ITエンジニアとして 知っておいてほしい、 電子メールという 大きな穴



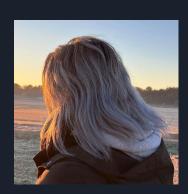
永見 拓人

X: GitHub: @logica0419

@logica0419

自己紹介

- 永見 拓人 (ハンドルネーム: logica)
- 千葉工業大学 情報科学部 情報ネットワーク学科 3年
- Webバックエンド / インフラの開発が専門
- セキュキャン歴
 - 2022 開発コース Y3ゼミ 修了
 - 山梨2022 受講
 - 2024 専門講義 Bクラス チューター



なぜ、今メールなのか

「メールなんて**今どき 使わないじゃん**」と 思っていませんか?

平成28年~令和2年のもの

罪名	14	15	16	17	18	17年から18年にかけての場面
	105	145	142	277	703	426 (153.8%)
コンピュータ・電磁的記録対象犯罪	30	55	55	73	129	56 (76.7%)
電子計算機使用詐欺	18	34	42	49	63	14 (28.6%)
電磁的記録不正作出・毀棄	8	12	8	17	56	39 (229.4%)
電子計算機損塊等業務妨害	4	9	5	7	10	3 (42.9%)
ネットワーク利用犯罪	1,471	1,649	1,884	2,811	3,593	782 (27.8%)
詐欺	514	521	542	1,408	1,597	189 (13.4%)
児童實春・児童ポルノ法違反(児童實春)	268	269	370	320	463	143 (44.7%)
児童質春・児童ポルノ法違反(児童ポルノ)	140	102	85	136	251	115 (84.6%)
商標法違反	37	95	82	109	218	109 (100.0%)
古少年保護育成条例違反	70	120	136	174	196	22 (12.6%)
わいせつ物頒布等	109	113	121	125	192	67 (53.6%)
著作権法 違反	66	87	174	128	138	10 (7.8%)
その他	267	342	374	411	538	127 (30.9%)
合計	1 606	1,849	2,081	3,161	4,425	1,264 (40.0%)

メールが何らかの形で絡んでいるものは どれでしょう?

罪名	14	15	16	17	18	17年から18年にかけての場面
	105	145	142	277	703	426 (153.8%)
コンピュータ・電磁的記録対象犯罪	30	55	55	73	129	56 (76.7%)
電子計算機使用詐欺	18	34	42	49	63	14 (28.6%)
電磁的記録不正作出・毀棄	8	12	8	17	56	39 (229.4%)
電子計算機損塊等業務妨害	4	9	5	7	10	3 (42.9%)
ネットワーク利用犯罪	1,471	1,649	1,884	2,811	3,593	782 (27.8%)
詐欺	514	521	542	1,408	1,597	189 (13.4%)
児童實春・児童ポルノ法違反(児童實春)	268	269	370	320	463	143 (44.7%)
児童質春・児童ポルノ法違反(児童ポルノ)	140	102	85	136	251	115 (84.6%)
商標法違反	37	95	82	109	218	109 (100.0%)
古少年保護育成条例違反	70	120	136	174	196	22 (12.6%)
わいせつ物頒布等	109	113	121	125	192	67 (53.6%)
著作権法 違反	66	87	174	128	138	10 (7.8%)
その他	267	342	374	411	538	127 (30.9%)
合計	1 606	1,849	2,081	3,161	4,425	1,264 (40.0%)

メールが何らかの形で絡んでいるものはどれでしょう?

罪名 年	14	15	16	17	18	17年から18年にかけての機動
不正アクセス禁止法違反	105	145	142	277	703	426 (153.8%)
コンピュータ・電磁的記録対象犯罪	30	55	55	73	129	56 (76.7%)
電子計算機使用詐欺	18	34	42	49	63	14 (28.6%)
電磁的記録不正作出・鈴蕪	8	12	8	17	56	39 (229.4%)
電子計算機損塊等業務妨害	4	9	5	7	10	3 (42.9%)
ネットワーク利用犯罪	1,471	1,649	1,884	2,811	3,593	782 (27.8%)
詐欺	514	521	542	1,408	1,597	189 (13.4%)
児童買春・児童ポルノ法違反(児童買春)	268	269	370	320	463	143 (44.7%)
児童買春・児童ポルノ法違反(児童ポルノ)	140	102	85	136	251	115 (84.6%)
商標法違反	37	95	82	109	218	109 (100.0%)
吉少年保護育成条例違反	70	120	136	174	196	22 (12.6%)
わいせつ物頒布等	109	113	121	125	192	67 (53.6%)
著作権法 違反	66	87	174	128	138	10 (7.8%)
その他	267	342	374	411	538	127 (30.9%)
合計	1 606	1 849	2,081	3,161	4,425	1,264 (40.0%)

メールが何らかの形で絡んでいるものは どれでしょう?

何気なく使っているメール、 サイバー犯罪の温床なんです! けての場面 153.8%

76.7%) 28.6%

229.4%

42.9%) 27.8%

13.4%)

(44.7%)(84.6%)

商標法違反	37	95	82	109	218	109 (100.0%)
青少年保護育成条例違反	70	120	136	6 174	196	22 (12.6%)
わいせつ物頒布等	109	113	121	125	192	67 (53.6%)
整作権法 違反	66	87	174	128	138	10 (7.8%)
その他	267	342	374	411	538	127 (30.9%)
th control of the con	1,606	1,849	2,081	3,161	4,425	1,264 (40.0%)

なぜ電子メールは 様々な**サイバー犯罪で 用いられる**のだろうか?

そのためには、メールの現状を知る必要がある

電子メールのおこり

- 1965年頃から存在する
 - 非常に長く使われている
 - インターネットがない時代からある
 - インターネットのもとになった技術の一つ
- 現在の <u>fuga@hogehoge.net</u> のような形のアドレスが 使われ始めたのは、1971年から
- 1980年代以降に、RFCとして送受信方法の共通化が 行われた

電子メールの特徴

- 特定のサービス等に依存しない通信
 - ここがSNSとは違う
- プライベートな 1対1 or 1対多 の通信
 - Webサイト等とは少し別の技術
- 非常に広く用いられる
 - メールアドレスを持たない人、周りでいますか?
 - ほとんどのサイトでログインに使用できる
 - 企業同士のやり取りは、多くが電子メール

電子メールのセキュリティ懸念

- 単純に**広く使われている**ので、犯罪に使いやすい
 - ターゲットが多いので、数打て戦法が有効に
 - SNSをあまり見ない**高齢者や企業さえ標的に**できる
- インターネットの起こり時代に定められた、非常に 脆弱で機能が少ない仕組みで送られている
 - 素の状態では送信元偽装・改ざんとかし放題
 - ≓ Secure by Defaultでない
 - ただ使っているだけで安全…とはならない

メールはどのように絡んでいるのか

罪名	14	15	16	17	18	17年から18年にかけての場別
不正アクセス禁止法違反	105	145	142	277	703	426 (153.8%)
コンピュータ・電磁的記録対象犯罪	30	55	55	73	129	56 (76.7%)
電子計算機使用詐欺	18	34	42	49	63	14 (28.6%)
電磁的記録不正作出・鏡棄	8	12	8	17	56	39 (229.4%)
電子計算機損塊等業務妨害	4	9	5	7	10	3 (42.9%)
vットワーク利用犯罪	1,471	1,649	1,884	2,811	3,593	782 (27.8%)
詐欺	514	521	542	1,408	1,597	189 (13.4%)
児童買春・児童ボルノ法違反(児童買春)	268	269	370	320	463	143 (44.7%)
児童實春・児童ポルノ法違反(児童ポルノ)	140	102	85	136	251	115 (84.6%)
商標法違反	37	95	82	109	218	109 (100.0%)
青少年保護育成条例違反	70	120	136	174	196	22 (12.6%)
わいせつ物頒布等	109	113	121	125	192	67 (53.6%)
茅作権法 違反	66	87	174	128	138	10 (7.8%)
その他	267	342	374	411	538	127 (30.9%)
合計	1,606	1,849	2,081	3,161	4,425	1,264 (40.0%)

メールはどのように絡んでいるのか

<u> </u>							
罪名 年	14		フィ	11=1	ノグメ	— II,7	7.
不正アクセス禁止法違反			ノ 」	ノ ノ ノ			-
コンピュータ・電磁的記録対象犯罪			パス「	フート	で奪	う	
電子計算機使用詐欺	ា ៦						
電磁的記録不正作出・鈴蕪 ^で	8		警告:	メール	しを装	ってす	ァカヴ
電子計算機損壞等業務妨害	4	_					
トットワーク利用犯罪	1,471		リセ	ットを	E強要	しアカ	」ウン
詐欺	514						
児童實春・児童ポルノ法違反(児童實春)	268		乗っ耳	収る			
児童賢春・児童ポルノ法違反(児童ポルノ)	140		•				
商標法達反	37						
青少年保護育成条例違反	70	120	136	174	196	22	(12.6%)
わいせつ物頒布等	109	113	121	125	192	67	(53.6%)
著作権 法違反	66	87	174	128	138	10	(7.8%)
その他	267	342	374	411	538	127	(30.9%)
合計	1,606	1,849	2,081	3,161	4,425	1 264	(40.0%)

| |メールはどのように絡んでいるのか

罪名 ————————————————————————————————————	14	15	16	17	18	17年から18年にかけての場割
正アクセス禁止法違反	105	145	142	277	703	426 (153.8%
ンピュータ・電磁的記録対象犯罪	30					
電子計算機使用詐欺 💂	15		メーリ	しの添	ミ付フ	ァイルでマ
電磁的記録不正作出・毀棄						
電子計算機損壞等業務妨害			ウェフ	アを送	り付	ける
ットワーク利用犯罪:	1,471					
詐欺	514	•	ヌーノ	レを媽	評介と	して広がっ
児童賢春・児童ポルノ法違反(児童賢春)	268					
1.5	140		\Box	ウィル	ノスタ	1/ H H//
児童買春・児童ポルノ法違反(児童ポルノ)	140	'	\cup ' \setminus ' .	/ / L	ノヽビ	1 F <i>PX</i>
児童買春・児童ポルノ法違反(児童ポルノ) 商標法違反	37	(V 1 \ .	- I / L		1 F <i>PX</i>
		120	V I .	1170	,	
商標法違反	37					22 (12.07
商標法達反 青少年保護育成条例違反	37 70	120	. 100	1170	1.00	67 (53.6%
商標法違反 者少年保護育成条例違反 わいせつ物類布等	37 70 109	113	121	125	192	67 (53.6%

15

145

16

142

55

42

メールはどのように絡んでいるのか

	罪名 年	14
不ī	Eアクセス禁止法違反	105
(د	/ピュータ・電磁的記録対象犯罪	30
	電子計算機使用詐欺	18
	電磁的記録不正作出・鈴棄	8
	電子計算機損塊等業務妨害	4
ት :	- トワーク利用犯罪	1,4
	詐欺	
	児童賢春・児童ポルノ法違反(児童賢春)	2.
	児童買春・児童ポルノ法違反(児童ポルノ)	140
	商標法違反	37
	青少年保護育成条例違反	70
	わいせつ物頒布等	109
	著作権法違反	66
	その他	267
88	F	1 606

• フィッシングメール

17

277

49

送り主を大手サービスだと偽り お金を払わせる

18

703

129

63

17年から18年にかけての場面

426 (153.8%)

56 (76.7%)

14 (28.6%)

家族のふりをしてメールを出し オレオレ詐欺をする

幅広い犯罪に利用されてしまう 電子メールという通信方法を 学ぶことは大事!

