

JNSA PKI相互運用WG・電子署名WG共催セミナー  
PKI Day 2015 サイバーセキュリティの要となるPKIを見直す

# SSL/TLS生誕20年、脆弱性と対策を振り返る

2015年4月10日(金) 13:40-14:15  
於：ヒューリックカンファレンス秋葉原ROOM1

富士ゼロックス株式会社 漆 嶋 賢二, CISSP



# 自己紹介: 漆 賢二(うるしま), CISSP

## 経歴

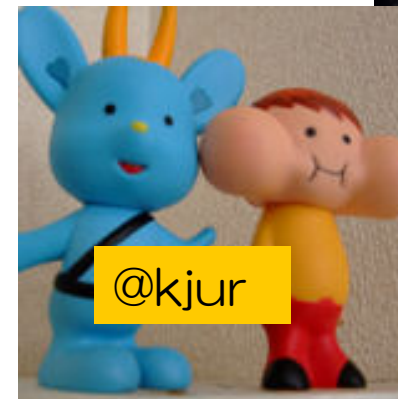
- 富士ゼロックス(2010~)
- エントラストジャパン(2005~2010)
- セコム(1988~2005)

## 興味:

PKI, TLS, 電子署名, SSO, 認証, 暗号, CSIRT, 脆弱性検査, フォレンジック, スマホ, プログラミング, ビットコイン

## 別名

- 証明書ハンター
- (TLS)暗号スイートウォッチャー
- 委員、標準化、認定基準、実証実験、普及啓蒙
- CRYPTREC, 日本データ通信協会
- IBECOM, PKI-J, JNSA, 欧州ETSI
- PKI, 長期署名, タイムスタンプ, TLS



ブログ: 自堕落な技術者の日記



jsrsasign - JavaScript 実装暗号ライブラリ

# アジェンダ

- SSL/TLSとは
- SSL/TLSの歴史
  - プロトコルの歴史
  - クライアントの歴史
  - サーバーの歴史
  - ライブラリの歴史
- 脆弱性の歴史
- 脆弱性への対応の整理(4つの分類)
- TLSやPKIの脆弱性対策技術の最新動向

# SSL/TLSとは

# アマゾンでの商品購入例とSSL/TLS

注文の確定 - Amazon.co.jp レジ - Windows Internet Explorer

https://www.amazon.co.jp/gp/buy/signin/handlers/continue.html?ie=UTF8&oldPurchaseId=&oldCust...

Amazon.co.jp

サインイン 宛先 商品確認 ギフト 配送 会計 確認

注文内容を確認・変更する

**お届け先住所:**  
漆島賢二  
**東京都中央区 1 - 1**  
電話番号: **03-xxx** [変更](#)

**支払い方法:**  
MasterCard 下4桁 **1234**  
Amazonギフト券・Amazonポイントなど [変更](#)  
**請求先住所:**  
発送先住所と同じ [変更](#)

Amazonギフト券・Amazonショッピングカードまたはクーポン:  
 [適用](#)

**注文を確定する**

**注文内容**

商品:	¥ 2,154
配送料・手数料:	¥ 0
合計:	¥ 2,154
Amazonギフト券:	-¥ 2,112

**注文合計: ¥ 42**

**お届け日: 2011年 12月5日**  
17 時間と 27 分 以内に確定されたご注文が対象です。 ([詳細](#))

**ハリー・ポッターと死の秘宝 PART 2 [DVD]**  
デイビッド・イェーツ  
¥ 2,154  
Prime  
数量: 1 [変更](#)  
在庫あり。  
販売: Amazon.com Int'l Sales, Inc.  
[ギフトの設定](#)

Amazon Primeのお得な配送方法:  
 無料 通常配送 (1~3営業日)  
 無料 お届け日時指定便  
ご希望のお届け日時を選択してください    
 無料 お急ぎ便 **明日 2011/12/5 月曜日にお届けします!** コンビニ・ATM・ネットバンキング・Edy払い(先払い)の支払い方法は「お急ぎ便」にはご利用いただけません。

**配送料について**  
ご注文に含まれるAmazonプライムの特典対象商品には、Amazonプライムの配送が適用されました。

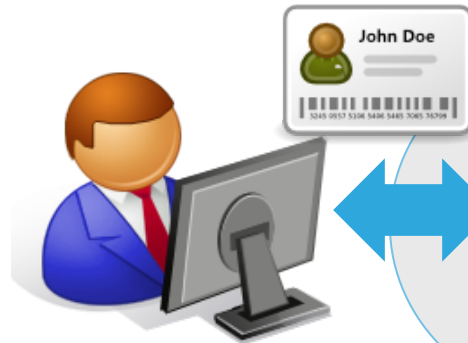
インターネット | 保護モード: 有効 100%

出典: アマゾン(www.amazon.co.jp)

# HTTPS暗号通信はなぜ必要なのか

## アマゾンでの商品購入例

amazon.com



クレジットカード番号  
住所・氏名  
秘密にしたい買い物



SSL/TLSが守る

今のネットに必須の機能

「カード番号、住所、氏名、買い物の内容」を途中で見られたくない

ニセのアマゾンサイトに「カード番号、住所」なんかを送りたくない。

途中で「届け先住所」を書き換えて商品を騙し取られたくない。

# SSL/TLSの3つの機能



「カード番号、住所、氏名、買い物の内容」を途中で見られたくない

二セのアマゾンサイトに「カード番号、住所」なんかを送りたくない。

途中で「届け先住所」を書き換えて商品を騙しとられたくない。

## 機密性

暗号通信により通信相手以外に通信内容を盗み見(盗聴)されないようにする

覗き見(盗聴)防止

共通鍵暗号を使う

## 相手認証

証明書(PKI)などを使い通信相手が正しい相手であるか認証する

なりすまし防止

PKI(公開鍵暗号)、パスワード、Kerberos認証を使う

## 完全性

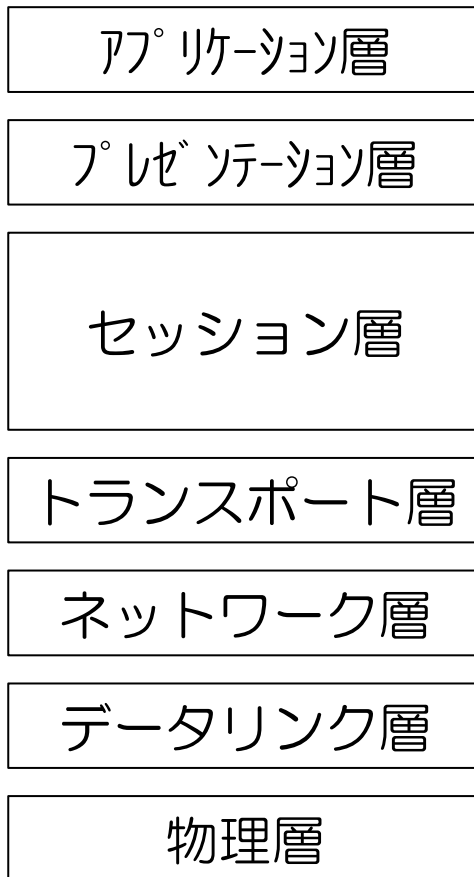
通信の途中でデータが書き換え(改ざん)されないよう、改ざん検知できる

改ざん防止

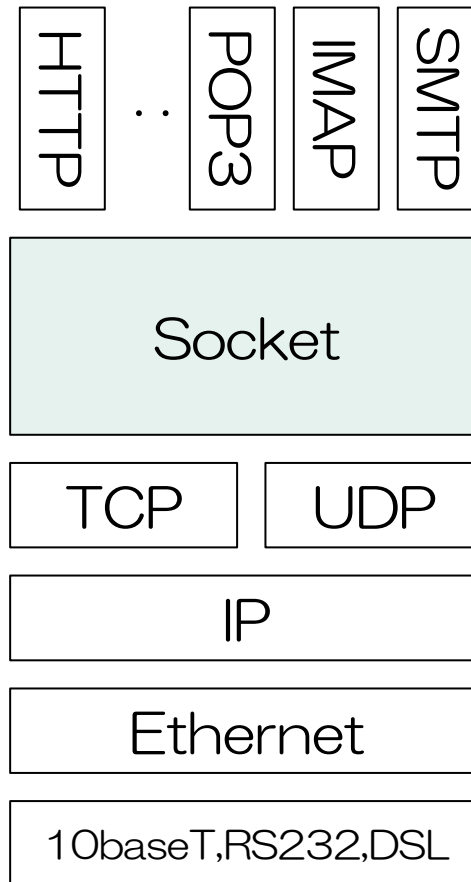
MAC(メッセージ認証コード)を使う

# SSL/TLSの特徴(HTTPでも何でも乗る)

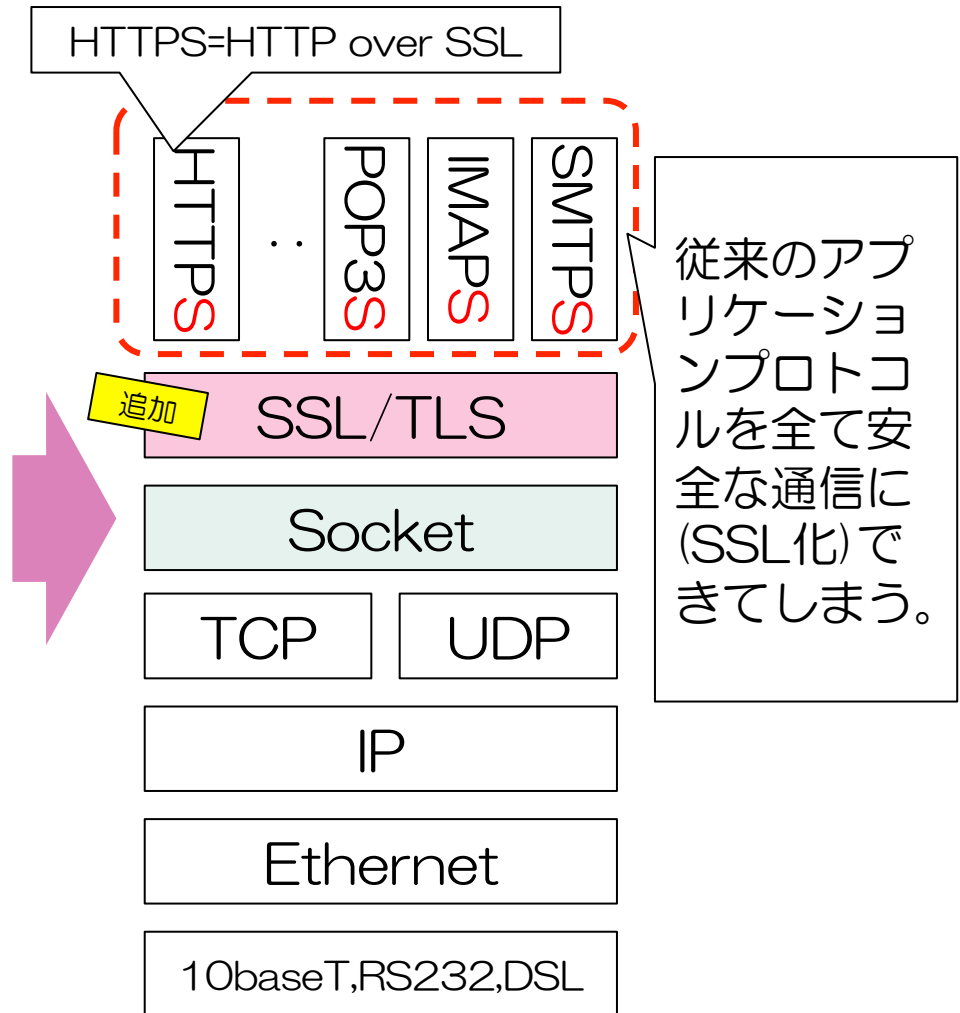
## OSI参照モデル



## 従来通信の例



## SSL化(※1)



※1: 他にLDAPs, FTPs, TELNETsとか



# SSL/TLSの歴史

# そもそもの始まり (SSL 2.0、SSL 3.0) 1995年

SSL 1.0  
1995年

SSL 2.0 Protocol Specification  
1995年2月  
Kipp E.B. Hickman / Netscape

SSL Protocol 3.0  
1996年11月18日  
Alan O. Freier / Netscape  
Philip Karlton / Netscape  
Paul C. Kocher

**SSL 0.2 PROTOCOL SPECIFICATION**

---

**THIS PROTOCOL SPECIFICATION WAS REVISED ON NOVEMBER 29TH, 1994:**

- a fundamental correction to the client-certificate authentication protocol,
- the removal of the username/password messages,
- corrections in some of the cryptographic terminology,
- the addition of a MAC to the messages [see section 1.2],
- the allowance for different kinds of message digest algorithms.

