

TECHNICAL OVERVIEW

# MukenVault

侵入後の被害拡大を抑制する  
Runtime Stability アーキテクチャ

---

**Superasystem 株式会社**

Technical Overview / Public Edition  
<https://superasystem.jp>

## CONTENTS

# 目次

Technical Overview の全体構成

CH 01	はじめに .....	P. 03
CH 02	世界の変化 .....	P. 04
CH 03	暗号化されたデータを守る、最後の領域 .....	P. 05
CH 04	侵入後に、起きること .....	P. 06
CH 05	既存対策との関係 .....	P. 07
CH 06	MukenVault とは .....	P. 08
CH 07	KeyLess Architecture .....	P. 09
CH 08	導入前と、導入後 .....	P. 10
CH 09	MukenVault が提供する価値 .....	P. 11
CH 10	Runtime Stability .....	P. 12
CH 11	検証結果 .....	P. 13
CH 12	特徴 .....	P. 14
APPX.	MukenVault が保護する対象 .....	P. 15
CH 13	About Superasystem .....	P. 16

## この資料について

本資料は MukenVault の技術概要を公開する **Public Edition** です。  
評価ライセンス・PoC・技術検証レポートなど詳細情報は、お問い合わせください。

## CHAPTER 01

# はじめに

侵入を防ぐだけでは、足りない時代へ

サイバー攻撃は、攻撃者が一つの弱点を見つければ成功します。しかし守る側は、すべてのシステムや設定を常に守り続けなければなりません。

さらに AI の登場により、攻撃者は情報収集や攻撃の自動化をこれまで以上に低コストで行えるようになっていきます。そのため「侵入を完全に防ぐ」ことは、ますます難しくなっています。