自分たちが使わないからこそ、狙われたりします

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

プロトコルスタックとRFC

プロトコル

- コンピューター同士が通信する時の約束事
 - どんなデータを渡す必要があるか
 - どんな順番でデータを渡すか
 - それぞれのデータはどんなサイズか
 - 。 …などなど
- ◆ 人間界の「言語」と同じものだと思ってください。
- 世界中のあらゆる機器 (どんな企業製でも) が相互通信できるインターネットを作るのに必要不可欠です

プロトコル

プロトコルを使って「会話」する



プロトコル

プロトコルが無いと…



言ってることが わからないので、 通信できない!



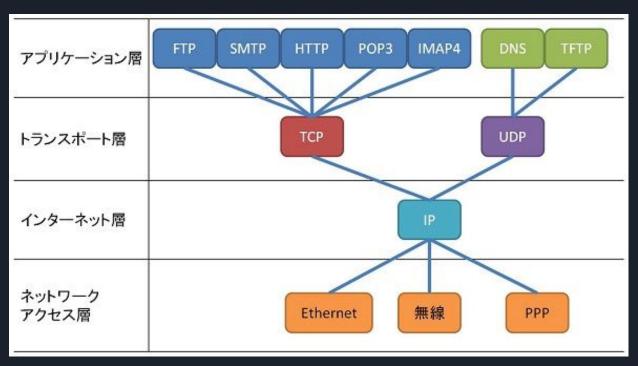


プロトコルは、**通信の** 「**カオス」を解決**してくれる すごいやつです

プロトコルという仕組みに感謝しましょう

プロトコルスタック

様々なプロトコルを、**積み重ねて**ひとまとめにしたもの



なぜ「積み重ねる」(階層化する)のか

- 通信に必要な知識は膨大
 - 極論を言えばみんなが指から電気信号を出して通信 しなければいけなくなる
 - 人間に分かりやすくするため / あらゆる場面に 対応するために、様々な知識が必要
- 関心の分離をし、専門分野を分ける
 - LANケーブルを扱う人と、Webページを作る人は 一緒ではない

OSI参照モデル

今のインターネットを支えるプロトコルの、**よく使われる階**

層分け方法

7	アプリケーション層
6	プレゼンテーション層
5	セッション層
4	トランスポート層
3	ネットワーク層
2	データリンク層
1	物理層

階層ごとの役割

7	アプリケーション層	アプリケーション間のやり取り
6	プレゼンテーション層	データの表現形式
5	セッション層	接続の手順
4	トランスポート層	データ通信の制御
3	ネットワーク層	インターネットワークでの通信
2	データリンク層	同一ネットワーク上での通信
1	物理層	ケーブルや電気信号やコネクタなど

それぞれの階層で、

様々な人が頑張って**プロトコルを 守る**ことで、インターネットは 成り立っています

インターネットはこれらの頑張りによる奇跡の産物です

プロトコルを決める過程 - RFC

- **RFC** (Request for Comments)
 - インターネットにおけるプロトコルの仕様書のこと
- RFCのフェーズ
 - Internet Draft 仕様を煮詰める
 - Proposed Standard 標準化すべきと認められた
 - Draft Standard 多くの機器で使える
 - Standard インターネットで広く使われる
- 「決めてから普及」ではなく、「普及したら標準」

プロトコルを決める過程 - RFC

- IETFという団体で議論され、標準化される
- 思想
 - **オープン**である Request for "Comments"
 - RFCはメーリングリストで決まる
 - 仕様よりも実装重視
 - RFCのフェーズも、この思想の下にある
- 内容は**改定できない**
 - 新しいRFCの発行と、古いRFCの無効化で変更

RFCの例 - http

<u>https://www.rfc-editor.org/rfc/rfc9110.html</u> で公開

RFC 9110 HTTP Semantics

Abstract

The Hypertext Transfer Protocol (HTTP) is a stateless application-level protocol for distributed, collaborative, hypertext information systems. This document describes the overall architecture of HTTP, establishes common terminology, and defines aspects of the protocol that are shared by all versions. In this definition are core protocol elements, extensibility mechanisms, and the "http" and "https" Uniform Resource Identifier (URI) schemes.

This document updates RFC 3864 and obsoletes RFCs 2818, 7231, 7232, 7233, 7235, 7538, 7615, 7694, and portions of 7230.

インターネットが好きなら 是非**プロトコルスタック**と RFCを理解して使いましょう!

RFCを読む会…なんてイベントもあるみたいですよ

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- ◆ メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

メールが送られる仕組み

hoge@gmail.com から fuga@icloud.com への メール送信を図解してみる

メールが送られる仕組みを理解しましょう

登場人物たち

ユーザー

メールサーバー

DNSサーバー







ユーザーが送信する (サブミッション)











ユーザーが送信する (サブミッション)



インターネット







メールサーバー

- メールの **送信 / 受信 / 保存** 全てを執り行う
- メールの受信
 - あらゆるメールを受け取る
 - 自分宛のメールならユーザーごとの メールボックスに保存し、提供する
- メールの送信
 - 自分宛でなければ正しい宛先に届ける
 - リレーと呼ばれる



宛先に届ける(リレー)



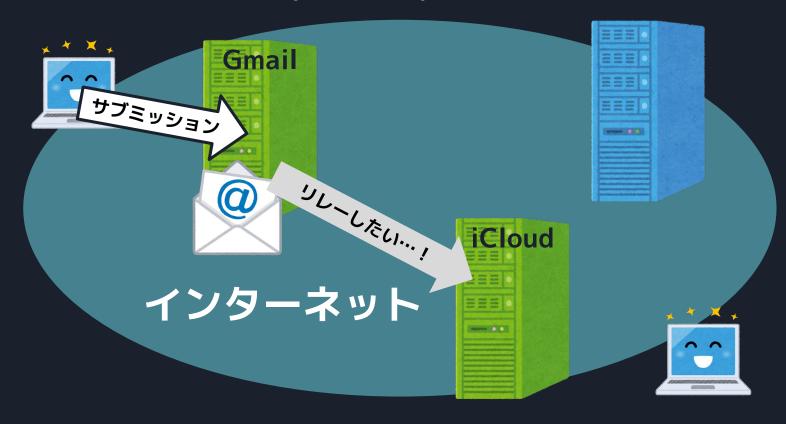
インターネット



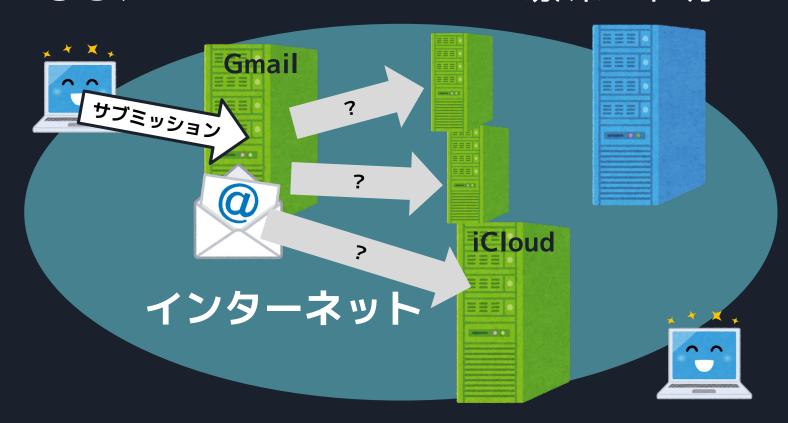




宛先に届ける(リレー)



でも、iCloudのサーバーの場所が不明…



IPアドレスとドメイン名

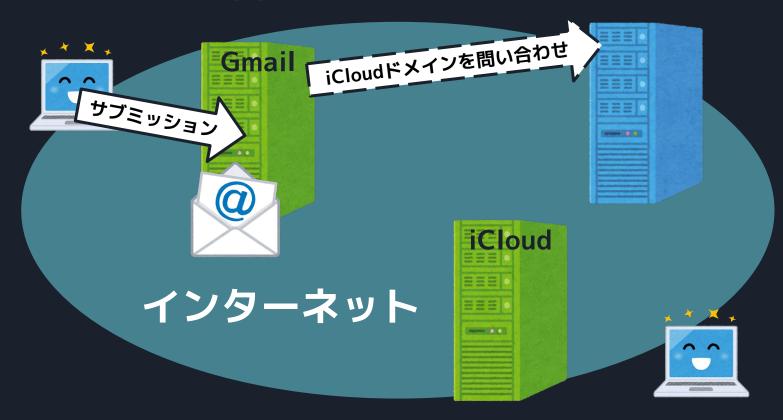
- IPアドレス
 - インターネットでそれぞれの機器が**一意に持つ住所**
- ドメイン名
 - IPアドレスは数字でわかりづらいので、これを わかりやすい文字列で表せるようにしたもの
 - o https://www.security-camp.or.jp/
 - o info@security-camp.or.jp
 - 実はメールアドレスにもドメインは入ってる!

