サイバーセキュリティ

ネットワーク接続性は組込みアプリケーションに大きな変化をもたらしています。静的、固定された機能で、デバイスに固有なアプリケーションは、「クラウド」接続した世界への移行により、監視、アップグレード、機能強化、そして補給を容易に提供する機会が大量にもたらされるようになりました。

そのマイナス面として、ネットワーク接続されたシステムにおいて、権限のないユーザーがアクセスできるあらゆる箇所(「攻撃対象領域」と総称されます)に脆弱性があると、悪意ある人物がシステムを危険にさらす可能性があります。さらに悪いことに、これらの攻撃者は、あなたやあなたのシステムに特別な関心を持っている必要はありません。ほとんどの場合、彼らは金銭的な利益から愉快犯的なものまで、あらゆる動機で、脆弱性がないかあらゆるものを調べます。そして、彼らの成功とあなたの不運は、あなたの製品と会社に計り知れない損害をもたらす可能性があります。おそらく、賠償請求、コンプライアンス違反の罰金、評判の失墜、パニックに陥った顧客のサポート要求などです。

サイバーセキュリティとは何ですか?

「サイバーセキュリティ」(cybersecurity あるいは cyber-security)という用語は、システム、ネットワーク、およびデータをデジタル攻撃、不正アクセス、損害、または盗難から保護する実践を指します。これには、情報の完全性、機密性、および可用性を保護するために設計された対策とテクノロジの実装が含まれます。重要な側面には、脆弱性の特定、脅威の監視、およびインシデントへの対応が含まれます。

組込みシステムにおけるサイバーセキュリティには、大規模な機械システムや電気システム内の特定の機能に特化した特殊なコンピューティングデバイスの保護が含まれます。自動車制御、医療機器、産業機械、IoT デバイスなどのアプリケーションに見られるこれらのシステムは、多くの場合、リソースが制限され、リアルタイム環境で動作します。これらのシステムのセキュリティを確保するには、処理能力やメモリの制限、リアルタイム操作の必要性など、固有の課題に対処する必要があります。

多層防御とは何ですか?

ネットワーク接続されたシステムに対する防御は 1 つだけでは完全に侵入不可能であることは保証できませんが、複数のレベルのセキュリティを適用することで、1 つのレベルが失敗しても他のレベルが防御を固めます。

この多層防御アプローチの一般的な例えとして、多数の塔、カーテンウォール（城壁）、堀、モット（小丘）、落とし格子、跳ね橋、殺人孔、その他の巧妙なメカニズムによって包囲攻撃から住人を守る中世の城が挙げられます。しかし、これらすべての防御にもかかわらず、城は侵略者によって危険にさらされました。
ネットワーク接続された組込みシステムの開発者が利用できる防御も多種多様です。城の防御と同様、どれも役立ちますが、侵入不可能な防御はありません。次のような防御があります。

- 保存データの保護
- セキュアブート
- ドメイン分離
- 複数の独立したセキュリティレベル(MILS)の設計原則
- 攻撃対象領域の縮小
- セキュアコーディング

LDRA のサイバーセキュリティへの主な貢献はセキュアコーディングに関するものですが、LDRA ツールのテスト機能は他の防御メカニズムを実行するためにも使用できます。

セキュアコーディングとは何ですか?

セキュアコーディングとは、最高のセキュリティ規格とベストプラクティスに準拠したコード設計の原則を指します。セキュリティを優先し、既知および未知の脆弱性からコードを保護します。

セキュアコーディングでは、コードのセキュリティの責任はセキュリティチームではなく開発者に委ねられます。

セーフティクリティカルな分野全体でネットワーク接続がより普及するにつれて、適切な防御の適用に関するガイダンスの必要性も比例して増加しています。

コーディング規約

セキュアコーディング規約は、脆弱性を最小限に抑え、潜在的な脅威から保護するようにソフトウェアが開発されることを保証することを目的としています。これらの規約は、ガイドラインとベストプラクティスを提供することで、一般的なコーディングの欠陥を防ぎ、コード全体の品質を向上させ、規制要件への準拠を確保するのに役立ちます。ソフトウェア開発ライフサイクル全体にわたって安全なプラクティスを推進することで、データの完全性と機密性を保護する上で重要な役割を果たします。

