JaSST'25 Kansai
帰ってきたセキュリ亭ジャス虎

踏み出せ！飛び出せ！セキュリティテストの巻

演目

1. そのテスト、攻撃に耐えられますか？
2. 「守り」と「攻め」のセキュリティテスト
3. セキュリティテストの実践ステップ
4. まとめ

そのテスト、攻撃に耐えられますか？

～バグを追うだけでは守れない、攻撃に備える新たな品質のカタチ～

時は202X年、セキュリティの嵐がテスト現場を襲う …

時は202X年

ソフトウェア開発はまるで戦（いくさ）。

テストエンジニアは日々、バグ退治に奔走しておりました。

ところが現れたのは「攻撃者」なる黒き影。

情報を奪い、サービスを止め、信頼までも土足で踏みにじる。

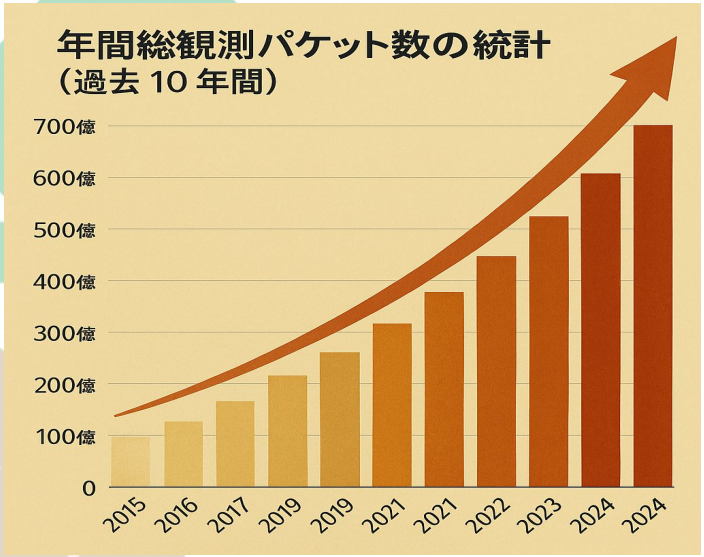
「守る」だけでは追いつかぬ。

これよりは「攻め」のテスト、始めるといたしましょう。

セキュリティの嵐　求められる内製の守り

サイバー攻撃の増加

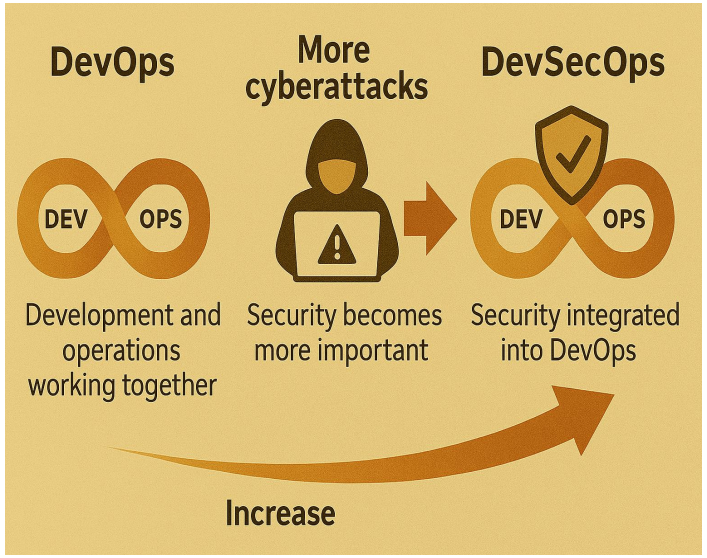
サイバー攻撃の増加に伴う、
セキュリティテストの重要性の増加。



NICT情報情報通信研究機構
NICTER観察レポート2024より

DevOPSからDevSecOPSへ

内製化の流れの中で、 DevOpsはDevSecOps
へと進化。開発・運用に加えセキュリティも組
み込む体制が求められている。



外注化に伴う課題

費用・納期・透明性に限界があり、内製化によ
る対応力と可視化が求められている。

セキュリティの嵐を乗り越える、テストチームの新たな挑戦

1. セキュリティ知識を習得し、脆弱性や攻撃手法を理解する。

2. テストの内製化に向け、診断・ファジング・スキャンの技術と　AI活用を習得する。

3. 開発チームと連携し、セキュリティを設計段階から支援する。

4. 運用・監視にも関与し、ログ分析やインシデント対応に貢献する。

「守り」と「攻め」のセキュリティテスト

「守り」と「攻め」のセキュリティテスト

- セキュリティテストとは、**「何を守るのか」「どこが弱点か」** を**「知る」**ことから始まる。
- 守るべき資産や基本的なセキュリティ対策の理解が**「守りの型」**
- 脅威を発見・検証する技術が**「攻めの型」**

