PACIFIC SYSTEMS

PACIFICサイバーセキュリティ研究所
研究所レポートVol.2

2026.04

解析で浮かび上がる脅威の輪郭

～ランサムウェア解析を企業防御に翻訳する～

PACIFICサイバーセキュリティ研究所
Email：PacificCSLab@pacific-systems.co.jp
TEL：048-845-2285

目次

PACIFIC サイバーセキュリティ研究所レポート Vol.2

解析で浮かび上がる脅威の輪郭

～ランサムウェア解析を企業防御に翻訳する～

1. はじめに

本レポートは、PACIFIC サイバーセキュリティ研究所が実施したランサムウェア LockBit 3.0 の分析結果を、企業防御にどう生かせるかという観点からまとめたものである。目的は、解析技術そのものを紹介することではなく、解析から何が分かり、それを防御・検知・初動対応にどう活かしていけるかを明らかにすることにある。

対象読者は、社内セキュリティ担当者、CSIRT 関係者、ならびに顧客向け提案や支援に関わる技術・営業担当である。技術的な事実は保持しつつも、各章で「何を確認したか」「何が分かったか」「防御上どう見るか」を明示し、マルウェア解析の専門家でなくても追いやすい構成とした。

なお、本レポートでは、攻撃の再現や悪用につながる情報の記載は避け、再現可能な操作手順や詳細な解析手法の記載は行わず、確認結果と企業防御上のポイントに焦点を当てる。

2. レポートサマリー

【目的】

LockBit 3.0 の分析結果をもとに、解析から何が分かり、それを防御・検知・初動対応にどう活かせるかを整理する。解析技術そのものを解説するのではなく、表面上の挙動だけでは見えにくい回避構造や、その背景にある設計を企業防御の観点で捉え直すことを目的とした。

【調査対象】

調査対象は、RaaS 型ランサムウェアとして広く流通した LockBit 3.0（Black）である。LockBit は代表的なランサムウェアの一つであり、特定の個別事例にとどまらず、企業防御の観点から広く示唆を引き出しやすい題材として選定した。

【調査手法】

公開脅威情報の確認と、実物サンプルの静的解析を組み合わせて分析した。公開脅威情報では外形的な挙動や処理の流れを確認し、静的解析では FLOSS、Capa、Ghidra を用いて、表面上の挙動だけでは見えにくい構造を補足した。

【主要な発見】

公開脅威情報上で「ほぼ動かない」ように見えた挙動の背景として、２億回のカウントダウンループを確認した。加えて、可読文字列の少なさ、非標準セクション、自己書き換え、動的 API 解決など、多層的な回避構造が確認された。これにより、見かけ上の静かな挙動の背後に、解析を難しくする設計が存在することが分かった。

【防御上の示唆】

十分な挙動が見えないことを安全の根拠にしてはならない。長時間待機、不自然な CPU 使用、非標準セクション、見えにくい API 解決なども危険シグナルとして扱い、見えにくさそのものを判断材料に含める視点が重要である。外形的な挙動だけで判断せず、静的解析など複数の観点を組み合わせて評価することが、防御・検知・初動対応の質を高める。

