、受刑者が敵対ギャングに属する別の受刑者を襲う事件が発生。
2013年5月	米セキュリティ企業が、オーストラリア・シドニーのオフィスのビル管理システムへの侵入テストを実施し、フロア空調やエネルギーメーター、アラームといったビル管理機能への侵入を実現。同ビルの設備管理に使われているデバイスは、世界中で数十万個利用されている。
2014年12月	ドイツの製鋼所のネットワークが標的型メールによるサイバー攻撃を受け、制御システムを乗っ取られた。その結果、プラントの各所に障害が発生し、溶鉱炉が制御不能となり、最終的に停止不能となった。
2016年1月	IBMのチームが商業オフィスのBASに対するペネトレーションテストを実施し、複数のビルを遠隔のBASで管理しているようなケースにおいて、全米の複数のビルの自動コントローラに対する完全な指揮権を入手出来ることを明らかにした。
2016年11月	フィンランド南東部の都市・ラッペーンランタのビルがDDos攻撃を受け、空調や温水管理をしていたコンピュータが不調をきたし、暖房が停止した。比較的早急に回復出来たが、外気温マイナス2度の環境で、しばらく暖房を利用できない状況となった。

本当の 攻撃者は誰?

- 組織内部 ： 大半
  - 社員は信用できない。 ➔ ゼロ トラスト
- 組織外部 ： 少数
- 保守的人種 (組織の内外を問わず)
  - 新しいもの(e.g.,技術)には危険/リスクはつきもの。。。。。でも、現状を維持したがる。

まず、、、、、

サイバーセキュリティー

- ✓完全(＝安全）は、存在しない。
- ✓暗号化技術は、超難しい数学を使うけど、結局は、「いいかげん」(＝安心）

セキュリティー の 「大枠」

- 何が問題なのだろう?
  - 「個人情報保護法」、「青少年ネット規制法」
  - 「組織犯罪処罰法」 (=「共謀罪法」、「テロ等準備罪法」
- 「(情報)セキュリティー」はどうあるべきか?
  - 安心してのびのび仕事ができるような環境

    (*) 事故が起こらないように 委縮した活動環境?

    (*) 実は、職場の「安全衛生管理」と同じ。
  - 「放蕩」と「箱入り」、どっちが強い? ➔ 公助&Firewallの罠

    ➔ 意図的な“Diversity”環境の構築

本当の 攻撃者は誰?

【注意が 必要な 常套手段】

◆ たくさんの 攻撃 に 遭遇 しています。
1. 攻撃者 を 発見 したいですよね。
2. 攻撃から 守って あけますよ。
3. 僕を信頼して 中身を 見せれば安心ですよ。

◆ 「繋いでいない」から 大丈夫です。
1. だから、セキュリティー対策は 不要 です。
2. でも、繋がるかもしれませんよね。。。
3. システムをアップデートすると 保証 できなくなってしまいますよ。
   （脅し、恫喝）

How do you think?

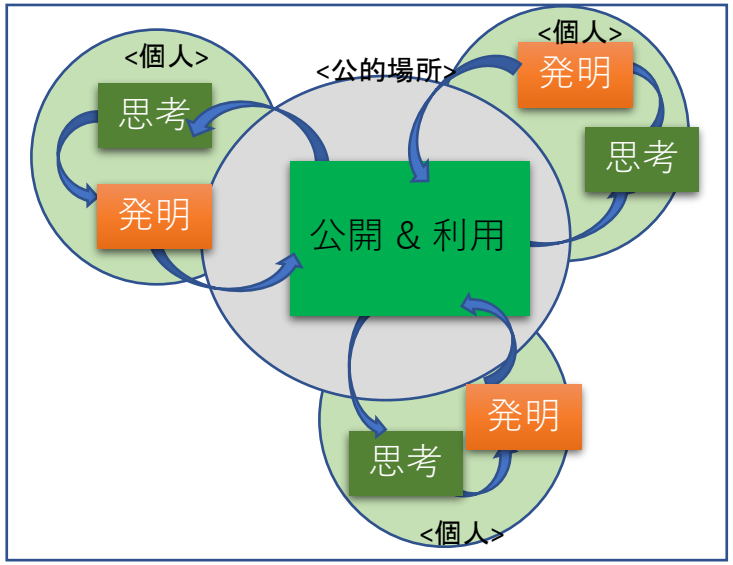
『項羽と劉邦』（司馬遼太郎著)より

。。。。やがて華何が死に、曹参は後任を命ぜられた。 彼は、斉の丞相の職を後任に譲るとき、
「それでは、斉の獄市を貴官にお渡しします」
と言った。 獄市とは商品の市場のことである。むろん、この時代といえども政治は多岐にわたっており、獄市のみではない。 後任者は不審に思い、政治にはほかにもっと大事なものがあるのではないでしょうか？ と反問すると、
「獄 と 市だけが、政治の要です」
と、曹参は言った。 曹参の考えは、牢獄も商業の場も、善悪ともに 受け容れるところです、これに対して**為政者が善悪に厳格でありすぎると、かえって ぐあいが悪くなります**、ということであった。

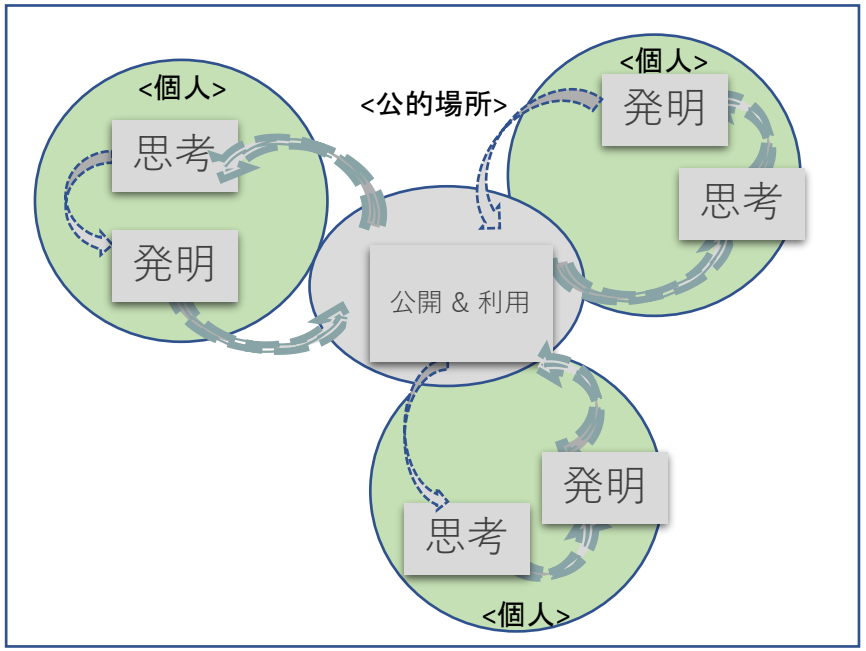
・・・(略)・・・・

曹参は、世の中には必ず姦人とい者がいる、という。 これをやわらかくつつむのが、曹参の社会に対する生理学的な認識のようであった。 そういう姦人たちは、司法の対象になるか、市場管理の対象になるかどちらかだが、この獄と市をあまり やかましく正しすぎると姦人は世に容れなくなり、必ず乱をおこし、国家そのものを毀損することになる、だから獄市は大切だと言ったのです、曹参は答えたと いう。

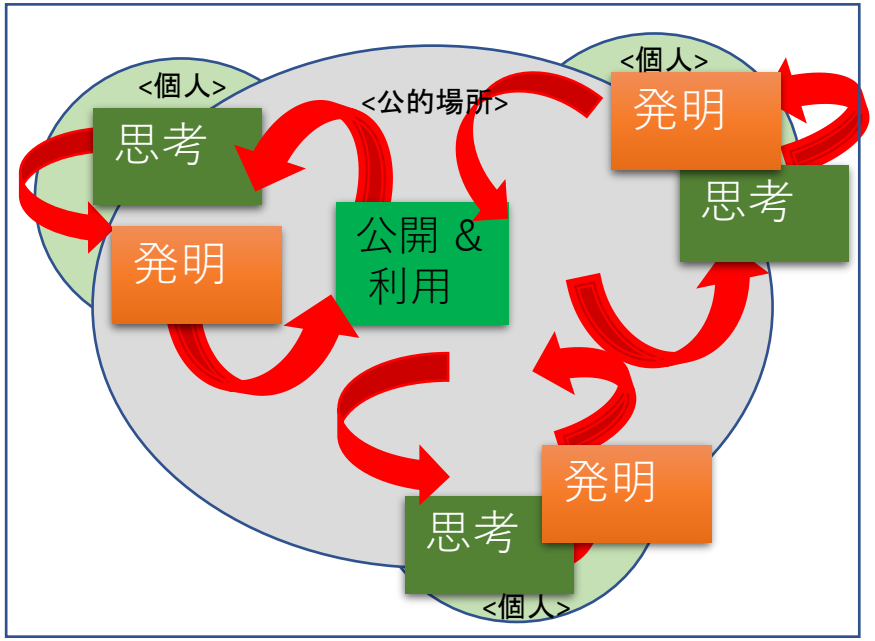
著作権・知的財産権の規制もしかり



過度な規制



交流の
促進



“セキュリティーの(正しい)意義

【誤】： ① 風紀委員の増強
② 安全(“ゼロ”)の実現

【正】： 1. “安心”(not 安全)の実現
2. “のびのびと”仕事
3. リスク対応(“non”-ゼロ)

“セキュ

つまり
「自分で守る」が基本&前提
👇
ゼロ・トラスト
(Zero Trust)

【誤】

② 安全 ゼ 実現

【正】： 1. “安心”(not 安全)の実現
2. “のびのびと”仕事
3. リスク対応(“non”-ゼロ)

暗号化・認証アルゴリズム

[1] **平文の認証** (一般には、N bits(入力) → m bits(出力),　N>m)

[2] **平文の暗号化**

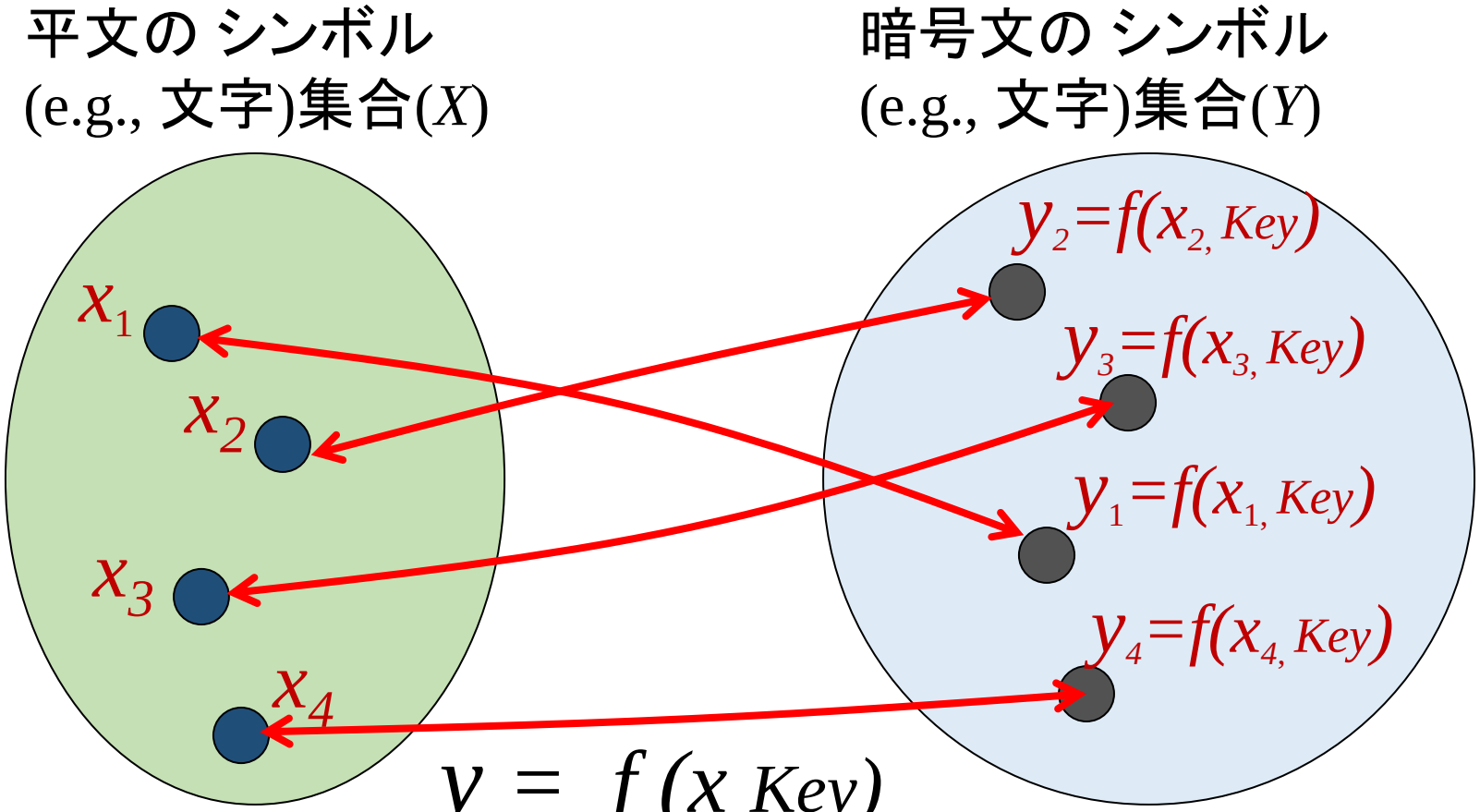
① DES(Data Encryption Standard) ; **秘密鍵方式**

② RSA(Rivest, Shamir, Adleman) ; **公開鍵方式**

さて、暗号化は何をしているのか。

- {平文}文字の空間 と {暗号}文字の空間 との間の 写像 {F(文字,パラメータ)} の計算を行っている。 {暗号}文を、**たくさん 眺めると**, 文字 と パラメータ が **見えてくる**。
- この {たくさん} が、**十分に大きければ よし とする**。
- {暗号}文の空間 も、写像関数F もほぼ、無限に存在する。
  - 写像関数 F の 逆関数 F^{-1} が存在すること
  - 必ず、1対1 の写像となっていること。