だからこそ重要なのは、**侵入された後に何を守るか**です。

たとえ侵入されても、パスワードや暗号鍵などの重要情報を盗まれない仕組みを作ること。  
**それが、これからのセキュリティに求められています。**

## CHAPTER 02

# 世界の変化

セキュリティは「侵入前提」へ

世界では現在、以下のような考え方が広がっています。

**Zero Trust** 「信頼しない」を前提に、すべてのアクセスを検証する

---

**Cyber Resilience** 攻撃を受けても、業務を継続できる強さを設計する

---

**Confidential Computing** 利用中のデータ(メモリ)を、外部から隔離する

---

呼び名は違ってても、本質は同じです。

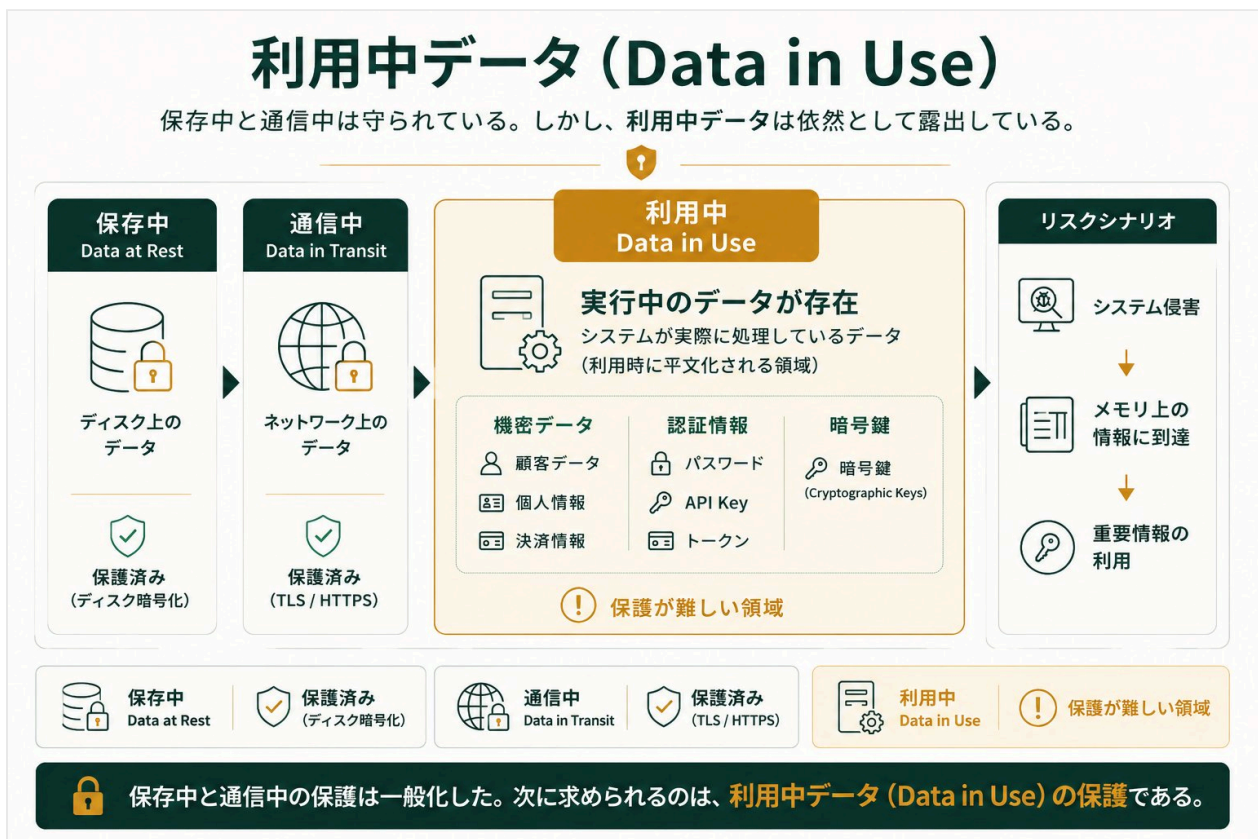
——**侵入されても、被害を最小化する。**

MukenVault も、同じ問いから生まれました。

## CHAPTER 03

# 暗号化されたデータを守る、最後の領域

保存中と通信中は守られている。利用中だけが、残されていた。



システムが実際に動いている間、認証情報・暗号鍵・API キー・トークンは、メモリ上に存在しています。**利用中のデータ (Data in Use)** —— ここが、これまで十分に保護されてこなかった領域です。

## 解けない構造

これまで誰も解決できなかった理由は、明確です。メモリを暗号化するには「鍵」が必要です。しかし、その鍵自体も、メモリに置くしかありません。

**メモリを守る鍵もまた、メモリに置くしかない。**  
この入れ子構造が、これまで解けない問題でした。

## CHAPTER 04

# 侵入後に、起きること

"正規ユーザー" と区別がつかない

攻撃者が、認証情報や暗号鍵を取得すると——

システムから見ると、すべて「正規アクセス」として見えます。

攻撃者はシステムから見ると、  
"正規ユーザー" と区別が付きません。

例えば、こうした操作が「正規」として通る

- 顧客データの閲覧
- バックアップの復号
- API の利用
- 管理操作の実行

システムから見れば、すべて「正常」です。

CHAPTER 05

# 既存対策との関係

外側を守る対策に、内側を守る視点を

Firewall、VPN、EDR、認証・認可——いずれも非常に重要な対策です。

ただ、これらが守っているのは、主に「侵入を防ぐ」領域です。



MukenVault は既存製品を置き換えるものではなく、**既存対策に加わる新しいセキュリティレイヤ**です。

「侵入を防ぐ」対策と組み合わせて、「侵入された後も守る」レイヤを担います。

## CHAPTER 06

# MukenVault とは

鍵を保存しない、独自の暗号化技術

従来の考え方

## 鍵を、より強く守る

(金庫を厚くする・監視を増やす)

MUKENVAULT の考え方

## 鍵を、置いておかない

(必要な瞬間だけ、その場で生み出す)