**THIS DOCUMENT WAS REVISED ON DECEMBER 22ND, 1994:**

- The spec now defines the order the clear key data and secret key data are combined to produce the master key.
- The spec now explicitly states th
- The spec is more clear on the as
- The spec is more clear on how r
- from the hash function.

**THIS DOCUMENT WAS REVISED ON**

- Defined the category to be infor
- Clarified ordering of data elem
- Defined DES-CBC cipher kind an
- Defined DES-EDE3-CBC cipher k

**THIS DOCUMENT WAS REVISED ON JANUARY 24TH, 1995:**

- Fixed bug in definition of CIPHER CHOICE in CLIENT MASTER KEY message. The previous spec erroneously indicated that the CIPHER CHOICE was an index into the servers CIPHER-SPECS-DATA array, when it was actually supposed to be the CIPHER-KIND value chosen by the client.
- Clarified the values of the KEY-ARG-DATA.

SSL Protocol V. 3.0

**The SSL Protocol**

Version 3.0

Internet Draft

March 1996 (Expires 9/96)

Alan O. Freier, Netscape Communications  
Philip Karlton, Netscape Communications  
Paul C. Kocher, Independent Consultant

---

**Table of Contents**

- [6.4 Numbers](#)
- [6.5 Enumerateds](#)
- [6.6 Constructed types](#)
  - [6.6.1 Variants](#)
- [6.7 Cryptographic attributes](#)
- [6.8 Constants](#)
- [7. SSL protocol](#)

幾つかの致命的な設計上の欠陥が発見されリリースされなかった

Elgamal暗号で有名な暗号学者で、「SSLの父」と呼ばれるTaher Elgamal氏のチームが1995年から1998年にNetscape Communicationsの主席科学者だった際に開発した。

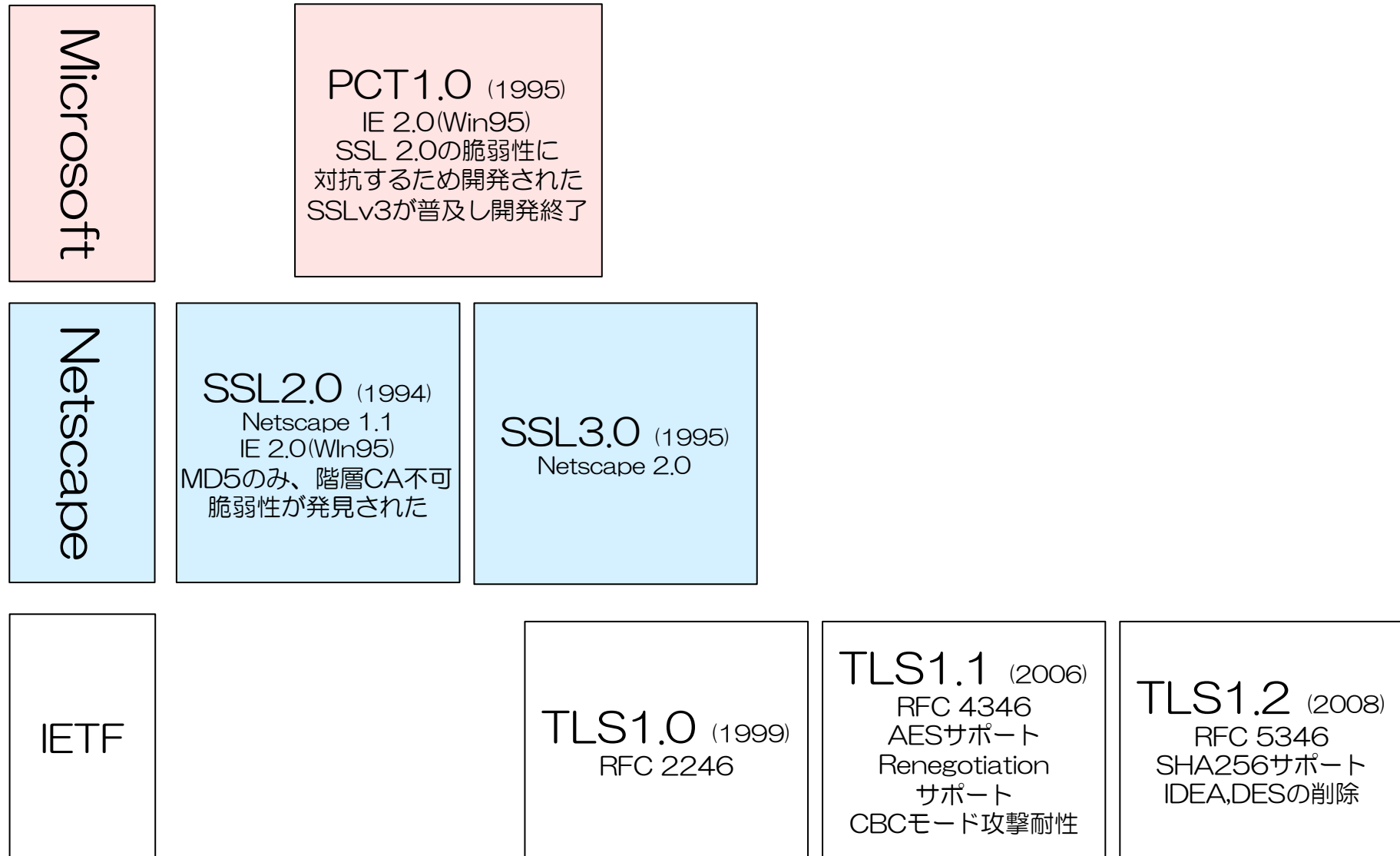
出典：<http://www-archive.mozilla.org/projects/security/pki/nss/ssl/draft02.html>  
<http://www.freesoft.org/CIE/Topics/ssl-draft/INDEX.HTM>

# SSL/TLSの20年の歴史のまとめ

	1995年	2000年	2005年	2010年	2015年	
プロトコル	<ul style="list-style-type: none"> <li>▲ SSL 1.0</li> <li>▲ SSL 2.0</li> <li>▲ PCT 1.0</li> <li>▲ SSL 3.0</li> </ul>	<ul style="list-style-type: none"> <li>▲ TLS 1.0</li> </ul>	<ul style="list-style-type: none"> <li>▲ TLS Extensions</li> </ul>	<ul style="list-style-type: none"> <li>▲ TLS 1.1</li> </ul>	<ul style="list-style-type: none"> <li>▲ TLS 1.2</li> </ul>	
ライブラリ	<ul style="list-style-type: none"> <li>▲ SSLeayリリース</li> <li>▲ IAIK JCE Toolkitリリース</li> </ul>	<ul style="list-style-type: none"> <li>▲ OpenSSL初リリース(0.9.1c)</li> <li>▲ RSA BSAFE SSL-Cリリース</li> <li>▲ Java 1.2/1.3+JSSE</li> <li>▲ SChannel on Win2000</li> </ul>	<ul style="list-style-type: none"> <li>▲ BouncyCastle JCEリリース</li> <li>▲ NSS</li> <li>▲ Gnutls</li> </ul>		<ul style="list-style-type: none"> <li>▲ OpenSSL 1.0.0リリース</li> <li>▲ LibreSSL</li> <li>▲ BoringSSL</li> </ul>	
ブラウザ	<ul style="list-style-type: none"> <li>▲ NetscapeNavigator 1.1 (初SSL対応)リリース</li> <li>▲ Internet Explorer 2 (初SSL対応)リリース</li> </ul>		<ul style="list-style-type: none"> <li>▲ Safariリリース</li> <li>▲ Firefoxリリース</li> </ul>	<ul style="list-style-type: none"> <li>▲ Chromeリリース</li> </ul>		
暗号危殆化		<ul style="list-style-type: none"> <li>▲ 米国暗号輸出規制緩和</li> <li>▲ SHA1攻撃成功</li> </ul>	<ul style="list-style-type: none"> <li>▲ NIST SHA1使用期限</li> <li>▲ MD5不正CA証明書</li> <li>▲ CBCブロック暗号モード脆弱性</li> <li>▲ RC4ストリーム暗号危殆化</li> </ul>	<ul style="list-style-type: none"> <li>▲ MS, Chrome SHA1証明書警告</li> <li>▲ factorableによる秘密鍵推測</li> <li>▲ FREAK攻撃</li> </ul>		
プロトコルの問題	<ul style="list-style-type: none"> <li>▲ SSL 2.0ダウングレード攻撃</li> </ul>	<ul style="list-style-type: none"> <li>▲ 以降、パディングオラクル攻撃・タイミング攻撃が増加していく</li> </ul>		<ul style="list-style-type: none"> <li>▲ リネゴシエーション脆弱性</li> <li>▲ SSLstrip中間者攻撃</li> </ul>	<ul style="list-style-type: none"> <li>▲ POODLE脆弱性</li> <li>▲ BLEACH脆弱性</li> <li>▲ Lucky13脆弱性</li> <li>▲ BEAST脆弱性</li> <li>▲ CRIME脆弱性</li> </ul>	
認証局の運用の問題			<ul style="list-style-type: none"> <li>▲ MD5不正CA証明書</li> </ul>	<ul style="list-style-type: none"> <li>▲ DigiNotar問題</li> <li>▲ Comodo問題</li> <li>▲ TURKTRUST問題</li> <li>▲ マレーシアDigicert Sdn問題</li> </ul>	<ul style="list-style-type: none"> <li>▲ live.fi不正発行</li> <li>▲ CNNIC/MCS不正発行</li> </ul>	
実装の問題		<ul style="list-style-type: none"> <li>▲ MS製品ASN.1重大バグ</li> </ul>	<ul style="list-style-type: none"> <li>▲ Debian OpenSSL 鍵生成問題</li> <li>▲ 識別名NULL 終端問題</li> </ul>		<ul style="list-style-type: none"> <li>▲ HeartBleed脆弱性</li> <li>▲ Apple Go to fail脆弱性</li> </ul>	