(*) 認証では、この条件が少し甘くできる。



$$y = f(x, Key)$$

$$x = f^{-1}(y, Key)$$

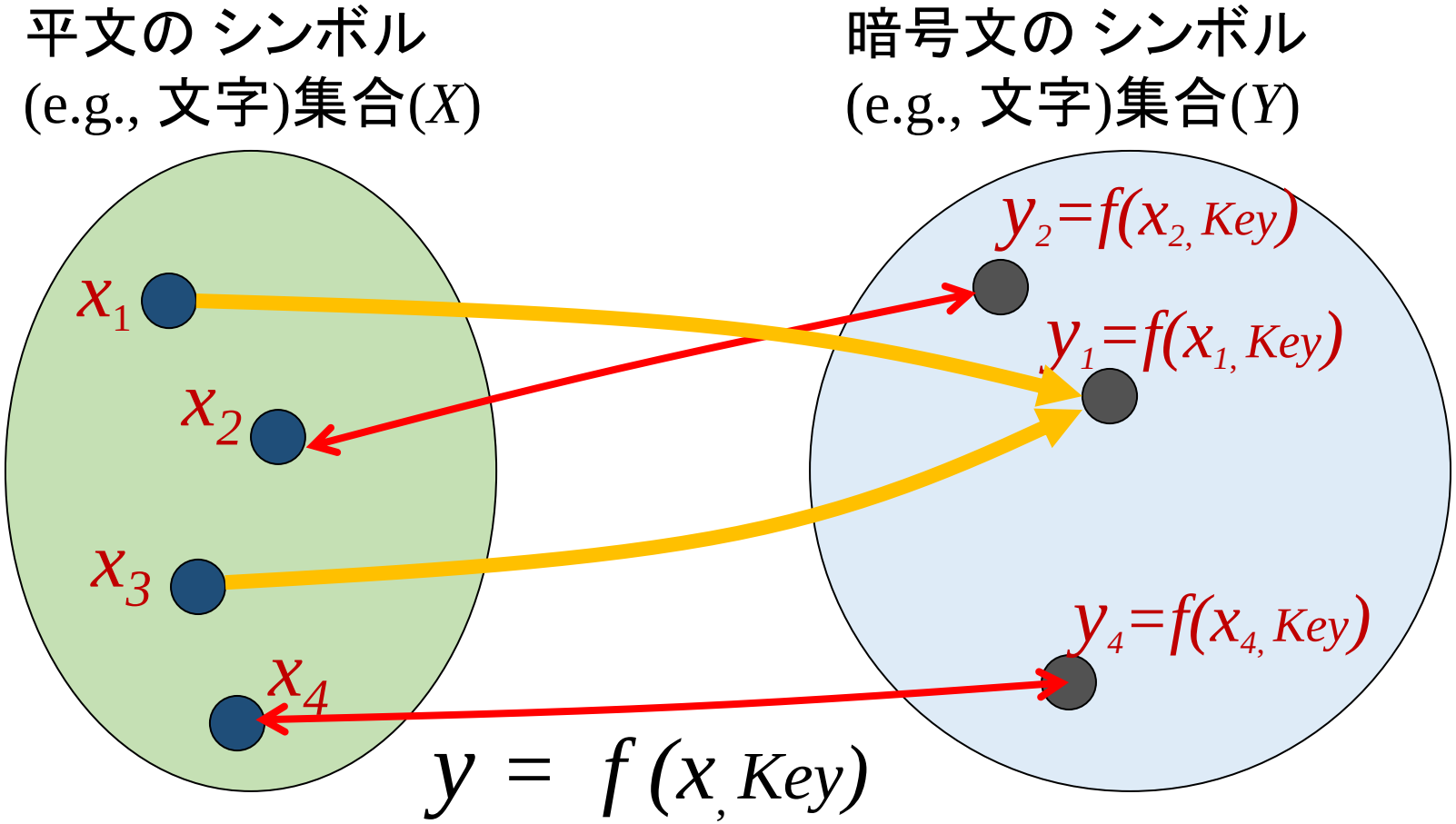
(*) 必ず、

(i) f^{-1} が存在

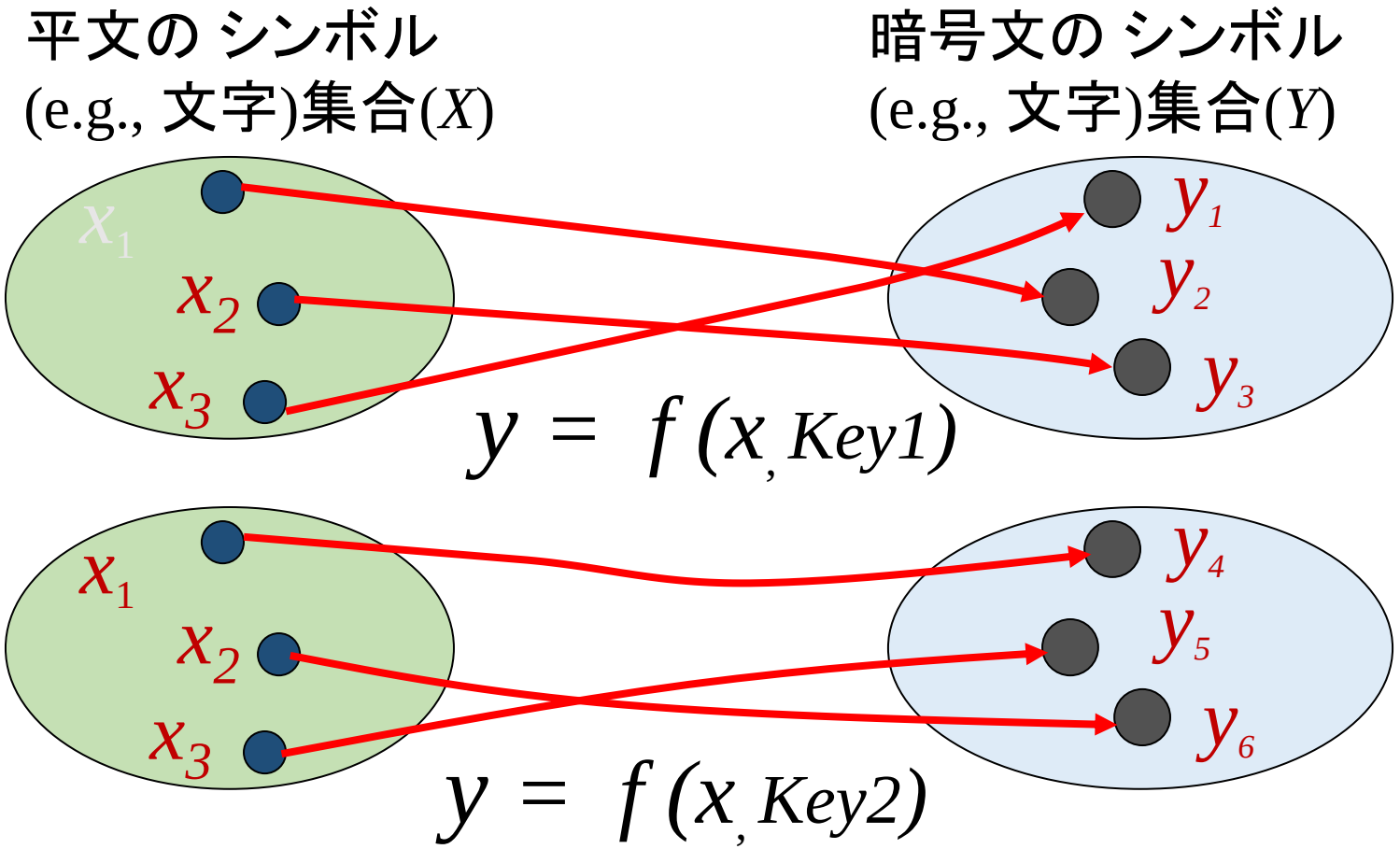
(ii) 写像は、1対1

$y_1 \neq y_2$ for any x_1, x_2

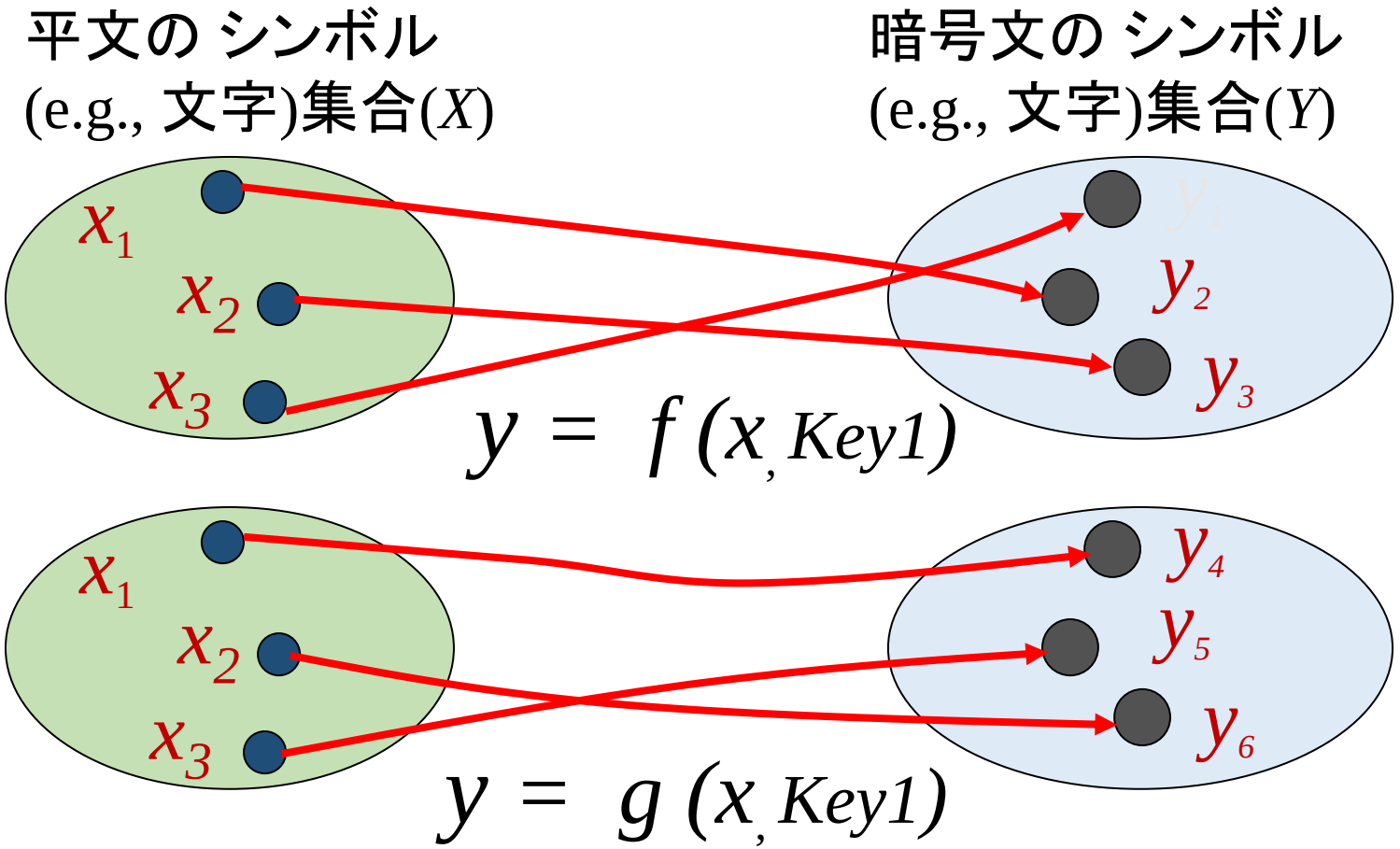
(*) *Key* は、ユーザごとに定義



(*) もし、同じ y に写像される x が複数存在すると f^{-1} が存在しない
ことになり、暗号の 複合 ができなくなる

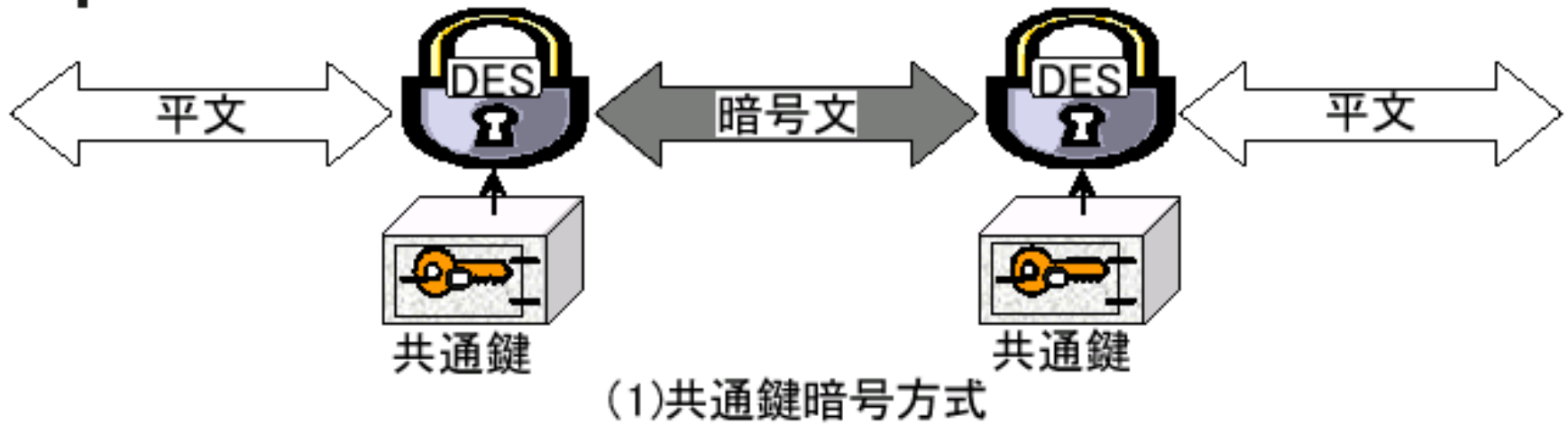


(*) 同じ写像関数(暗号化関数) f を用いていても、関数 f のパラメータである Key が異なれば、写像値(暗号文のシンボル)も異なる。

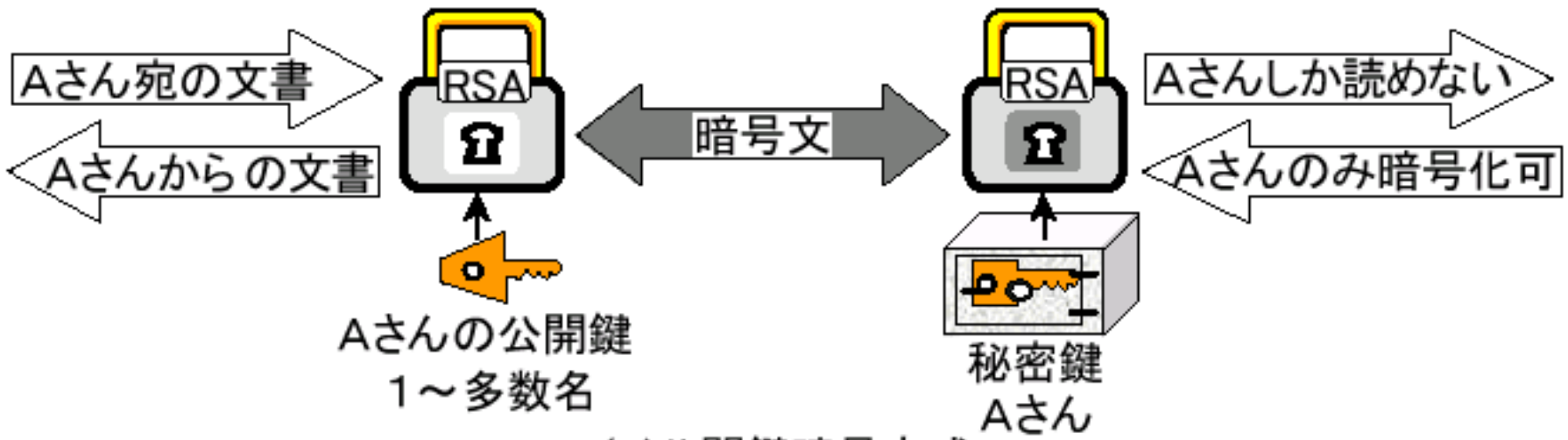


