LINE BOT セキュリティガイドライン

作成部署 :LINE 株式会社 セキュリティ室

作成日:2018/4/1

1. 文書の目的

Contents Provider Server（以下CPサーバーといいます）は、LINEの公式アカウントとしてLINE CHANNELと連動する機能を保持しています。本文書は、LINE CHANNELと連動するBusiness Connect（以下BCといいます）のCPサーバーに対するベストプラクティスに基づいたベースラインを定義し、適正なセキュリティ状況を維持することを目的としています。

2. 文書の適用範囲

本文書はLINE CHANNELを利用したBCのCPサーバーの開発、運用時に適用するものとします。

但し、以下のベストプラクティスの構築、運用、評価の各詳細項目に対して御社内部に該当指針、対象チェックリスト項目が細分化された場合、その指針、チェックリストを追加参考資料として参照しても構いません。

3. ガイドライン対象

LINE CHANNELと連動するBCのCPサーバー（バックアップデータも含んだバイナリデータ）、及びBCのCPサーバーを介した通信が対象。

4. ベストプラクティス項目

4.1. BC の CP サーバー構築

4.2.1. インストール、環境設定

① セキュアな OS をインストールすること（サポート対象の OS インストールなど）

② セキュアなサーバーアプリケーションをインストールすること（なるべく最新バージョンの使用など）

③ 適切な OS、及びサーバーアプリケーションの環境設定を行うこと（使用しないサービスは無効など）

④ 適切な認証、認可機能適用すること（パスワードポリシー、アカウントロック機能など）

（注意事項）

上記の環境設定（サーバーチェックリスト）の参考資料として添付した「LINE_SecurityChecklist_JP」を参考にし、構築すること

4.2.2. 個人情報を含む内部ファイルの暗号化実施

① 個人情報などをデータベース形式などでファイルシステムへ保存する際、暗号化を施した後、暗号化されたデータを内部ファイルに保存すること 別紙 1 別紙 2 別紙 3

4.2.3. LINE Channel Gateway との安全な通信経路確保

① LINE Channel Gateway と CP サーバー通信は必ず SSL(HTTPS)で暗号化すること

② 個人情報などを含む通信を CP サーバーから貴社の他のサーバー、または他のマシンに対して送信・受信する際、暗号化を施した通信で個人情報を送信・受信すること

③ LINE Channel Gateway、CP サーバーとの通信は、正常なリクエストであるかを確認するため、デジタル署名の有効性の検証を行うこと(Signature Validation)

④ LINE Channel Gateway、CP サーバーとの通信は、指定された正しい通信であるかどうかを確認するため、リクエストヘッダに指定された認証が用いられている。特に、Channel access token など認証情報が外部などに漏洩された場合、成りすましなどが発生する恐れがあるため、認証情報にはしっかり管理すること。(アクセス制御管理、及び通信、保存時、暗号化実施など)

4.2.4. アクセスコントロール機能実施

① CP サーバーのアカウント、及び権限は必要最低限な範囲で付与すること

② 適切な認証プロセスを実施すること(複雑なパスワード組み合わせ、アカウントロック機能など)

③ 適切なリモート接続機能を使用すること(暗号化された通信によるリモート接続など)

4.2.5. 変更検知監視機能実施

① (可能であれば、)不審なプロセスなどによるファイル変更検知、および監視機能導入、運用し、事前に検知できるようにすること(既存の改ざん検知ツール導入、運用)

4.2.6. ウイルス対策製品導入

① 対象サーバーに対する資産の重要性(サービス停止時の影響度など)を基準にし、使用環境(対象 OS、対象ネットワーク広域、稼動中のサービス)などを考慮した上で(可能であれば、)ウイルス対策製品を導入し、運用すること

4.2.7. ネットワークセキュリティ対策製品導入

① ファイアウォールにより保護されたネットワーク広域に CP サーバーを設置、運用すること

② 不正侵入やウイルスなど不審な通信の検知、防御を行うため、IDS(Intrusion Detection System)、IPS(Intrusion Protection System)製品を導入、運用すること