セキュアコーディング規約の例は次のとおりです。

- MISRA
- CERT
- CWE
- NIST 500-268

航空宇宙、防衛分野におけるセキュアな組込みソフトウェア開発

民間航空

民間航空当局は、航空宇宙産業全体にわたるサイバーセキュリティに対する総合的かつ統合的なアプローチを提供する航空宇宙セキュリティフレームワークを適用することで、組込みシステムのサイバーセキュリティの懸念に対処しています。

DO-326A/ED-202A および DO-356A/ED-203A は、航空機のサイバーセキュリティと耐空性を確保するための詳細なガイドラインと方法を提供する特定の規格です。DO-326A は、航空機システムにおけるサイバーセキュリティリスクを管理するためのフレームワークを確立します。DO-356A は、セキュリティ評価の実施、保護対策の実装、セキュリティ要件の検証など、セキュリティ対策を適用するための具体的な方法、技術、ベストプラクティスを提供します。

防衛

多くの防衛プロジェクトでは民間航空の規格を採用していますが、分野固有のセキュリティ規格も存在し、その多くは特定の軍事組織に関連しています。たとえば、DISA STIG は、米国国防総省(DoD)内で使用されるさまざまな情報システムとソフトウェアを構成および保護するための詳細なガイドラインを提供しており、AppSecDev STIG は組込みソフトウェア開発者にとって特に興味深いものです。

自動車分野におけるセキュアな組込みソフトウェア開発

自動車分野では、最近まで ISO 26262 機能安全規格を SAE J3061（サイバーフィジカル車両システム向けサイバーセキュリティガイドブック）が補完していました。これは、ネットワーク接続され自動化が進む車両によって増大するサイバーセキュリティの脅威に対処するための自動車業界の基礎文書として機能しました。SAE J3061 は現在、同様の目的を果たす ISO/SAE 21434（自動車 ー サイバーセキュリティエンジニアリング）に置き換えられています。

製造、プロセス、エネルギー分野におけるセキュアな組込みソフトウェア開発

ISA/IEC 62443 シリーズは、産業オートメーションおよび制御システム(IACS)の安全な開発のための、多岐にわたる業界標準の集合です。サイバーセキュリティの脅威から産業ネットワークを守るための一連のサイバーセキュリティ保護方法と技術を定義します。これらの技術を分類し、メーカー、資産所有者、サプライヤーを含むすべての関係者に適用します。

医療機器分野におけるセキュアな組込みソフトウェア開発

医療機器分野におけるサイバーセキュリティの推奨事項としては、米国 FDA の「医療機器におけるサイバーセキュリティ：品質システムの考慮事項と市販前申請の内容」が挙げられ、新たな取り組みとしては EU の European Health Data Space（欧州ヘルスデータスペース）が挙げられます。

鉄道（GTS）分野におけるセキュアな組込みソフトウェア開発

鉄道および GTS 業界では、クローズドな有線の分離ネットワークからオープンな相互接続ネットワークへの進化により、新たな脅威が生まれています。EN 50128 および EN 5012x シリーズは、サイバーセキュリティを間接的にしか扱っておらず（このような脅威が安全性に影響を及ぼす場合）、対処方法について具体的なガイダンスは提供していません。そのため、多くの GTS および鉄道プロジェクトでは、IEC 62443 規格（上記で説明）への準拠を指定しています。

CLC/TS 50701「鉄道アプリケーション － サイバーセキュリティ」は、鉄道分野におけるサイバーセキュリティの要件と推奨事項を定義します。この規格の目的は、悪意のある人物が鉄道システムの RAMS 特性を侵害できないようにすることです。この規格で説明されているセキュリティモデル、概念、およびリスク評価プロセスは、IEC 62443 シリーズ標準に基づいているか、またはそこから派生したもので、鉄道固有の状況に合わせて調整されています。

宇宙分野におけるセキュアな組込みソフトウェア開発

宇宙での安全なソフトウェアアプリケーションの開発に関連する規格には、「宇宙システム － サイバーセキュリティ管理の要件と推奨事項」の草案 ISO/DTS 20517 や、NASA 手順要件 NPR 7150.2D の一部の要素が含まれます。