(*) 仮に、暗号化に必要な 鍵 (key) が同じでも、異なる写像関数 (暗号化関数) を用いることも可能。

暗号方式の種類

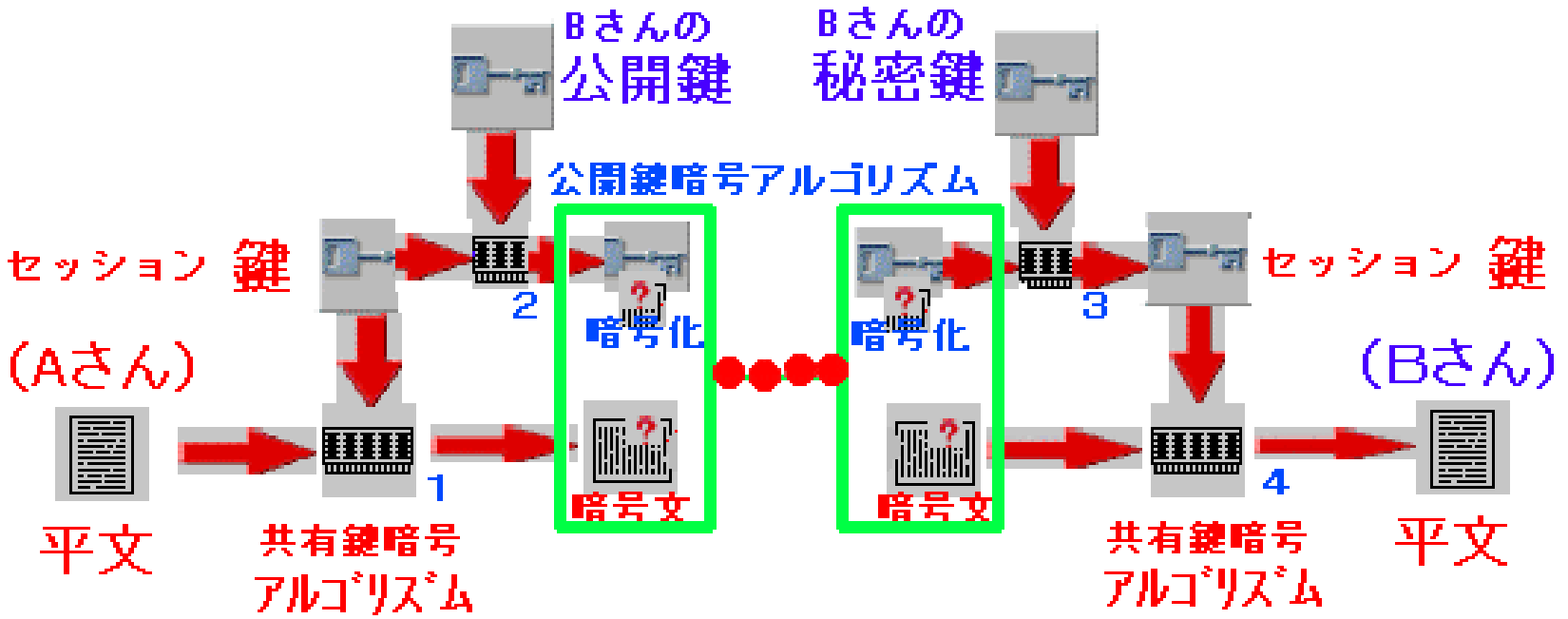


(1)共通鍵暗号方式



(2)公開鍵暗号方式

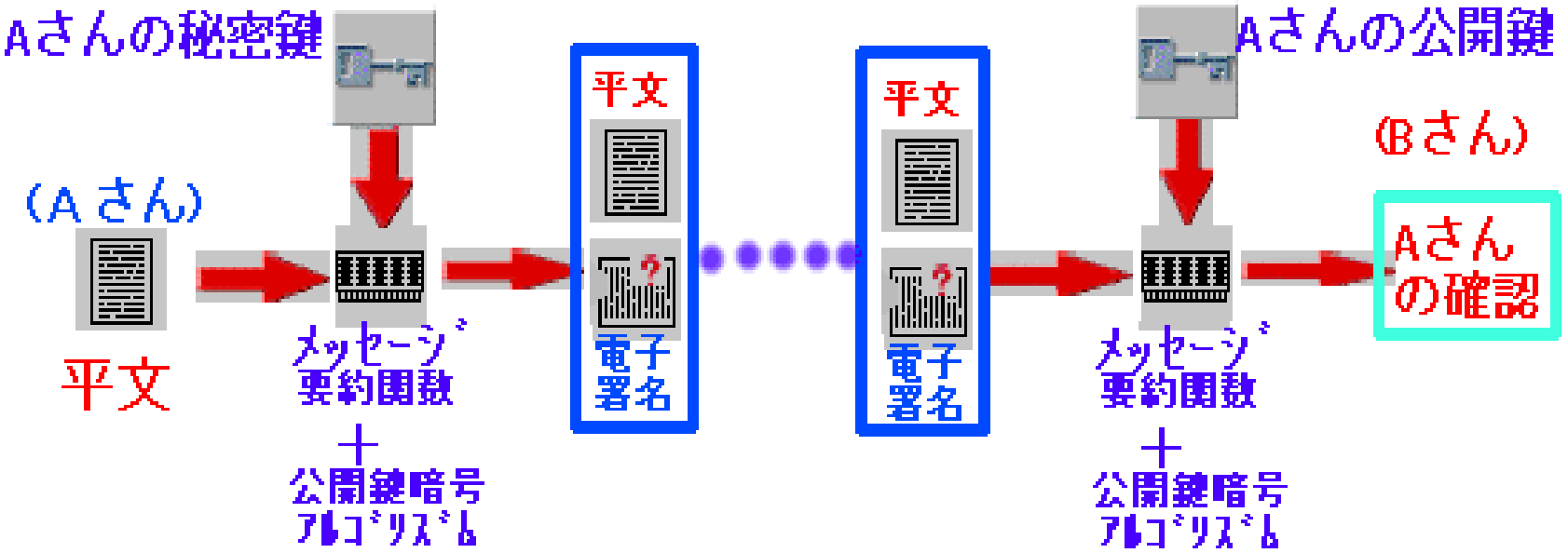
公開鍵暗号方式



例；ssh (Secured Shelll)

出典: http://www.psn.or.jp/TROUBLE/security.html

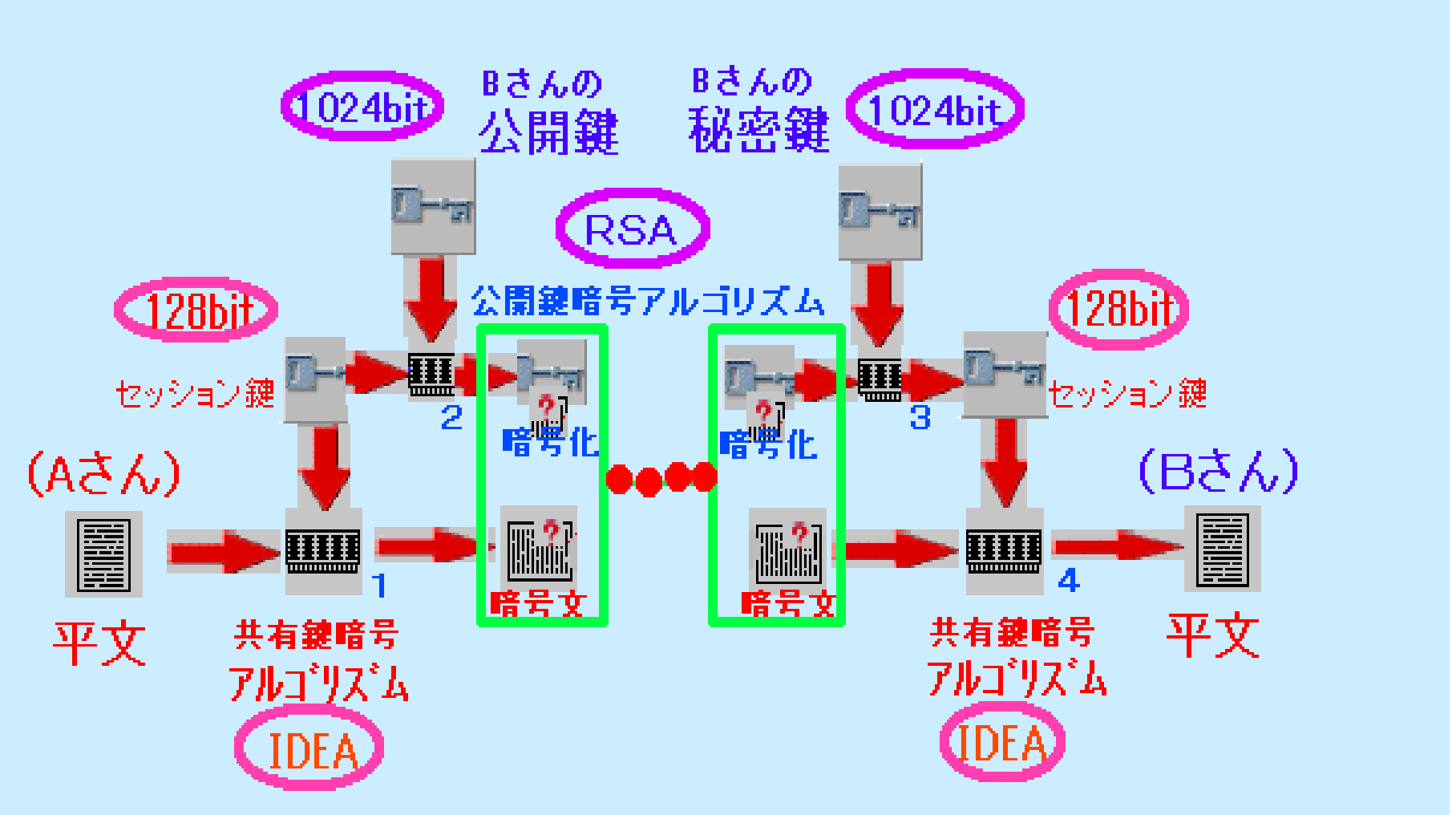
電子署名方式



- MD5 (128 bits)
- SHA (160 bits)

出典: http://www.psn.or.jp/TROUBLE/security.html

PGP暗号化メール



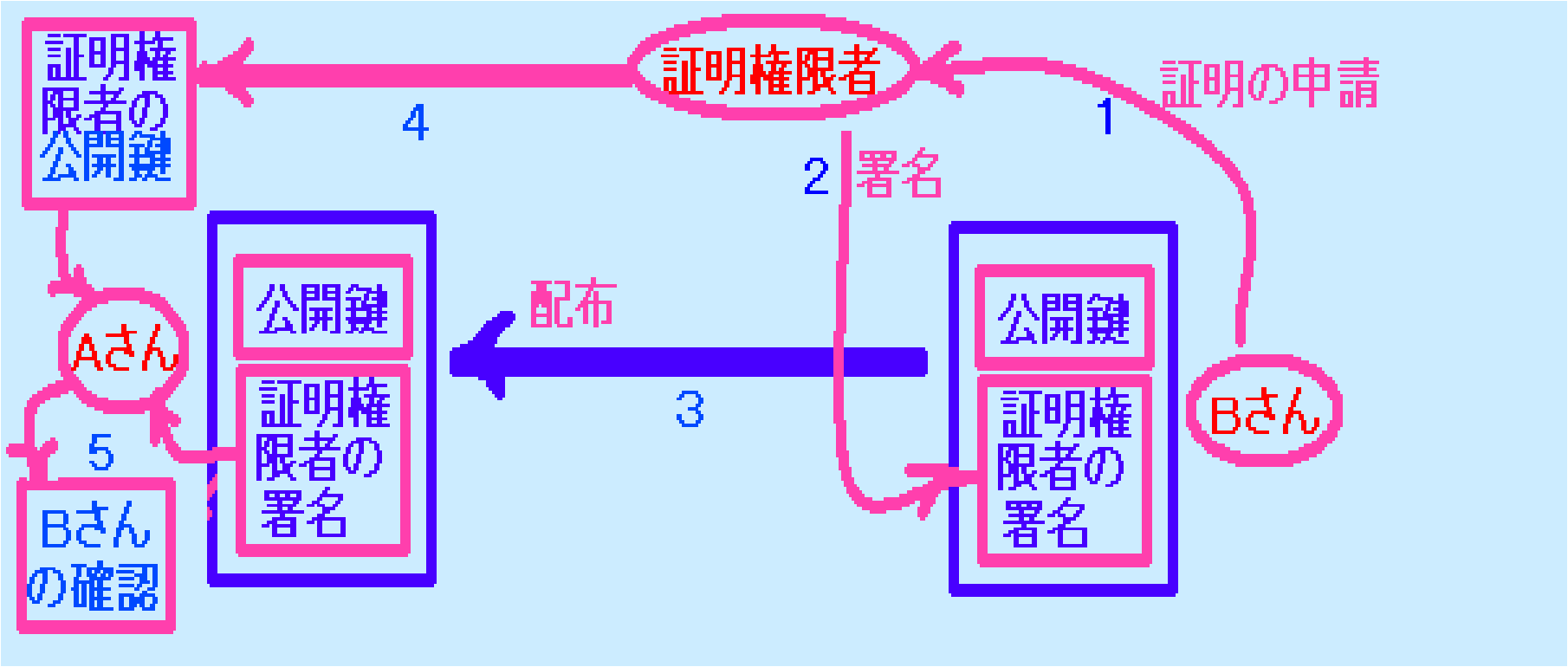
公開鍵暗号化方式(暗号化方式；RSA+IDEA)