セキュリティテストにおける「守り」の対象

- **データ資産**：漏えいすると**信頼・法的リスク**が発生。
- **サービス資産**：攻撃対象になりやすく、**業務継続**に直結。
- **ユーザ資産**：奪われると不正アクセスの**入口**になる。

資産	例	リスクの一例
データ資産	個人情報、業務ログなど	情報漏えい、法的責任
サービス資産	Webアプリ、APIなど	不正操作、サービス停止
ユーザ資産	ユーザID、パスワードなど	なりすまし、内部情報の流出

セキュリティテストにおける「攻撃」の対象

- **認証**：アクセスする人が**「誰か」**を確認する第一関門。
- **認可**：認証後、**「何をできるか」**を制御して誤操作・悪用を防止。
- **暗号化**：データを守る**「鍵」**。盗まれても中身が分からなければ安全。
- **監視とログ**：異常が起きたときに**「何が起きたか」**を追える仕組み。

要素	意味	攻めの視点 テストで突くポイント
認証	**ユーザが本人かを確認する仕組み**	**パスワードの強度チェック、多要素の有無、総当たり攻撃可否**
認可	**アクセス範囲・操作権限の制御**	権限の越権操作、なりすましアクセスの可否
暗号化	データを安全にやり取りする技術	通信・保存データが平文か、SSL/TLSの適用有無
監視・ログ	**操作や異常を記録し検知する仕組み**	**不正操作の検知可否、ログ改ざんや取得漏れ**

守りと攻めの視点で挑むセキュリティテスト

「守り」の視点＝正常を保証する

- 仕様通りのセキュリティ機能が正しく動作しているか？
- 守るべき資産（データ・サービス・ユーザー）に対する適切な防御があるか？
- 設定漏れ・実装ミスによる意図しない穴がないか？

「攻め」の視点＝異常を突く

- 認証・認可・暗号・監視に対して、どう突破できそうか？
- 「想定外の使い方」「予期せぬ入力」「設定の抜け道」など脆さを探る
- 攻撃者の立場で、破られる可能性を意識して試す

正常も異常も疑って試す。仕様通りを守り、仕様外を突く。

「守りの開発者」と「攻めのテストエンジニア」がチームで補完し合う。

セキュリティテストの実践ステップ

既知も未知も見逃さない、3つの「攻め」

「既知の脆弱性」・「未知の脆弱性」に対応する3種のテスト手法

- **脆弱性診断　：「知られた危険」**をつぶす。
- **ファジング　：「想定外の振る舞い」**を発見。
- **ポートスキャン：「狙われやすい穴」**を防ぐ。

テスト手法	「攻め」の対象	見つける脆弱性	主な役割
脆弱性診断	既知の攻撃パターン	既知の脆弱性	**設定ミスや古いライブラリの脆弱性。**
ファジング	想定外の入力	未知の脆弱性	**境界条件や異常系でのクラッシュ検出。**
ポートスキャン	開放された通信口	不必要なサービス	**攻撃経路となる「入口」の検出。**

セキュリティテスト × AI 分析システムの概要

- システムの目的
  - 攻撃者視点でシステムの弱点を見つけ、事前に対応策を講じる。
  - 専門的なテスト結果をわかりやすく分析し、対応を迅速化する。
  - 未知の脆弱性も含めた幅広いリスクをカバーし、安心できる運用を実現。

テストの実行

- 脆弱性診断
- ファジング
- ポートスキャン

AI-Agentによる分析

- AIがログを解析し、重要な脆弱性や傾向を抽出。

レポート作成・対策立案

- テストエンジニアや開発者が具体的対策を迅速に判断・実行。

セキュリティテスト × AI 分析システム構成

ツール名	役割・目的	ツールの説明	実行環境	ツール情報
Tenable Nessus® Essentials	既知の脆弱性診断	既知の脆弱性診断。	Windows PC上でネイティブ実行。	https://www.tenable.com/products/nessus/nessus-essentials
ffuf	ファジング（入力パターンテスト）	Webアプリの不正入力・パラメータを高速に総当たりテスト。	WindowsのWSL (Ubuntu等)上で実行。	https://github.com/ffuf/ffuf
Zenmap (nmap GUI)	ポートスキャン・ネットワーク調査	ネットワークの開放ポートを検出し、稼働中サービスを調査。	Windowsネイティブ実行。	https://nmap.org/
Dify (AI-Agent)	AIによるテスト結果分析・要約	各ツールの出力ログを自然言語でわかりやすく解説。	Windows PC上のDocker環境またはクラウドサービス	https://dify.ai/

いざ、実践！

Nessus 脆弱性診断結果の確認ポイントと AI分析の注意点

CVSSスコアとVPRの確認 (脆弱性リスクと優先度の把握)

CVSS
(Common Vulnerability Scoring System)

- 0～10の数値で脆弱性の深刻度を示す。
- 高スコア（例：7以上）は優先対応すべきリスク。

VPR
(Vulnerability Priority Rating)

