情報リテラシー（第4回 後期）

サイバーセキュリティ2

今日のねらい

- 暗号化の仕組みと共通鍵・公開鍵暗号方式の違いを理解できる
- デジタル署名と電子認証の役割を説明できる
- SSL/TLSによる安全な通信の仕組みを理解できる
- 電子透かしと誤り検出符号の目的と使い方を説明できる

暗号化とは

暗号化の基本概念

暗号化

- **意味：** データを第三者が読めない形式に変換すること
- **目的：** 情報の機密性を保護する
- **具体例：** メール、パスワード、クレジットカード番号の保護

復号

- **意味：** 暗号化されたデータを元の形式に戻すこと
- **必要なもの：** 正しい鍵

平文と暗号文

- **平文** → 暗号化前の元のデータ
- **暗号文** → 暗号化後の読めないデータ

共通鍵暗号方式

仕組み

- **同じ鍵**で暗号化と復号を行う
- 送信者と受信者が**事前に秘密の鍵を共有**

特徴

- **メリット：** 高速処理、計算負荷が軽い
- **デメリット：** 鍵の配送が困難、相手ごとに鍵が必要
- **代表的な方式：** AES（最も安全で広く使われている）
- **注意：** 古い方式のDESは解読可能なため使用禁止

利用例

- ファイルの暗号化、ハードディスク全体の暗号化
- VPN通信、無線LANの暗号化（WPA2/WPA3）

公開鍵暗号方式

仕組み

- **2つの鍵**を使用：公開鍵と秘密鍵
- **公開鍵**で暗号化 → **秘密鍵**で復号
- 公開鍵は誰でも入手可能、秘密鍵は本人だけが保持

特徴

- **メリット：** 鍵配送問題の解決、相手ごとの鍵管理不要
- **デメリット：** 処理速度が遅い、計算負荷が高い
- **代表的な方式：** RSA

利用例

- メール暗号化、デジタル署名、SSL/TLS通信

ハイブリッド暗号方式

両方の利点を組み合わせる

1. **共通鍵**でデータ本体を暗号化（高速処理）
2. その**共通鍵を公開鍵**で暗号化（安全な鍵配送）
3. 受信者は**秘密鍵**で共通鍵を復号
4. 復号した共通鍵でデータ本体を復号

→ HTTPS通信（SSL/TLS）で実際に使われている方式

デジタル署名

デジタル署名とは

- 電子文書の**作成者を証明**し、**改ざん検出**する技術
- 紙の文書における「印鑑」「サイン」に相当

仕組み

1. 送信者が秘密鍵で文書に署名
2. 受信者が送信者の公開鍵で署名を検証
3. 検証成功 → 本人が作成、改ざんなし

役割

- **認証：** 送信者が本人であることの証明
- **完全性：** 文書が改ざんされていないことの保証
- **否認防止：** 送信者が「送っていない」と否定できない

ハッシュ関数とデジタル署名

ハッシュ関数

- 任意のデータから**固定長の値（ハッシュ値）**を生成
- **同じデータ** → 必ず**同じハッシュ値**
- **少しでも変更** → **全く異なるハッシュ値**