サイバーセキュリティと LDRA ツール

さまざまなセクターで提供されるアドバイスは、内容、詳細、成熟度が異なりますが、共通するテーマも多数あります。軽減策が効果的であることを確認するために検証および妥当性確認の手法が使用され、その多くは LDRA ツールスイートによって支えられています。

シフトレフトの原則で採用されている概念は、セーフティクリティカルなアプリケーションを開発する個人やチームには馴染みのあるものです。長年、機能安全規格では同様のアプローチが求められてきました。その結果、機能安全分野で実証された多くのベストプラクティスが、セキュリティクリティカルなアプリケーションにも適用されます。

- 最初に、または各イテレーションの前に機能要件とセキュリティ要件を確立する
- 開発のすべての段階にまで双方向に要件をトレースする
- セキュアな言語サブセットを使用する

- セキュリティ規格を遵守する
- SAST（静的）および DAST（動的）セキュリティテストのプロセスを自動化する
- テイント解析を使用して防御メカニズムをテストする
- 早めに頻繁にテストする

最初に要件を確立する

要件が文書化されていないと、あらゆる側面でコミュニケーションエラーが生じ、やり直し、変更、バグ修正、さらにはセキュリティの脆弱性が生じます。プロジェクト開発をスムーズに進めるには、製品のすべての部分とその開発プロセスをすべてのチームメンバーが同じように理解する必要があります。明確に定義された機能要件とセキュリティ要件は、これを実現するのに役立ちます。

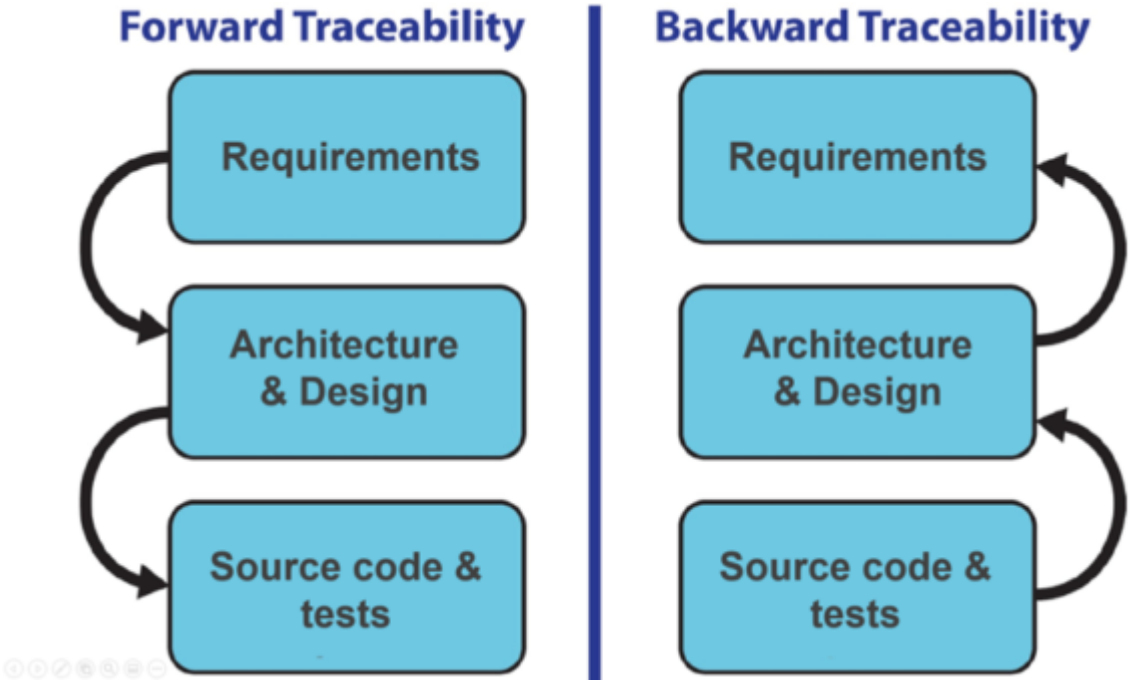
V 字モデルの開発者にとって、このような要件は完全なシステムを定義するものになる可能性があり、DevSecOps を適用する開発者にとっては 1 回の繰返しに過ぎませんが、原則は同じです。これは、ソフトウェアが概念実証を作成するための「知的モデリングクレイ」として決して使用できないと言っているわけではなく、このような実験の最終結果は、明確に定義された要件、そして、それらを満たすよう適切に開発された製品コードでなければならないということです。