3. 用語の簡易説明

本レポートで使用する主な専門用語を以下に示す。マルウェア解析に馴染みのない読者はこの章を先に参照されたい。

用語	説明
公開脅威情報 ANY.RUN	クラウド上で疑似的に実行環境を提供し、マルウェアの外形的な挙動や通信を観察できる解析環境。本レポートでは ANY.RUN を利用した。
FLOSS	実行ファイル内の文字列や、単純な難読化を解除した後の文字列を確認するための解析ツール。実行せずに内部の文字列情報を抽出できる。
Capa	マルウェアが持つ機能や能力を静的に分類するツール。MITRE ATT&CK や MBC（Malware Behavior Catalog：マルウェアの振る舞いを整理した分類体系）との対応を自動で示す。
Ghidra	実行ファイルを逆アセンブル・逆コンパイルし、コード構造を確認するための解析ツール。NSA （National Security Agency：アメリカ国家安全保障局）が開発した OSS。
RaaS	Ransomware-as-a-Service。ランサムウェアをサービスとして提供し、攻撃実行者に利用させる運用形態。
自己書き換え	実行時に自分自身のコードやデータを書き換える構造。静的解析を難しくする要因となる。LockBit 3.0 では本体の復号に使用されていた。
ATT&CK	MITRE が整理した攻撃者の戦術・テクニック・手順のフレームワーク。T1027（難読化）等の番号で参照される。
IOC	Indicators of Compromise。侵害の痕跡を示すファイルハッシュ・IP アドレス・ドメイン名等。
TLP	Traffic Light Protocol。情報共有の範囲を示す分類（例：RED＝受信者限り、AMBER＝限定共有、GREEN＝コミュニティ内共有、CLEAR＝公開可）。

4. 調査対象と分析アプローチ

4.1 なぜ LockBit 3.0 を調査対象としたのか

調査対象はランサムウェアの LockBit 3.0（Black）である。LockBit は、攻撃基盤を複数の実行者に提供する RaaS（Ransomware-as-a-Service）型として広く流通した代表的なランサムウェアであり、ランサムウェア作成用ツールの流出後は、類似した亜種や派生版も多く確認されている。企業防御の観点からも、特定のランサムウェアだけに限られない教訓を引き出しやすい題材といえる。

4.2 公開脅威情報と実物解析をどう組み合わせるか

分析は、公開脅威情報の内容と、実物サンプルに対する静的解析（マルウェアを実際に動かさずに行う解析）を組み合わせて行った。前者では外形的な挙動、処理の流れ、ネットワーク通信、ATT&CK 上の位置づけを確認し、後者では文字列、能力分類、コードレベルの構造から、公開脅威情報だけでは説明しにくい「見えにくさの理由」を補足した。なお、実物の解析には分析用に公開されているサンプルを使用した。

4.3 各手法で何を確認するのか

本レポートにおける着眼点は、(1) 公開脅威情報上の挙動、(2) 実物の静的解析で確認できる構造、(3) 両者の照合によって説明可能になる回避挙動の三点である。

調査全体の見取り図

分類	分析手法	主な確認対象	この手法で分かること
公開脅威情報	**公開脅威情報の確認（ANY.RUN）**	外形的な挙動・処理の流れ・ネットワーク通信・ATT&CK マッピング	何が起きそうかを先に把握する
実物解析	**文字列抽出（FLOSS）/ 能力分類（Capa）**	文字列・API・暗号化処理・見えにくさの構造	見えにくさの手がかりを捉える
	コードレベル解析（Ghidra）	遅延実行・自己書き換え・API 動的解決	公開脅威情報の確認で説明できなかった停止要因を補足する

5. 公開脅威情報の確認結果

5.1 公開脅威情報で確認できたこと

公開脅威情報では、復旧妨害や設定変更に関係する操作、身代金要求につながる挙動、ファイル・プロセスの変化など、ランサムウェアとして典型的な動きが確認できた。一方で、対象サンプルは期待したほど十分な挙動を示さず、見かけ上「ほぼ何もしない」ように見えるケースがあった。

この「十分に動作しない」という確認結果は、単なる解析失敗ではなく、後続の静的解析につながる重要な手がかりとなった。公開脅威情報の確認だけでは、なぜ十分な動作が出ないのかを説明しきれず、ここに実物の静的解析を組み合わせる意義があった。

公開脅威情報で確認された主な挙動

カテゴリ	確認された挙動	ATT&CK テクニック
復旧妨害	ボリュームシャドウの削除・回復モードの無効化	T1490
ログ削除	セキュリティ・システムイベントログの消去	T1070.001
プロセス停止	セキュリティ製品プロセスの強制終了	T1562.001
ファイル暗号化	拡張子の変更・身代金ノートの生成	T1486
ネットワーク通信	Tor.onion ドメインへの HTTPS 通信	T1071
難読化・回避	「ほぼ動かない」現象（遅延実行の可能性）	T1027