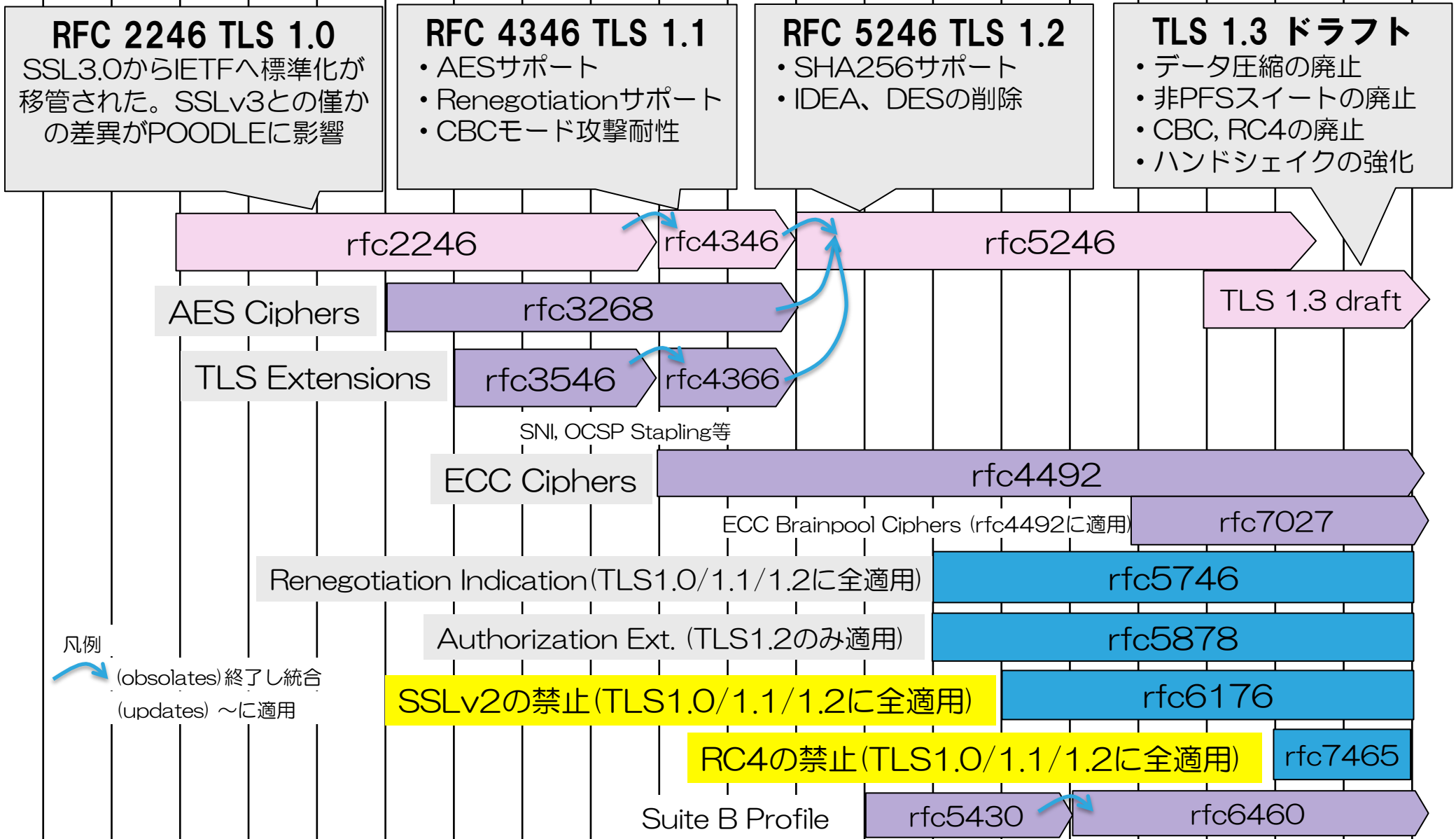
# SSL/TLSプロトコルの歴史

# SSL/TLSの標準仕様の歴史の概要



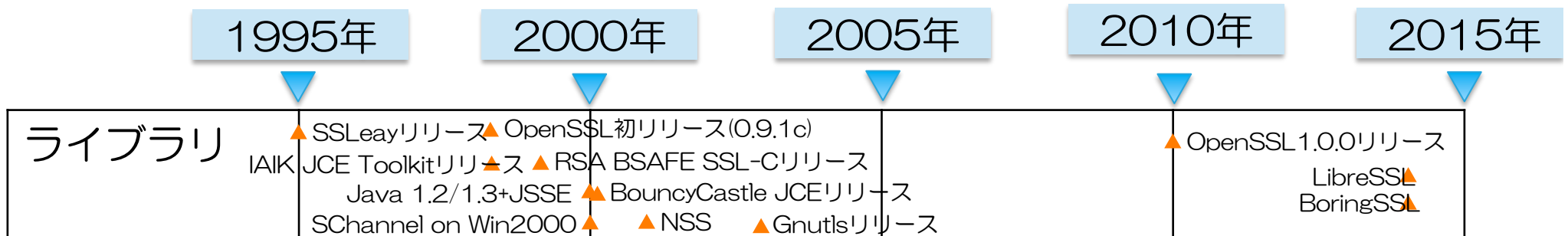
# IETF RFCのSSL/TLS標準仕様

1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017



# SSL/TLSライブラリの歴史

# SSL/TLSライブラリの歴史



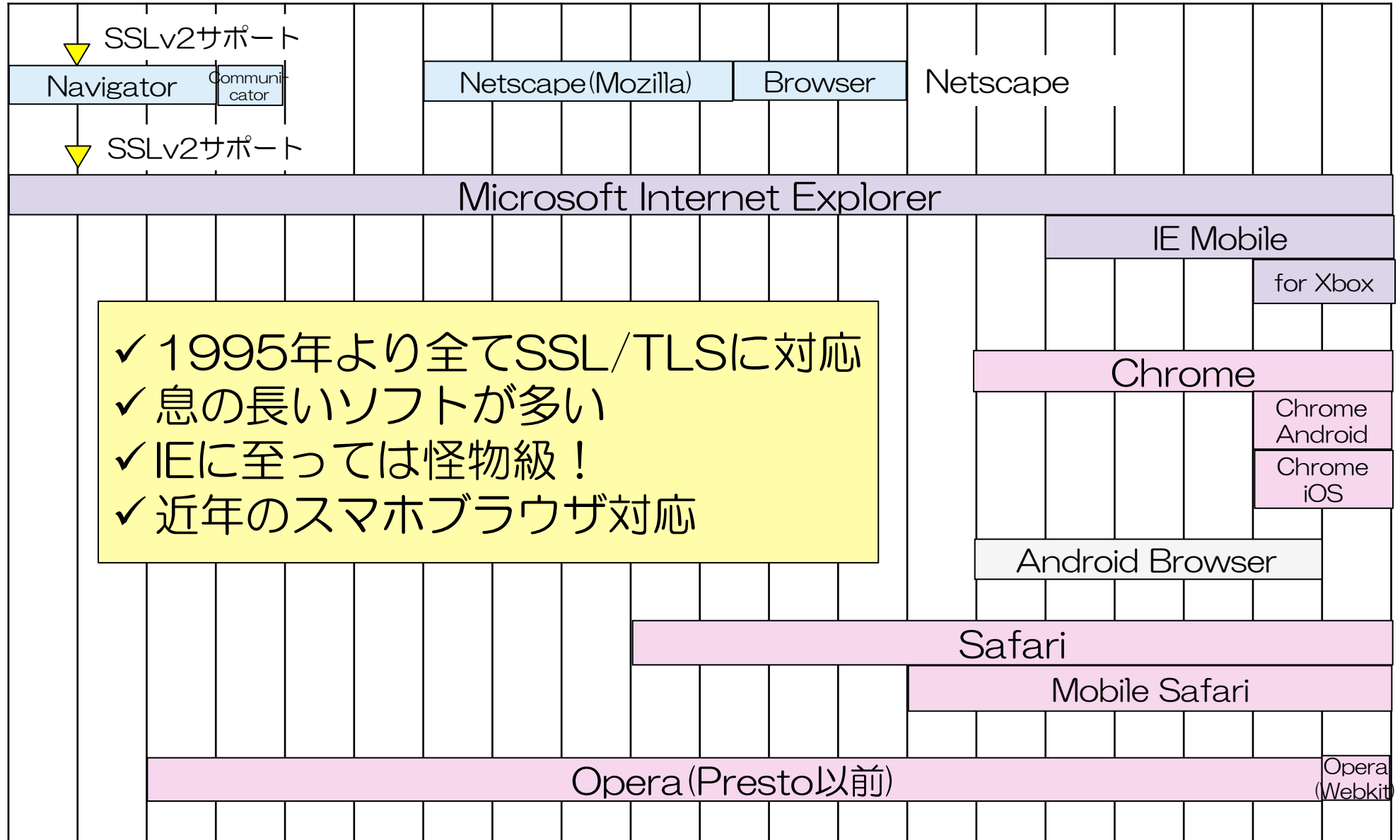
- 1995年、世界最初のSSLライブラリSSLLeayリリース
- 1998年、SSLLeayの後継としてOpenSSL 0.9.1c発リリース
- 1999年、SSLLeayを元に商用ライブラリRSA BSAFE SSL-Cリリース
  - 以降、ゲーム機や組み込みブラウザなど組み込み製品には数多く利用されている
- 2000年、WindowsのSSL実装Schannel on Win2000リリース
- Javaでは暗号API(JCE)とSSL拡張(JSSE)の共通API仕様を策定
  - Sun 標準JCE、JSSE
  - 1998年、IAIK JCE Toolkit (商用ライブラリ 研究無償、企業有償)
  - 2000年、BouncyCastle JCEライブラリ(オープンソースライブラリ)
- 2001年、MozillaがNSSライブラリをリリース
- 2003年、GNUがgnutls 1.0をリリース
- OpenSSLのソースコードのメンテナンス性の低さ、脆弱性によりブランチ
  - 2014年、OpenBSDによりLibreSSLが、GoogleによりBoringSSLがリリース