双方向のトレーサビリティを提供する

IEEE 標準ソフトウェアエンジニアリング用語集では、トレーサビリティを「開発プロセスの 2 つ以上の成果物、特に相互に先行-後続または主従関係を持つ成果物間で関係を確立できる度合い」と定義しています。双方向トレーサビリティとは、トレーサビリティパスが前方と後方の両方で維持されることを意味します。自動化によって、変化するプロジェクト環境でトレーサビリティを維持することがはるかに簡単になります。

前方トレーサビリティは、実装とテストを含む開発プロセスの各段階ですべての要件が反映されていることを示します。要件の変更や失敗したテストケースの影響は、影響分析を適用することで評価でき、その後対処できます。その結果として得られた実装を再テストして、双方向トレーサビリティの原則が継続的に遵守されているという証拠を提示できます。

同様に重要なのは、指定された要件をまったく満たさないコードを強調表示する後方トレーサビリティです。見落とし、ロジックの誤り、過度の機能追加、悪意のあるバックドアメソッドの挿入はすべて、セキュリティの脆弱性やエラーを引き起こす可能性があります。



双方向トレーサビリティ

セキュアな組込みシステム成果物のライフサイクルは、現場で最後の例が使用されなくなるまで継続することを覚えておくことが重要です。このような成果物が侵害されると、対応、変更された要件、または新しい要件が必要となり、開発エンジニアが長い間触れていないソースコードに対して即時の対応が必要になります。このような状況では、LDRA ツールスイートの TBmanager コンポーネントによって実現される自動トレーサビリティによって、必要なものを分離し、影響を受ける関数のみに自動テストを実施することが可能になります。

セキュアな言語サブセットを使用する

C や C++での開発では、ソフトウェアの欠陥の約 80%が言語の約 20%の誤った使用に起因していることが、長年の研究でわかっています。これに対処するために、開発者は問題のある構造を禁止することで安全性とセキュリティの両方を向上させる言語サブセットを使用できます。

2 つの一般的なサブセットは MISRA C と、カーネギーメロン大学ソフトウェア工学研究所（SEI）の CERT C で、どちらも開発者がセキュアなコードを作成するのに役立ちます。この 2 つの規格の目標は似ていますが、実装方法が異なります。最大の違いは、要求される準拠の度合いです。

一般的に、MISRA C で新しいコードを開発すると、第一原理に基づいて定義されたより厳格で決定可能なルールがあるため、コーディングエラーが少なくなります。MISRA C コーディング規約を参照してソフトウェアを迅速かつ簡単に分析する機能により、コードの品質

と一貫性が向上し、デプロイまでの時間が短縮されます。対照的に、開発者が遡及的にコードにルールを適用する必要がある場合、CERT C が実用的な選択肢となる可能性があります。CERT C を適用してコードを分析すると、ほとんどのソフトウェアセキュリティ攻撃の背後にある一般的なプログラミングエラーが特定されます。

MISRA C または CERT C のいずれかを適用すると、より安全なコードになります。大規模なコードベースにこのような規約を手動で適用するのは現実的ではないため、作業を自動化する静的解析ツールが必要です。MISRA C と CERT C の違いの概要を以下に示します。これには、LDRA ツールスイートの TBvision や LDRArules の両方が役立ちます。

	MISRA C	CERT C
用途	高い一貫性や高信頼性が要求される組込みシステムの開発向けに設計されている 自動車・医療・航空電子・鉄道・原子力・通信・IoT・IIoT など、重要分野で幅広く使用される	多くの場合、ウェブ開発を含め、厳しい要件を満たすセキュリティクリティカルなエンタープライズソフトウェアの開発で使用される
遡及的利用	プロジェクトの最初から採用すべきである 確立され実証済みのコードベースに遡及的に適用すると、問題を引き起こす可能性が高い	主な目的は、新規コード開発をサポートすること その次に優先されるのは、古いコードの修復
決定可能性と自動化	静的解析ツールで自動的にチェックできる ポータブルなコードが生成される	ツールでは、開発者が CERT C の多くの側面を理解し、実装しているかチェックすることはできない

