03 全8ページ

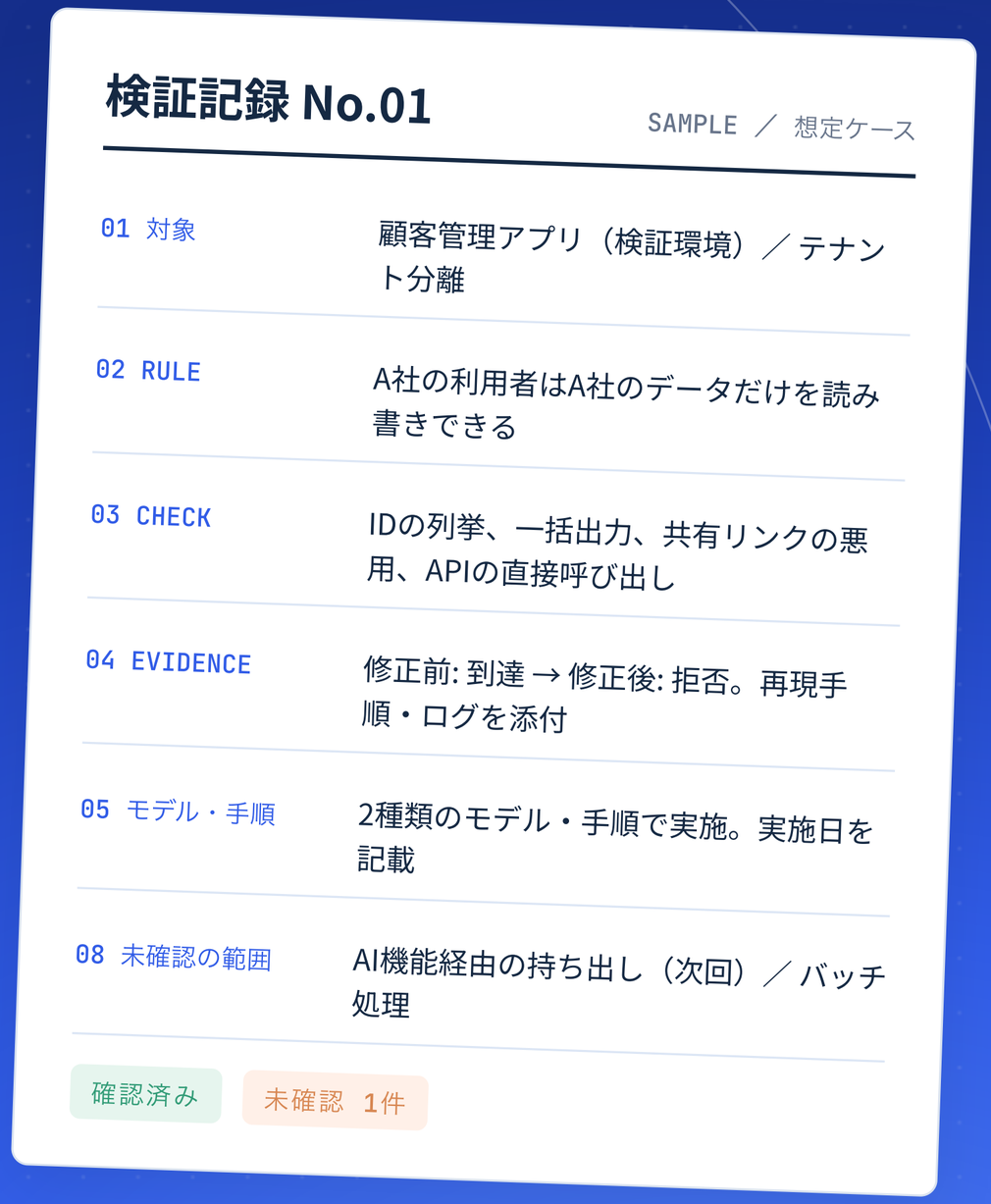
SaaS・ソフトウェア向け

法人導入審査・チェックシートに備える

検証記録の作り方

確認できた事項と未確認の事項を分けて記録し、
提出先に説明する

Securiy セキュリー



WHEN YOU NEED IT

「はい」の根拠になるのは、試した記録だけ

チェックリストに「はい」と書けても、実際に試した記録がなければ説明できません。この冊子は、記録の型と、提出先への整理の仕方をまとめたものです。

SECURITY CHECK SHEET ／ 提出先から届く質問票（想定）　回答前

No.	質問	回答	根拠資料	備考
23	テナント間のアクセス制御は適切に実装されていますか	はい		
24	認証情報・顧客データの取り扱いは文書化されていますか	はい		
25	変更時に再検証する仕組みはありますか	はい		

根拠は？
試した記録を添付

記録が求められる場面

法人顧客のセキュリティ審査

チェックシートの「はい」の裏付けとして、実際に試した記録を求められる

大型案件の契約前レビュー

情報システム部門から、テナント分離をどう検証したかを聞かれる

SOC 2・ISMS の内部監査

認可の設計と検証を、証跡つきで示す必要がある

対応づけの例として使う文書: ISO/IEC 27001 附属書A「情報へのアクセス制限」、SOC 2 の論理アクセス（CC6）（版・項番は提出先の指定に従う）

FORMAT

記録1枚の解剖図

1つの操作につき1枚。守るべき条件・試した操作・結果の3つが揃っていない記録は、後から「何を確かめたのか」が分かりません。

検証記録 No.01

SAMPLE ／ 想定ケース

01 対象	**顧客管理アプリ（検証環境）／テナント分離**
02 RULE	**A社の利用者はA社のデータだけを読み書きできる**
03 CHECK	**IDの列挙、一括出力、共有リンクの悪用、APIの直接呼び出し**
04 EVIDENCE	**修正前: 到達 → 修正後: 拒否。再現手順・ログを添付**
05 モデル・手順	**2種類のモデル・手順で実施。実施日を記載**
08 未確認の範囲	**AI機能経由の持ち出し（次回）／バッチ処理**