# SSL/TLSブラウザの歴史

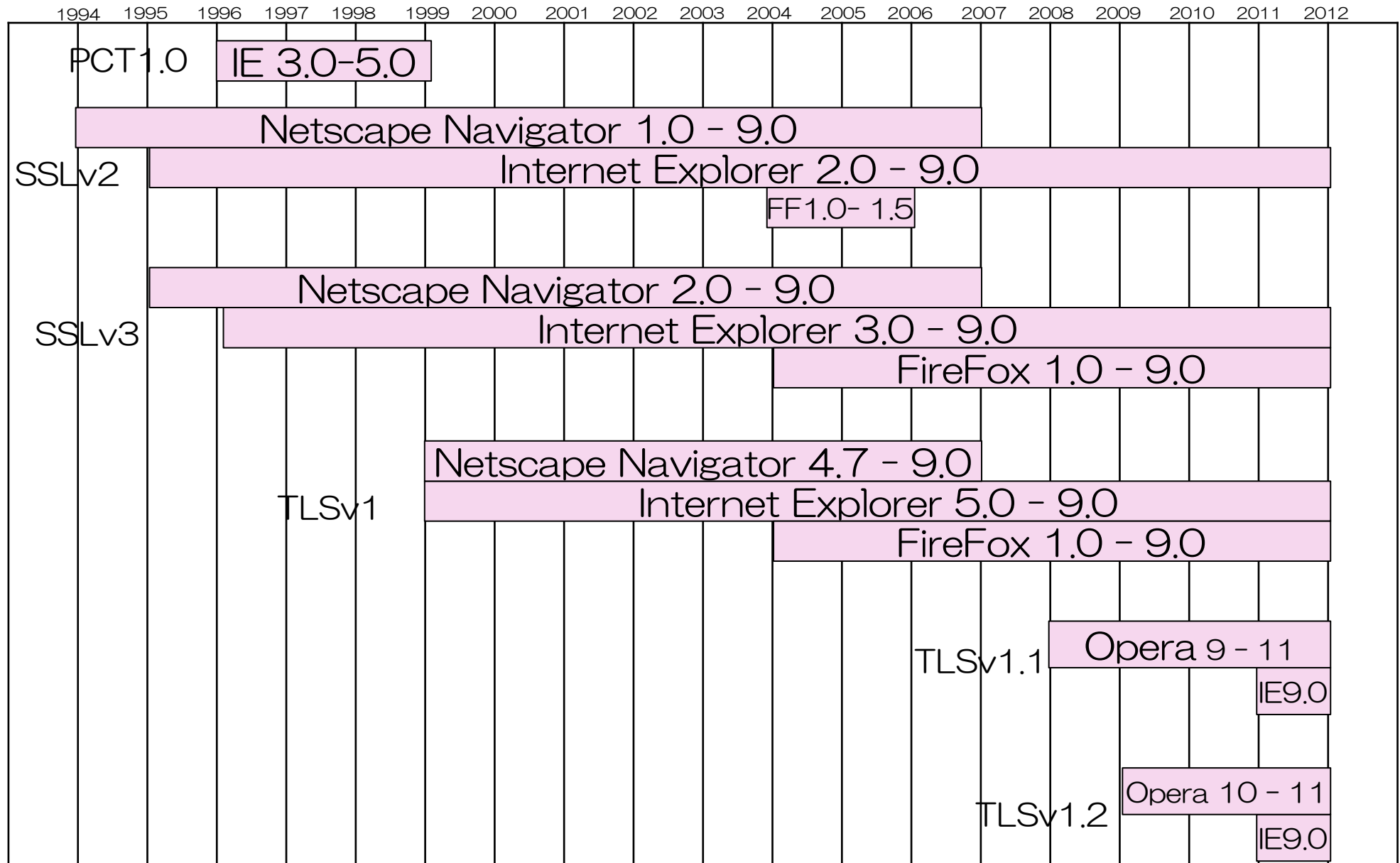
# ブラウザの歴史

1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014



- ✓ 1995年より全てSSL/TLSに対応
- ✓ 息の長いソフトが多い
- ✓ IEに至っては怪物級！
- ✓ 近年のスマホブラウザ対応

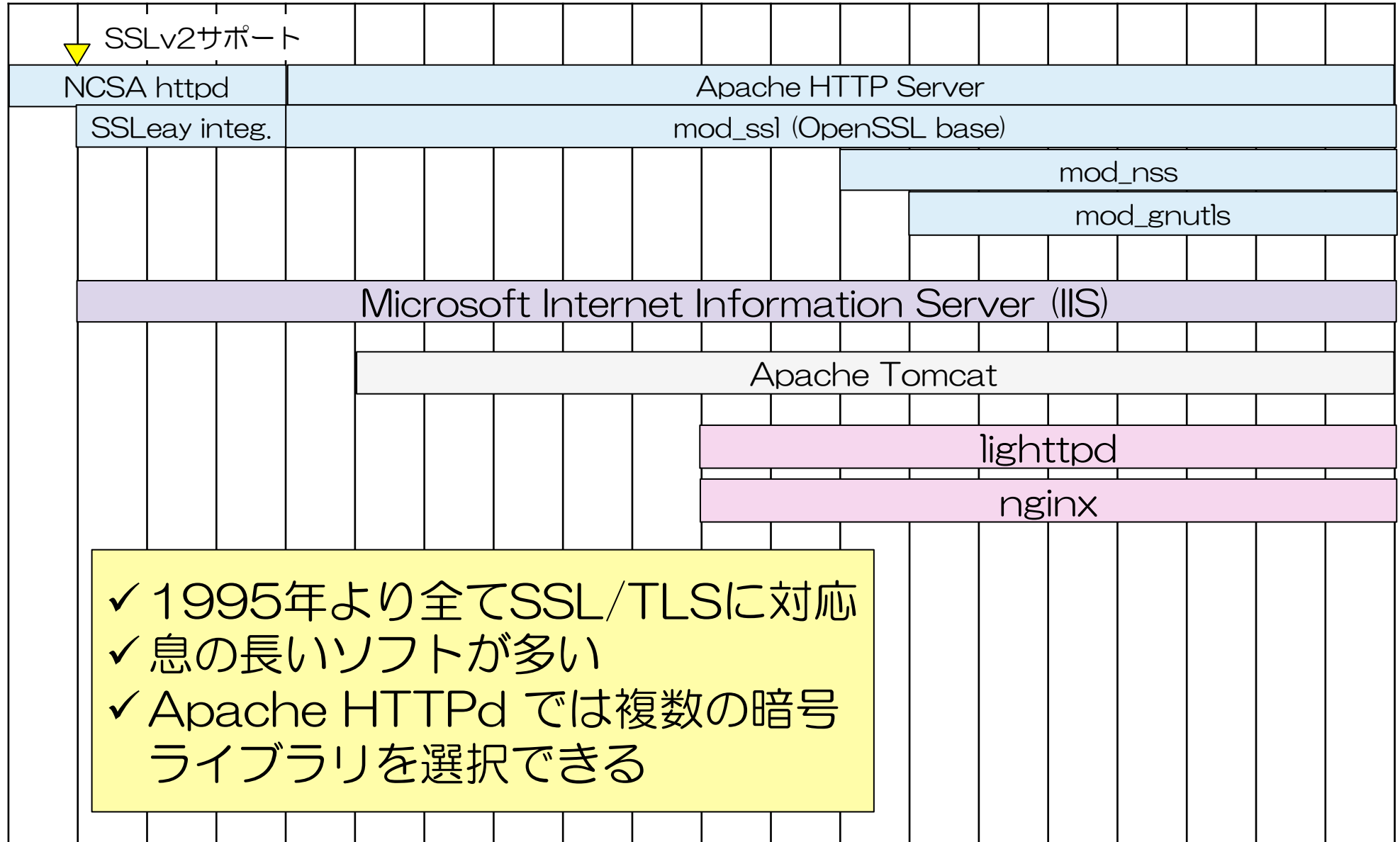
# SSL/TLSバージョン対応とブラウザ(参考：2012年まで)



# SSL/TLS対応 ウェブサーバーの歴史

# SSL対応ウェブサーバー、コンテナの歴史

1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014



- ✓ 1995年より全てSSL/TLSに対応
- ✓ 息の長いソフトが多い
- ✓ Apache HTTPd では複数の暗号ライブラリを選択できる

# SSL/TLS関連の脆弱性の歴史 と脆弱性対応の整理

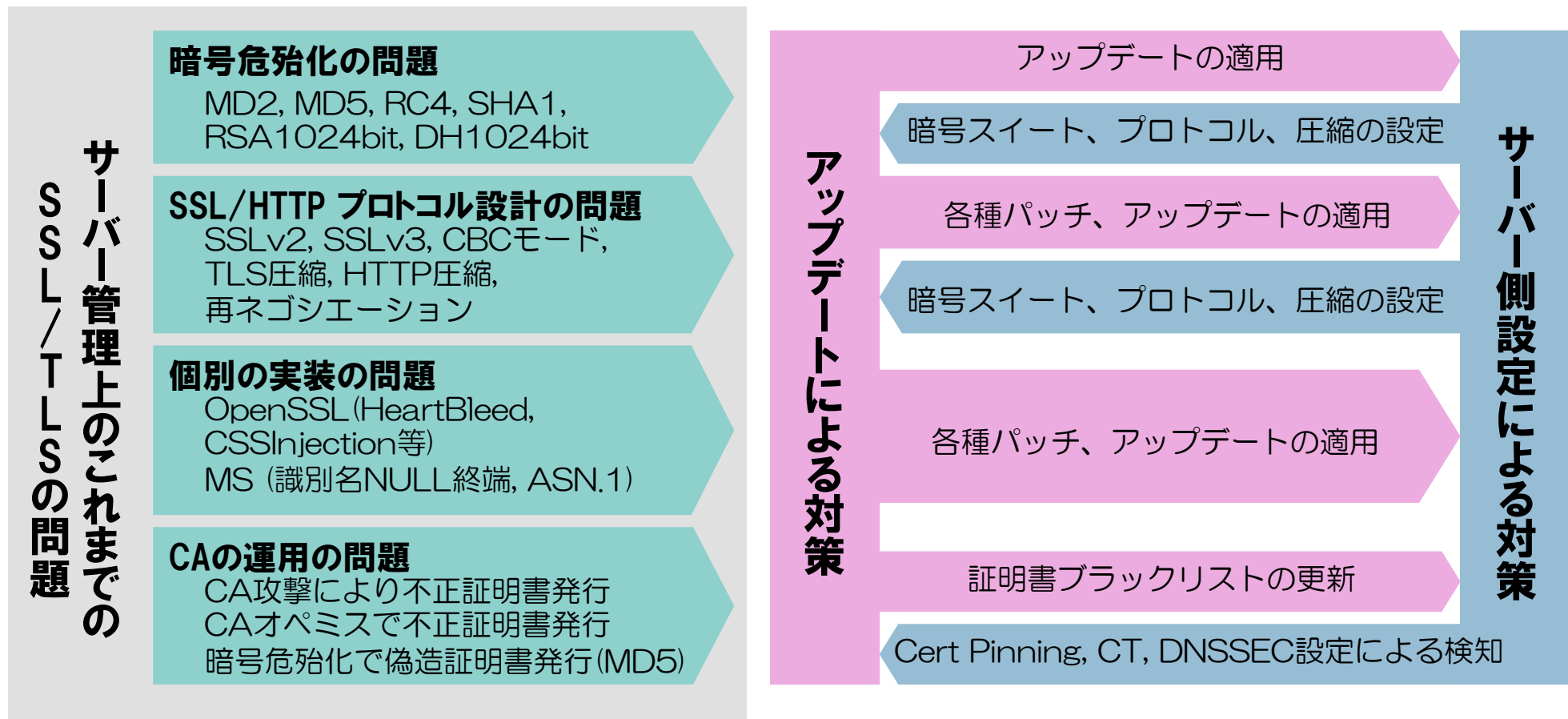
# SSL/TLSの過去の主要な問題と対応

**サーバー設定で回避し  
続けられないものも多数**

**古い脆弱性であっても、  
アップデートだけでは  
解決しない問題が  
多数残っている**

時期	問題・事件	対策
2005.11	OpenSSL SSLv2バージョンロールバック	アップデート
2009.01	RapidSSL MD5衝突偽造中間CA ★	アップデートやPinning
2009.07	NULL終端による証明書ホスト名一致不備 ★	アップデートやPinning
2009.11	再ネゴシエーション脆弱性	アップデート
2011.03	Comodo不正証明書発行(RA攻撃) ★	アップデートやPinning
2011.08	DigiNotar不正証明書発行(RA攻撃)	アップデートやPinning
2011.09	BEAST攻撃 ★	暗号スイート/プロトコル設定(非CBC)
2011.11	Digicert Sdn不正証明書発行(RSA512)	アップデートやPinning
2012.05	FLAMEマルウェア用Windows Terminal ServerによるMD5衝突偽造中間CA, Windows Update攻撃 ★	アップデートやPinning
2012.09	CRIME攻撃	圧縮解除設定(SSL)
2013.01	Lucky13攻撃	暗号スイート/プロトコル設定(GCM利用)
2013.01	TURKTRUST不正証明書発行(オペミス)	アップデートやPinning
2013.03	SSLにおけるRC4暗号危殆化	暗号スイート(非RC4)
2013.03	TIME攻撃	圧縮解除設定(SSL)
2013.06	BREACH攻撃	圧縮解除設定(HTTP gzip)
2013.06	スノーデン氏暴露(NSAの全SSL通信保管)とPFS	暗号スイート/プロトコル設定(ECDHE,DHE使用)
2014.02	Apple OSX, iOS goto fail 問題 ★	アップデート
2014.04	HeartBleed攻撃 ★	アップデート
2014.06	CSSInjection攻撃	アップデート
2014.10	POODLE攻撃	暗号スイート/プロトコル設定(非SSLv3,CBC)
2015.03	FREAK攻撃	暗号スイート(非EXPORT)
2015.03	live.fi、CNNIC/MCS不正証明書発行(運用不備)	アップデートやPinning(CABF BR要件に課題も)
2015.03	パスワード盗聴可能なRC4暗号スイート攻撃	暗号スイート設定(非RC4)