デジタル署名での利用

1. 文書全体ではなく**ハッシュ値を暗号化**して署名
2. データ量削減、処理の高速化
3. 受信側でハッシュ値を比較して改ざん検出

→ 効率的で確実な本人確認と改ざん検出を実現

電子証明書と認証局

電子証明書とは

- **公開鍵が本当に本人のものか**を証明する電子文書
- 運転免許証やパスポートのような「身分証明書」

認証局

- 電子証明書を発行する**信頼できる第三者機関**
- 本人確認を行い、公開鍵と所有者を結びつける

証明書に含まれる情報

- 所有者の情報（名前、組織名など）
- 公開鍵、有効期限、発行者（認証局）の署名

→ なりすましを防ぐ重要な仕組み

SSL/TLSとは

SSL/TLS（Secure Sockets Layer / Transport Layer Security）

- インターネット通信を**暗号化**するプロトコル
- Webサイト（HTTPS）、メール、VPNなどで使用

3つの役割

1. **暗号化：** 通信内容の盗聴防止
2. **認証：** 通信相手が本物か確認
3. **完全性：** データの改ざん検出

HTTPSとHTTP

- **HTTP：** 暗号化なし、盗聴・改ざんのリスク
- **HTTPS：** SSL/TLSで保護、URLが「https://」で始まる

SSL/TLS通信の流れ

ハンドシェイク（接続確立）

1. クライアントがサーバーに接続要求
2. サーバーが**電子証明書**を送信
3. クライアントが証明書を検証（認証局の署名確認）
4. 共通鍵を生成して交換（公開鍵暗号で保護）

データ通信

5. 生成した共通鍵で通信を暗号化
6. 高速な暗号化通信を実現

→ ハイブリッド暗号方式を活用した安全な通信

演習1：証明書の確認

ブラウザでWebサイトの証明書を確認してみよう。

確認方法（Edge）

1. HTTPSサイトのURLバー左側の**鍵マーク**をクリック
2. 「この接続は保護されています」→「証明書」をクリック

確認すべき項目

- 発行先（サイトのドメイン名が正しいか）
- 発行者（信頼できる認証局か）
- 有効期限（期限切れでないか）

偽サイトは証明書エラーが表示される！

電子透かし

電子透かしとは

- デジタルコンテンツ（画像、音声、動画）に**見えない情報を埋め込む**技術
- 人間には知覚できないが、専用ソフトで検出可能

目的

- **著作権保護：** 作成者や所有者の情報を埋め込む
- **不正コピー追跡：** 流出元の特定
- **改ざん検出：** コンテンツの変更を検知

特徴

- **頑健性：** 圧縮や加工に耐える
- **不可視性：** 品質を損なわない
- **検出可能性：** 必要時に確実に検出できる

電子透かしの種類と応用

可視型電子透かし

- 目に見える形で情報を埋め込む
- 例：テレビ放送の局ロゴ、写真サイトの透かし文字

不可視型電子透かし

- 目に見えない形で情報を埋め込む
- 例：音楽ファイルの著作権情報、機密文書の追跡情報

実際の応用例

- 音楽・映画の著作権管理
- 紙幣・パスポートの偽造防止
- 機密文書の漏洩元特定

→ デジタル時代の権利保護に不可欠な技術

誤り検出符号

誤り検出符号とは

- データ通信や記録時のエラーを検出する技術
- データに冗長な情報（検査用ビット）を付加

パリティチェック

- データのビット数を偶数（または奇数）にする1ビットを追加
- **例：** 1011010 → 10110100（パリティビット0を追加）
- 受信側でビット数を確認してエラー検出

特徴

- **利点：** シンプルで実装が容易
- **限界：** 2ビット以上のエラーは検出不可
- **発展：** より高度な検査方式も存在

演習2：パリティチェックでエラーを見つけよう

偶数パリティで送信された次のデータを確認してください。
エラーがあるものはどれ？

A. 10110100
B. 11010101
C. 00111000
D. 11110000

ヒント：1のビット数を数えて、偶数かどうか確認しよう！

演習2の解答

各データの1のビット数を数える

A. 10110100 → 1が**4個**（偶数）✓ 正常
B. 11010101 → 1が**5個**（奇数）✗ **エラー！**
C. 00111000 → 1が**3個**（奇数）✗ **エラー！**
D. 11110000 → 1が**4個**（偶数）✓ 正常

まとめ

- 偶数パリティでは1のビット数が偶数であるべき
- BとCは奇数なのでエラーを検出
- 実際の通信では、エラーを検出したら**再送要求**を行う

→ シンプルだが効果的なエラー検出方法

まとめ

- **暗号化**は共通鍵・公開鍵・ハイブリッドを使い分ける
- **デジタル署名**で本人確認と改ざん検出を実現
- **SSL/TLS**により安全なWeb通信が可能に
- **電子透かし**で著作権保護、**誤り検出符号**でデータの完全性を保証
- 複数の技術を組み合わせた**多層防御**が重要

感想をチャット欄に書こう！