確認済み　未確認 1件

RULE	「誰が・何を・どの条件で」を業務の言葉で1行。悪い例「認可が正しいこと」／良い例「A社の利用者はA社のデータだけを読み書きできる」
CHECK	再現できる粒度の手順。使ったモデル・手順の種類と実施日を添える
EVIDENCE	修正前と修正後の両方。到達・実行の可否、ログ・画面
未確認の範囲	試していない経路・条件と理由。次回の予定

VERIFIED ／ NOT YET ／ OUT OF SCOPE

確認済み・未確認・対象外を、経路×シーンで塗り分ける

「安全」の総合判定はしません。初回の2操作（テナント分離・管理者権限の奪取）で確認できた範囲と、まだ試していない範囲を同じ表で示すと、提出先の判断が速くなります。



確認済みに書くこと

再現手順、モデル・手順の種類、実施日、修正差分、再検証の結果

未確認に書くこと

理由（スコープ外／環境が無い）、次回の予定、優先度。例: AI機能経由の持ち出し、バッチ処理、モバイル向けAPI

対象外に書くこと

その経路・機能が無いこと（例: AI機能を導入していない）

TEMPLATE ／ 10 ITEMS

記録に含める10項目（この帳票をそのまま使えます）

共同検証の検証記録もこの10項目でお渡しします。自社で検証するときも同じ見出しにすると、提出先への説明が揃います。

検証記録 No._____

作成日 ____／____　レビュー者 _____

01 対象	アプリ名・環境（検証用）・対象の機能と操作	02 RULE	守るべき条件を1行で（誰が・何を・どの条件で）
03 CHECK	試した操作と手順（再現できる粒度）	04 EVIDENCE	結果（到達の可否・実行の可否）とログ・画面の記録
05 モデル・手順	使ったAIモデルと手順の種類、実施日	06 修正差分	直した箇所と、直し方の要点
07 再検証	修正後に拒否されること／通常操作が通ること	08 未確認の範囲	試していない経路・条件と、その理由
09 残課題	次回に回した項目と優先度	10 判断の記録	誰が・いつ・何を根拠にリリースを判断したか

SAMPLE ／ FIX & RETEST

修正差分と再検証チェックは、この粒度で残す

SaaS・ソフトウェアの「テナント分離」を例にした想定ケースです。修正前後を同じ表で対比し、リリース判断（誰が・いつ・何を根拠に）を添えます。

PULL REQUEST ／ 修正案（想定） app/records.py

```
1     def get_records(request, target_id)
2         rec = Record.get(target_id)
3 -       return rec
4 +       # テナントIDと利用者の権限を照合
5 +       if not belongs(request.user, rec
6 +           raise Forbidden()
7 +       # 一括出力に権限確認を追加
8 +       return rec
```

再検証チェック

No.01 ／ 修正後

- ✓ **他テナントのデータは拒否（修正前: 到達）**
- ✓ **正常な共有リンクは継続**
- ✓ **別の手順・モデルでも同じ結果**
- **AI機能経由の持ち出し（次回）** ☐

判断: ☐ リリース可 ☐ 条件つき ☐ 保留　　判断者 _____　　日付 ____／____

HOW TO EXPLAIN

提出先には、要件番号と記録番号の対応表で答える

提出先が知りたいのは「どの要件を、何で確かめたか」です。要件番号と記録番号の対応表を先に作ると、質問の往復が減ります。

SECURITY CHECK SHEET ／ 提出先から届く質問票（想定） 回答後

No.	質問	回答	根拠資料	備考
23	テナント間のアクセス制御は適切に実装されていますか	はい（2操作を検証済み）	検証記録 01・02	未確認: バッチ処理・モバイルAPI（次回予定）
24	認証情報・顧客データの取り扱いは文書化されていますか	はい	規程 第4章	
25	変更時に再検証する仕組みはありますか	はい	再実行用テスト（記録 02）	

整理の3点

審査票の項目と記録を対応づける

例:「情報へのアクセス制限」の項目 → RULE-01／CHECK-01〜04／EVIDENCE-01。項目ごとに記録の番号を書き添えます（対応づけの例であり、適合の判断ではありません）。

未確認の範囲を先に言う

試していない経路（例: AI機能、モバイル向けAPI）を明記し、次回の予定を添えます。

第三者認証の代わりにしない

検証記録は審査の補助資料です。認証取得の証明ではないことを説明します。

記載の検証シーンは共同検証のテーマ例で、業界別の導入実績・認証取得を示すものではありません。対象は、検証と修正を行える自社アプリ・独自APIです。

記録の型を、
共同検証で一緒に作る。

初回スコープ（1アプリ・2操作・10営業日）で、この冊子の型に沿った検証記録・修正案・再検証チェック・再実行用テストをお渡しします。本番の認証情報や実データは不要です。

Securiy は Tempura technologies株式会社のセキュリティ事業です。

FREE CONSULTATION

30分無料相談を予約する

提出先と求められている項目を伺い、どの操作から記録を作るかを一緒に決めます。

予約ページへ →

SET

01 検証シーン集 ／ 02 プログラムのご案内

SaaS・ソフトウェアの3シーンは 01、対象・期間・費用・体制は 02 へ。