**KeyLess は、一般的な CPU 環境で利用可能な、鍵を保存しない独自の暗号化技術です。**

従来の暗号化では、暗号化や復号に使用する鍵をメモリやストレージに保存する必要があります。しかし、メモリ上の秘密情報を保護する場合、その鍵自体の保護が課題となります。過去には、CPU 内部の特殊な領域に鍵を保存し、メモリを暗号化する研究も行われましたが、運用上の課題から広く普及するには至りませんでした。それでも、「汎用 CPU でもメモリ暗号化は可能である」という重要な可能性を示した研究でした。

KeyLess は、**CPU や実行環境の状態から再生成可能な鍵を、必要な時だけ生成し、利用後に削除**します。これにより、暗号鍵を保存することなく、ランタイム中の秘密情報を保護します。

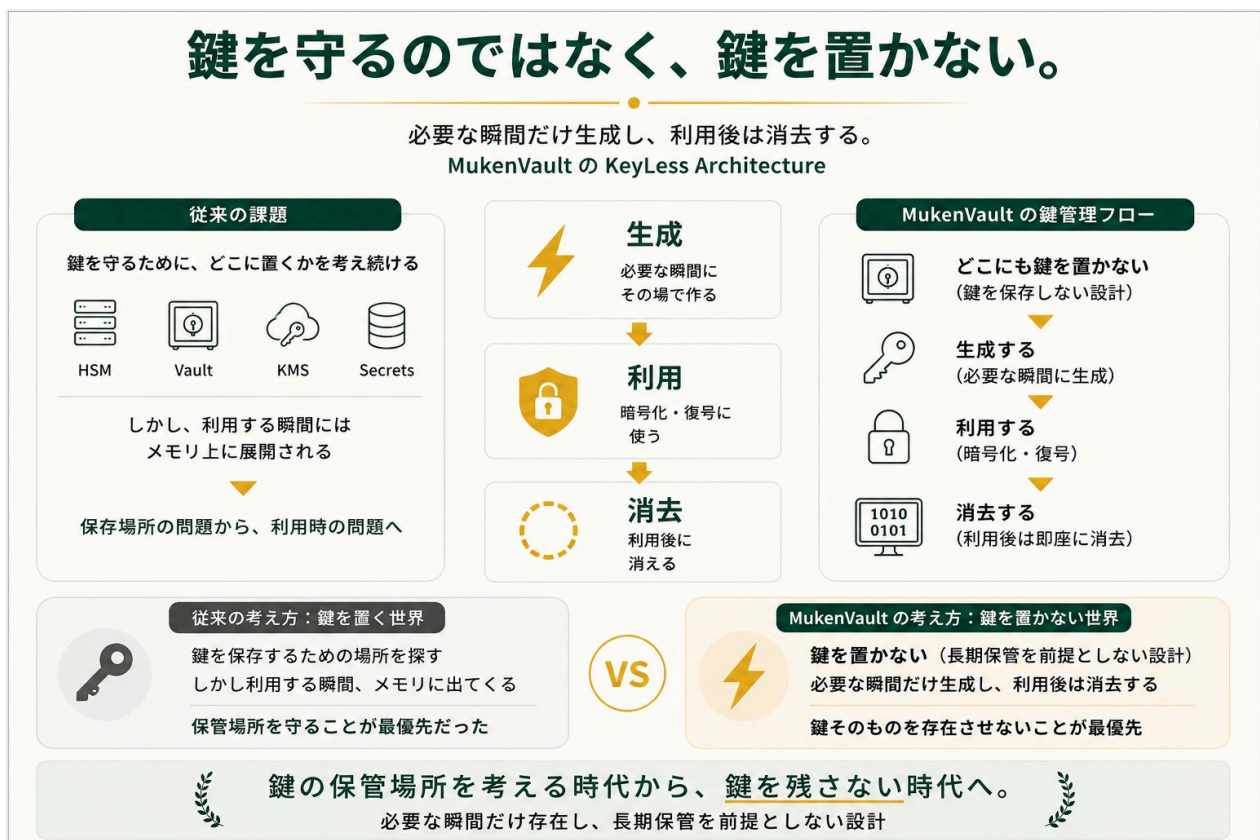
専用のセキュリティチップや特殊な CPU 機能を必要とせず、一般的な CPU 環境で利用できることを特徴としています。MukenVault は、こうした仕組みにより、**アプリケーション単位・ユーザー単位・セッション単位**でのメモリ保護に挑戦しています。