# サーバー管理上のこれまでのSSL/TLSの問題と対策の整理



古い脆弱性であっても、アップデートだけでは解決しない問題が多数残っている  
デフォルト設定でなく、きめ細かい設定で問題に対処する必要がある



## 第6位：個別の実装の問題から

### Apple Mac OS X、iOSのgoto fail脆弱性(2014年2月)

```
    : 中略
if (err = Hash.update(値1)) != 0)
    goto fail;
    goto fail;
if (err = Hash.final(ハッシュ結果) != 0)
    goto fail;
err = VerifySignature(公開鍵, データ, 署名値)
```

この部分は処理せずに  
即ち、公開鍵で認証せずに  
すべて成功して

```
fail:
    メモリの開放等後処理
    return err;          値を返す
```

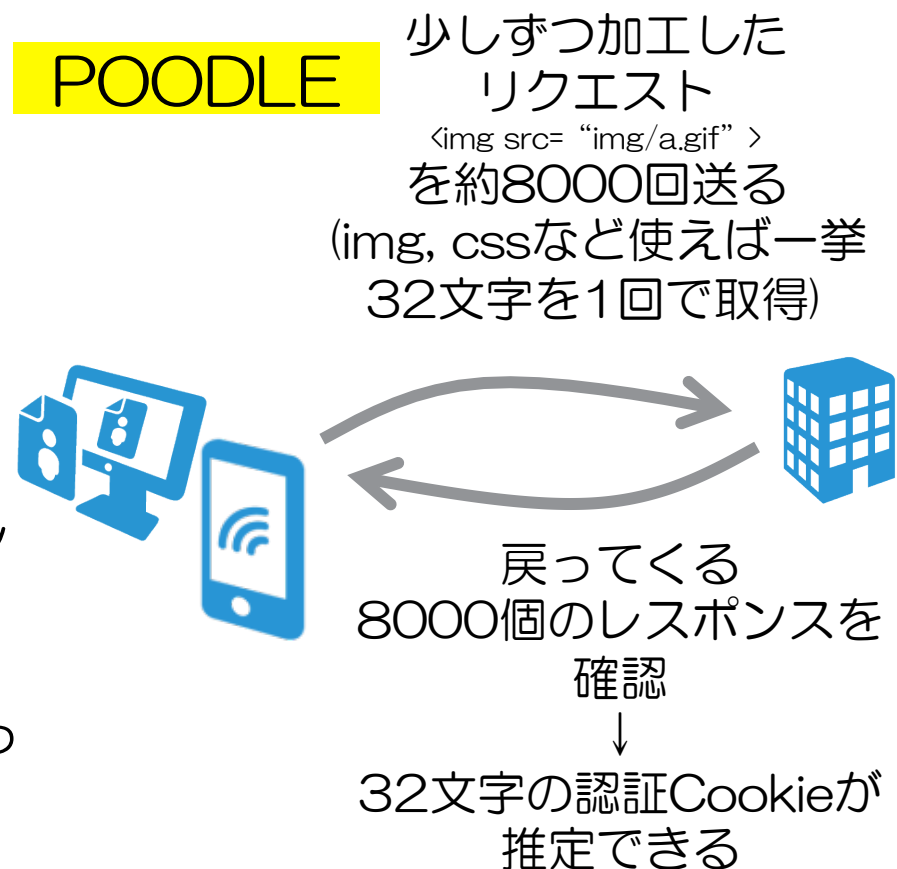
- Apple Secure Transportライブラリのバグ
- 多分、コピペミス？
- DHE、ECDHEを使いSSLv3～TLSv1.1で影響有
- iOSは 6.0.0 ~ 7.0.5までが脆弱
- Mac OS X Mavericks は 10.9.2 で修正
- 2013年10月からあったらしい

## 第5位：個別の実装の問題から(2009年7月) NULL終端による証明書ホスト名一致確認不備

- ① 攻撃対象のドメインが shop.com だったとする。
- ② 適当なドメイン foo.co.jp を取得する。
- ③ CN= “shop.com¥0.foo.co.jp” の証明書をCSRを作り発行要求
- ④ CNの文字種を確認しないCAはそのまま証明書を発行
- ⑤ C言語では ¥0 は文字列の終端記号なので誤った実装では、¥0 を文字列終端として扱うために、④で発行された証明書を shop.com 用の証明書として利用できてしまう。
  
- ⑥ 実際にMicrosoftやFirefoxなど幾つかの実装に影響があり、アップデートが公開された。

# 第4位：プロトコル設計の問題から POODLE攻撃(2014年10月)、BEAST(2011年9月)攻撃など

- BEAST
  - CBCモードの不備を突いて、クッキーを盗む
  - デモではJavaアプレットの脆弱性を突いて実施した。
  - FlashでもWebSocketでも何でも良かった
- POODLE
  - CBCモードとSSLv3のパディング検証をしない事を利用して、BEASTより格段に効率的にクッキーを盗むことができるようになった



参考：SSL v3.0の脆弱性「POODLE」ってかわいい名前だけど何??Padding Oracle On Downgraded Legacy Encryptionの仕組み  
<http://developers.mobage.jp/blog/poodle>

## 第3位：認証局の運用の問題から 認証局の問題による不正証明書発行 (Comodo、DigiNotar、TURKTRUST、CNNICなど)

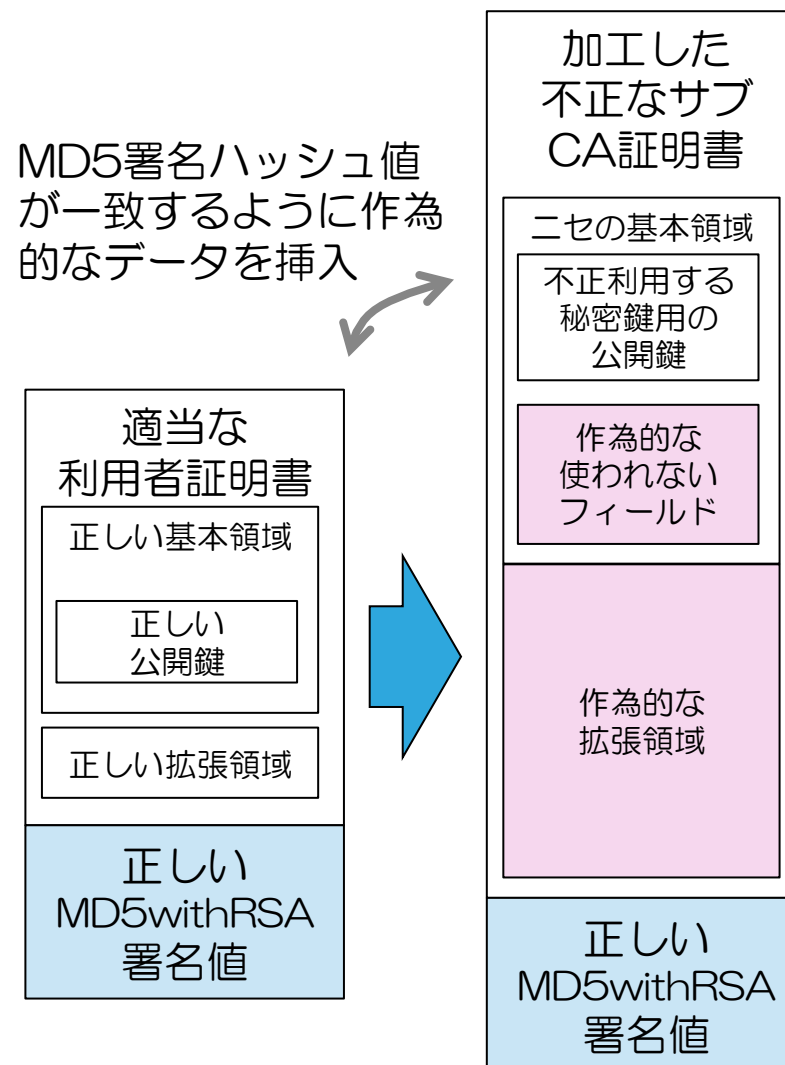
- 登録局(RA)のソフトウェアの脆弱性をつかれて証明書を不正発行
  - 2011年3月、ComodoはRA Windowsアプリの脆弱性を突かれた
  - 2011年8月、DigiNotarも同様にRAの攻撃により不正発行
- 運用のオペレーションミスにより不正発行
  - 2013年1月、トルコのTURKTRUST
- 弱い鍵により
  - 2011年11月、マレーシアDigicert SdnはRSA512bit CAだった
- ガイドラインの不備により
  - 2015年3月、Comodoはドメイン管理者と勘違いし\*.live.fiを発行
- 暗号を破るより簡単に低コストで有効なサーバー証明書が入手できる
- \*.google.com、\*.gmail.com、\*.yahoo.com、\*.microsoft.comなどメジャーなサイトのワイルドが人気があり、盗聴に使われる。

## 第2位：暗号危殆化から

## MD5ハッシュ衝突による証明書偽造

(RapidSSL 2009年1月, FRAMEマルウェア2012年5月)

- RapidSSL
  - 2009年元旦あたり、本物の認証局(RapidSSL)を使って不正サブCA証明書が作れることを示したグループ。
  - 200台のPS3クラスタ
  - 2日以下でできた
- FLAMEマルウェア
  - Windows Terminalサーバー用の証明書から不正証明書を作った
  - 実際にFLAMEマルウェアのコード署名、\*.google.comのサーバー証明書を作成
  - イラクISPの通信が盗聴
- 今は暗号破るよりRA攻撃が簡単？



# 第1位：個別の実装の問題から

## HeartBleed脆弱性(2014年4月 CVE-2014-0160)

RFC 6520 TLS HeartBeat拡張という死活監視機能を悪用して、OpenSSLのサーバーでは、誰でもメモリの一部を取り出すことができ、運が良ければ認証クッキーやIDパスワードなどの情報が盗めた。