出典: http://www.psn.or.jp/TROUBLE/security.html

鍵の証明と配布

・証明を発行する機関・組織を利用

例；Netscape Webサーバ 暗号化情報交換



出典: http://www.psn.or.jp/TROUBLE/security.html

資料提供：
TORUS 代表取締役社長 木村 幹夫 氏

インターネットの裏方
公開鍵方式

私達が仕事でネットを使えるのは
重要なデータのやりとりができるから

TORUS

重要な文書を安心して注1送れるのは
他人にのぞき見られたり
改ざんされたりしないから

注1：安全ではないが安心

TORUS

暗号化技術が
インターネットを支えている

どんなハイテクが使われているだろう？

TORUS

その前にちょいと歴史を

人類は古代〜１９７０年代まで
暗号のときは一種類の鍵でやりくりしていた

「暗号化が大変なら
複合化も同じくらい大変だよね！」
という発想

できるだけ難しい暗号化を考えよう

でも
暗号の鍵と復号の鍵は同じ

どんなに難しい暗号の仕組みを作っても・・

それって

1.送受信の間に鍵を盗まれたらアウトだよね

2. たくさん通信したら鍵がばれやすくなる 注2

注2: 鍵を毎回変えればいい。毎回違う新しい鍵(Key-2)を鍵(Key-1)で知らせれば、はるかに ばれにくくなる。

暗号以前に
鍵をどうやって安全に受け渡しするか？
の根本問題があった

TORUS

何とかならないのか？

TORUS

鍵を分けたら良いんじゃないか？

「暗号化だけ」できる鍵
&
「複合化だけ」できる鍵

書類を送ってもらう相手に
「この鍵で暗号化してね」
とメッセージする

暗号化した文書が送られてくる

解読できるのは
複合化の鍵が分かっている自分だけ

TORUS

復号の鍵は送らないので、
文書と暗号鍵が盗まれても、解読されない

TORUS

なかなかCOOL！

でも、暗号化の鍵から
複合化の鍵が類推できるなら
意味がないよね？

暗号化できる鍵を知っていても
複合化できる鍵が作れなかったら良いはず

TORUS

そんな都合の良い話があるのか？

TORUS

実はあった！

TORUS

これは何と何のかけ算でしょう？

659034559238322856095735929958183900010927490610419750233482165335776109951761306403675261951798419420935094109639541495275099522919133770168486933189068146990122638578681299724765786565109905538811117702587975850622614126727499487983786842423997021756695295025796649478667676404148103987626858941473356361788586236135672196552610020609265955149053426632114506705895083471047870101868583556307814843666327903699778617915563355578695800173273721038751403858104606550440301478455725610068184364896003720453626412783077489676217576497167936071279345055214835626603055023099638880769303507014025608886118587281648134837016434816928387584508117909444519620291361953315564941839065903698573664980615465313228356506045226628690844289616409404902047153870357570926608886305369101757419970753453065490075267186190015713949569023134120195069556224421918840251752047137476642488289261842750842716747427383529023842769107104409539201621874543017935433181379867168752703712683141770310524994307667356988327004423732386199971777413394221463720977807998802523366723615609282666143427913434093078378117599039754843645916071464436281955016155687395 3

注3: RSA暗号

こたえ[注3]

素数p

313991399371199131139799331911377147529895941991587879456361416793343797754289852575517133312684269943695978946644516863648961536981354977375935673418795287369494189373478623641239162919379269294319941871985794933399739235523691657154837889117834232678974449658279117129522895488222612449716435651112797868118722475112367318718359954332756851152845673554343833423958324129279242571543956244312159149656971499164148747227159798119915531789396889314926554998567389189177184378411356887579966732519395769634484946484155736859195773976485587598811713196922772648319742413259665798111566314845954551344321292792178583218155711143611735499324729469232679643212644511755544726594454683193623626957711324895114496128478896375157597659974246467315936911531792288239249136494329788845728831611728857639343337449493221561738959339141347119138332653219119612984163669317356624631952956188127648784846583361813646131913157456632928169513747231224138425962243343371145487745954412587484837933238642278851955148574512595199696856122454391187376263997421961437425778191179173199799997773713113719997933931

と素数q

209889366574405864861512642566102225938639 21

この組み合わせを見つけるのは

計算機を使っても大変！
何年もかかる

桁数を増やせば、1000年でも解けない
問題を作るのは簡単だよね

TORUS

元ネタの２つ（pとq）を知っていれば、
かけ算（p*q）は一瞬
でも結果(p*q)から元ネタ(pとq)の推定は
膨大な時間を削られる

TORUS

かけ算の結果 ＝ 暗号鍵

元ネタ ＝ 復号鍵

になっていれば良い

かけ算の結果 ＝ 暗号鍵～（公開鍵）

元ネタ ＝ 復号鍵～（秘密鍵）

TORUS

任意の素数を２つピックアップすれば良いので
鍵は即座にいくらでも作れる

素数は無限にある

TORUS

その鍵を解くのは膨大な時間がかかる

計算機を使っても追いつかない

だから公開鍵方式は安全[注4]

注4: 正確には安心。計算機の能力(e.g.,量子コンピューティング)が向上すると暗号が解読される確率が大きくなる。

プライバシー

“Privacy by Design” (PbD)
7つの原則

1. リアクティブ(事後)でなくプロアクティブ(事前)
2. ディフォルト設定でプライバシー保護
3. 設定時に組み込むプライバシー対策
4. ゼロサム ではなく ポジティブサム
5. エンド・ツー・エンド でのセキュリティー
6. 可視化と透明性 (オープン性・公開性)
7. 個人のプライバシー尊重(個人を主体に)

プライバシー

- (他人からの干渉を受けない個人の)私生活
- 秘密, 内密
- 隠遁(いんとん)
- Private の名詞形

かなり、主観的な面が強い。。。。。

➔ 結局は、セクハラと 類似している

プライバシー

- 結局は、セクハラ、アカハラ、ドクハラ と似ている。
- 同じ事象でも、問題がない場合と、問題になる場合とがある。
- 基準も時間とともに 変化していく

- 必ず 騒ぎ出す ひとがいる。
  - 最初から話をするべし！

プライバシー

- 対策は、責任の所在を明らかにすること
- 基本的には、個人で守るしかない。
- しかし、個人情報に 合った サービスを 顧客はありがたがる。

  ➔ 情報の 2次流通、2次利用に関するコンセンサスの必要性

プライバシー

- 結局は、誰が(Who)、何の目的で(Why)、どのように(How & Where)、いつ(When) 個人の情報を利用するか。
  ➔ なんだ、5W 1H じゃん。
- デジタルもアナログも基本的には同じ。デジタルは、情報の流通速度を大きくする。

プライバシー

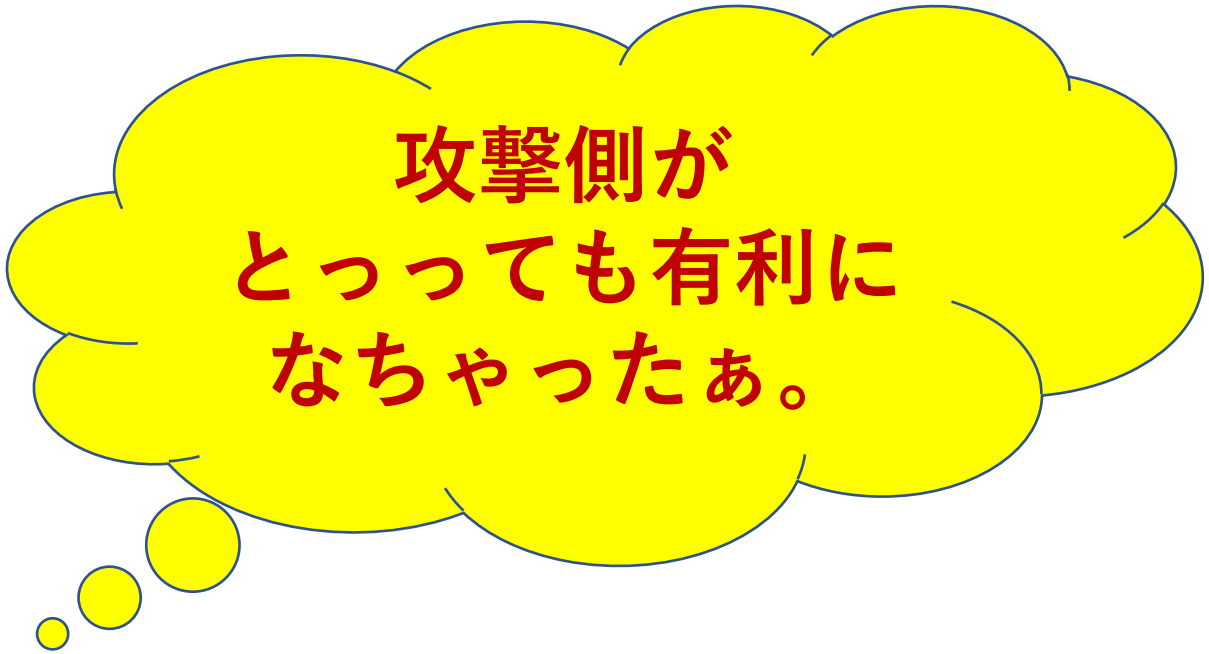
- ところが、、、、、、
  - 情報理論：

    秘密の情報(＝貴重な情報) ほど、高い価値がある。
  - プライバシー

    公然の事実（＝普通の情報）になると、プライバシーではなくなる

プライバシー

- デジタル技術の効果
  - 複製
  - 流通
  - 共有
  - 加工



が **簡単に できちゃう !!!**

ところで、通信の秘匿性

- 何のために、秘匿性が必要なのだろう？
- 「匿名性」 は 何の ために必要なのだろう？

- なぜ、「ブロッキング」 は、大きな 議論の対象になるのだろう？

セキュリティー

セキュリティーと辞書で引くと

- 安全, 無事
  - public ～ 治安, 公安/in ～ 安全に, 無事に.
- 安心, 心丈夫
- (財政上の)安定, 保障:⇒social security.
- 〔危険・危害などに対する〕防衛(手段), 警備(態勢), 安全保障〔against, for〕

- (負債の支払いに対する)保証, 担保, 抵当(物件)、保証人、有価証券:government securities 政府発行の有価証券(e.g., 国債・公債)

安心

- 個人の**主観的**な判断に大きく依存するものである。当懇談会では安心について、人が知識・経験を通じて予測している状況と大きく異なる状況にならないと信じていること、自分が予想していないことは起きないと信じ何かあったとしても受容できると信じていること、といった見方が挙げられた。 人々の安心を得るための前提として、安全の確保に関わる組織と人々の間に信頼を醸成することが必要である。互いの信頼がなければ、安全を確保し、さらにそのことをいくら伝えたとしても相手が安心することは困難だからである。よって、安心とは、安全・安心に関係する者の間で、**社会的に合意されるレベルの安全を確保しつつ、信頼が築かれる状態である。**完全に安心した状態は逆に油断を招き、いざというときの危険性が高いと考えられる。よって、**人々が完全に安心する状態ではなく、安全についてよく理解し、いざというときの心構えを忘れず、それが保たれている状態こそ、安心が実現しているといえる。**

2004年4月 文部科学省
「安全・安心な社会の構築に資する科学技術政策に関する懇談会」の報告書

安全

- 人とその共同体への損傷、ならびに人、組織、公共の所有物に損害がないと**「客観的に」**判断されることである。 ここでいう所有物には、無形のものも含む。世の中で起こりうる全ての出来事を人間が想定することは不可能であり、安全が想定外の出来事により脅かされる可能性は常に残されている。そこで、リスクを社会が受容可能なレベルまで極小化している状態を安全であるとする。同時に、社会とのコミュニケーションを継続的に行う努力をすることにより、情勢に応じて変動しうる社会のリスク受容レベルに対応する必要がある。 **安全を高めようとすればするほど、利便性や経済的利益、個人の行動の自由等が制約され、プライバシーが損なわれる可能性がある。**よって、安全性を向上させる際には、このようなトレードオフの関係を考慮する必要がある。しかしながら、より高いレベルの安全を実現するためには、安全と自由のトレードオフの次元にとどまらず、安全性と行動の自由やプライバシーを並立させる努力を続けることが重要となってくる。

2004年4月 文部科学省
「安全・安心な社会の構築に資する科学技術政策に関する懇談会」の報告書

セキュリティー

- 経済面での観察
  1. (とても) お金がかかる
     さらに、、、、、完全性が保証できない。。。。

  2. お金を払いたがらない
     事故が起こらないと必要性を認識しない。
     ➔ 適度な事故が発生しないとお金を払わない。

- ということで、セキュリティーのビジネスが成立する社会
  ＝**“適度に”**事故がある

セキュリティー

- 必要なこと / 実現すべきこと

守るべき物に関して、以下の2つを実現。

1. **破壊・変更(改竄)**されない
   再生コストを必要とする事故

2. **盗難**されない & **不正利用**されない
   a. 再生コストを必要する 事故
   b. 他人のコスト負担を必要とする事故

セキュリティー

- 可能な対策
  1. (修復)コストを保障/補償（保険, insurance)
  2. 事故を未然に防ぐ (Pro-active)
  3. 事故発生時の対応策
     a. 事前策 (Pro-active)
     b. 事後策 (Re-active)
- 注意すべき点
  1. 完全性は要求不可能
     (*) 事前、事後の両面において
  2. “完全な” 社会ではセキュリティービジネス自体が成立しない

セキュリティー

- 例えば、デブ への生命保険
  - 困ること
    - 死亡 ➔ 収入の喪失
    - 疾病発生➔治療費

    (*) つまりは、収入の減少
  - 保障すべき物
    - お金
  - 最近保険会社がやっていること
    - デブにならないようにする方法を奨める ⬅ **ここに AI と IoT**

      i.e., 予防策の伝授 ➔ 保険利用の確率を下げる。

さらに、「法律で規制」に。
By 効果があることを証明
（導入で事故が減少する）

昔の先例：
火災報知器 と スプリンクラー 消火器
with 火災保険

最近の例：
自動車 衝突回避機能、次に 運転記録
with 損害保険

いろいろなモデルがある。。。。

- オープンソース至上主義
  - すべての ソフトウェアは、オープンソースでなければならない。
  - ブラックボックス VS オープンソース
    (Microsoft)　　(Linux)

  - ブラックボックスがないとセキュリティーは実現不可能
  - ブラックボックスだと、手が出せない
  - オープンソースは、誰でも設計図を持ってしまう。

ところで、対策コストはどうなる？

- 手段の共通化共有化が容易
  - 陽： 守りの強さを常時同レベルに維持可能
  - 陰： 攻めるための情報

- 鍵&方法 の複製コストが低下
  - 陽： いつでも、鍵&方法は変えられる
  - 陰： 鍵&方法は すぐに 複製され流通可能

プライバシー vs セキュリティーの事例

- 尖閣諸島中国漁船衝突映像の流出事件 ： 2010年11月4日
  1. 海上保安庁 石垣海上保安部 が録画
  2. 沖縄地方検察庁が映像ファイルを保管。
  3. 神戸市内所属の海上保安官が、映像ファイルをダウンロード、インターネットカフェから本画像ファイルYouTube に をアップロード。
- 問題点
  - サイバーセキュリティ対策
    1. 不十分な 海上保安庁、検察庁の サイバーセキュリティ対策 ← 安全な環境との認識？
    2. 当該 保安官 の 逮捕を 見送る。
    3. 保安官を配置換え、2010年12月に 「退職願」を提出。
    4. 「退職願」は一旦拒否されたが、その後 「受理される」 !! ← 甘すぎる処分
  - プライバシー確保
    1. 新聞記者は、取材先の 秘匿性を守っていた。 ← 表現の自由の確保のための原則
    2. ISP/CSP が、 **{令状なしに}** アクセスログ情報を提供した。。。。← 通信の秘匿性