DNSサーバー

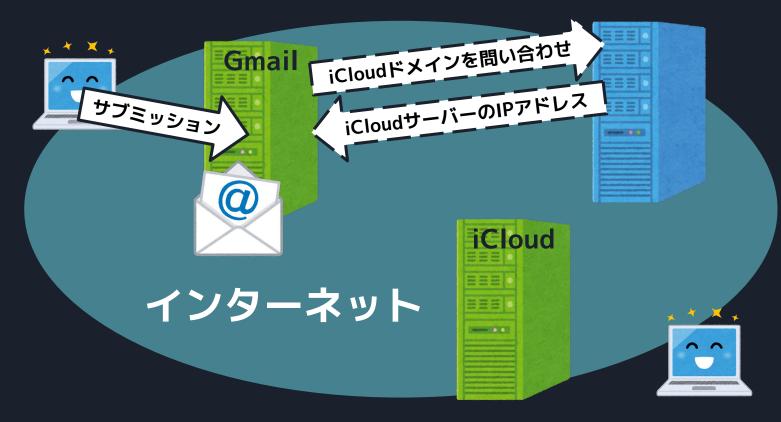
- ドメイン名とIPアドレスを紐付けるための仕組みを 提供する
 - これ無しでドメイン名は成り立たない
- おまけ機能として、ドメイン名と任意の テキストデータを紐づける仕組みも 提供している
 - これが、メールを安全にするためのプロトコルで大活躍します



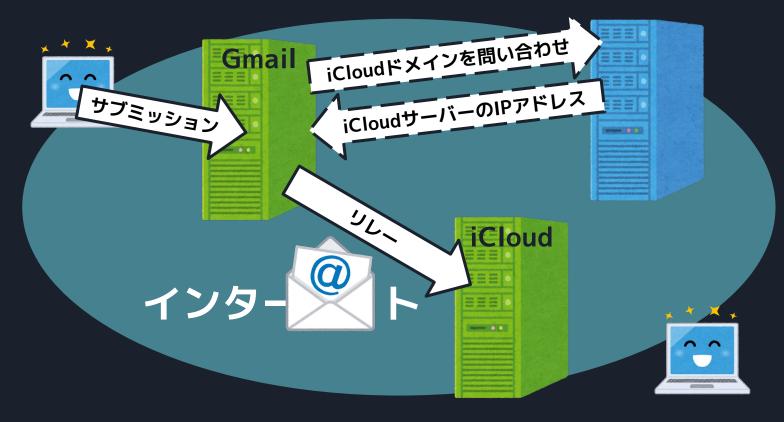
DNSに問い合わせ



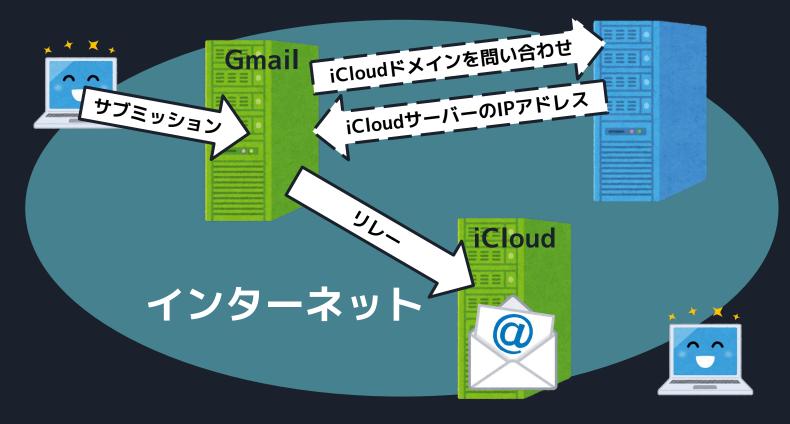
DNSに問い合わせ



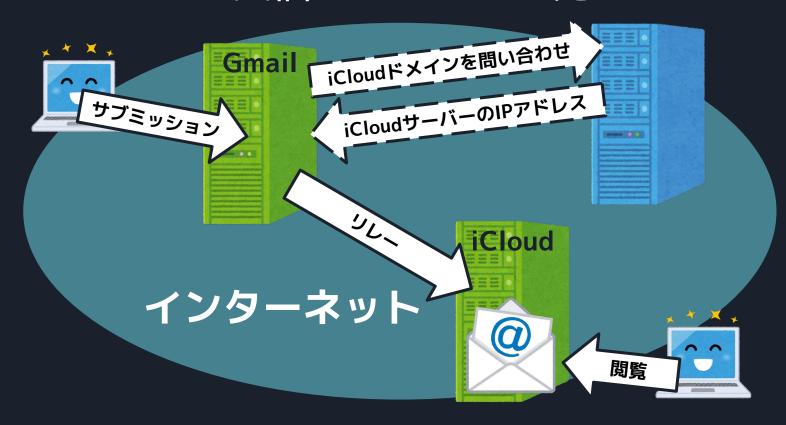
宛先に届ける(リレー)



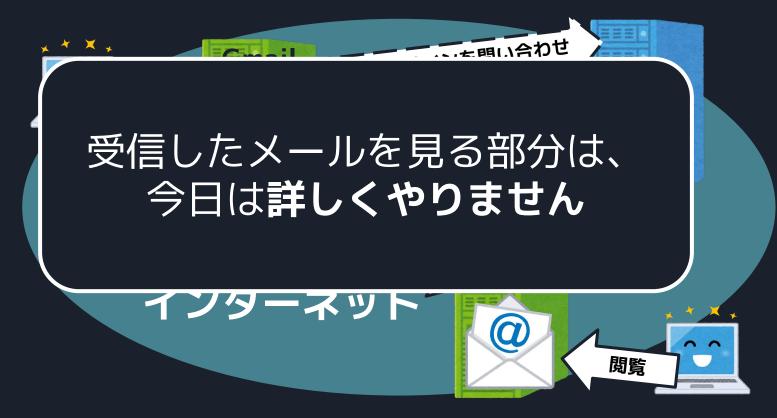
iCloudがメールを受信



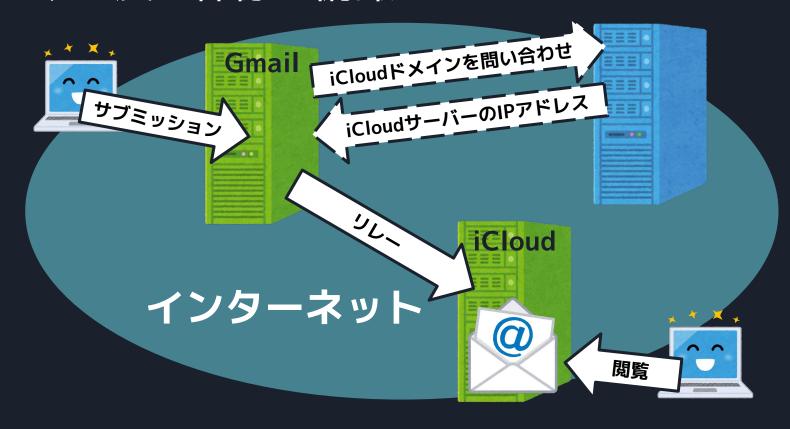
<u>ユーザー</u>が受信したメールを見る



ユーザーが受信したメールを見る



今一度全体像を俯瞰してみよう



アジェンダ

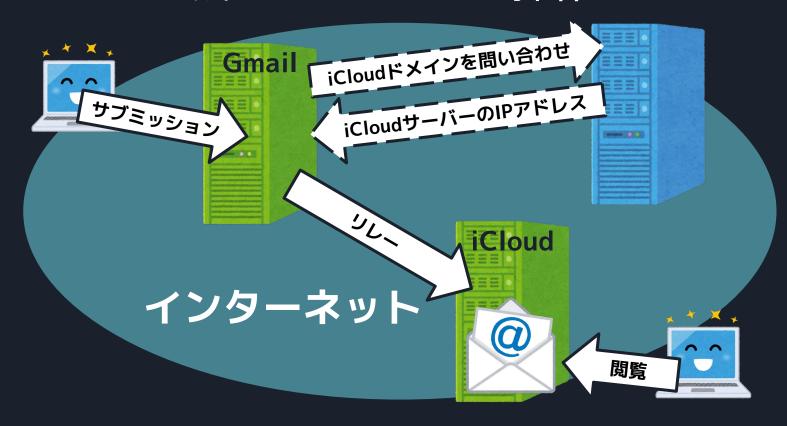
- なぜ、今メールなのか
- プロトコルスタックとRFC
- ◆ メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

アジェンダ

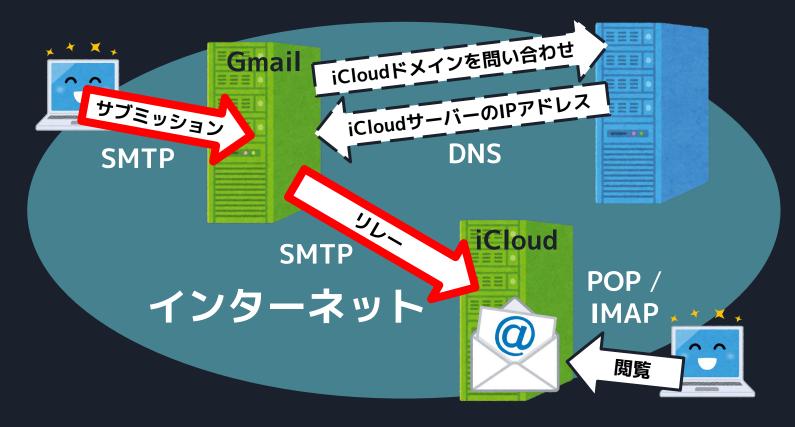
- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

SMTPプロトコル

メールを成り立たせている操作



一番大事な部分を支えるのが、SMTP!



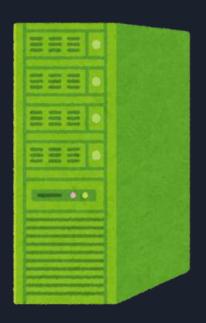
SMTPとは

- **SMTP** (Simple Mail Transfer Protocol)
 - **メールの送信**に特化したプロトコル
 - 1982年に制定され、大枠は今も変わっていない
- 対話型のプロトコル
 - httpは、行って帰ってきたら終わり
 - 「ページのデータ下さい」「はい」で表示
 - 自分の情報 / メールの情報を一つずつ渡し、それ ぞれに対してサーバーが応答を返してくれます

プロトコルとしての立ち位置

- OSI参照モデルの中では以下の二つに位置する
 - 7: アプリケーション層
 - 6: プレゼンテーション層
 - 「メール」というデータの表現の仕方と、メールの 送受信についてをどちらも定義している
- プロトコルとして
 - TCP (4: トランスポート層) の上のプロトコル
 - 関連プロトコルがたくさんある(後述)

送信側

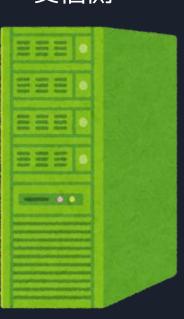




送信側

自己紹介 (HELO)





送信側



自己紹介 (HELO) OK



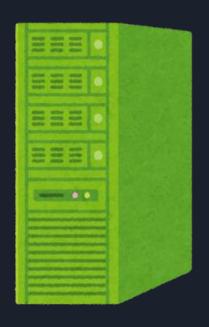
送信側



自己紹介 (HELO) OK 送り主 (MAIL FROM)



送信側







送信側







送信側







送信側







送信側







送信側





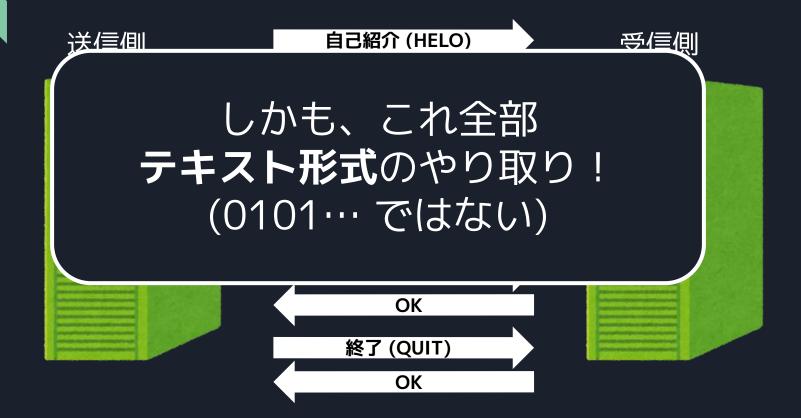


送信側









自分でも書けそうな 気がしてきたでしょ?