MISRA C と CERT C の比較

セキュリティ重視のプロセス規格を遵守する

上で説明したように、セキュリティ重視の規格は、セキュアな開発ソリューションのもう 1 つの要素を提供します。セーフティクリティカルな分野では、このような規格が機能安全に重点を置いた規格を補完することがよくあります。たとえば、ISO/SAE 21434「自動車 ― サイバーセキュリティエンジニアリング」は、自動車の ISO 26262 機能安全規格を補完します。LDRA ツールスイートの TBmanager コンポーネントは、必要に応じて、セキュリティクリティカルと機能安全の開発者ワークフローを同時に統合できます。

SAST（静的）、および DAST（動的）テストプロセスを自動化する

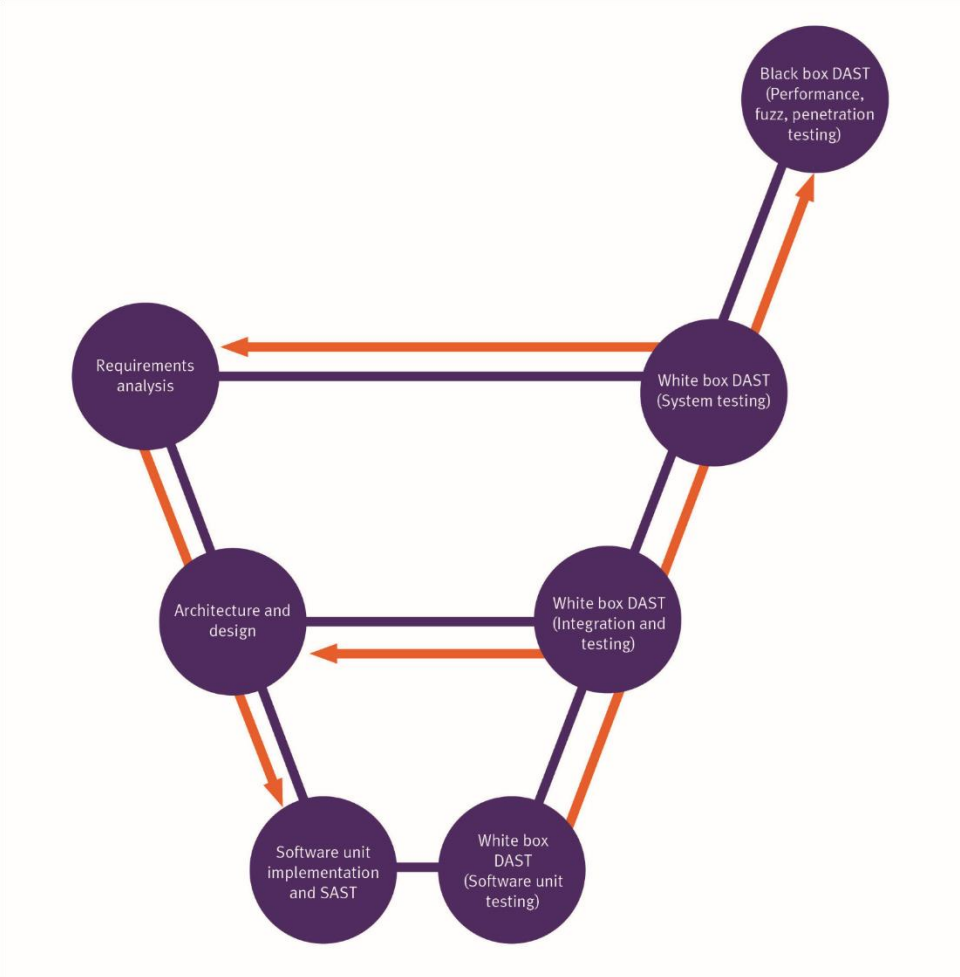
静的解析は、ソースコードの自動検査を伴うテストレジームの総称です。対照的に、動的解析では、ソースコードの一部またはすべてを実行します。このような手法をセキュリティ

の問題に焦点を当てた結果、それぞれ静的解析(またはアプリケーション)セキュリティテスト(SAST)と動的解析(またはアプリケーション)セキュリティテスト(DAST)になります。