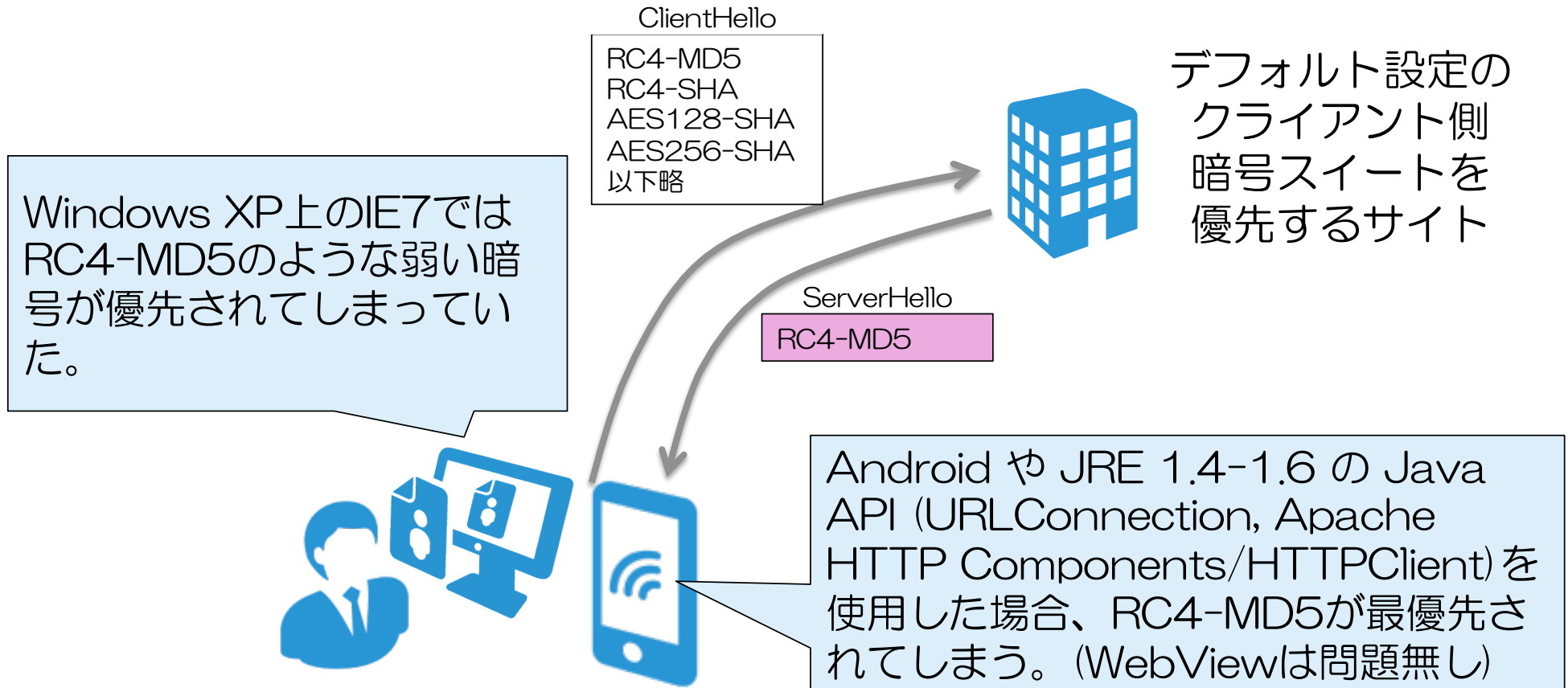
- OpenSSLだけの問題
- HeartBeat要求に対し任意の長さを受け付けるようになっていたので、バッファオーバーランしてサーバーのメモリにある情報が盗めた。
- 運が良ければ認証クッキーやアカウント情報が通常のHTTPSアクセスで誰でも盗めた。



# SSL/TLS関連の脆弱性への 対策技術の最新動向

# 暗号スイートのサーバー側優先

クライアントから送る暗号スイート一覧の順序を優先して接続すると弱い暗号が使われることがある



SSLHonorCipherOrder On等設定してサーバー側を優先する設定を

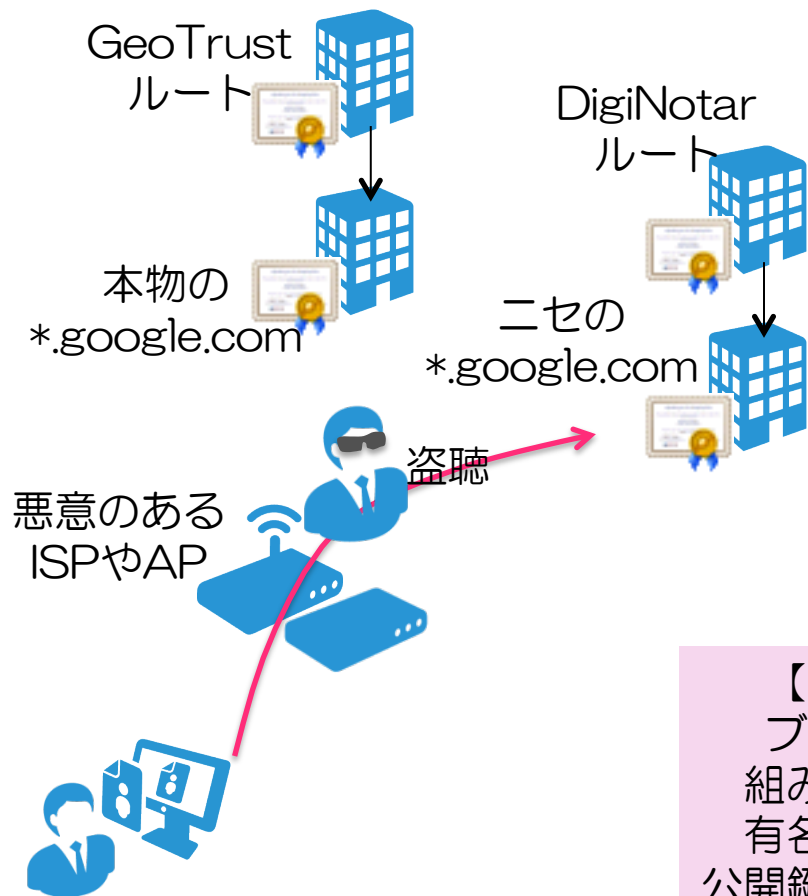
参考 PKIDay 2011 NTT武藤氏 : SSLにおける暗号危殆化サンプル調査の報告 [http://www.jnsa.org/seminar/pki-day/2011/data/03\\_mutoh.pdf](http://www.jnsa.org/seminar/pki-day/2011/data/03_mutoh.pdf)  
自堕落な技術者の日記 JRE 1.4-1.6やAndroidのAPIを使ったHTTPS接続のCipherSuitesのRC4-MD5優先 [http://blog.livedoor.jp/k\\_urushima/archives/1727793.html](http://blog.livedoor.jp/k_urushima/archives/1727793.html)



# Certificate Pinning/Public Key Pinning (不正証明書の拒否)

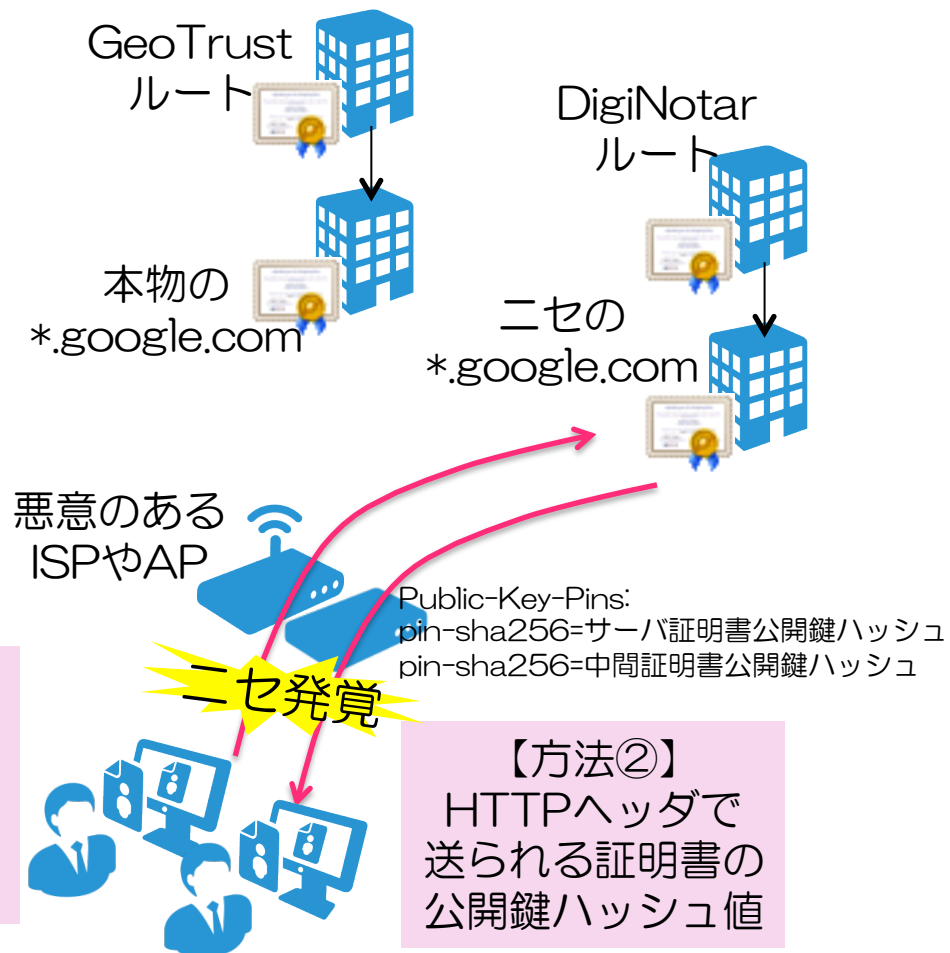
## Pinningは二重証明書対策に効果がある

### Pinningがない場合



【方法①】  
ブラウザに  
組み込まれた  
有名サイトの  
公開鍵ハッシュ値

### Pinningがある場合



【方法②】  
HTTPヘッダで  
送られる証明書の  
公開鍵ハッシュ値

# Certificate Pinning/Public Key Pinning (不正証明書の拒否)

- インターネットドラフトで規定されている  
<https://tools.ietf.org/html/draft-ietf-websec-key-pinning-21>
- HTTPヘッダを追加することで使用する証明書チェーンの不正入替を防ぐ
- ヘッダのキー：Public-Key-Pins
- ヘッダの値(例)：pin-sha256=証明書の公開鍵のSHA256ハッシュ値のBase64値
- ヘッダを作成してくれるサイトを活用するとよい  
<https://projects.dm.id.lv/s/pkp-online/calculator.html>

### JavaScript Public-Key-Pins (HPKP) calculator v1.0.2

[Project website](#) | [GitHub page](#) | Based on [Internet standard Draft](#)

Copyright © 2014 Dāvis Mošenkovs

Certificates and/or Certificate Signing Requests (in PEM format; newline separated) containing public keys to be

```
VRBFB0wOzA5bG9uY29uHR8cDovL2kybC51c2VydHJ1c3QuY29tL1VUFT11VURVS  
Rnlyc3Q1SCFyZhdnbnNlY3J1eHQGCCsGAQUFBwEwBBggwZJA9BgggrBgEFBQcwAoYx  
oHR8cDovL2kydC51c2VydHJ1c3QuY29tL1VUFTkfkZFRyd0N0L2VydrrVyXN0BmNm  
oDAlBggrBgEFBQcwAYTZ6lIR8cDovL29jc3Aud0NlcnRyd0N0LmhbvTANBgkqhkiG  
9w8BAQJFAADCAQEATIPuS3z2hYt0xApuc54ym0e0gTrZs8gyTAGEKM/yXA4HRJML  
lhoh45g81A5nSYEvj0NTrDa76AxtPxv8916M0TgQ7ahY8QzLGDYkLUNrANrkT  
QLmT7BR5LFPkI+idyf0hcpxrVqDDFY1opYcfc53rWn08aXFABFxc0EUIEL4eNe9  
itg5xt8Jt1ooqQ04K00tzb88G1oRPjj028s0ec8z0glI9rJjnbUcRkLy7wVYc0FV  
r7BmXt0mdcCekbYrDyql0QIN4+mi.tF3A884sooL4dmHGSYKrlb0Cpr10nCIY+2v+  
lhb/NX5UR6g83EMnqZsF157ANED0MNDywxFa4Q=
```

-----END CERTIFICATE-----

Time for clients to remember these pins:

Pin these keys also for all subdomains (use with caution!)