出来ます。 やりましょう。

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

ハンズオン 「SMTPを手書きしてみよう!」



別で**ハンズオン資料を 用意してあります**ので、 それに従って進めて下さい!

https://github.com/logica0419/security-minicampyamanashi-2024/blob/main/handson.md

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

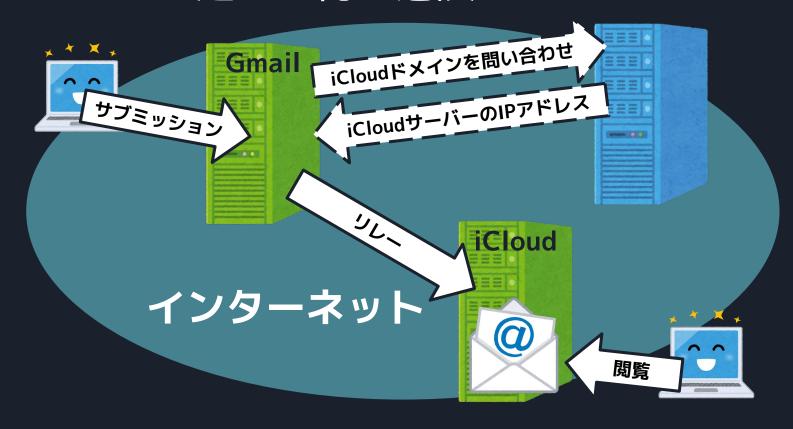
アジェンダ

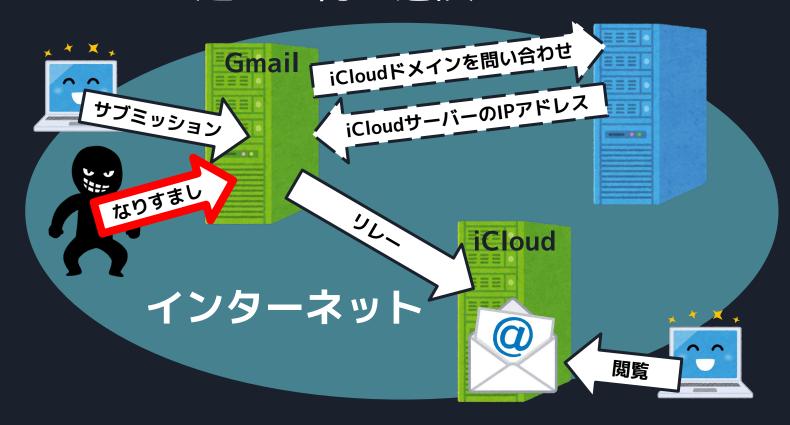
- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

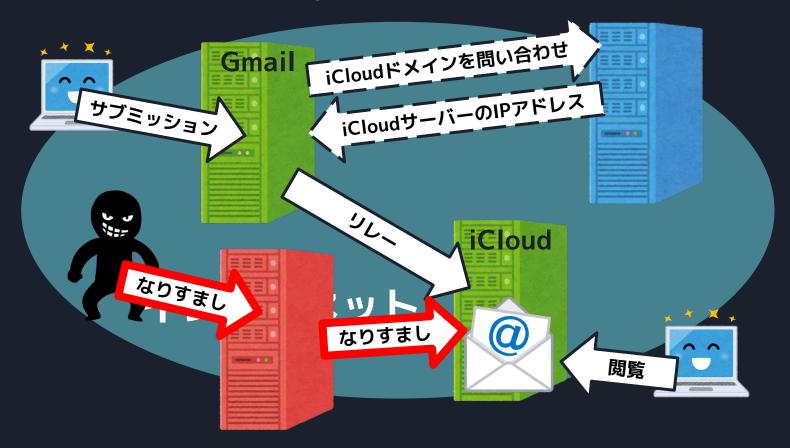
SMTPを安全にするための仕組み

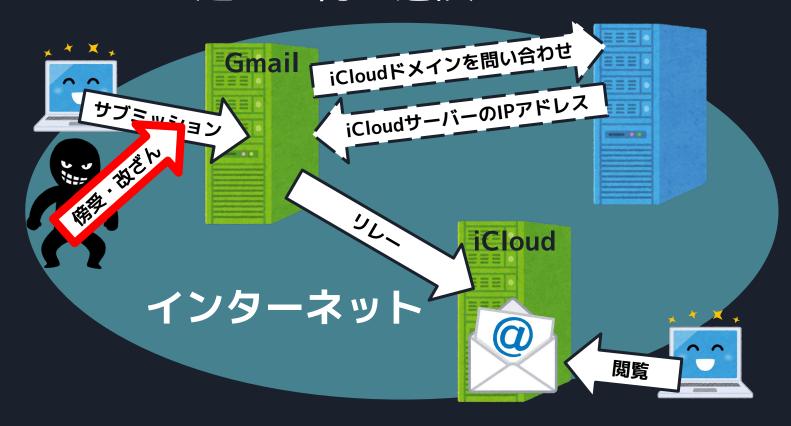
SMTPというプロトコル、 素の状態では大量の 危険にさらされている

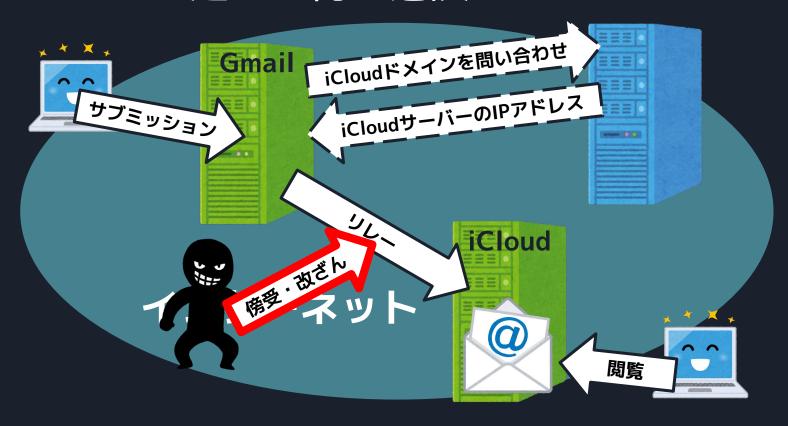
ハンズオンでも一部体験出来ましたね!

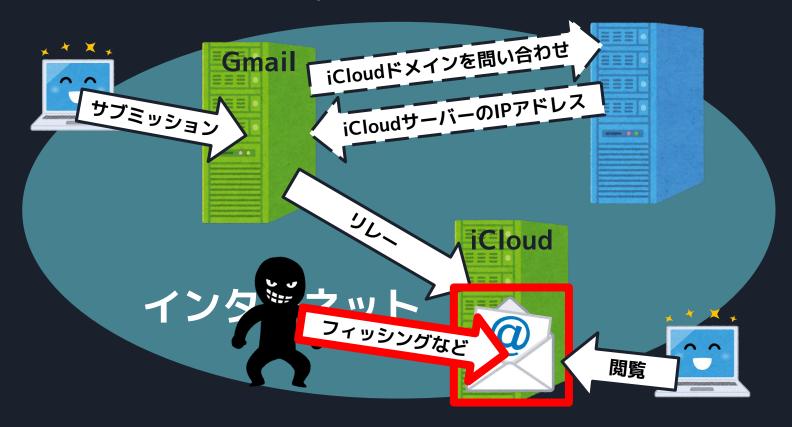












対策は2つ

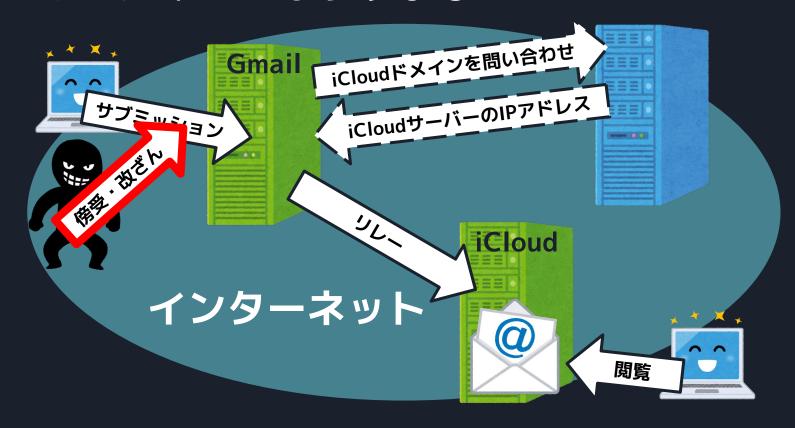
プロトコル

- プロトコル(仕様)としてセキュリティ対策を定義
- 広く使ってもらいやすい
- 汎用的でなければいけないため、効力が薄いことも
- メールサーバー独自の対策
 - メールサーバーが独自に対策を実装
 - 目的に特化させやすい
 - 実装依存なため、統一化はしにくい

脅威への対策プロトコル

~ユーザーなりすまし編

このタイプのなりすまし



POP before SMTP

ユーザーが送信する (サブミッション)









…の前に









からの









Pop before SMTP

- 「受信できる人間はこのメールサーバーの利用者だね」 問う考え方に基づく
 - 自分が受信できるメールアドレスを持っていれば、 メールサーバを利用できる
- Fromの書き換えを検知する仕組みではないので、なりすましの強固な予防策とはならない
- このあとのSMTP AUTHの方がよく使われているので、 マイナーなプロトコル

SMTP AUTH (SASL認証)

ユーザーが送信する (サブミッション)