6. 実物の静的解析①：FLOSS による文字列抽出

6.1 文字列抽出で何を見ようとしたか

公開脅威情報の解析だけでは把握しにくかったサンプル内部の特徴を補うため、文字列情報およびファイル構造の観点から静的解析を実施した。特に、可読文字列の有無、外部機能を呼び出すための API の露出状況、実行ファイル内部の区画であるセクション構成に通常と異なる点がないかを確認することで、主要な処理部分がどの程度表層から見えにくい構造を持つかを評価した。

6.2 可読情報が少なく、ランダム文字列が多かった

文字列抽出の結果、可読な情報は限定的であり、多くはランダム性の高い文字列として観測された。また、難読化解除によって得られた情報もごく少量にとどまり、主要な処理部分がそのまま文字列として露出していないことが確認された。

加えて、GetTickCount、GetProcAddress、LoadLibraryExA などの API も確認された。これらは、それぞれ時刻情報の取得、実行時の関数特定、外部ライブラリの読み込みに関わるものであり、プログラムが動作する環境を見ながら処理を分岐させたり、利用する機能を後から組み立てたりする可能性を示している。こうした作りは、静的解析の段階で処理のつながりを把握しにくくする要因になりうる。

6.3 .xyz セクションは何を意味するのか

ファイル構造上の特徴としては、Windows の実行ファイルで一般的に見られる .text（コード領域）、.rdata（読み取り専用データ）、.data（書き換え可能データ）などの PE セクションに加えて、.xyz という通常はあまり見られないセクション名が確認された。こうした独自のセクションは、解析を分かりにくくする目的で追加されることがあり、本件でも後続の解析結果とあわせると、主要な処理部分のコードまたは暗号化済みデータの格納領域である可能性が高いと判断した。

FLOSS 観察結果サマリー

確認項目	観察結果	解釈
静的文字列（2,078 件）	ほぼランダムな文字列	本体が暗号化されている可能性を示す所見
難読化解除文字列	1 件（"vE29" のみ）	FLOSS では復元不可のレベルの難読化
注目 API	GetTickCount / GetProcAddress / LoadLibraryExA	タイミング検出・動的 API 解決が存在する可能性
非標準セクション	.xyz セクションを確認	暗号化された本体コードの格納場所の可能性

6.4 文字列から見えた「見えにくさ」の特徴

本結果は、本サンプルが単純な文字列抽出や一般的な静的確認だけでは全体像を把握しにくい構造を持つことを示している。大量のランダム文字列と限定的な可読情報という組み合わせは、主要な処理部分が何らかの形で暗号化または難読化された状態で保持されている可能性を示している。

6.5 この結果から何に注意すべきか

⚠ 初動対応の一時判断で、何を不審の手がかりと見るべきか

- 今回のサンプルでは、可読な文字列が少なく、多くがランダム文字列として観測されたうえ、通常あまり見られない .xyz セクションも確認された。
- こうした構造は、明確な不正文字列や典型的な API 呼び出しだけでは、不審性を十分に見抜けない場合があることを示している。
- そのため、通常とは異なるセクション構成、可読情報の乏しさ、ランダム文字列の多さといった不自然な特徴そのものを、検知・一時判断上の重要な手がかりとして扱う必要がある。
- つまり、「何が悪いか」がすぐに分からなくても、通常の実行ファイルと比べて何が不自然かを見る視点が重要である。

7. 実物の静的解析②：Capa による能力マッピング

7.1 Capa で何を確認したか

サンプルの機能的特徴を静的に把握するため、能力分類の観点から解析を実施した。目的は、暗号化、難読化、防御回避、永続化など、ランサムウェアとして想定される機能のうち、どの範囲が表層的に把握可能かを確認することである。