具体的な 対策

ネットワークセキュリティー

実世界の“価値ある”情報がネットワーク上に存在し交換流通されている。

→ 情報の保護、利用者の認証、不正アクセスの防止
　サービスの継続

保護対象;

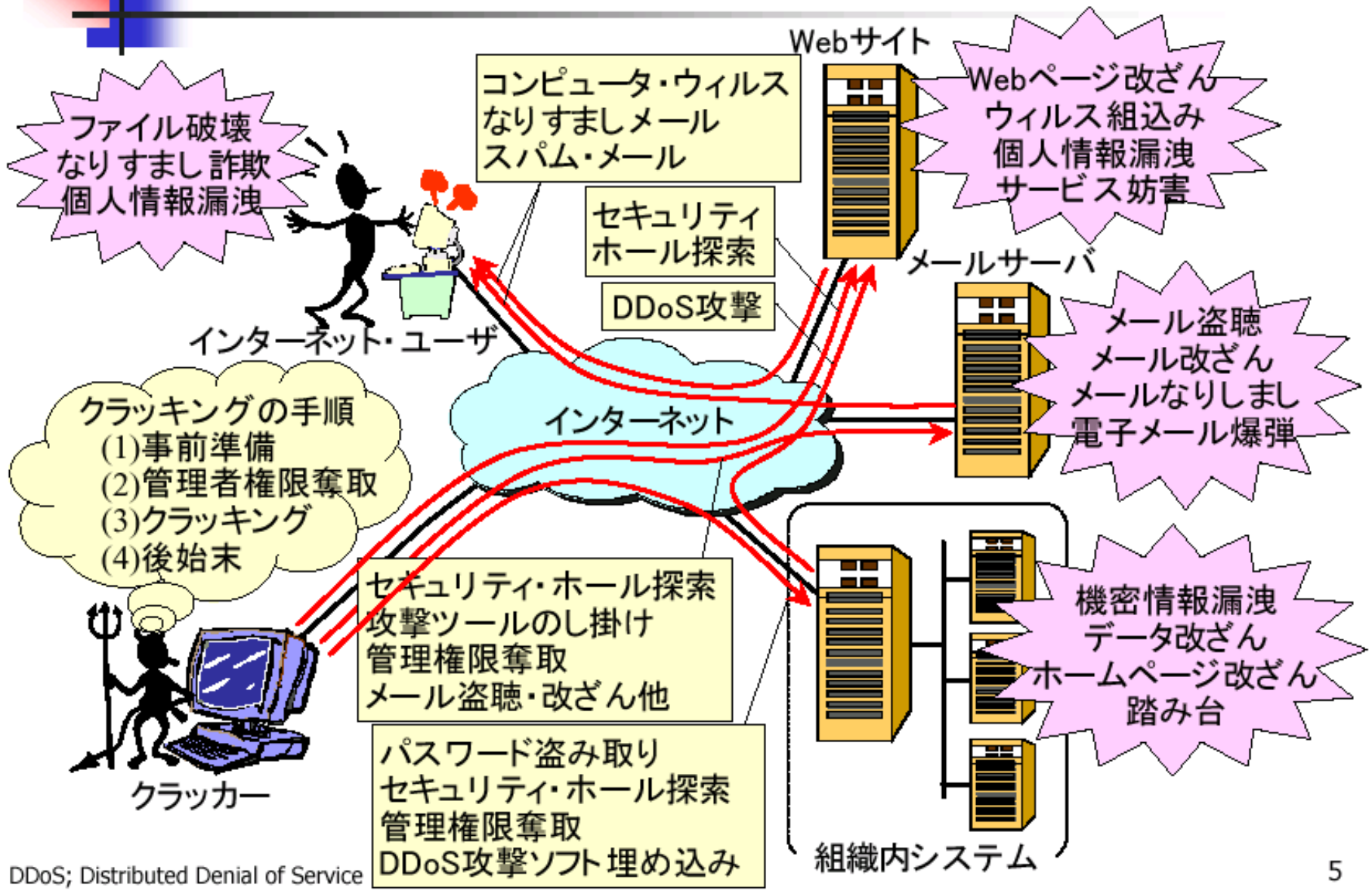
(1) データ
- (a) 機密性(他人に知られたくない情報) ・・・・ **C : Confidentiality**
- (b) 保全性(他人に変更されたくない情報) ・・・・ **I : Integrity**
- (c) 可用性(管理を必要とする情報) ・・・・・ **A : Availability**

(2) 資源 (CPU、メモリなど)