…の前に

ユーザ名・パスワード







からの

ユーザ名・パスワード









SMTP AUTH

- サブミッション前に、何らかの形でユーザー名とパス ワードをメールサーバーに送って認証
 - いわゆるアカウントの認証
- 認証情報の送り方によってタイプがある
 - SMTP AUTH **PLAIN** まとめる、弱い
 - SMTP AUTH **CRAM-MD5** まとめる、強いが面倒
 - SMTP AUTH **LOGIN** 2つをバラバラに、弱い

SMTP AUTH = SASL認証

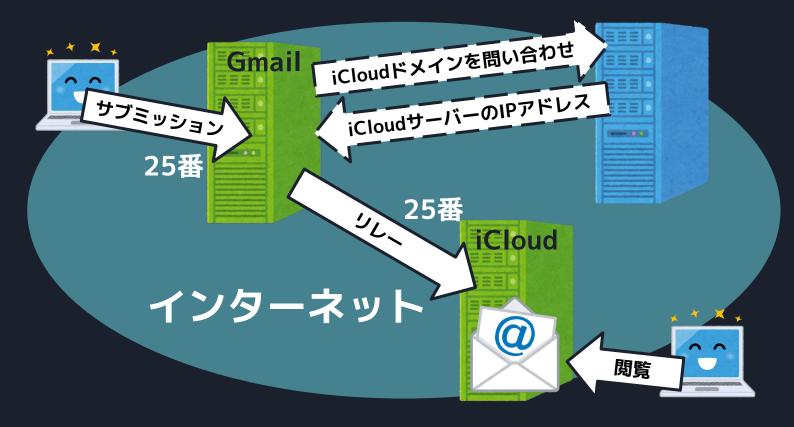
- **SASL** (Simple Authentication and Security Layer)
 - インターネットにおける、認証とセキュリティの ためのフレームワーク
 - 基本的にどんなプロトコルとも組み合わせて使うことができる
- 紹介したもの以外にも、ワンタイムパスワードなどが サポートされている

OP25B

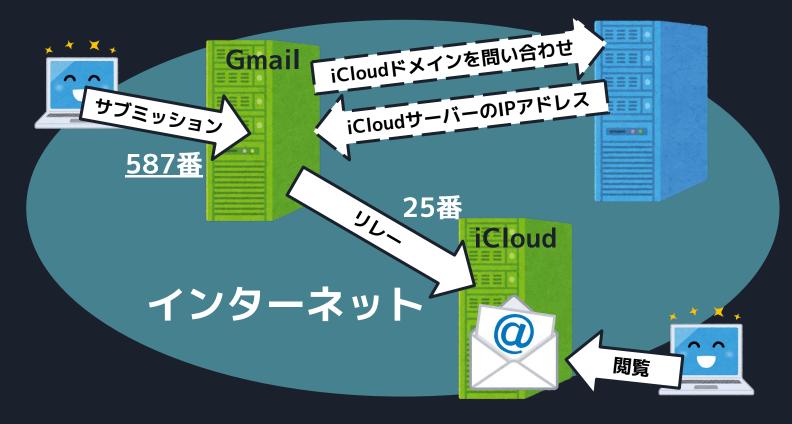
OP25B

- リレー用ポートとサブミッション用ポートを分けよう!という取り組み
- リレーでいちいち認証は出来ない
 - 全てのメールサーバーが、他の全てのサーバーへの 認証情報を持たなくてはいけなくなる
- サブミッションは認証を必ずさせたい
 - セキュリティをガチガチにすることを強制させたい

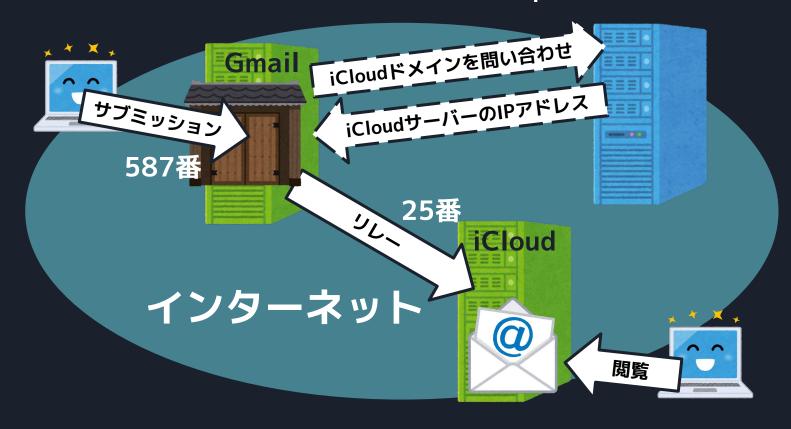
こうだったのを



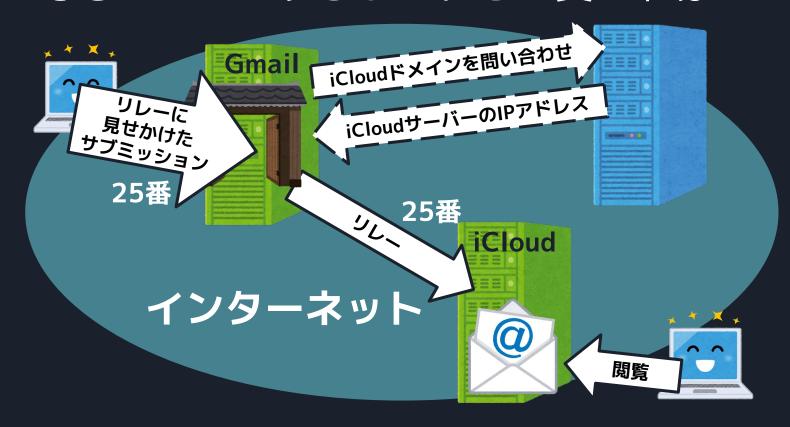
ちゃんと分けた



サブミッションポートは堅牢にする



でもアクセスするポートさえ変えれば…

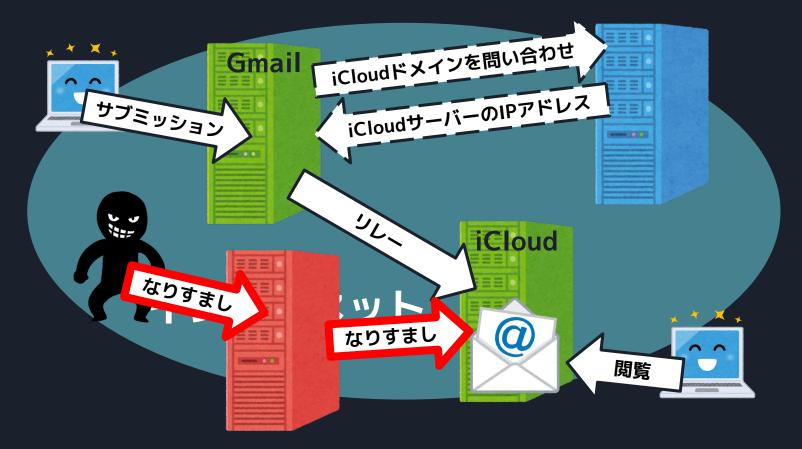


OP25B

- 一般ユーザーに587番ポートしか使わせないために、 最終的にISP (通信事業者) がインターネット経由で 25番ポートにアクセスする行為を封じた
 - これがOP25B
- 「OP25B」と調べると「外部の25番ポートへの接続を 出来ないようにする」という内容しか出てこない
 - 背景情報をきちんと知ることで、初めてきちんと 理解できるタイプの取り組み

脅威への対策**プロトコル** 〜サーバー / ドメイン なりすまし編

このタイプのなりすまし



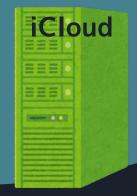
SPF (Sender Policy Framework)

下準備



インターネット







下準備

Gmailのサーバーは x.x.x.xにあります!





インターネット





おさらい: DNSサーバー

- ドメイン名とIPアドレスを紐付けるための仕組みを 提供する
- おまけ機能として、<u>ドメイン名と任意の</u>テキストデータを紐づける仕組みも提供している
 - これが、メールを安全にするためのプロトコルで大活躍します



おさらい: DNSサーバー

● ドメインタとIPアドレスを紹付けるための仕組みを

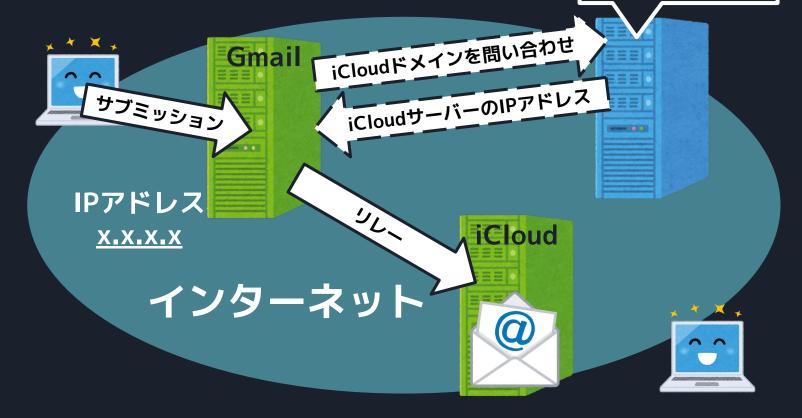
SPFにおいては、サーバーのIPを この「**任意テキスト**」として DNSに保存しておく

CTUDY NEXTED BICOM

プロトコルで大活躍します

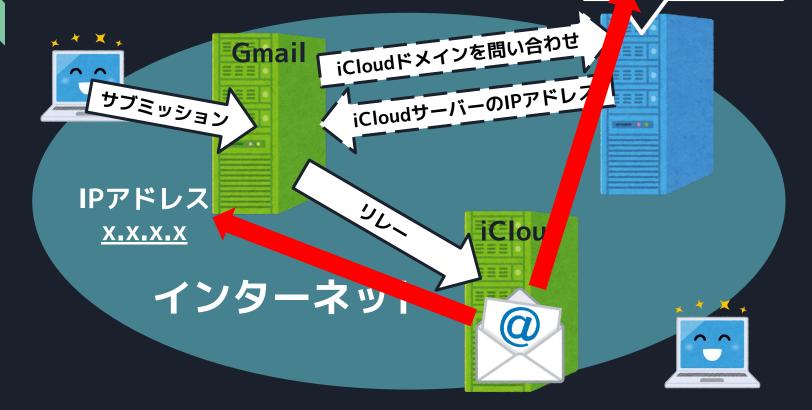
受信後

Gmailのサーバーは x.x.x.xにあります!



一致するか確かめる

Gmailのサーバーは x.x.x.xにあります!