Create also HSTS header (with same values)

I have read and accepted [license agreement \(GNU General Public License\)](#)

SSLサーバー証明書から最上位  
の中間CA証明書までの証明書  
チェーンをPEM形式で入力すれ  
ばヘッダを自動作成してくれる

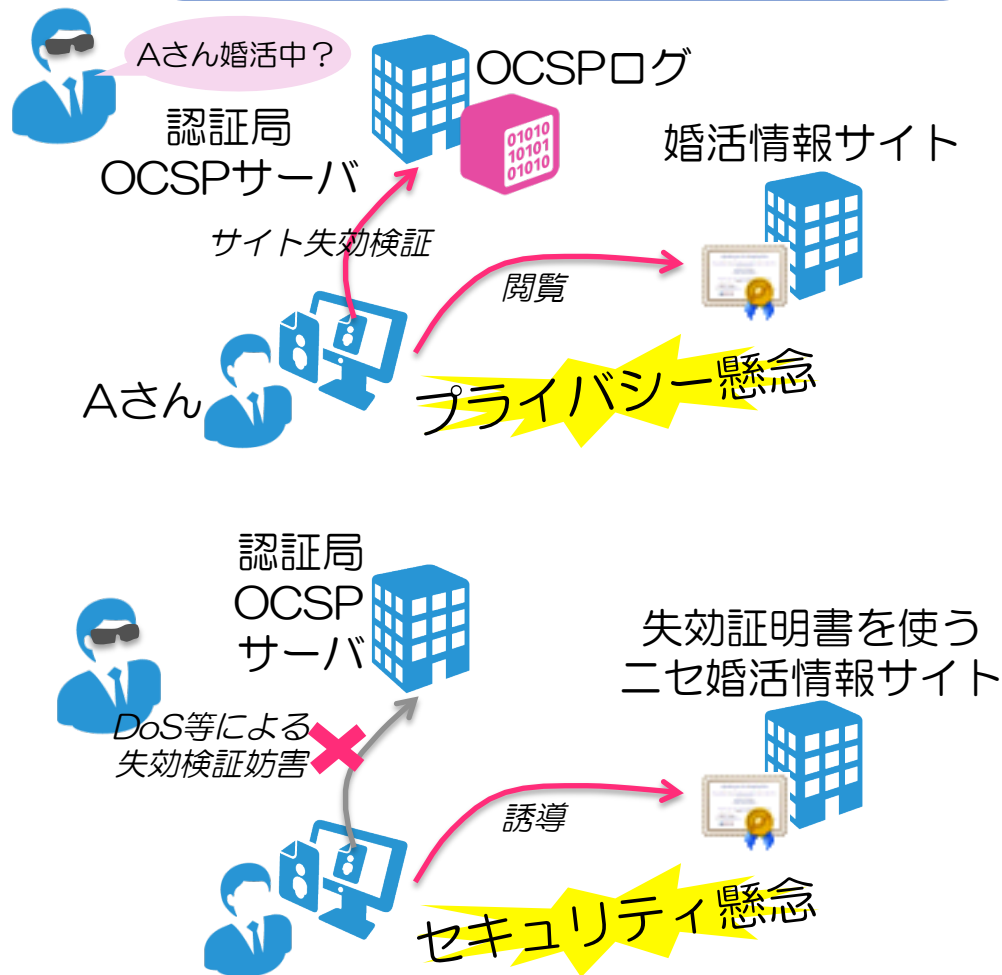
### Public-Key-Pins HTTP header:

```
Public-Key-Pins: pin-sha256="lCpPfbkrLJ3EcVFakeip0+44VaoJUymbn0aEUK7tEU=";  
pin-sha256="TUDnr0MEoJ3of7+YliBMBVFB4/gJsv5z07Ix0D9+YoWI=";  
pin-sha256="68YTDKOLH2QWSUUVgfmnsnkvGFAtbM7Ljsrzn3v/gfY=";  
pin-sha256="owdYGO+3vooMgfjI2BhXs6mm1c5NtAlEjd/QncQNbKQ="; max-age=31536000;  
includeSubDomains
```

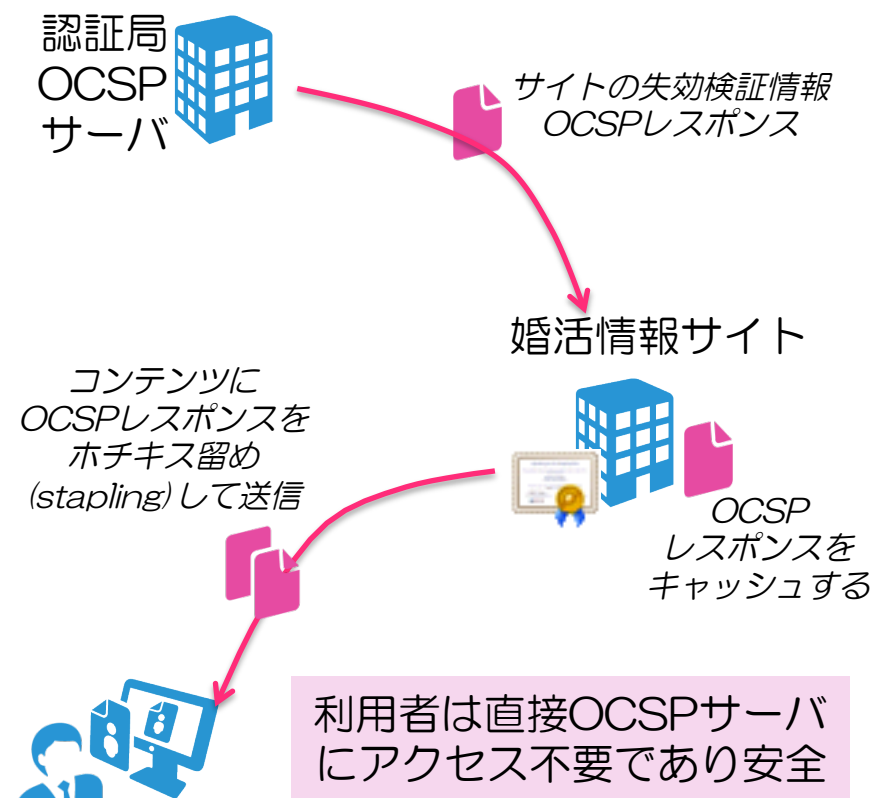
# OCSP Stapling(RFC 6066TLS拡張8章)の意義と仕組み

- ① 近年、CRLは数十MBまで膨大しモバイル/組込み環境の失効検証に向かずOCSPを使う方向に
  - ② ブラウザがOCSPに接続できないと失効如何にかかわらず証明書有効になってしまう
  - ③ OCSPサーバのログで、どのIPの人がどのサイトを閲覧したかという情報を認証局も知ってしまう
- OCSP Staplingでこれを解決

## OCSP Staplingがない場合

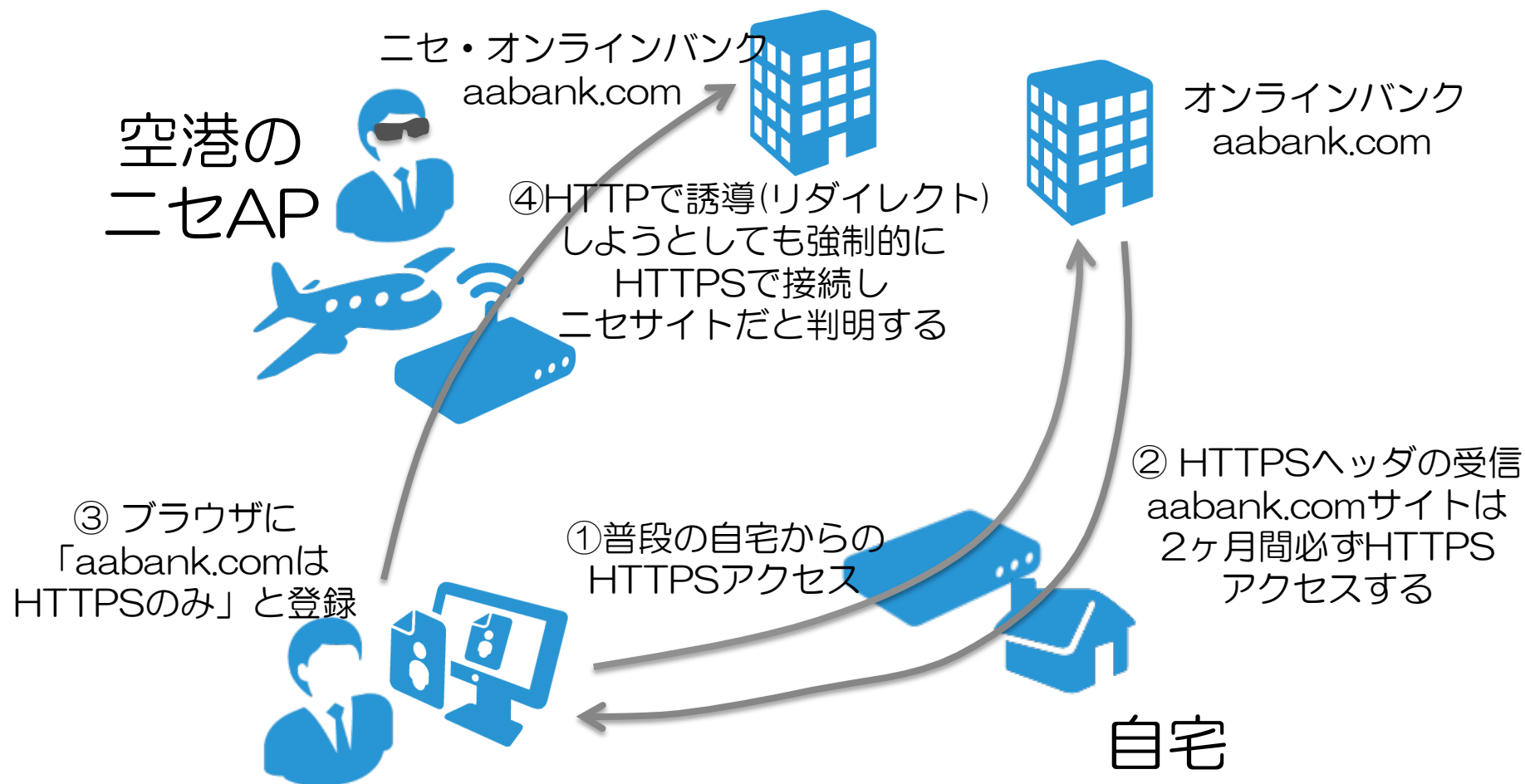


## OCSP Staplingがある場合























# HSTSによる強制的なHTTPS接続

RFC 6797 HTTP Strict Transport Security (HSTS)  
普段、HTTPSでアクセスしているオンライン banking サイトに、空港など外出先でアクセスした場合、そのアクセスポイントが不正APで、ニセのHTTPで作ったサイトに誘導しようとしても、HSTS機能が有効であれば、二回目以降の接続は自動的にHTTPSサイトへ接続します。