CHAPTER 07

# KeyLess Architecture

再生成可能な鍵を、必要な瞬間だけ生成する

MukenVault の核心は、「再生成可能な鍵を、必要な瞬間だけ生成する」という設計にあります。サーバー固有の物理的特徴から再生成可能な鍵を、必要な瞬間に生成し、使い終わればその場で消去します。



「鍵を守るための鍵」を別途用意する必要がなくなります。  
必要な瞬間だけ存在し、保存しない——これが KeyLess アーキテクチャです。

CHAPTER 08

# 導入前と、導入後

導入する前と後で、何がかわるか

同じ攻撃を受けたとき、メモリから取得できるデータが、根本から変わります。

## 突破されても、価値を渡さない。

同じ攻撃を受けても、攻撃者が得られる成果が変わる

### KeyLess OFF (導入前)

取得されたデータがそのまま利用可能

攻撃者が取得した内容 (例)	メモリダンプの実例 (一部)
<b>ユーザー認証情報</b> (例) password=Admin@123 api_key=sk-abcd1234...	アドレス      16進ダンプ (抜粋)      ASCII (抜粋) 56172c714c40    6f 72 70 31 2e 30     orp1.0...  56172c714c50    6f 64 75 63 74 69     oduction  56172c714c60    6f 6e 2e 65 78 61     n.exa...  56172c714c70    6d 70 6c 65 2e 63     mple...  56172c714c80    6f 6d 00 00 00 00     om.....  ...
<b>暗号鍵・秘密鍵</b> (例) tls_key=BEGIN PRIVATE KEY... ssh_rsa AAAAB3NzaClyc2FAAAADAQ...	561735a1b2c0    70 61 73 73 77 6f     paswo...  561735a1b2d0    72 64 3d 41 64 6d     rd=Adm...  561735a1b2e0    69 6e 40 31 32 33     !n@123...  561735a1b2f0    21 00 00 00 00 00     !.....  ...
<b>クレジットカード情報</b> (例) credit_card=4222-4242-4242-4242	561735a1b2c0    42 45 47 49 4e 20     BEGIN...  561735b3c4e0    50 52 49 96 41 54     PRIVAT...  561735b3c4f0    45 20 52 53 41 20     E RSA...  561735b3c500    50 52 49 96 41 54     PRIVAT...  561735b3c510    45 20 4b 45 59 2d     E KEY...  ...
<b>セッショントークン</b> (例) session=eyJhbGciOiJIUzI1NiIs...	...
<b>データベース接続情報</b> (例) db_password=DbPassword123! user=admin	...

顧客情報を利用できる   
 秘密鍵を悪用できる   
 なりすましが可能

### KeyLess ON (導入後)

取得されても利用できないデータのみ検出

攻撃者が取得した内容 (例)	メモリダンプの実例 (一部)
..... 意味のないバイト列 ..... (例) a8f3b2c1d4e5...	アドレス      16進ダンプ (抜粋)      ASCII (抜粋) 7fd41c90c00    f9 09 b9 a2 cf 48     ...H...  7fd41c90c10    25 4d c6 68 34 d6     RM.h4...  7fd41c90c20    7d 9a 8d 4a 63 0d     }.Jc...  7fd41c90c30    8c 7a 13 25 b8 20     .z.%...  7fd41c90c40    eb 31 62 aa 85 4b     .1b..K  ...
..... 暗号化されたデータ ..... (例) 9c7e1f0a3b6d...	...
..... ランダムなデータ ..... (例) 2e4d5f8a9b1c...	7fd41d2e320    3f db d0 df cb b3     ?......  7fd41d2e330    ee 45 32 1d 6a 4f     .E2_30...  7fd41d2e340    9c 95 06 2f 5d 34     .../ J4  7fd41d2e350    2d 6b 42 bd e0 93     -kb...  7fd41d2e360    28 0b 65 8e 6a 95     (.e.j...  ...
..... 保護されたデータ ..... は取得不可 (例) どのデータが機密情報が特定できない	...