7.2 確認できた能力は一部に限られていた

Capa を用いて、サンプルが持つ機能や能力を静的に分類し、どのような振る舞いが想定されるかを確認した。ここでは、難読化、暗号化、データ処理といった観点から、実行ファイル内に表れている特徴を確認している。結果として、難読化・パッキング、RC4 暗号化、XOR エンコードに関する特徴が検出された一方、anti-analysis、impact、persistence など本体展開後に明確化しやすい能力については確認できなかった。これは、本サンプルが未展開の状態であり、解析時点で全体の機能が露出していなかったためと考えられる。

分類カテゴリ	検出された能力	ATT&CK / MBC
DEFENSE EVASION	難読化・パッキング (Obfuscated Files)	T1027 / E1027.m02
CRYPTOGRAPHY	RC4 暗号化 (RC4 KSA)	C0027.009 / C0028.002
DATA	XOR エンコード (2 か所で検出)	C0026.002
(未検出)	anti-analysis / impact / persistence 等	— (本体が未展開のため)

7.3 「未検出」は何を意味するのか

「未検出」は、その機能が存在しないことを意味するものではない。今回のように本体が未展開の状態では、通常は検出されやすい機能まで表面化していない可能性がある。すなわち、本サンプルは主要な本体機能を初期状態で直接露出させず、解析ツールから把握しにくい形で保持している可能性が高い。

FLOSS で観測されたランダム文字列群や非標準セクションの存在とあわせると、本体が暗号化・展開前の状態で格納されており、能力分類ツールからは暗号化処理の断片のみが検出された可能性が高い。

7.4 この結果をどう受け止めるべきか

⚠ マルウェア解析の初期段階で、見えている機能が少ない場合の考え方

- 今回のサンプルでは、難読化・暗号化・データ処理に関する特徴は確認できた一方で、典型的な機能の多くは表面化していなかった。
- このことは、能力分類ツールの出力が少ない場合でも、脅威性が低いとは限らないことを示している。
- 特に、暗号化関連だけが浮かび上がる場合は、本体が未展開または隠蔽されている可能性を考慮する必要がある。
- そのため、能力分類ツールの結果は単独で判断せず、文字列抽出やコードレベル解析とあわせて読むことが重要である。

8. 実物の静的解析③：Ghidra によるコードレベル解析

8.1 コードレベル解析で何を確認したか

公開脅威情報上で十分な動作が観測されなかった理由を補足するため、コードレベルで本体展開処理および解析回避の可能性がある処理を確認した。特に、公開脅威情報の上で「止まって見える」現象の技術的背景を明らかにすることを目的とした。

8.2 観察結果①―最重要発見：２億回のループ

コードレベルの解析では、実行開始直後に到達する entry 関数から呼び出される主要関数の冒頭に、２億回のカウントダウンループが確認された。これは、公開脅威情報上で観測された「ほぼ動かない」挙動を説明しうる、今回の最重要発見である。

また、この主要関数の周辺では、メモリ上のコード領域を書き換える自己書き換え的な処理が繰り返し観測され、実行前提で本体を展開する構造もうかがえた。

8.3 観察結果② ―― なぜ止まって見えたのか

✔ 本調査で最も重要な発見

本体処理の起点となる関数の冒頭に、２億回のカウントダウンループが確認された。
この処理は、公開脅威情報上で観測された「十分な動作が出ない」「ほぼ何もしないように見える」という現象と整合する。
公開脅威情報上での「停止」は単純な非動作ではなく、解析環境の観察時間内に本体挙動を表面化させないための遅延設計である可能性が高い。
公開脅威情報の確認で得られた現象と、コードレベル解析で得られた構造的根拠が結びついたことで、サンドボックス回避の可能性をより具体的に説明できた。

8.4 観察結果③ — 自己書き換えと動的 API 解決が示すもの

コードレベルの解析では、自己書き換えを伴う処理に加え、Windows の機能呼び出しをその場で組み立てるような構造が確認された。これは、利用する機能を表面上見えにくくするものであり、通常の文字列抽出やインポート情報の確認だけでは、実際の挙動を追いにくくする。