これらのグループ分けにはさまざまなバリエーションがあります。たとえば、侵入テスト、機能テスト、ファズテストは、すべてブラックボックス DAST テストであり、機能を実行するためにソースコードにアクセスする必要がありません。ホワイトボックス DAST テストはユニットテスト、統合テスト、システムテストを含み、動的解析によってアプリケーションソースコードの脆弱性を明らかにします。ブラックボックス DAST テストはホワイトボックス DAST テストを補完します。

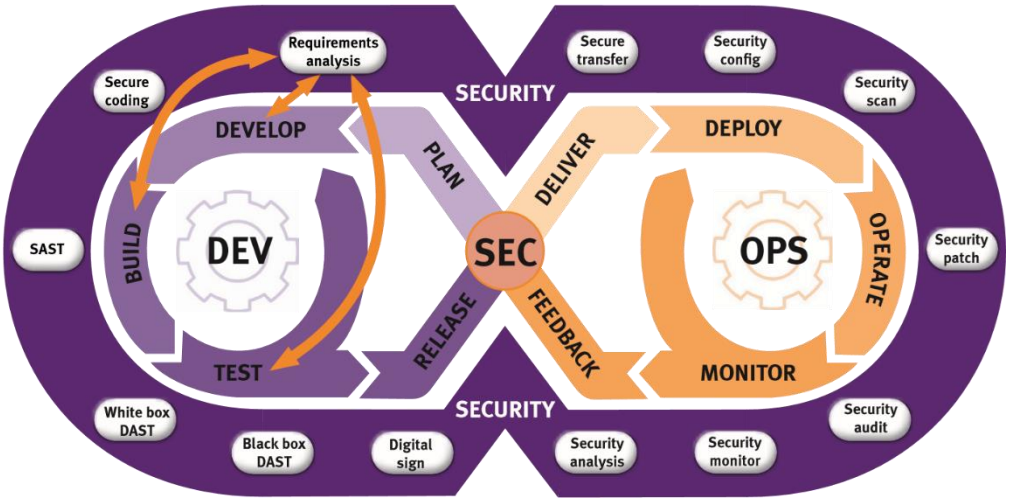
早めに頻繁にテストする

ここで説明するセキュリティ関連のツール、テスト、および技法はすべて、各ライフサイクルモデルで活用できます。V 字モデルでは、これらは機能安全アプリケーション開発に通常関連するプロセスとほぼ類似しており、補完的です。



V 字モデルベースのセキュアソフトウェア開発ライフサイクルでのセキュリティテスト

DevSecOps モデルでは、継続的な開発プロセス全体を通じて、DevOps ライフサイクルにセキュリティ関連のアクティビティが重ね合わされます。



DevSecOps プロセスモデルでのセキュリティテスト

要件のトレーサビリティは、V 字モデルの場合は開発プロセス全体を通じて維持され、DevSecOps モデルの場合は開発の反復ごとに維持されます(オレンジ色で表示)。