SPF (Sender Policy Framework)

- DNSサーバーに正しいメールサーバーのアドレスを保存 しておくことで、受信後メールを送ってきたサーバーと 正しいメールサーバーのアドレスが一致するか確認 できるようにするプロトコル
 - 不審なメールサーバーを検知できる
- サーバー運用者の設定が面倒
- 間にサーバーが挟まってしまうと意味を成さない

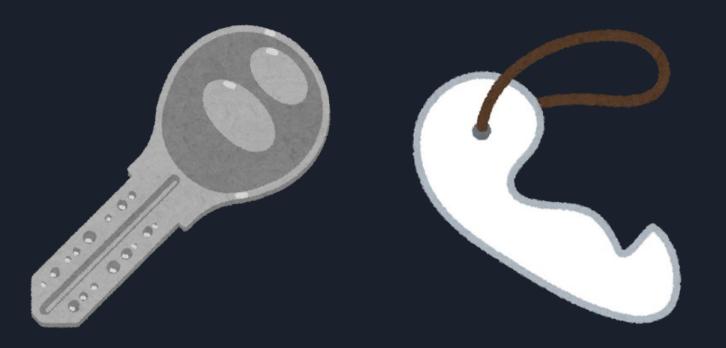
DKIM (DomainKeys Identified Mail)

前提知識: 公開鍵 / 秘密鍵 暗号

- 公開鍵 / 秘密鍵という、みんなに**知られても良い鍵**と **知られてはいけない鍵**の2種類を用意する暗号化
- 鍵 = 一定のルールで作られたランダムなテキスト
- 特徴
 - 公開鍵で暗号化した文書は、秘密鍵でしか復号 できない
 - 秘密鍵で暗号化した文書は、公開鍵で必ず復号 できる

特徴の部分をより分かりやすく図解

秘密鍵(知られてはいけない) 公開鍵(知られても良い)



下準備

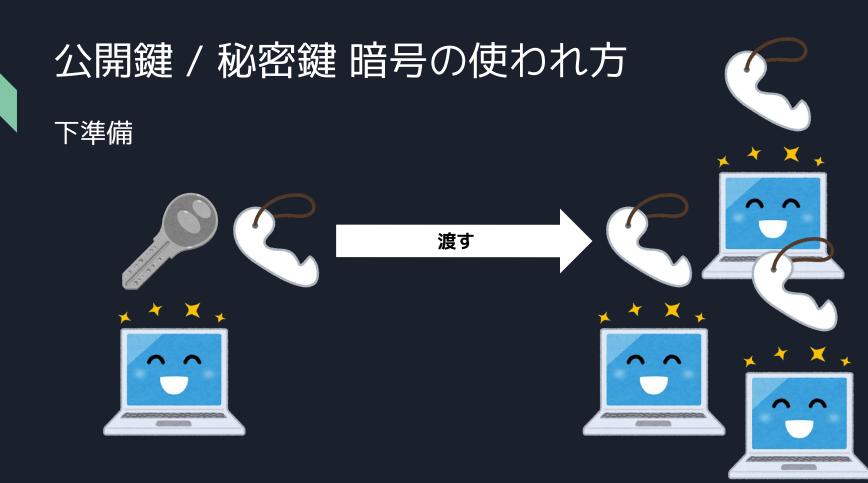


← 必ずセットで作られる



下準備











受け手



送り手







送り手

① 秘密鍵を持つ人しかわからないデータを作る(保護)

公開鍵で暗号化した文書は、 秘密鍵でしか復号できない







受け手

送り手







送り手

① 秘密鍵を持つ人しかわからないデータを作る (保護)

これが**皆さんの想像する** 「**暗号**」だと思う











送り手



受け手



送り手



受け手





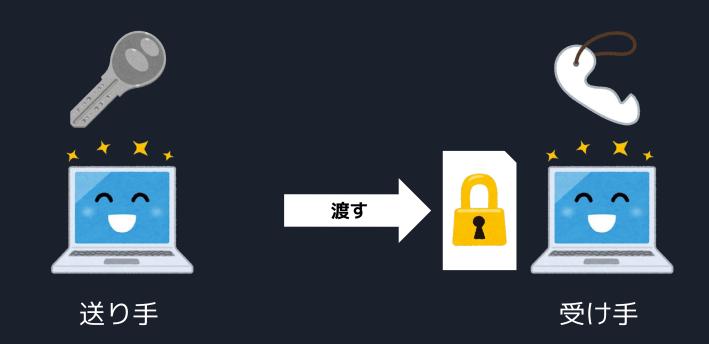
② 送り主が秘密鍵を持っているか確かめる (証明)

実は「**見れなくする**」 だけが暗号じゃないんです



送り手







送り手



受け手

② 送り主が秘密鍵を持っているか確かめる(証明)

正しく復号できた = **正しい秘密鍵で暗号化**された データだとわかる!





公開鍵 / 秘密鍵は 情報の保護にも証明にも使える 面白い仕組みです

あくまでイメージなので、詳しい人には怒られるかも…

本題のDKIM

秘密鍵 公開鍵 を用意



下準備



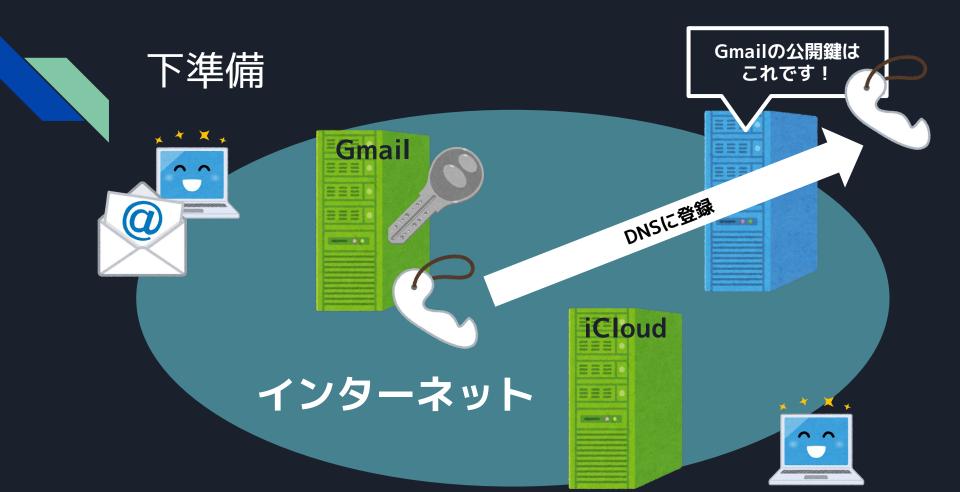


インターネット









送信後



インターネット







暗号化したものを「署名」に

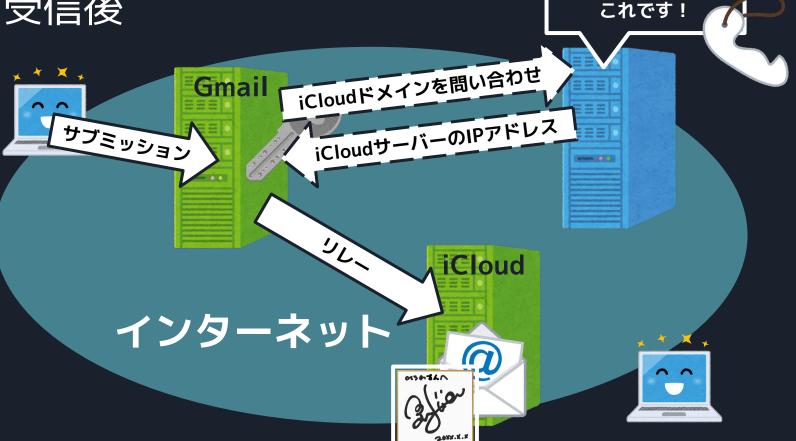








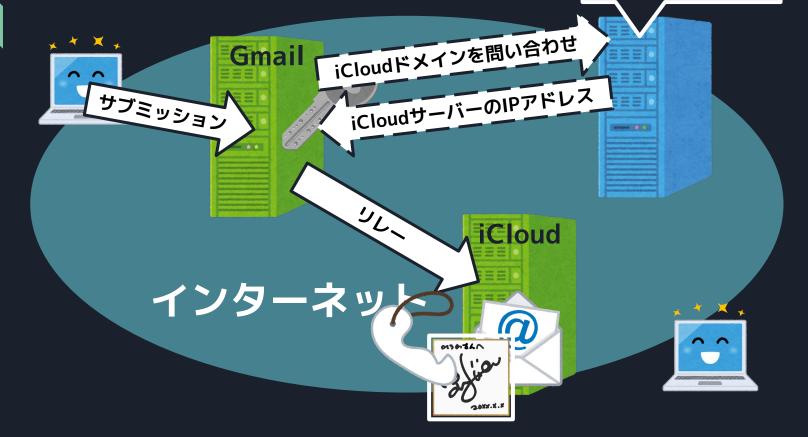
受信後



Gmailの公開鍵は

署名を復号!

Gmailの公開鍵は これです!



DKIM

- 秘密鍵をメールサーバーに持ち、公開鍵をDNSで公開 しておく
- 秘密鍵でメール全体を暗号化したものを「署名」として メールに付けることで、正しいメールサーバーから 送られてきたことを保証する仕組み
- サーバー運用者の設定が面倒
- サーバーを提供するサービス側の実装が面倒

DMARC
(Domain-based Message
Authentication Reporting
and Conformance)

DMARC

- SPF及びDKIMでメールサーバーが正しいと判定できなかった時「どのようにメールを処理するか」を決めることができるプロトコル
 - 同じくDNSにポリシーとして定める
- 「自分のドメインの成りすましが出た」という場合のみ 対処できる
 - それ以外のパターンは考慮していない
- サーバー運用者の設定が面倒

DMARCのポリシー

- None
 - メッセージは通常通り送信される
 - 「DMARCレポート」がドメインの所有者に来る
- Quarantine
 - メッセージが別のフォルダに隔離される
- Reject
 - メッセージはその場で破棄される

DMARCまでやると、 現時点で「そのドメインは 安全」と言えるライン

自分でサーバーを運用するときは、是非設定しましょう

BIMI
(Brand Indicators for Message Identification)

要はこれのこと



BIMI

- DNSで、そのドメインからのメールに表示されるべきブランドロゴを設定できるプロトコル
 - どちらかというと企業とか向け
 - Gmail / Yahooメールなどは表示してくれる
- DMARCの検証が通って、初めて表示される
- サーバー運用者の設定が面倒
- 受信するサービス側の実装が面倒

ARC (Authenticated Received Chain)

SPF・DKIM (・DMARC) の弱点

- 複数サーバーのリレー / メーリングリストなどで検証が うまく行かなくなることがある
- SPF
 - 検証時、送り元のサーバーは**直前のサーバー**のため
- DKIM
 - リレーの過程で**ヘッダが変更**されることがある
 - メーリングリストでもヘッダが変更される