また、実行ファイルが自分自身の内部構造を参照しながら、各セクションを確認する処理も確認された。FLOSS で観測された .xyz セクションの存在とあわせると、本サンプルは、内部に格納した本体コードまたは暗号化済みデータを必要に応じて取り出して扱う設計である可能性が高い。

Ghidra で確認された回避ロジックの構造

発見した構造	技術的根拠	現象との対応
２億回カウントダウンループ	中心関数の冒頭で確認（200,000,000 回）	ANY.RUN で「止まる」現象の技術的説明
自己書き換えコード	entry 関数でメモリ上コードを連続書き換え	静的解析ツールが制御フローを追えない
ハッシュベース API 解決	数値定数（ハッシュ）での API 特定	FLOSS が呼び出し元を検出できない理由
PE 自己解析	自分のセクションヘッダーを走査	.xyz セクションを動的に特定・展開

8.5 この発見が重要な理由

以上の結果から、本サンプルは、(1) 初期状態で本体を見えにくく保持し、(2) 実行時に自己書き換えや復号を伴って本体を展開し、(3) その前段で遅延処理により解析環境のタイムアウトを誘発する、という多層的な回避構造を持つと考えられる。

特に ２億回ループの確認は、公開脅威情報側で確認された「止まる」現象の技術的説明として重要である。今回のレポートで最も大きな成果は、公開脅威情報の確認だけでは原因不明だった現象を、実物の静的解析によって回避ロジックとして説明可能にした点にある。

8.6 公開脅威情報と実物解析をどう読むか

⚠ 解析を進める中で、マルウェア的な挙動が見えにくい場合の判断ポイント

- 公開脅威情報で十分な挙動が確認できないサンプルを、早い段階で「無害」または「不活性」と判断しないことが重要である。
- 見かけ上は大きな動きがなくても、内部では実行条件の待機や本体展開前の準備が進んでいる場合がある。
- 遅延実行、自己書き換え、動的 API 解決のような構造を持つ場合、公開脅威情報の確認だけでは本体挙動に到達できないことがある。
- そのため、表面上の挙動が乏しいこと自体を、回避設計の可能性として捉える視点が必要である。
- 公開脅威情報の確認に加え、静的解析で背景構造を補うことで、「なぜ動かないのか」「何を隠しているのか」を説明しやすくなる。
- 長時間待機、不自然な CPU 使用、非標準セクション、見えにくい API 解決は、通常の実行ファイルと異なる不自然さとして、補助的な危険シグナルにできる。

9. 統合分析：何が分かり、どう読むべきか

9.1 「ほぼ動かない」挙動は２億回ループで説明できた

今回の分析で価値が大きかったのは、公開脅威情報の確認結果と実物の静的解析を相互に照合できた点である。公開脅威情報では原因不明だった「ほぼ動かない」挙動を、Ghidra で確認した ２億回ループが技術的に説明したことで、観察結果とコード上の構造が一つの流れとしてつながった。

9.2 今回の LockBit 3.0 は多層的に「見えにくさ」を設計していた

また、FLOSS で見えたランダム文字列、Capa で見えた限定的な能力、Ghidra で見えた自己書き換えと API の動的解決は、すべて「静的解析では把握しにくい」「短時間の観察では表面化しにくい」という同じ方向性を示している。今回の LockBit 3.0 は、単一の回避技術ではなく、多層的に「見えにくさ」を設計していたと考えられる。

9.3 比較表で見る各ツールの役割の違い

本表は、各ツールの優劣ではなく、確認できる情報の種類の違いを示したものである。ANY.RUN は外形的な挙動の把握に強く、FLOSS は文字列、Capa は機能断片の分類、Ghidra は内部構造や実行ロジックの確認に強みを持つ。今回のサンプルでは、ANY.RUN だけでは停止要因を説明できなかったが、複数ツールを組み合わせることで、見えにくさの理由と本体構造を補完的に把握できた。