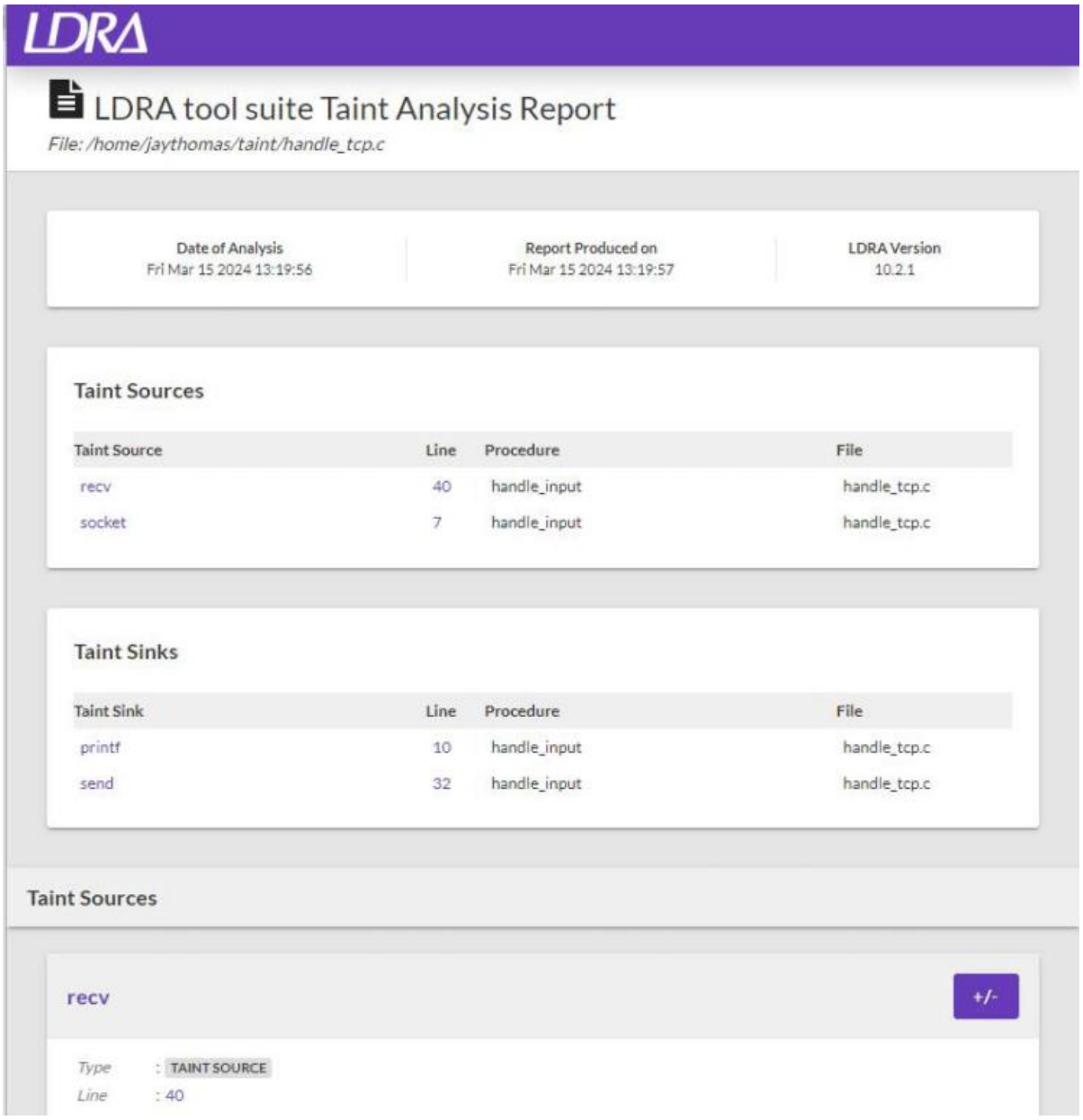
一部の SAST ツールは、コーディング規約への準拠を確認し、複雑さを最小限に抑え、コードが保守可能であることをチェックするために使用されます。その他のツールは、セキュリティの脆弱性をチェックするために使用されますが、実行環境のコンテキストなしでソースコードに対してそのようなチェックが可能な範囲に限られます。

ホワイトボックス DAST を使用すると、コンパイル・実行されたコードを開発環境で、またはさらに良いことには、ターゲットハードウェア上でテストできます。LDRA ツールが提供するコードカバレッジ機能は、コードがすべてのセキュリティおよびその他の要件を満たしていること、およびすべてのコードが 1 つ以上の要件を満たしていることを確認します。システムの重要度に応じて、これらのチェックをオブジェクトコードレベルまで実行することもできます。LDRA ツールスイートはテイント解析(下図:データの汚染源と、そのデータの伝搬先をレポート)をサポートしており、システムが悪意のある人物によるデータ操作から適切に保護されていることを示すのに役立ちます。

TBextreme モジュールが提供する堅牢性テストは、 LDRA ツールスイートの TBrun コンポーネントが提供するユニットテスト環境内で使用できます。これにより、特定の関数が単独でも、コールツリーのコンテキストでも、レジリエントであることを証明できます。

従来、ソフトウェアセキュリティに関連付けられているファズテストと侵入ブラックボックステストの手法は依然として大きな価値がありますが、このコンテキストでは、セキュリティを

基盤として設計および開発されたシステムの堅牢性を確認および実証するために使用されます。



テイント解析：データの汚染源と、そのデータの伝搬先をレポート

この記事は https://ldra.com/cybersecurity/ をベースに日本語化したものです。

富士設備工業株式会社 電子機器事業部
https://www.fuji-setsu.co.jp