4.2. CP サーバー運用

4.3.1. タイムスタンプ

① 外部ネットワーク機器からのログ、内部システムからのログなどでの整合性を確保するため、Network Time Protocol などを利用した正しいタイムスタンプ設定を行い、運用すること

4.3.2. パッチマネージメント

① OS、及びサーバーアプリケーションに対するパッチ適用、更新のための適切なパッチマネージメント導入し、運用すること

4.3.3. ログ

① 適切なログ対象設定（適用）をした後、ログ管理を行うこと（監査、セキュリティログの拡張など）

② 適切なログ保存期間を設定（適用）した後、ログ管理を行うこと（各国の該当通信法律に準拠すること）

③ 適切なログ監視機能構築、導入、及びログ管理を行うこと

4.3.4. サーバー、及びサービスの脆弱性診断

① OS、OS に含まれているサービスに対する脆弱性診断（自動診断ツールによる診断など）、及び更新（運用）を行うこと

② 構築したサービスに対する脆弱性診断（自動診断ツール、及び手動による診断）、及び更新、修正（運用）を行うこと

4.3.5. バックアップ

① 適切に指定されたバックアップ周期毎にデータをバックアップすること（対象データ、バックアップポリシーなどは各国の該当通信法律に準拠すること）

② 個人情報などを含むデータをバックアップする際、暗号化バックアップメディアを使用し、バックアップすること

③ 個人情報などを含むデータをバックアップする際、暗号化された情報としてバックアップすること

4.3.6. インシデントレスポンス

① 適切なインシデントの事前対応プロセスの構築、及びインシデントレスポンスの運用（実施）を行うこと

4.3. CP サーバー評価、及び改善

4.3.1. CP サーバー構築、運用に対する評価、及び改善

上記の「4.1. CP サーバー構築」、及び「4.2. CP サーバー運用」に記述された各項目に対して適切に指定された評価期間毎に各項目を評価されること。その評価により発見された問題点、改善点がある場合、適切な手段で速やかに改善すること。

別紙 1

◆ハッシュ関数

ハッシュ 関数名	ハッシュ長	推奨有無	備考
SHA256	256 bits	推奨	長さ 256bits 以上のハッシュ使用推奨
SHA384	384 bits	推奨	長さ 256bits 以上のハッシュ使用推奨
SHA512	512 bits	推奨	長さ 256bits 以上のハッシュ使用推奨
MD5	128 bits	非推奨	長さ 256bits 未満のハッシュ使用 （ハッシュコードの衝突の恐れがあり）
SHA1	160 bits	非推奨	長さ 256bits 未満のハッシュ使用 （ハッシュコードの衝突の恐れがあり）

別紙 2

◆共通鍵暗号

共通鍵暗号名	暗号ブロック	推奨有無	備考
AES-128/192/256	128,192,256 bits	推奨	128 bits 以上のブロック使用推奨
Camellia-128/192/256	128,192,256 bits	推奨	2000 年に NTT と三菱電機により共同開発されたブロック暗号
KCipher-2	ストリーム	推奨	DDI 研と九州大学により設計、実装されたストリーム暗号
DES	64 bits	非推奨	
2-key Triple DES	64 bits	非推奨	
3-key Triple DES	64 bits	非推奨	

別紙 3

◆SSL protocol や CipherSuite 設定の例

-SSL verion 3 以下を削除する

-DES/3DES を削除する

[apache(httpd.conf)の例]

SSLProtocol All -SSLv2 -SSLv3 //可能であれば TLSv1.2 のみを使う

SSLCipherSuite ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES256-SHA:AES128-GCM-SHA256:AES128-SHA256:AES128-SHA

[nginx(nginx.conf)の例]

ssl_protocols TLSv1 TLSv1.1 TLSv1.2; //可能であれば TLSv1.2 のみを使う

ssl_ciphers "ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES256-SHA:AES128-GCM-SHA256:AES128-SHA256:AES128-SHA";