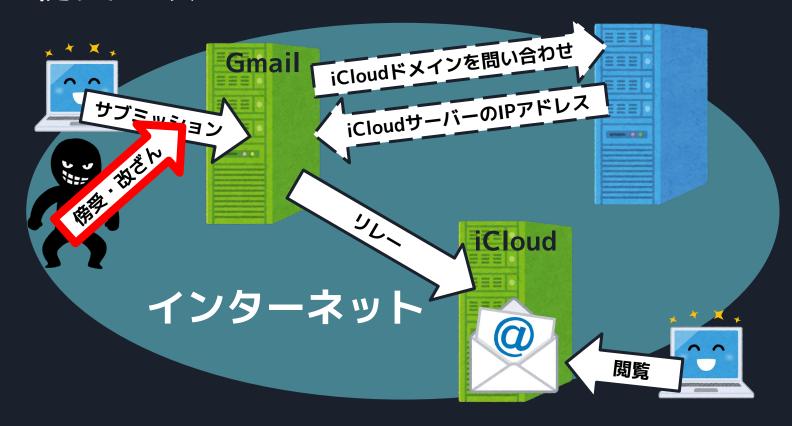
ARC

- リレーやメーリングリストに一斉転送する際、中間の サーバーでSPF・DKIMを検証して、その検証結果を メールに付けて次のサーバーに送る
 - SPF・DKIMがダメならARCのヘッダを見る
- 検証結果が偽造できないように、署名を付ける
 - 署名はDKIMと全く同じ仕組みなのでDNSに追加の 設定をする必要ナシ

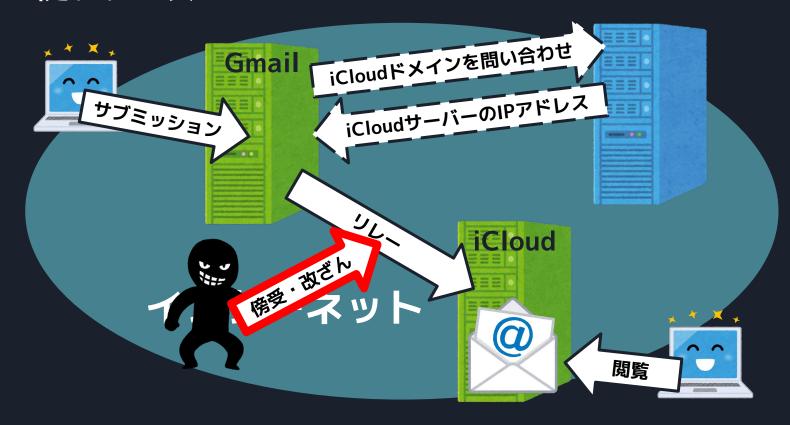
脅威への対策プロトコル

~傍受・改ざん編

傍受・改ざん



傍受・改ざん



STARTTLS

STARTTLS'

- TLS (Transport Layer Security) をメールでも扱える ようにしたもの
 - 「https」の「s」の技術
 - メール特有の技術ではない
- サブミッション / リレーなど一つ一つの通信経路を 暗号化する
- 悪意のあるメールサーバーが間に入った場合は対処 できない

MTA-STS (MTA Strict Transport Security)

MTA-STS

- 認証 (SMTP AUTH) やSTARTTLSを、メールサーバー利用者に強制することができるしくみ
 - より強固なメールサーバーを作るために必要
- Gmailなどが導入している
- DNSで「ポリシーがあるよ!」と宣言し、HTTPで ポリシー本体を公開する
 - 設定のハードルが高い

OpenPGP · S/MIME (Secure / Multipurpose Internet Mail Extensions)

OpenPGP · S/MIME

- エンド・ツー・エンドでメールを暗号化する仕組み
 - 「保護」的な使い方に当たる
 - 送信者が暗号化し、受信者が複合する
- エンド・ツー・エンドでメールを署名する仕組み
 - 「証明」的な使い方に当たる
 - 送信者が署名し、受信者が検証する
- 暗号化・署名で鍵が2ペアでき、しかもそれぞれが 逆の方を持つという面白いプロトコル

下準備













鍵を準備(暗号化サイド)



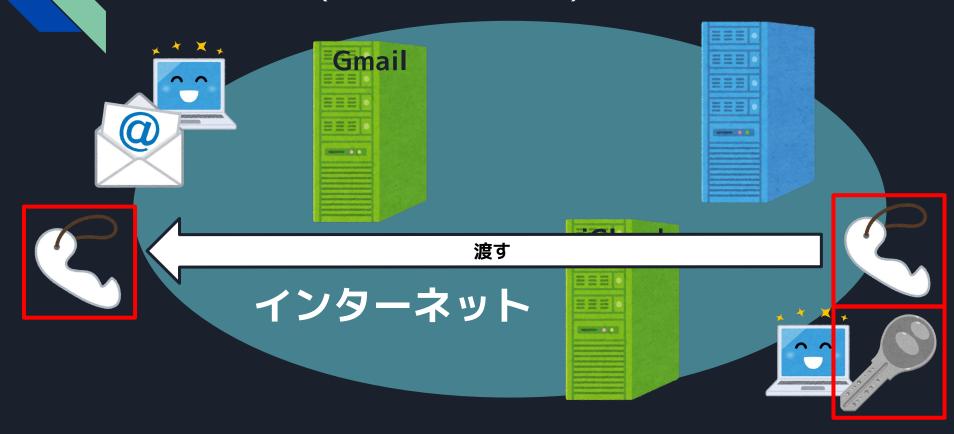


インターネット





鍵を準備(暗号化サイド)



鍵を準備(署名サイド)









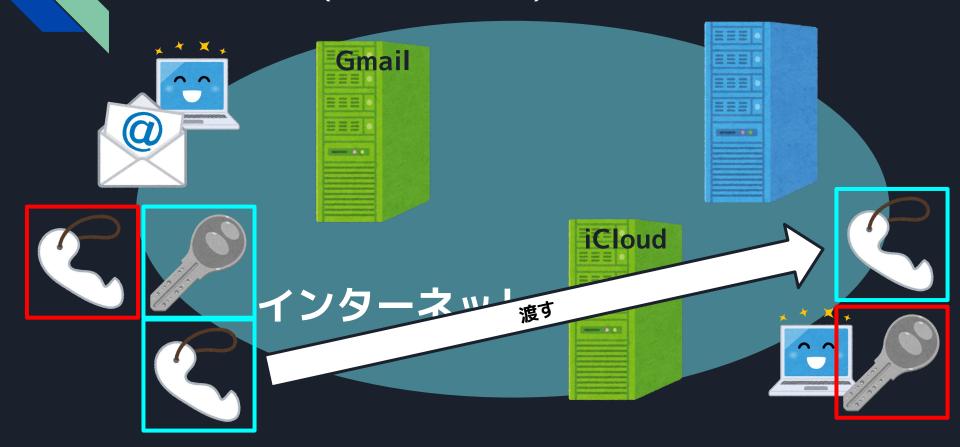


インターネット

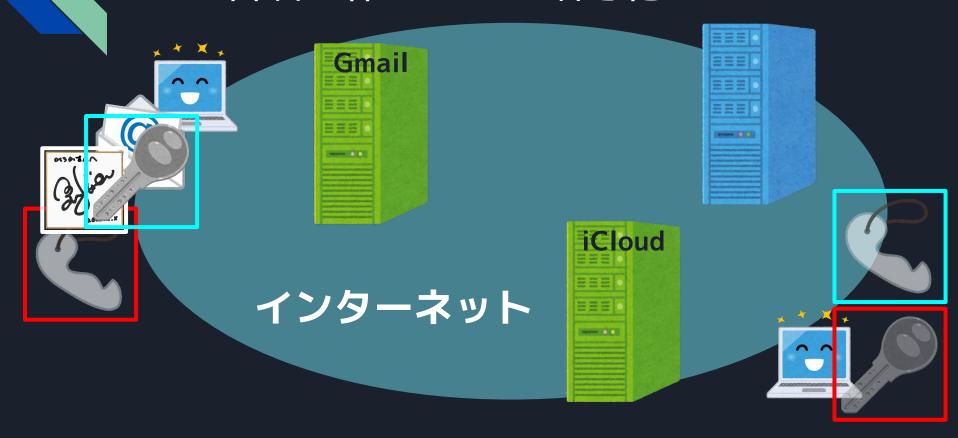




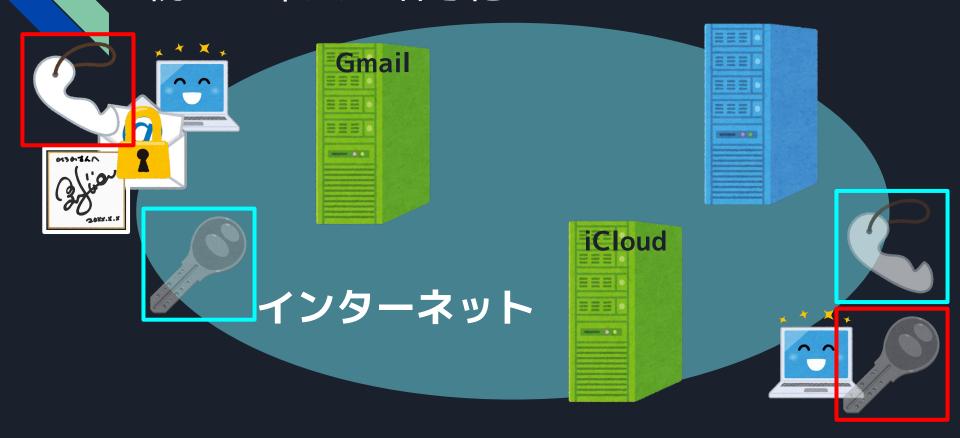
鍵を準備(署名サイド)