(3) 評判

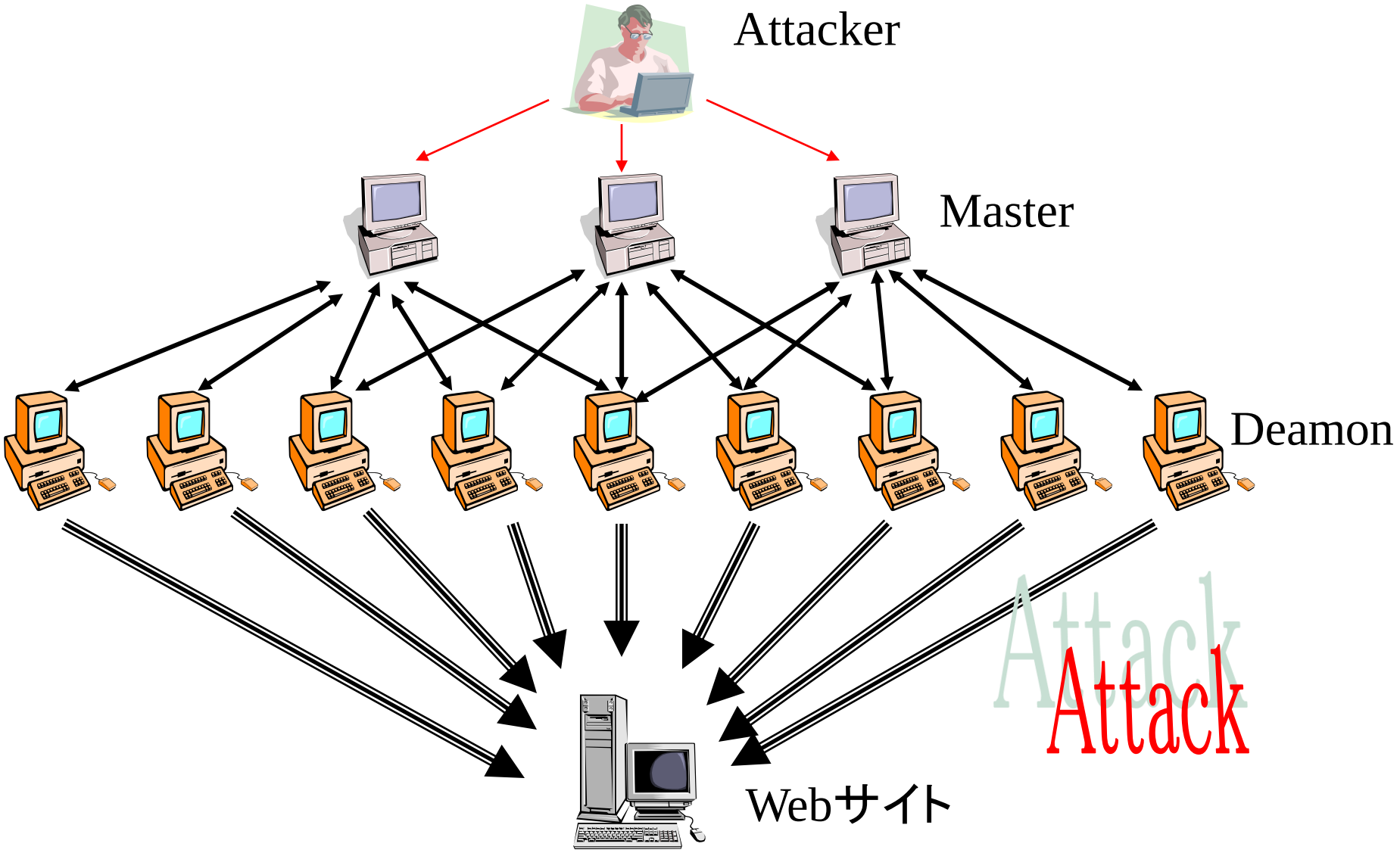
(4) サービス

不正アクセス行為の手口と悪事

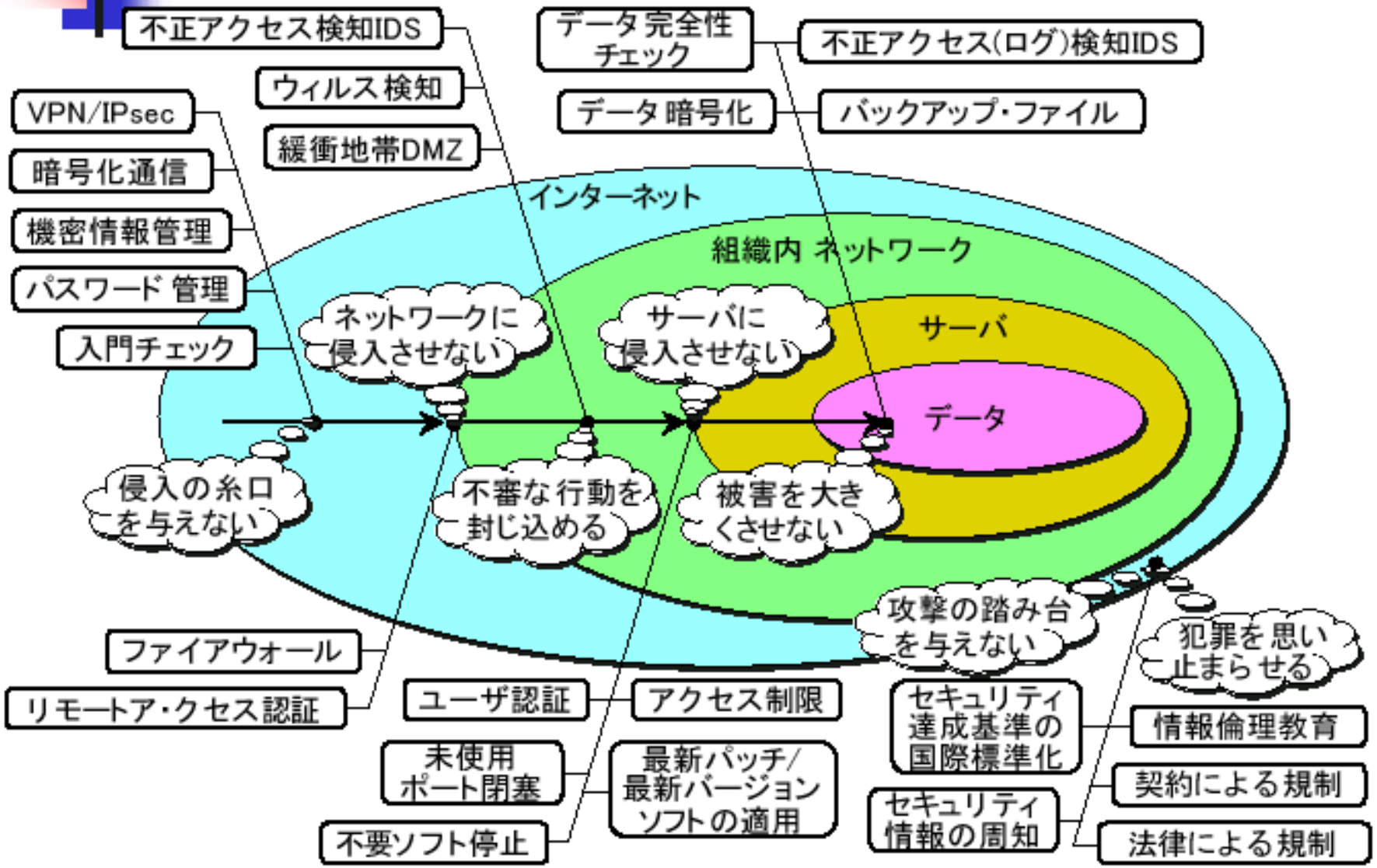


DDoS; Distributed Denial of Service

DDoS (Distributed Denial of Service) 攻撃の仕組み



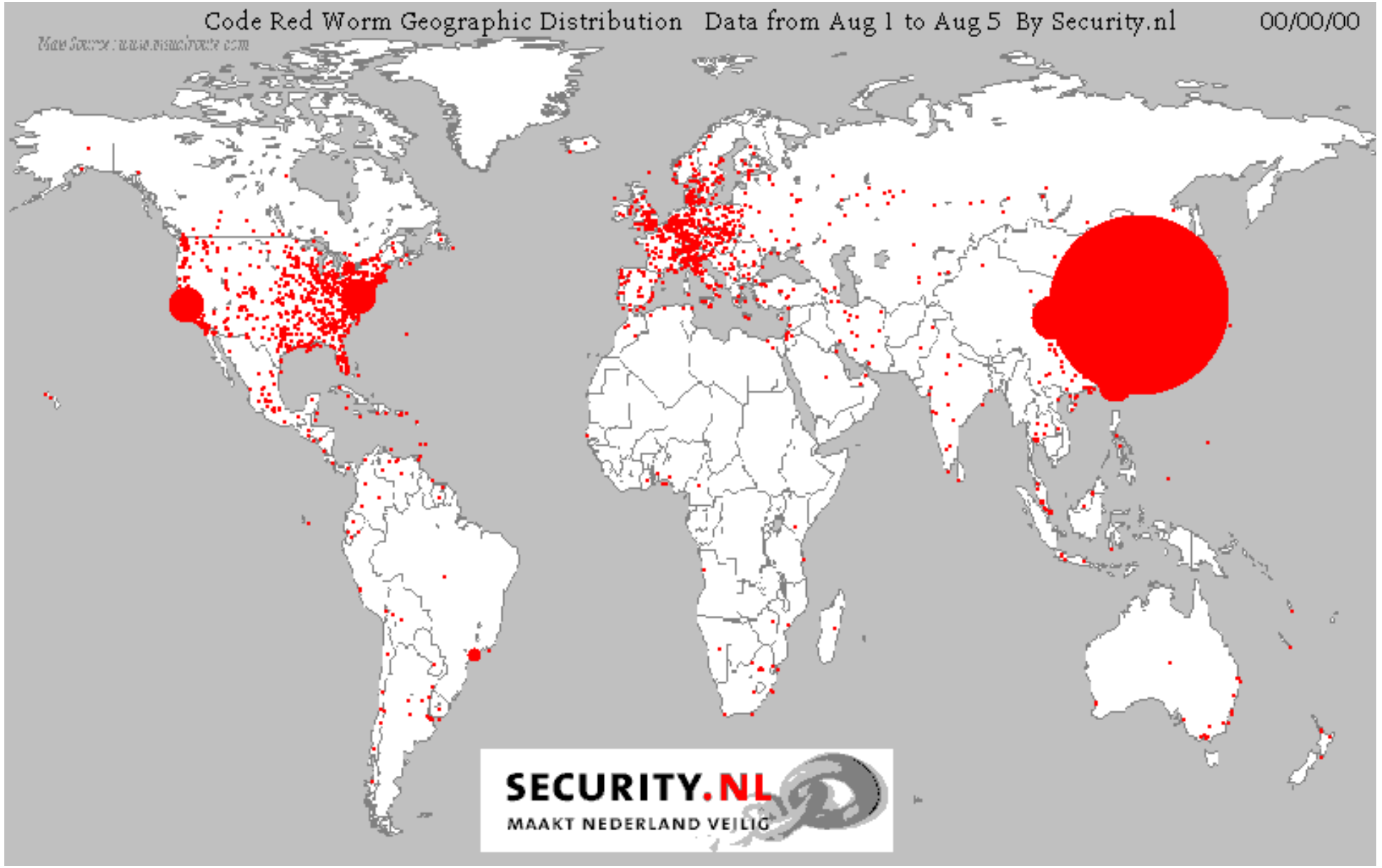
セキュリティ防衛策



IDS; Intrusion Detection System, DMZ; DeMiliarized Zone

最近のウィルス/ワームとその対策

- コードレッドⅡ(‘01.7), ニムダ(’01.9)
  - 感染力強く(複数の感染ルート), 変幻自在
  - パターンファイル流布前に, 感染が拡大
  - インターネットからの攻撃防衛だけでは不備
    例) 感染したモバイルPCから社内ネットワークに侵入
- 対策のポイント
  - 早期発見:異常トラフィック, 不信メール/ファイル・・・・
  - 正確な情報の早期入手(アンチウィルスベンダーから)
  - 感染経路遮断:ケーブル引抜, プロトコルの制限・・・・
  - 対応態勢(通常/警戒/非常事態)の整備
  - 多くはMS系ソフトを攻撃対象⇒UNIX系マシン？


Code Red Worm Geographic Distribution Data from Aug 1 to Aug 5 By Security.nl 00/00/00
SECURITY.NL
MAAKT NEDERLAND VEILIG

ネットワークセキュリティー

1. セキュリティーポリシーを決める
2. ユーザーの認証(パスワード管理)
3. ファイルの保護
4. Secureなホストへの アクセスの方法
5. アクセス制御(xinetd，TCP wrappers)
6. 暗号化 (IPsec)
7. Firewall (e.g., socks)
8. 電子メールシステム

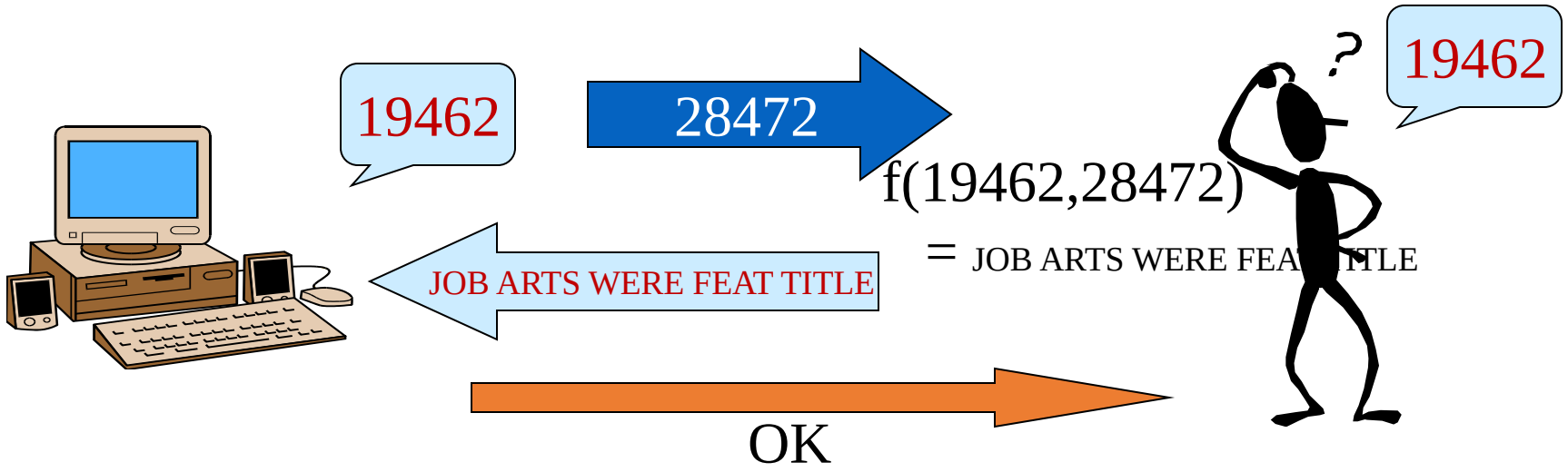
ユーザの認証(アカウント管理)

1. 想像しにくいパスワードの使用
   -すべて小文字、すべて大文字とか、、、、
2. パスワードファイル(/etc/passwd)の保護
   - Shadow Password File；/etc/master.passwd
3. 使い捨てパスワード(OTP；One Time Password)
   - Challenge & Response 型 with 共有鍵
4. ssh；Secure Shell
   - 暗号化された Remote Login with 公開鍵+秘密鍵
5. RADIUS (Remote Authentication Dial In User Services)
   - ダイアルアップユーザに対する認証処理
6. 2段階認証 (信頼できるアンカーポイントの存在を前提に)

使い捨てパスワード ； OTP

- 毎回異なるパスワードを利用する。
- Challenge & Response 型の認証
- 同一の鍵をサーバーとクライアント間で共有する。
- 例えば、

  ftp://ftp.nrl.navy.mil/pub/security/opie/opie-2.3.tar.gz
- Windowsのクライアントソフトもある。



安全なRemote Shell
- ssh ； secure shell -

- 公開鍵暗号方式；公開鍵＋秘密鍵
- フィンランド ヘルシンキ大学で開発
- RSA、IDEA、DES 方式による暗号化
- 例えば； http://www.cs.hut.fi/ssh
- Windows版クライアントもあります(Teraterm)
- sshd (secure shell daemon)
- ssh (rlogin)、scp (rcp)、ssh-keygen
- 鍵情報ファイル

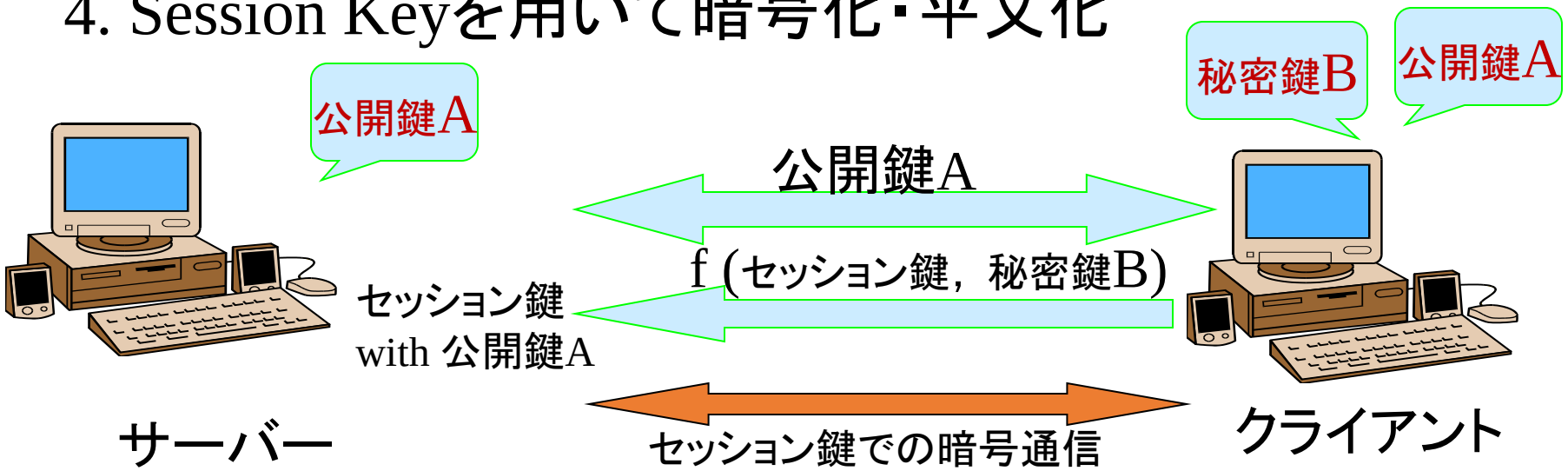
  .ssh/identity

  .ssh/identity.pub (client) => .ssh/authorized_keys (server)

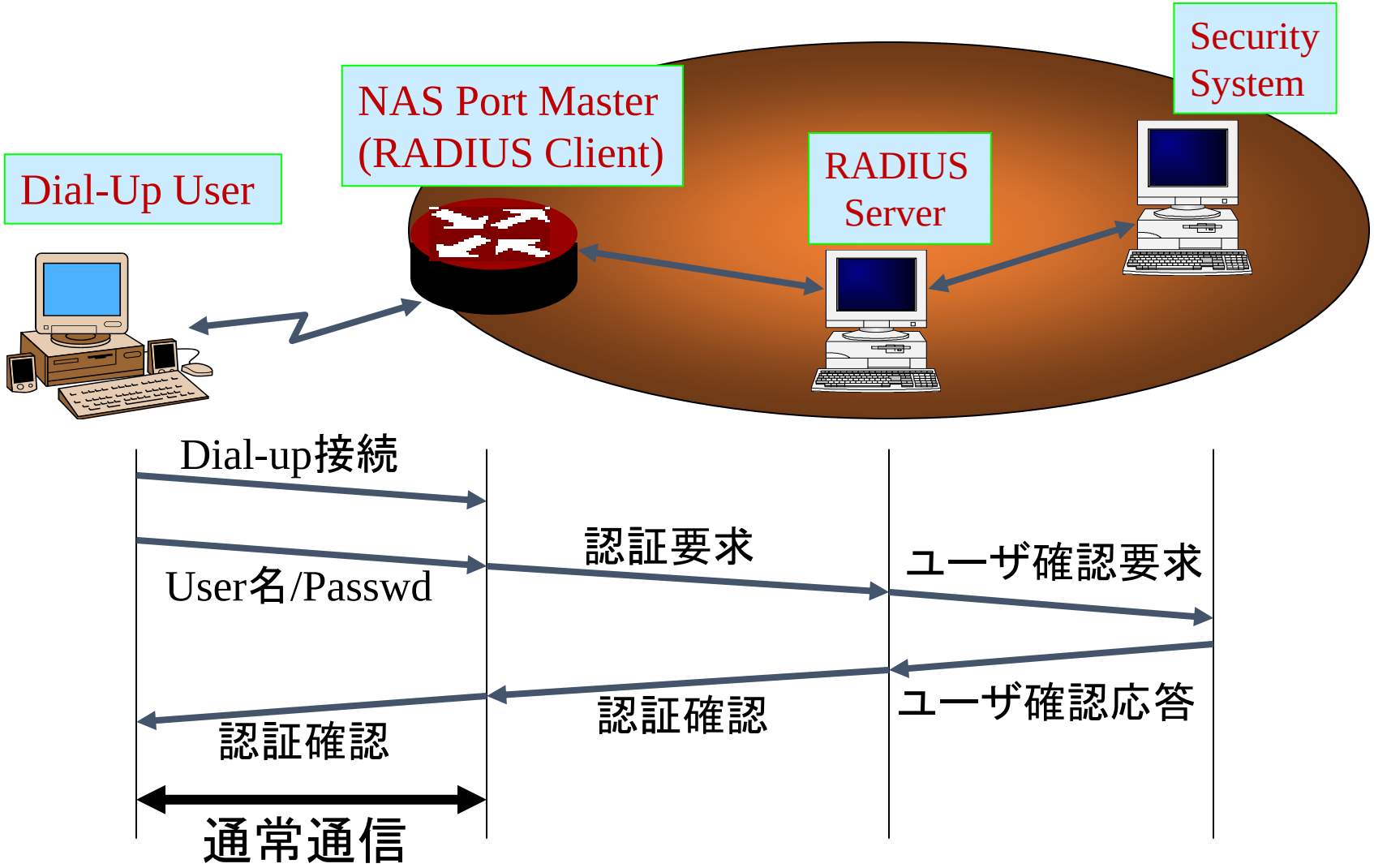
  .ssh/ssh_known_hosts, /etc/ssh_known_hosts

安全なRemote Shell
- ssh ； secure shell -

1. 公開鍵(Public key)の交換とチェック
   /etc/ssh_known_hosts、..ssh/known_hosts
2. クライアントは秘密鍵を使って Session Keyを送る
   (*) Session keyは毎回変わる。
3. サーバは公開鍵を使って Session Key を平文化する
4. Session Keyを用いて暗号化・平文化



RADIUSの動作(広域システムでの認証方式)



ネットワークセキュリティー

1. セキュリティーポリシーを決める
2. ユーザーの認証(パスワード管理)
→ 3. ファイルの保護
   (i) ファイルアクセス
   (ii) Freeソフトのインストール
4. Secureなホストへの アクセスの方法
5. アクセス制御(xinetd TCP wrappers)
6. 暗号化 (IPsec)
7. Firewall

ファイルアクセス権

3つのレベルのファイルアクセス権

(1) Global, Group(SGID), User(SUID)
(2) Readable, Writable

・root でのアクセスを極力防止する。

例 ;

ユーザ hiroshi **の** `.login` **が** Group (elab) **で**writable。
`/etc`**が** Group Owner**にとって**writable (root&sysem)
→ elab **グループの誰かに**login**されると、**hiroshi **の**
`.login`**に次のコマンドを潜り込ませることが可能。**

```
rm -f /etc/passwd
cp /tmp/data526  /etc/passwd
```

フリーソフトウェアの導入

フリーソフトの導入を行うときには、次のようなポイントに注意してインストール作業を行う方が良い。

(1) 特権ユーザ(e.g., root)としてはテストを行わない。
(2) ソースコードをコンパイルして使用する。
　　＝ バイナリコードは使用しない。
(3) Archiveをunpackするときは、内容を確かめてから(`tar tf file`)
(4) テンポラリーなディレクトリーでの unpack
(5) file コマンドでファイルの内容のチェック
　　（バイナリファイルは注意）
(6) ソースコードを眺める