顧客情報を利用できない   
 秘密鍵を悪用できない   
 なりすましができない

同じ攻撃。取得できるデータだけが変わる。

サービスは止まらない。正常に稼働しながら、メモリ上の機密情報だけを自動的に保護します。

**KeyLess Architecture**  
必要な瞬間だけ存在し、長期間保持しない設計。

**KeyLess は「取得されても価値がない世界」を実現し、攻撃下でも制御できるシステムを守り続けます**

	従来	MukenVault
取得される情報	認証情報・鍵が利用可能	暗号化された状態のみ
攻撃者の次の行動	"正規アクセス"として活動	取得しても活用しにくい
被害の広がり方	内部で連鎖しやすい	構造的に抑制される

MukenVault の KeyLess アーキテクチャにより、被害の連鎖が構造的に抑制されます。

CHAPTER 09

# MukenVault が提供する価値

攻撃の連鎖は、どこで止まるか

攻撃者は侵入後、メモリやプロセスから認証情報を探します。  
 ——その時点で、結末が分かります。



攻撃の入口を完全に閉じることは、難しいかもしれません。だからこそ、**攻撃成果を構造的に小さく**することが、これからのセキュリティの重心になります。

CHAPTER 10

# Runtime Stability

攻撃下でも、制御を失わない

MukenVault は、**Runtime Stability** という設計思想に基づいています。

目的は攻撃をゼロにすることではなく、攻撃を受けた状態でも業務継続性を保てる設計にあります。

**攻撃を防ぐだけではない。攻撃下でも、制御を失わない。**

Runtime Stability (ランタイムスタビリティ) の思想

**従来のセキュリティ**

侵入  
↓  
被害拡大  
↓  
停止  
サービス停止・業務停止

止めるしか選択肢がない

**Runtime Stability**

侵入  
↓  
被害抑制  
攻撃の影響を最小化  
↓  
継続運用  
サービス継続・制御継続

制御を維持し、守りながら走り続ける

**ABS (アンチロック・ブレーキ・システム) に学ぶ**

車が滑ってもABSが各輪を制御し、横滑りを防ぎながらハンドル操作を可能にします。

Runtime Stabilityも同じ発想。

攻撃を完全に防ぐことはできなくても、攻撃下でもシステムの**制御性**を維持します。

**KeyLess のアプローチ：メモリ内の情報を無価値化**

導入前：メモリに平文で存在	導入後：メモリ内で暗号化
プロセスメモリ (平文)	プロセスメモリ (暗号化)
IPアドレス: 192.168.1.10	IPアドレス: 3f7a8c2d6e...
パスワード: Passw0rd!	パスワード: a9b1d47fc2...
認証トークン: abcd1234efgh	認証トークン: 9d6e3b7a1c...

不正アクセス・横展開・情報漏洩のリスク大

**現実の攻撃：メモリダンプで秘密情報が取得可能**

OpenSSH (ssh-agent) 秘密鍵を復元 シールド機能のAES鍵がメモリに平文で存在し、秘密鍵を完全復元	Nginx (TLS) TLS秘密鍵を復元 RSA秘密鍵 (d, p, q) がメモリに平文で存在し、完全復元
MariaDB (TLS) TLS秘密鍵を復元 RSA秘密鍵の全パラメータがメモリに平文で存在し、完全復元	Kubernetes (Secrets) シークレットが平文 環境変数・etcdストレージに平文で存在 プロセスメモリにも残存