- 脆弱性の現実的リスクを評価、CVSSだけでなく最新の攻撃動向を反映。
- VPR高→即時対応優先度アップ。

INFOカテゴリの確認ポイント

INFOは**「情報提供」**レベルだが**侮れない**。

設定ミスや古いバージョン、未保護の設定などが含まれることがある。
これらは攻撃の足掛かりになることが多いので、ログや設定の詳細をしっかり確認。また、改善や監視強化の必要性を見落とさないようにすること。

AI-Agentでの分析における注意点

脆弱性の重大度（CVSS、VPR）を適切に区別し、重要度順に並べる指示を明確にNFOカテゴリの意味合いを理解し、潜在的リスクを含めて解説させる。

大量の診断結果から重要情報を抽出し、わかりやすいレポートを作るよう促す。

疑問点や補足解説が必要な場合は質問形式でフォローするよう指示。

ffuf ファジング結果の確認ポイントと AI分析の注意点

結果確認ポイント

異常なHTTPステータスコード（例:500系、403系）の検出。

通常と異なるレスポンスサイズや応答時間の変化を注視。

複数回試行し、安定的に再現する異常を重点的に確認。

ファジングによる誤検知もあるため、検証作業が重要。

ファジングで使うFazzデータ（辞書）の入手と内容

Fazz（Fuzzing Attack Zoo）はオープンなファジング用単語リスト集。GitHubなどで公開されており、ffufと組み合わせて利用可能。

代表例: https://github.com/fuzzdb-project/fuzzdb

内容例: SQLインジェクション文字列、XSS用コード、パス名、一般的な攻撃ペイロードなど多種多様。

AI-Agentでの分析における注意点

ファジング特有の誤検知リスクを考慮して分析を行うよう指示。

レスポンスコードやサイズの差異を異常の可能性として解説させる

一時的な異常か再現性のある問題か区別できるよう、**結果の信頼性について言及させる。**

辞書の種類や使ったファジング手法も簡潔に説明するよう促す。

Zenmap (nmap GUI)結果の確認ポイントと AI分析の注意点

スキャン結果確認ポイント

開いているポートと対応サービスの特定。

不要なサービスや意図しないポート開放の検出。

スキャン応答の有無や応答時間の異常確認。

スキャン種別ごとの違いを理解し、検査対象に適した手法を選択。

主なスキャン種別と用途

ICMPスキャン（Pingスキャン）

ホストの生存確認。ネットワーク上に応答する機器があるか調査。

TCPスキャン

TCPポートの開閉確認。サービス稼働の有無を特定。

SYNスキャン（半開きスキャン）など複数の種類があり、高速かつ隠密性がある。

UDPスキャン

UDPポートの開閉確認。DNSやSNMPなどUDPサービスの検出。

TCPより応答が得にくく時間がかかることが多い。

AI-Agentでの分析における注意点

開いているポートのリスクと重要度の説明を促す。

意図しないポート開放の指摘と対処例を示すよう指示。

スキャン種別の特徴を理解させ、検査結果の背景を解説するように促す。

大量のポート情報から重要ポイントを抽出し、簡潔にまとめることを指示。

AI-Agent 活用の拡張例 ～分析を超えた「助言者」として～

活用領域の広がり (分析＋意思決定支援):

AI-Agentは「試験結果の解釈」だけでなく、「次のアクション」を導く支援にも活用できる。

活用内容	説明内容やメリット
修正方法の提示	攻撃ベクトルごとの修正パターンを提示攻撃ベクトルごとの修正パターンを提示（例：XSSならHTMLエスケープ）
原因切り分け支援	ログ・試験結果から原因の推定ロジックを提示し、優先調査箇所を特定
修正のメリット・デメリット提示	修正による影響（例：認証強化 →利便性低下）をリスト化して開発判断を支援
修正コスト・難度の評価	類似プロジェクトのナレッジを元に、修正工数や担当レベルの目安を提示
関連ナレッジ・規格のレコメンド	OWASPやNISTなど関連ガイドラインを適宜リンク・引用して示唆を与える

まとめ

～踏み出せ！飛び出せ！セキュリティテスト～

テストチームの新たな「守り」と「攻め」の役割

- **守り**
  - データ・サービス・ユーザを守る視点を持ち、テスト設計に組み込む。
  - 認証・認可・暗号・ログの「基本4要素」をチェックリスト化して活用。
- **攻め**
  - 脆弱性診断・ファジング・ポートスキャンで攻撃者の視点を持ち、既知・未知のリスクを検出。
  - AIを活用し、試験結果から次のアクション（修正・相談・判断）を導く。
- **AI-Agent活用がセキュリティ内製化を加速させる鍵**
  - 分析・説明・助言の自動化。
  - 外部連携（専門家や外注）との橋渡しにも。

今日の一歩が、テストチームの未来を変える！