フリーソフトウェアの導入

（7）makeファイルを眺める。make －n
　　変なディレクトリーをアクセスしていないか
（8）string コマンドでObjectファイルをチェック
（9）可能な限りlocalファイルでテストする
（10）make －n install でmakeプロセスをチェック
（11）SUID、SGIDコマンドがあったら、要注意

ネットワークセキュリティー

1. セキュリティーポリシーを決める
2. ユーザーの認証(パスワード管理)
3. ファイルの保護
→ 4. Secureなホストへの アクセスの方法
   - Secure Shell； `ssh`
5. アクセス制御(xinetd，TCP wrappers)
6. 暗号化 (IPsec)
7. Firewall

ネットワークセキュリティー

1. セキュリティーポリシーを決める
2. ユーザーの認証(パスワード管理)
3. ファイルの保護
4. Secureなホストへの アクセスの方法
→ 5. アクセス制御(xinetd，TCP wrappers)
   (i) 不要なサービスをシャットダウン
   (ii) ログをとるプログラム(tcpd)
   (iii) ホスト・サービスのアクセス制御
6. 暗号化 (IPsec)
7. Firewall

アクセス制御

1. 必要なサービスのみにしてしまう。
   ＝ 不要なサービスはシャットダウンする。
   ＝＞ /etc/inetd. conf の不要な行をコメントアウト

   （1）共通; conmsat, finger, ntalk,
   （2）クライアント; pop, imap
   （3）ssh導入時; login, exec, shell, ftp

アクセス制御

2. *tcp_wrapper*

http://csrc/nist.gov/tools/tools.htm

(1) */etc/inetd.conf*

Before;

```
#service socket protocol wait?  User  program           arguments
ftp      stream  tcp    nowait  root /usr/sbin/ftpd       ftpd
telnet   stream  tcp    nowait  root /usr/sbin/telnetd    telnetd
shell    stream  tcp    nowait  root /usr/sbin/rshd       rshd
login    stream  tcp    nowait  root /usr/sbin/logind     logind
```

After;

```
#service socket protocol wait?  User  program           arguments
ftp      stream  tcp    nowait  root /usr/sbin/tcps       ftpd
telnet   stream  tcp    nowait  root /usr/sbin/tcpd       telnetd
shell    stream  tcp    nowait  root /usr/sbin/tcpd       rshd
login    stream  tcp    nowait  root /usr/sbin/tcpdd      logind
```

(2) inetd のreread

```
# kill -HUP pid-of-inetd-process
```

アクセス制御

3. ホスト・サービスのアクセス制御

(1) /etc/hosts.allow

```
fingerd      : ophelia hamlet laertes
rshd,rlogind: LOCAL EXCEPT hamlet
telnetd,ftpd: LOCAL, .expcons.com, 192.1.4
```

(2) /etc/host.deny

```
tftpd : ALL : (/usr/sbin/safe_finger -l @%h |
                /usr/sbin/mail -s %d-%h root) &
ALL   : ALL
```

アクセス制御

1. セキュリティーポリシーを決める
2. ユーザーの認証(パスワード管理)
3. ファイルの保護
4. Secureなホストへの アクセスの方法
5. アクセス制御(xinetd, TCP wrappers)

→ 6. 暗号化 (IPsec)
   (1) 暗号メール
   (2) トランスポート層での暗号化
   (3) IPsec

7. Firewall

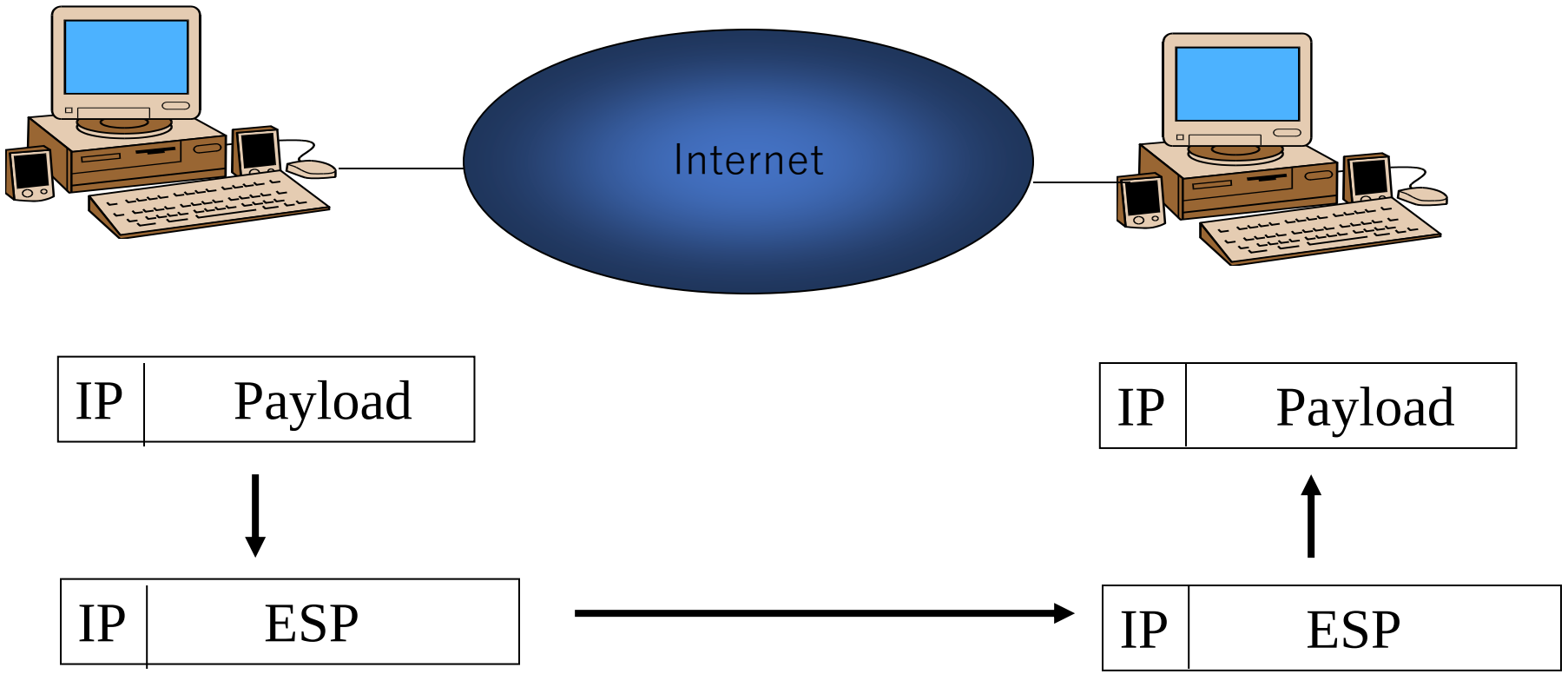
暗号化・ＩＰｓｅｃ

1.暗号メール
- PEM, MOSS
- S/MIME
- PGP (Pretty Good Privacy)

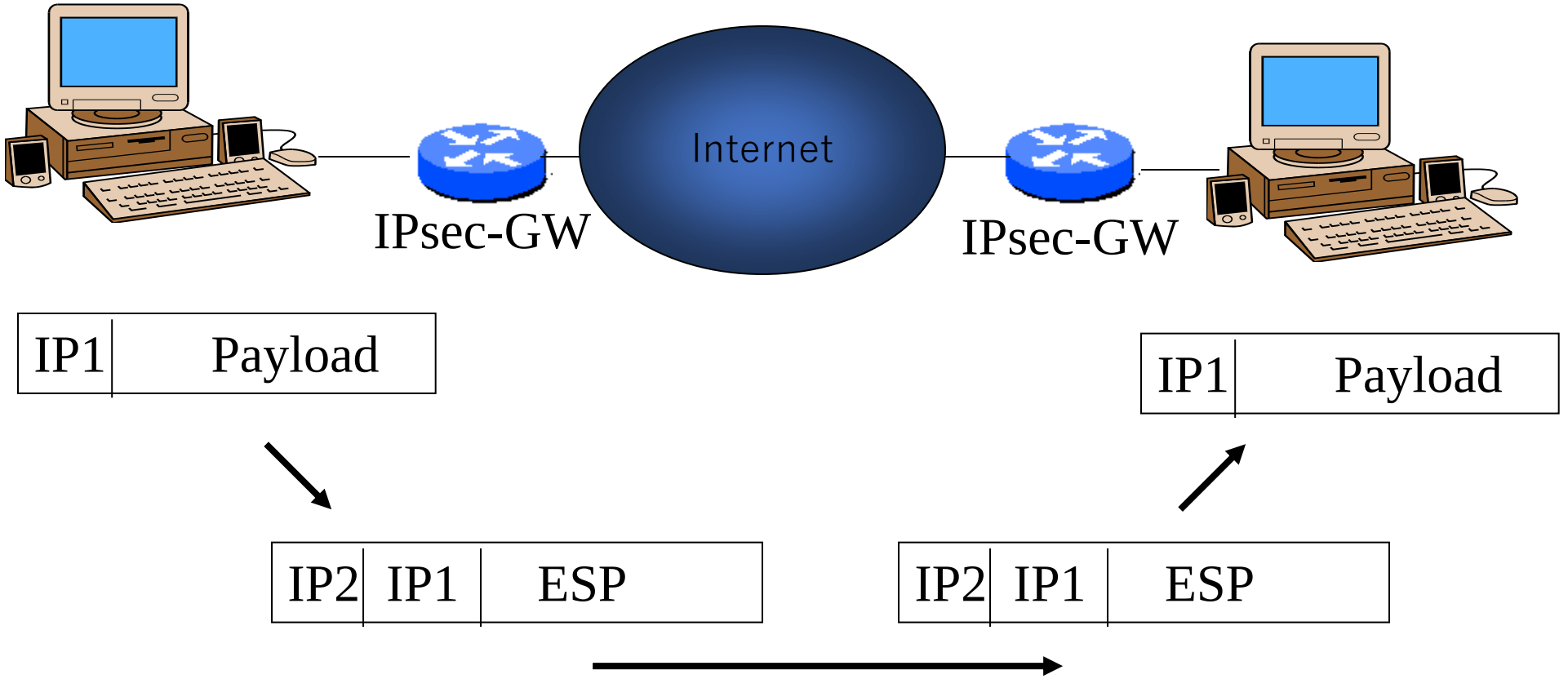
2. トランスポート層での暗号化
- SOCKS (http://www.socks.nec.com/)

3. IPsec
- 認証ヘッダ(AH: Authentication Header)
- 暗号ペイロード(Encapsulating Security Payload)
- 鍵交換プロトコル(Internet Key Exchange)

暗号化・ＩＰｓｅｃ
- トランスポートモード -



暗号化・ＩＰｓｅｃ
- トンネルポートモード -



暗号化・認証アルゴリズム (既述)

[1] **平文の認証** (一般には、N bits(入力) → m bits(出力),　N>m)

[2] **平文の暗号化**

① DES(Data Encryption Standard) ; **秘密鍵方式**

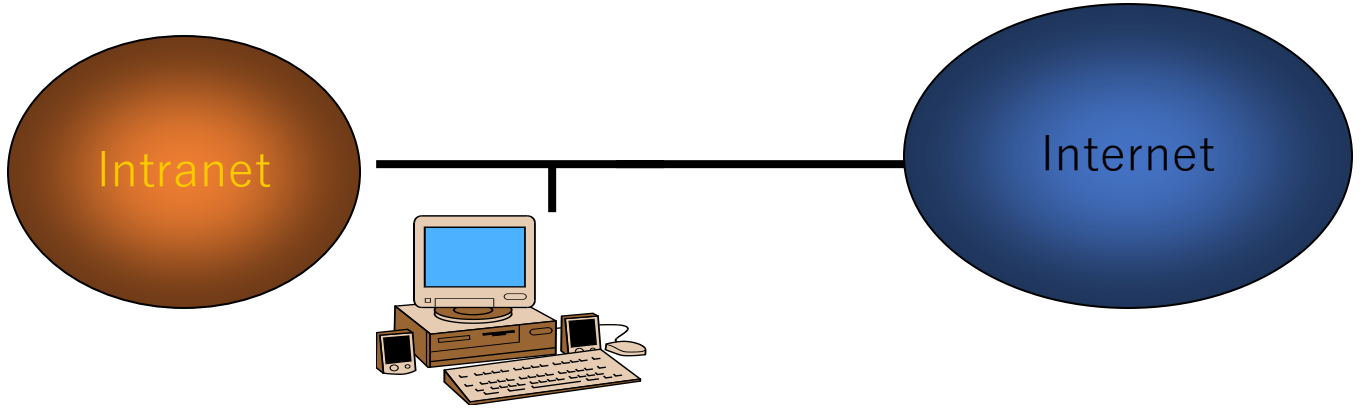
② RSA(Rivest, Shamir, Adleman) ; **公開鍵方式**

ネットワークセキュリティー

1. セキュリティーポリシーを決める
2. ユーザーの認証(パスワード管理)
3. ファイルの保護
4. Secureなホストへの アクセスの方法
   - Secure Shell； `ssh`
5. アクセス制御(xinetd，TCP wrappers)
6. 暗号化 (IPsec)