現実の攻撃では、多くの情報がメモリから取得されています

**KeyLess は「取得されても価値がない世界」を実現し、攻撃下でも制御できるシステムを守り続けます**

## CHAPTER 11

# 検証結果

思想だけでなく、実環境で

MukenVault は、実際の OSS 環境で検証を進めています。

## 27

検証済 OSS

Web / DB / VPN / 認証 等

## 0

鍵抽出の成功例

メモリダンプ攻撃 / 社内検証

### 検証対象 OSS (一部)

Web サーバー	Nginx / Apache
データベース	MariaDB / PostgreSQL / Redis
VPN	OpenVPN / WireGuard
認証	OpenSSH / Authentik / KeePassXC
基盤	Kubernetes / Docker
その他	Nextcloud / OAuth2 Proxy など

### 検証結果 (概要)

- 鍵の滞在時間 (メモリ上に鍵が存在する時間) : 実測値は技術検証レポートに記載
- メモリダンプ攻撃 (gcore //proc/[pid]/mem 等) における鍵抽出の成功例 : 0 件
- 既存アプリケーションへの動作影響 : 軽微

### 知的財産

特許出願済 特願 2025-241853 (KeyLess アーキテクチャ)

国際出願 PCT 進行中

詳細は技術検証レポートにてお伝えします。

## CHAPTER 12

# 特徴

導入と運用の前提

既存アプリ改修	不要
実装形態	ソフトウェアベース
動作要件	特定ベンダー製品に依存しない / AES-NI 搭載 CPU で動作
保護単位	プロセス単位
鍵管理	KeyLess アーキテクチャ
設計思想	Runtime Stability

## APPENDIX

# MukenVault が保護する対象

「うちの〇〇も守れますか？」への回答

MukenVault は、システム稼働中にメモリ上で利用される、各種の機密情報を保護対象とします。

## 主な保護対象

認証情報	ユーザー名・パスワード・ハッシュ値 など
暗号鍵	TLS 秘密鍵・対称鍵・署名鍵 など
API キー	外部 SaaS・AI サービス・決済 API のキー
OAuth トークン	アクセストークン・リフレッシュトークン
セッショントークン	Web アプリケーションのセッション識別子
クラウド認証情報	クラウド基盤のアクセスキー・サービスアカウント情報

上記以外の機密情報についても、MukenVault の保護対象に含まれる場合があります。

「うちの〇〇は守れますか？」については、お気軽にご相談ください。

対象範囲や対応可否は、システム構成・OS・ミドルウェアによって異なります。詳細は技術検証時に個別にご確認いたします。

## CHAPTER 13

# About Superasystem

侵入後も価値を守る、Runtime Stability の研究・開発

AI 時代において、侵入を完全に防ぐことは、ますます難しくなっています。だからこそ私たちは、侵入後も価値を守り、継続的な運用を支えるための **Runtime Stability アーキテクチャ** を研究・開発しています。

**AI 時代の Cyber Resilience を支える、  
Runtime Stability Platform へ。**

**安心という土台の上に、挑戦が生まれる社会へ。**  
A foundation of safety, on which challenge can grow.

## 次のステップ

### パートナーをご検討の企業様

Sler・セキュリティベンダー・クラウド事業者など、販売・実装パートナーとして検討いただける企業様を募集しています。

- 01 オンライン情報交換 (NDA 対応)
- 02 技術詳細・ビジネスモデルのご説明
- 03 評価ライセンスでの技術検証
- 04 パートナーシップのご検討

### 導入をご検討のお客様

MukenVault の導入は、提携パートナー経由でご案内しております。お客様の業界・地域に応じた最適なパートナーをご紹介します。

## Contact

上記いずれの場合も、まずはお気軽にお問い合わせください。

**MukenVault に関するお問い合わせ** [support@mukenvault.com](mailto:support@mukenvault.com)