Claude Code のサンドボックスモード

安全で効率的な開発環境

（社内LT会でMarp入門の説明をした際にデモで作ったスライドです）

Sandbox モードとは？

Claude Code のサンドボックスモードは、**OS レベルの保護機構**です。

ファイルシステムとネットワークの両方を分離し、安全にコマンドを実行できます。

```
🔒 Sandbox Mode
├── Filesystem Isolation
└── Network Isolation
```

技術的な実装

Sandbox は OS レベルのプリミティブを活用：

- **Linux**: bubblewrap を使用
- **macOS**: seatbelt を使用

これらにより、直接的なやり取りだけでなく、**スクリプトやサブプロセスまで対象**に制限が適用されます。

ファイルシステム隔離

現在の作業ディレクトリへの読み書きアクセスのみを許可

```
プロジェクトディレクトリ
├── ✅ 読み書き可能
├── ✅ アクセス許可ファイル
└── ❌ 他のディレクトリへのアクセスをブロック
```

外部のファイル修正を防止します。

ネットワーク隔離

UNIX ドメインソケット経由の制御されたアクセス

- **許可**: ドメインホワイトリスト内の接続
- **プロキシサーバー**: ドメイン接続を制限
- **防止**: 認可されていないドメインへのデータ流出

Sandbox モードの利点

1. セキュリティ向上

- 🛡️ プロンプトインジェクション攻撃への耐性強化
- 🛡️ 悪意のある依存関係やスクリプトからの保護
- 🛡️ SSH 鍵盗出やマルウェアダウンロード防止

Sandbox モードの利点

2. 承認疲れの軽減

内部測定では以下の成果が報告されています：

```
Approval Prompts: 84% 削減

結果:
✅ 開発生産性の向上
✅ ユーザー体験の改善
✅ セキュリティと効率の両立
```

使用例 1: settings.json での設定

```
{
  "permissions": {
    "allow": [
      "Bash(marp:*)",
      "WebFetch(domain:github.com)",
      "Skill(marp)"
    ]
  },
  "sandbox": {
    "enabled": true,
    "network": {
      "allowLocalBinding": true
    }
  }
}
```

使用例 2: 実装設定例

```
{
  "sandbox": {
    "enabled": true,
    "autoAllowBashIfSandboxed": true,
    "allowUnsandboxedCommands": true,
    "network": {
      "allowUnixSockets": [
        "/var/run/docker.sock"
      ],
      "allowLocalBinding": true
    },
    "excludedCommands": [
      "docker", "git", "ruby"
    ]
  }
}
```

Sandbox モードの初期化と管理

- `/sandbox` コマンドで初期化
- `settings.json` でアクセス許可ディレクトリを定義
- ネットワークドメインをホワイトリストで指定
- ツールがドメインアクセスをリクエスト時に許可を与えると、将来のアクセスが自動承認

簡潔なコマンドと明確な設定で、安全で効率的な開発が実現します。

まとめ

Sandbox モード = セキュリティ × 生産性

安全に、効率的に、開発を進めましょう。