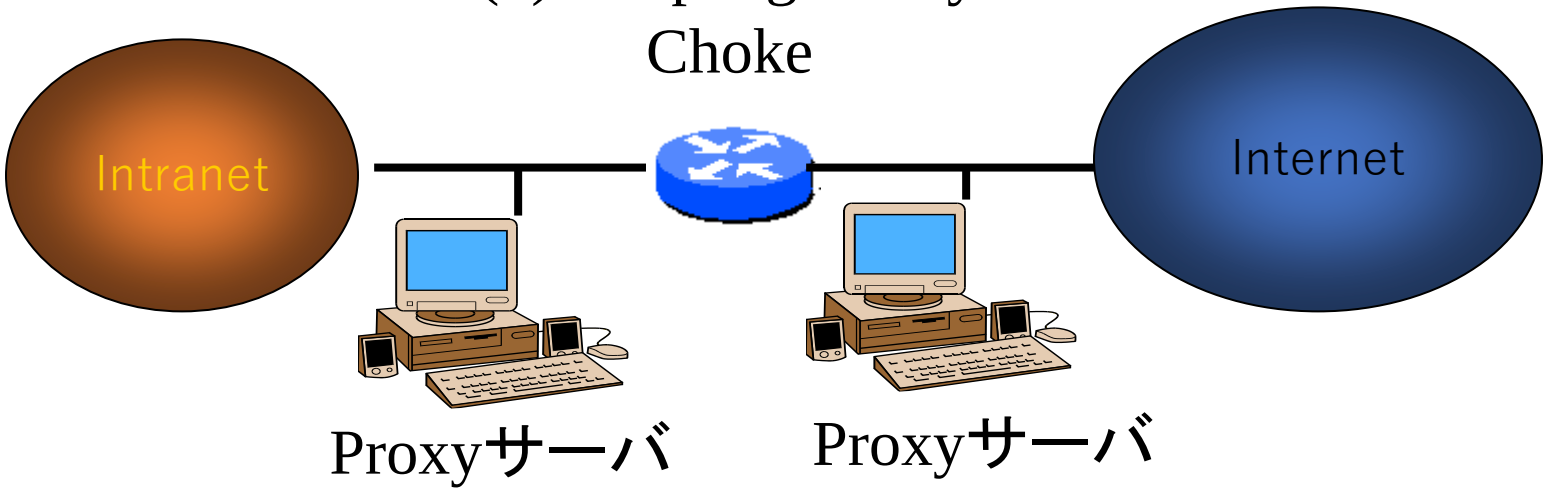
→ 7. Firewall

4 Levels of Firewall Configurations



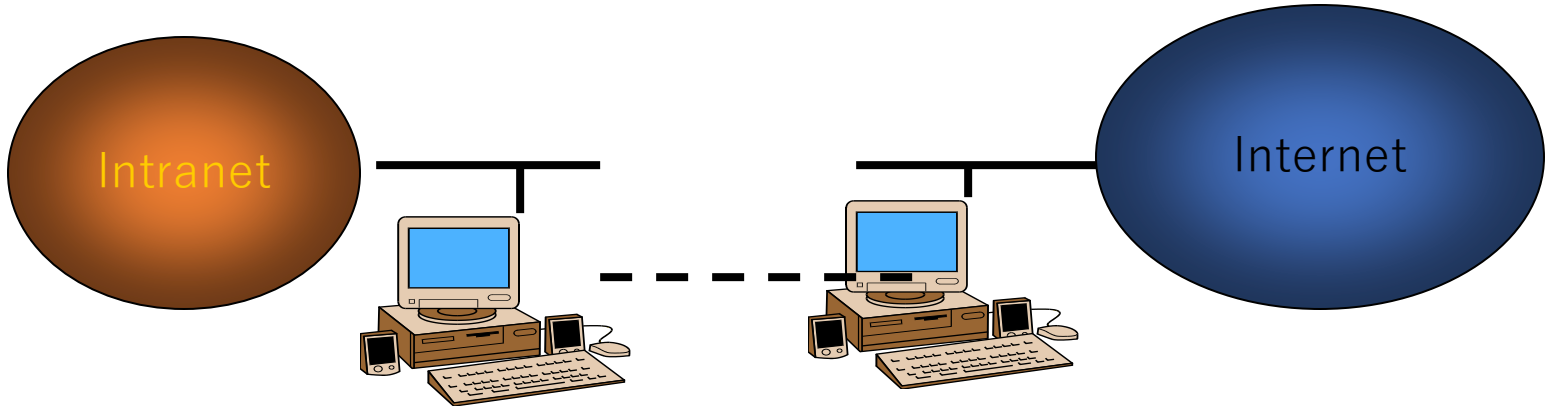
(1) Simple gateway

Choke



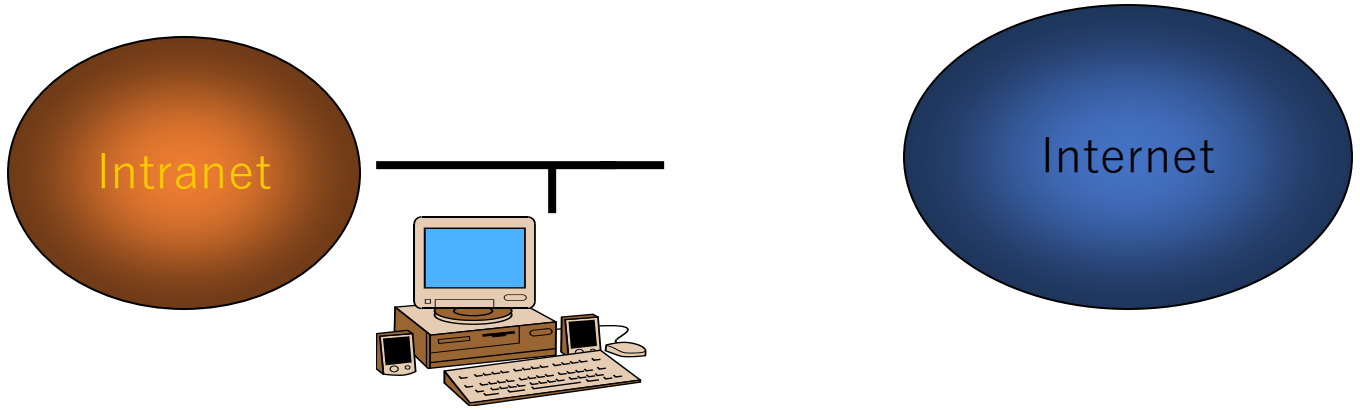
(2) Belt and Suspender

4 Levels of Firewall Configurations



Proxyサーバ

(3) TIPなどによる接続



(4) Disconnect

ファイアウォール

1. 経路情報の交換を停止
   → FW内の経路情報は外部に広告されない。
   (注) Source routingで進入可能

2. パケットフィルタリング
   socket{src_IP, src_port, dsrt_IP, dst_port}の情報でフィルタリングを行う。
   (注) - ftpなど相性がよくないアプリケーションが存在
   (a) 公開されるべきサービス
   WWW, anonymous-ftp, IRC
   (b) 公開されるべきではないサービス
   NIS, NFS, PRC, TFTP, SNMP
   (c) 内向きを禁止すべきサービス
   SMTP, NNTP, HTTP, FTP

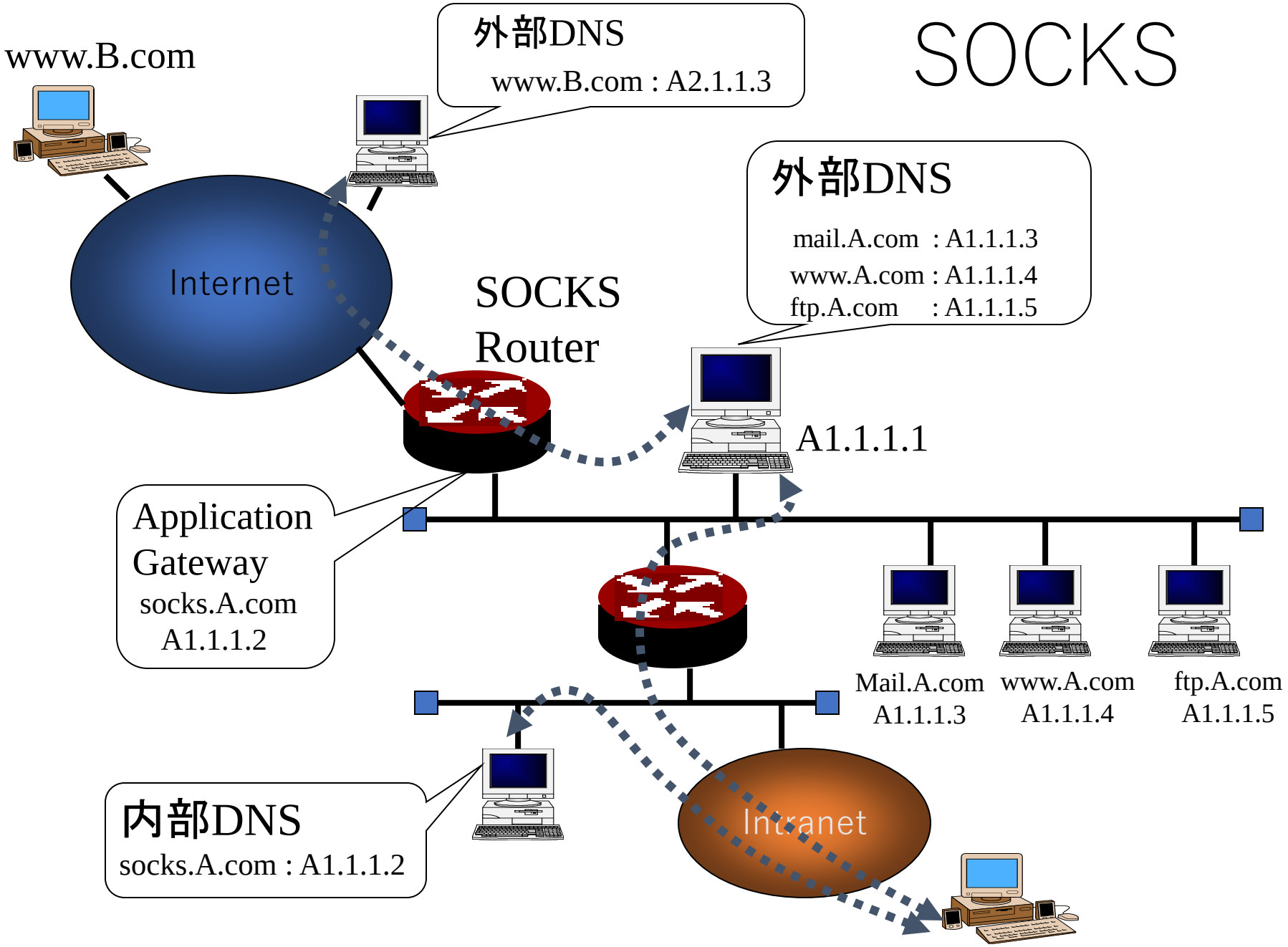
ファイアウォール

3. アプリケーション ゲートウェイ ; Proxyサーバ
   → 各アプリケーションでProxyサービスを提供する。

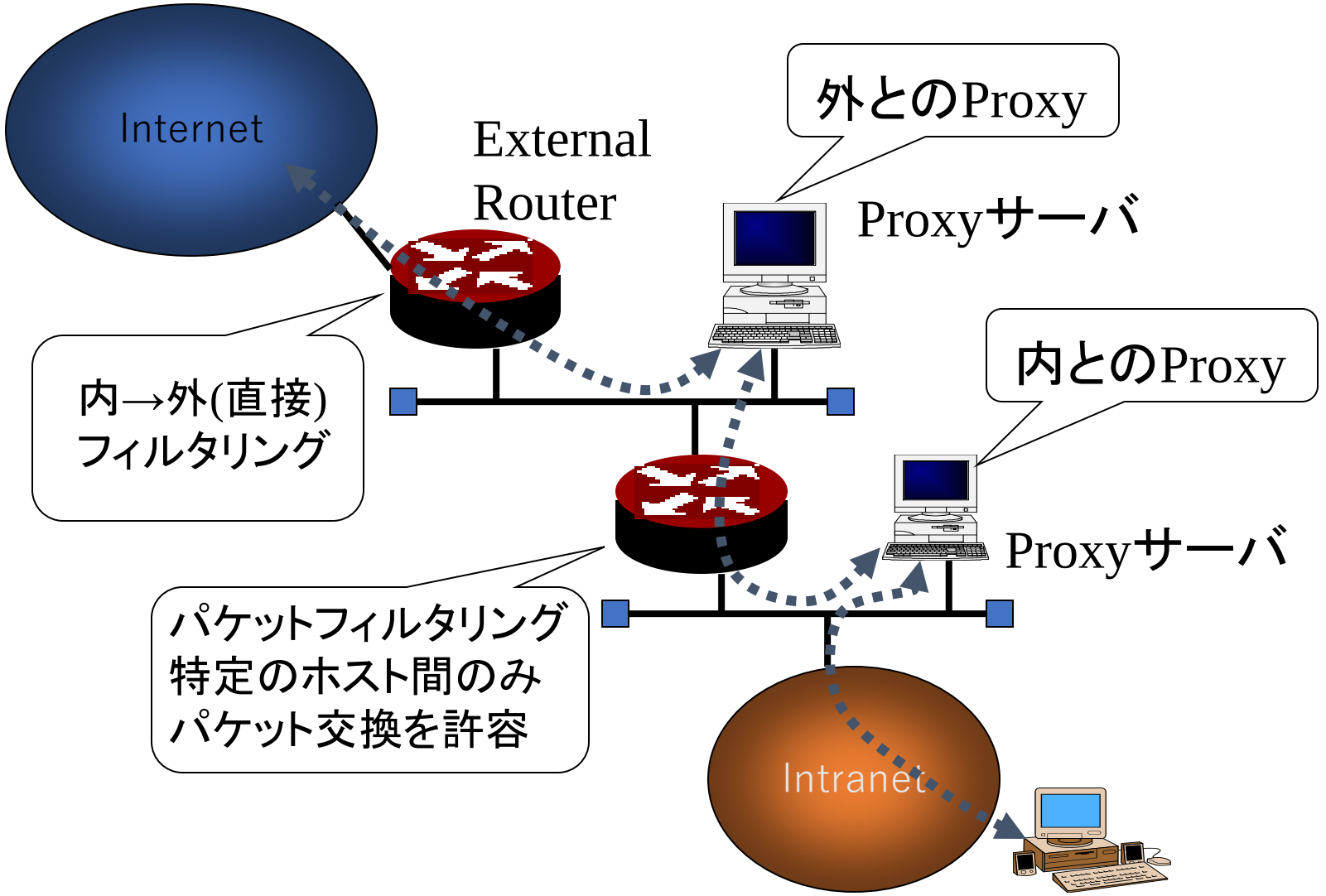
   e.g., SOCKS

   ftp://ftp.nec.com/pub/security/socks.cstc/socks.cstc.4.2.tar.gz

SOCKS


www.B.com
外部DNS
www.B.com : A2.1.1.3
Internet
SOCKS
Router
外部DNS
mail.A.com : A1.1.1.3
www.A.com : A1.1.1.4
ftp.A.com : A1.1.1.5
A1.1.1.1
Application
Gateway
socks.A.com
A1.1.1.2
Mail.A.com
A1.1.1.3
www.A.com
A1.1.1.4
ftp.A.com
A1.1.1.5
内部DNS
socks.A.com : A1.1.1.2
Intranet

Firewall System Configuration



でも、、、、、

- Firewall で、システムは 守れるのだろうか?

ゼロ トラスト ネットワーキングへ

クラウド技術 ＋ 携帯網技術

最後に

そもそも、、、、、

サイバーセキュリティー

- ✓完全(＝安全）は、存在しない。
- ✓暗号化技術は、超難しい数学を使うけど、結局は、「いいかげん」(＝安心）

セキュリティに対する考え方

1. **グローバル**に考え、**ローカルな施策**を行う
2. 「原理主義」ではなく**「実践主義」**で進める
3. 強制する・制限するのではなく、**活動の活力向上**を応援する
4. **「過保護」**は、かえってリスクを増大させる
5. 「やらされる」ではなく、**「やりたくなる」**を目指す
6. セキュリティ対策を、**品質向上のための投資**と捉える
7. 経験と知見の**「共有」**を行う
8. インシデントの経験者は、**「被害者」**として**「保護・支援」**する
9. **「匿名性」の堅持** と プライバシーの保護
10. **まずは自助、次に共助、最後に公助**

https://igcj.jp/meetings/concept-for-security.pdf