まず署名を作るための暗号化



続いて本文を暗号化



続いて本文を暗号化



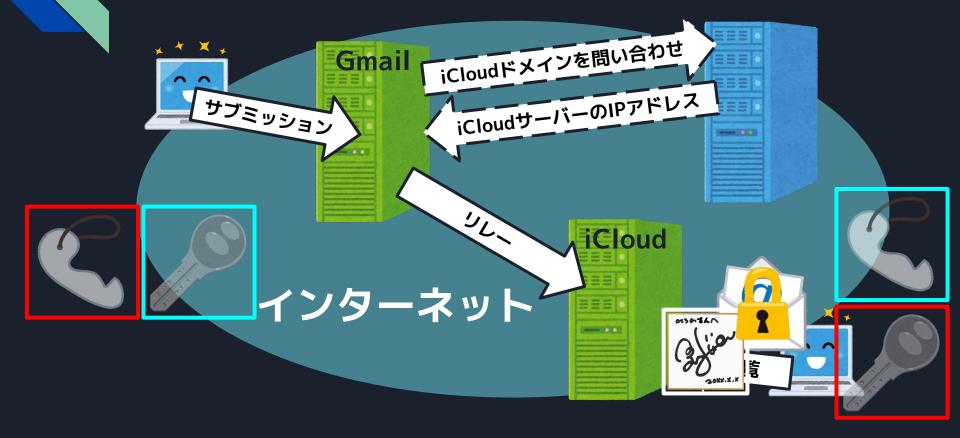
なお、正確には共通鍵を混ぜた もっと複雑な暗号化ですが ここでは割愛します

インターネット

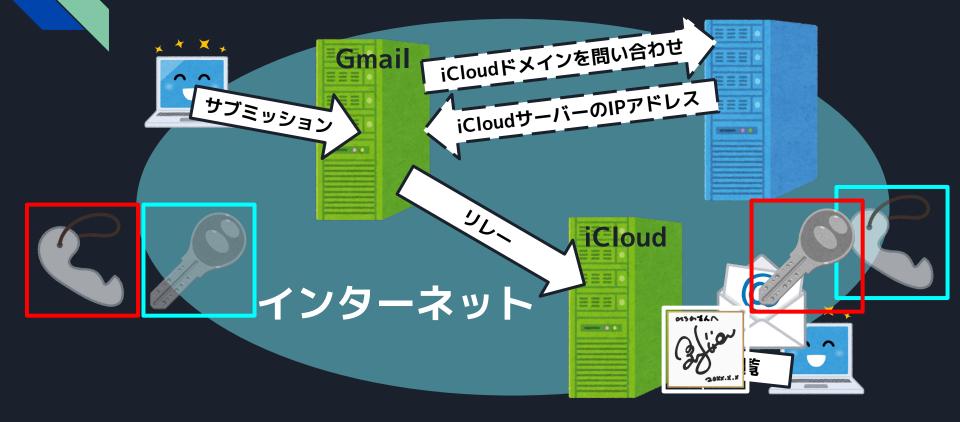




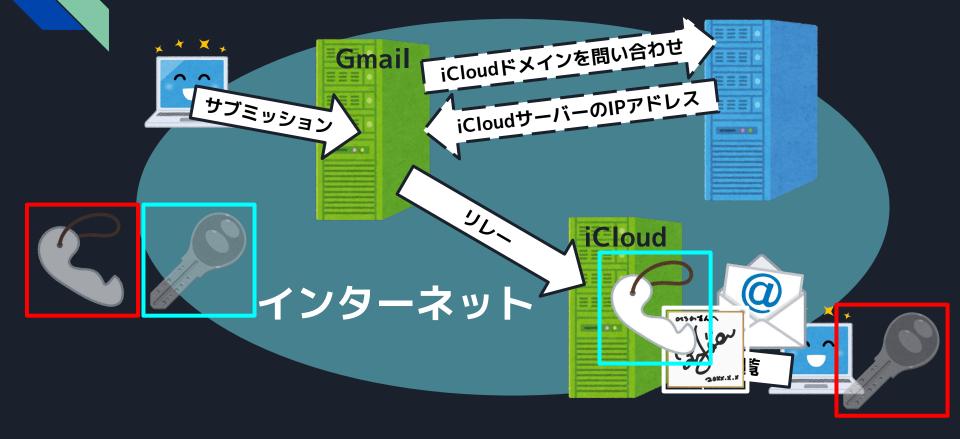
送信後、受信者の手元で



まず本文を復号



最後に署名を復号して検証

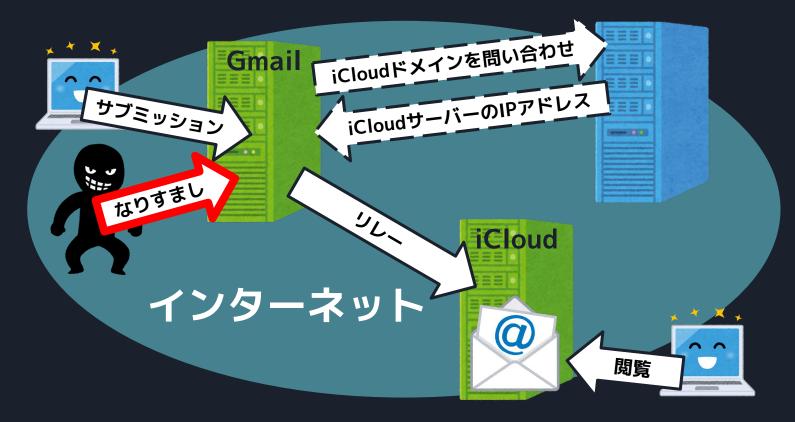


OpenPGP・S/MIMEの問題と解決法

- 署名用の公開鍵の信頼性が担保されない
 - 本人のでなかったら、**他人が署名してる**のでマズい
 - 暗号化は間違ってたら復号できないだけなので良い
- OpenPGP
 - 多くの人に信頼されている送信者は信頼
- S/MIME
 - 認証局が信頼してるから信頼
 - 認証局にお金を払って、公開鍵を信頼してもらう

メールサービス独自の脅威対策

ユーザーなりすまし



送信メールアドレス制限

- ハンズオンで皆さんが体験したやつです。
- 「自分が持っていること」が確認できるアドレスから しか送信できないようにする
 - メールアドレス認証
 - ドメインを持っていることを証明する
- Gmailや今回使ったBrevoをはじめ、様々なメールサー ビスが導入している

メール内容の危険性判定



メールフィルタリング

- 機械学習などを用いて、「怪しいメールのパターン」を特定して危険なメールを判断する
 - かなり昔から研究が進んでいる分野
 - 機械学習の分野の先駆け
 - 「ベイジアンフィルタ」というアルゴリズムが有名
- Gmailなど、大手メールサービスを中心に導入
 - Gmailはニューラルネットワークまで手を出してる という噂がある

このように、様々な **セキュアにするための 取り組み**があります

良く知って使いましょう

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

メールとの向き合い方

なぜここまで**さまざまな 対策が考案**されているのに、
電子メールは**未だ様々な サイバー犯罪**に使われるのか?

ここからは、 **事実に基づいた憶測**です

人間が対策することなので、 「正しさ」は保証されません

強固な対策が取りにくい

- 対策で、メールを送れなくすることは基本的にない
 - 正しい送信元か確かめたり、改ざん防止だったり…
- プライベートなやり取りであるのが難しい
 - Webページは、よく特定のユーザーにアクセス 制限をかける
 - 情報の提供者の階層が利用者と別れるため、一元的な指標で不正行為などを定められる
 - メールでは、「何を不正とするか」が曖昧

共通化の難しさ

- Webページのセキュリティは奇跡に近いクオリティ
 - Chromiumという、Google Chromeの核になる 部分が**ほぼ全てのブラウザ**で使われている
- 簡単さや社内政治のせいで、**メールサービスは乱立**
 - セキュリティ対策をしなくてもメールは送れて しまうので、セキュリティ対策をしないメール サービスが多い
- 何かしらの方法で対策を強要する必要がある

Gmailのセキュリティ基準強化

メール送信者のガイドライン

この記事のガイドラインに沿った対応を行うことで、個人用 Gmail アカウントにメールが正常に送信、配信されるようになります。2024 年以降は、Gmail 個人用アカウントにメールを送信する場合に、こちらに記載の要件を満たす必要があります。個人用 Gmail アカウントとは、末尾が @gmail.com または @googlemail.com のアカウントを指します。

メール認証の要件とガイドライン

ドメインに、以下のようにメール認証方式を設定する必要があります。

- すべての送信者: SPF または DKIM
- 一括送信者: SPF、DKIM、DMARC

Gmailのセキュリティ基準強化

- 「SPF / DKIMを利用していないと、メールを弾く 可能性があるよ」と警告
 - 個人メールにおいて**圧倒的シェア**を持つ影響力
- ◆ さくらのメールサーバーをはじめとする、様々なメールサービスが対応を迫られる結果に
 - 国内のかなり大手のサービスでもまだ仕組みに対応 していないことが判明した
- セキュリティ面では、非常に大きな一歩

メールから逃れられない人類

- セキュリティを考えれば、メールは排除すべき存在
 - ここまで対策しても**完全にセキュア**とは言えない
- ここから20~30年はメールから逃れられないであろう現状が存在する
 - **依存するもの**が多すぎる
 - 企業間のやり取り、アカウント、etc…
- 逃げられないのであれば「まだマシ」な運用をするよう 関わる全員が心掛けなくてはならない

アジェンダ

- なぜ、今メールなのか
- プロトコルスタックとRFC
- メールが送られる仕組み
- SMTPプロトコル
- ハンズオン 「SMTPを手書きしてみよう!」
- SMTPを安全にするための仕組み
- メールとの向き合い方

まとめ

今日学んだこと

- メールが様々な犯罪に用いられること
- RFC (プロトコル) がインターネットを支えていること
- メールがSMTPを使って送られていること
- SMTPが手書きで書けてしまうほど簡単なこと
- なりすましが容易にできること
- メールを安全にするために様々な取り組みが行われていること
- メールを安全にするのはとても難しいこと

メールの仕組み**そのものが 脆弱**なのは否定できない 「**まだマシ**」なメールを運用する ために知識を付けよう

ありがとう ございました

怪しいメールには 気を付けよう!

参考ページたち

- https://www.irasutoya.com/
- https://www.npa.go.jp/hakusyo/r03/honbun/html/xf211000.html
- https://sendgrid.kke.co.jp/blog/?p=10300
- 東京工業大学 情報理工学院 情報工学系 講義

「コンピューターネットワーク」

- https://atmarkit.itmedia.co.jp/ait/articles/1411/13/news019.html
- https://thinkit.co.jp/story/2015/04/28/5799
- https://www.gmo.jp/security/brandsecurity/dns/blog/dns-server/
- https://wa3.i-3-i.info/
- https://www.tohoho-web.com/

参考ページたち

- https://ja.wikipedia.org/
- https://www.itmanage.co.jp/column/osi-reference-model/
- https://www.proofpoint.com/jp/threat-reference/dmarc
- https://www.naritai.jp/
- https://support.google.com/a/answer/9261504
- https://qiita.com/hirachan/items/d0028da3ebb80b138404
- https://jp.globalsign.com/managed-pki/about_smime.html
- https://www.prins.co.jp/knowledge/column/20181101-557/
- https://support.google.com/a/answer/81126
- https://www.sakura.ad.jp/corporate/information/announcements/ 2023/12/19/1968214527/