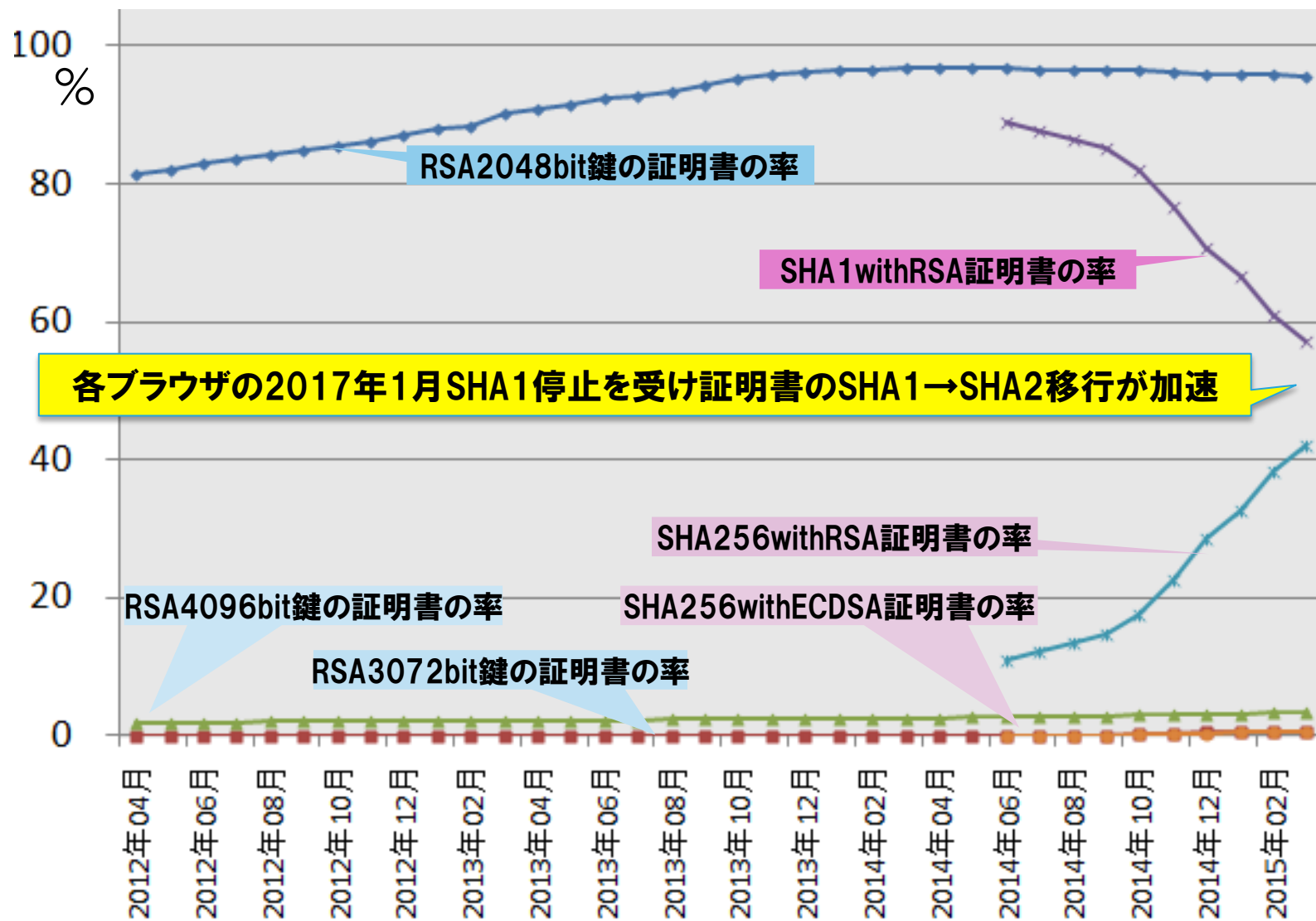
## Microsoft製品、Google ChromeのSHA1証明書からの移行(1/2)

- Windowsルート証明書プログラムのルートCA配下は2016年1月1日以降、SHA1証明書を発行できない。
- Windows製品では有効期限が2017年1月1日以降の証明書を受理しないためエラーとなる。
- Google ChromeはSHA1証明書の有効期限により、2015年1月以降のリリース版より下表の警告表示を開始し、2017年1月以降は受理しない。

Google Chrome		SHA1証明書の有効期限			
Chromeバージョン	安定版リリース日	2015.12.31まで	2016.05.31まで	2016.12.31まで	2017.01以降
<b>38</b>	2014.10.08	 https://	 https://	 https://	 https://
<b>39</b>	2014.11.18	 https://	 https://	 https://	 https://
<b>40</b>	2015.01.21	 https://	 https://	 https:// ☆	 https:// ☆
<b>42</b>	2015.04中旬	 https://	 https:// ☆	 https:// ☆	 https:// ☆
2017.01直後	2017.01中旬	 https:// ☆	 https:// ☆	 https:// ☆	 https:// ☆

記号：☆：Googleのポリシーにより、★：証明書期限切れにより

# Microsoft製品、Google ChromeのSHA1証明書への警告(2/2)



データ出典：SSL Pulse(<https://www.trustworthyinternet.org/ssl-pulse/>)

2015年6月頃にはSHA1証明書とSHA256証明書の利用数が逆転しそう

# Certificate Transparency(1)

- ✓ 2011年、DigiCertやComodoなどで google.comドメイン不正証明書が発行されイラクの通信が盗聴  
その対策の一環として提案。
- ✓ 証明書発行の公開監査情報として、  
証明書チェーンと登録日時を追記し署名される
- ✓ 2011年のGoogleの人の論文に基づく
- ✓ 実験RFC 6962になった
- ✓ Google Chromeのみが実験に対応
- ✓ ChromeはEV証明書の発行要件にした
- ✓ DigiCert, GlobalSignが先行対応
- ✓ 他の海外認証ベンダも対応中

# CT対応のサイト (=GlobalSign or DigiCert)

鍵を右クリック

例の出典  
https://www.digicert.com/



「公開監査が可能」  
=  
Certificate Transparency  
対応のサイト





# CT対応のサイト(=GlobalSign or DigiCert)

Signed Certificate Timestamp (SCT)  
(RFC 3161とは無関係の  
CTログエントリに対する署名情報)

DigiCert、GlobalSignには  
見慣れない拡張が！！  
(embeddedSCT)

署名付き証明書タイムスタンプビューア

2: 埋め込み済み、確認済み

検証ステータス	確認済み
発行元:	埋め込み済み
バージョン	V1
ログ名	Google 'Aviator' log
ログ ID	68 F6 98 F8 1F 64 82 BE 3A 8C EE B9 28 1D 4C FC 71 51 5D 67 93 D4 44 D1 0A 67 AC BB 4F 4F FB C4
発行:	2014年10月15日水曜日 8:25:08
ハッシュアルゴリズム	SHA-256
署名アルゴリズム	ECDSA
署名データ	30 44 02 20 11 34 9A 59 2C 9D 3B D3 8B 9A 58 18 37 24 55 F3 9D 0E CA 98 96 6B 8F C7 A2 E4 D8 BF 00 CE 40 FD 02 20 11 24 11 AB 62 7F B2 88 F0 6D 70 C0 FD A0 65 B5 B6 03 46 1F 10 30 ED F5 6D 7E 89 7B BA 20 32 64

閉じる

証明書

表示(S): <すべて>

フィールド	
サブジェクトの別名	DNS Name=www.digicert.com,...
拡張キー使用法	サーバー認証 (1.3.6.1.5.5.7.3.1), ...
CRL 配布ポイント	[1]CRL Distribution Point: Dis...
証明書ポリシー	[1]Certificate Policy:Policy Id...
機関情報アクセス	[1]Authority Info Access: Ass...
1.3.6.1.4.1.11129.2.4.2	04 81 f1 00 ef 00 76 00 a4 b9...
キー使用法	Digital Signature, Key Enciph...

04 81 f1 00 ef 00 76 00 a4 b9 09 90 b4 18 58 14  
87 bb 13 a2 cc 67 70 0a  
3c 35 98 04 f9 1b df b8 <5.....  
e3 77 cd 0e c8 0d dc 10 w.....  
00 00 01 49 10 fa bd 89 .I.....  
00 00 04 03 00 47 30 45 ...G0E  
02 20 66 d7 67 79 f4 aa .fgy..  
d3 b8 c6 9f 03 01 bf cd .....

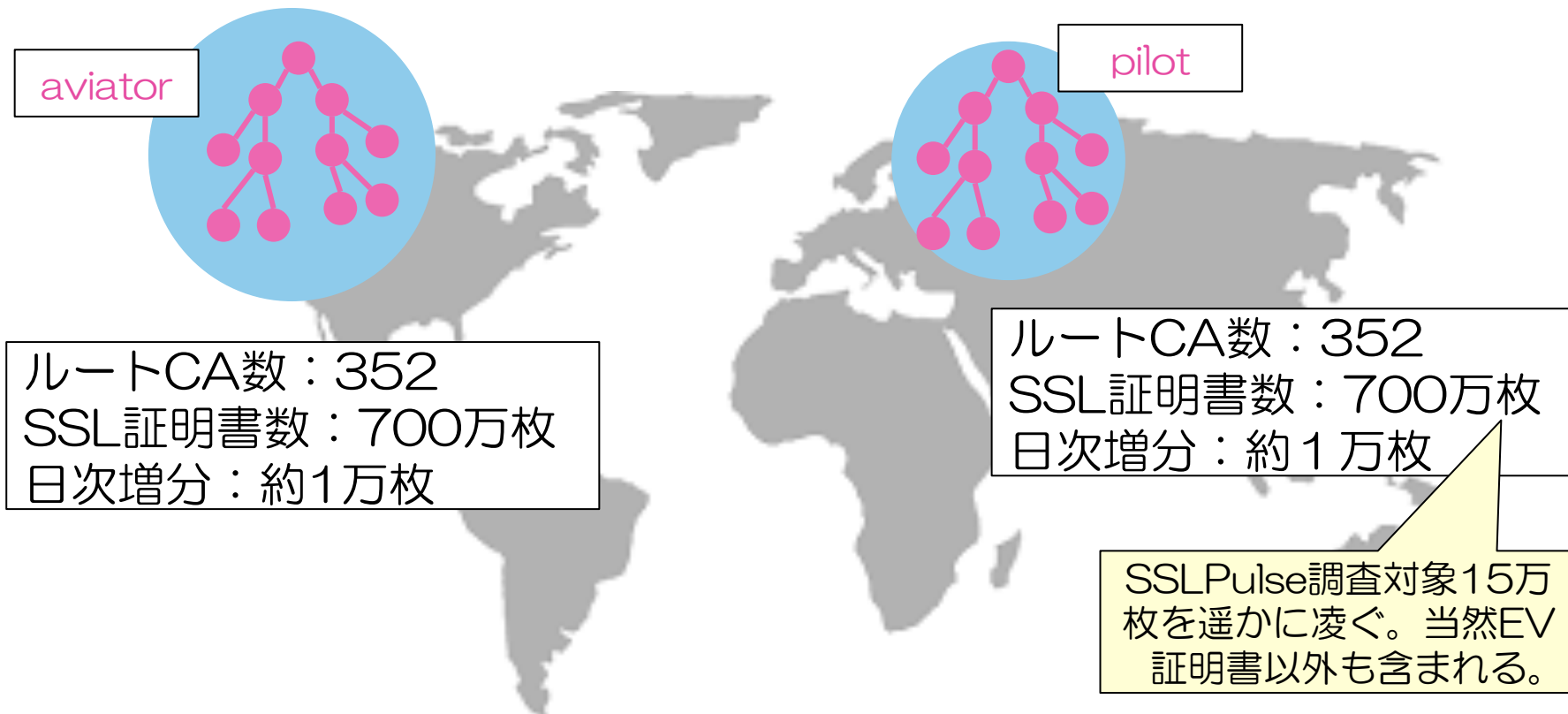
プロパティの編集(E)... ファイルにコピー(C)...

証明書の詳細について表示します。

OK

# Certificate Transparencyログサーバー(2015年4月2日時点)

Googleは世界2拠点でログサーバーをテスト運用している



Googleは3拠点を計画、他の組織がログサーバーを提供してもよい

# まとめ



# まとめ

- SSL/TLS生誕20周年でその歴史を振り返った
  - プロトコル、ライブラリ、ブラウザ、サーバー
- SSL/TLSの脆弱性の歴史
- SSL/TLSの脆弱性対策の整理
  - 4種類に分類できる、種類毎に対策方法も違う
    - 暗号期危殆化の問題
    - TLS/HTTPSプロトコル設計の問題
    - 個別の実装の問題 (OpenSSL、Windows、Java(JSSE, JCE))
    - CA運用上の問題
- SSL/TLSの脆弱性対策技術の最新動向
  - HSTS, OCSP stapling, Certificate Transparency等の解説

20歳になり成人して、社会の荒波に揉まれているかのようです。この1年、SSL/TLSやPKIに関して脆弱性が数多く出ている傾向にあります。一つ一つ手当をして今後も使われていくでしょう。

PKI Day 2015の私の講演とパネルの補足情報ページは以下にあります。

<http://www9.atwiki.jp/kurushima/pages/123.html>

本講演後、リンク等補足情報を提供する予定です。参考になさってください。