各ツールで確認できた情報の比較

回避ロジック・特徴	ANY.RUN	FLOSS	Capa	Ghidra
２億回ループ（タイミング回避）	止まった（原因不明）	×	×	★ 特定
自己書き換えコード	×	×	×	★ 確認
ハッシュベース API 解決	×	API 名は見える	×	★ 特定
RC4+XOR 暗号化	×	ランダム文字列	★ 検出	★ 確認
.xyz カスタムセクション	×	★ 発見	×	★ 役割確認
プロセスツリー・IOC	★ 観察	×	×	×
TTPs（ATT&CK）	★ 自動生成	×	部分的	部分的
ネットワーク通信	★ 観察	×	×	×

※凡例：★＝当該ツールで確認・特定できた事項、×＝確認できなかった事項

10. 防御側が具体的に強めるべき点

10.1 十分な挙動が確認できない検体を安易に無害と判断しない

公開脅威情報で十分な挙動が見えないサンプルを、早い段階で無害または不活性と判断しないことが重要である。見かけ上は大きな動きがなくても、内部では遅延実行や本体展開前の準備が進んでいる可能性がある。

10.2 見えにくさそのものを危険シグナルとして扱う

トリアージでは、明確な悪性文字列や既知シグネチャだけでなく、長時間待機、不自然な CPU 使用、非標準セクション、見えにくい API 解決といった特徴も危険シグナルとして扱う必要がある。表面上の挙動が乏しくても、内部構造には不審な特徴が表れている場合がある。

10.3 単一ツールではなく複数の観点で判断する

サンプル評価では、単一ツールの結果だけに依存せず、外形的な挙動、文字列情報、能力分類、コードレベル解析を組み合わせて確認することが重要である。公開脅威情報で判断がつかない場合は、そのまま打ち切らず、追加の静的確認を行うか、社内の責任者や外部支援先に相談する運用を決めておく必要がある。

10.4 不審な兆候が見えた段階で警戒度を引き上げる

初動対応では、暗号化挙動や拡張子変更が見えてからでは遅い可能性がある。たとえば、端末やサーバで不自然な負荷上昇が続く、ログ削除や復旧妨害に関わる操作が見られるといった段階でも、警戒度を引き上げて追加確認に進む視点が重要である。今回の分析結果は、表面上は静かに見える段階でも、内部では攻撃準備が進んでいる場合があることを示している。企業防御においても、明確な悪性挙動が見えないことを安全の根拠にせず、見えにくさそのものを判断材料に含める視点が重要である。

11. まとめ

11.1 今回の分析で最も重要だった点

本調査では、LockBit 3.0 を題材に、公開脅威情報の確認結果と実物の静的解析を組み合わせることで、公開脅威情報だけでは見えにくかった特徴を補足した。特に、２億回ループの確認により、「ほぼ動かない」挙動の背景を技術的に説明できた点は、今回のレポートの核となる発見である。

11.2 見えにくさをどう読むか

自己書き換え、本体展開、ハッシュベースの API 解決などは、いずれも「見えにくさ」を重視した設計として整合していた。企業防御では、見えていないから安全なのではなく、見えにくい段階でも危険を読み取る視点が重要である。

11.3 当研究所として今後どう取り組むか

当研究所としては、こうした解析知見を、顧客への提言、検知・監視設計、初動訓練へ結び付けていくことに加え、社会的責任の観点からも、広く防御力向上に資する形で体系化し、発信していくことが重要であると考えている。今後は、既知サンプルの分析結果と現行 RaaS の動向を対比しながら比較分析を進め、実務と社会の双方に還元できる知見として蓄積していく。

連絡先・問い合わせ先

PACIFIC サイバーセキュリティ研究所
Email：PacificCSLab@pacific-systems.co.jp
電話番号：048-845-2285

PACIFIC サイバーセキュリティ研究所レポート Vol.2

解析で浮かび上がる脅威の輪郭

～ランサムウェア解析を企業防御に翻訳する～

発行日：2026 年 4 月 24 日

著者：PACIFIC サイバーセキュリティ研究所 鈴木

"One Step Forward, One Step Beyond."