

AINET 0.1：制御された不正ノードによる試験と可用性の修正

Two Brain

技術報告 TB-AINET-SEC-001・改訂1・2026年10月31日・未査読

要旨

AINET 0.1は既存のIPネットワーク上で署名付きObjectを交換する実験的な規約である。相手認証と作者署名だけでは、通信相手が処理を進めることや資源を公平に使うことを保証できない。本研究では、単一Linux上の制御された不正・停止ノードを用い、実装の信頼境界を調べた。独立プロセスの試験で、登録済み1ノードが全処理枠を占有できる問題と、受信本文の停止が保存領域全体のロックを保持する問題を再現した。別のGo API試験では、ストリーム確立後に呼び出し元の期限が読み取りを中断しない問題を再現した。相手ごとの受け入れ数、容量予約を伴うロック外での受信、確立済みストリームへのキャンセル伝播を修正した。同じプロセス試験80項目は修正前70項目、修正後80項目が成功し、可用性2条件の正常取得はそれぞれ期限内0/5件から5/5件へ変わった。これは特定の実装不具合とその局所的な回帰試験の結果であり、新しい暗号原理、分散DoSへの耐性、HTTPS・IPFS・Nostrに対する優位を示すものではない。

1 研究の目的

Two Brainは、ソフトウェアやAIが情報を扱う際に、作者の識別、出典に関する申告、公開の判断を明確にできる情報交換を研究している。サイバー攻撃はその信頼境界を調べる動機になるが、攻撃が起こることからIPやWebが時代遅れだとは結論できない。Webでも構造化データ、アプリケーション署名、認可、複製を扱える。AINETも現在はIP、標準暗号、go-libp2pを利用しており、安全性は独自の名前や形式から推定せず、具体的な境界ごとに評価する必要がある。

先行する技術報告TB-AINET-TR-001はローカルの情報交換と混雑時の拒否を記録した[1]。その多くは結合試験・故障注入であり、コピーに保持された署名が検証できること自体は既知の性質である。本研究では、**暗号学的には正しい登録済み相手が通信を進めなくなった場合、無関係な正常取得を妨げられるか**を問う。また、不正資料の正式保存、受領が不確かなメッセージの重複、確立済み操作のキャンセルを確認する。成果はこの実装の原因分析と修正であり、独自プロトコルの必要性を確立したという主張ではない。

2 対象・脅威・方法

対象のAINET 0.1はTCP、Noise、Yamux、Ed25519、SHA-256、制限した決定的CBORを使う。公開には署名付きPublic条件と提供ノードの明示公開を必要とする。相手の登録は手動で、実装が許す接続先はリテラルのループバックアドレスに限る。自動探索、任意コード実行、AIモデルは本研究に含まれない。

表1：脅威モデルと測定範囲。

相手・資源	扱い
未登録の相手	登録と鍵の対応を検査、接続を拒否
登録済み不正提供元	Record・本文・応答対応・長さを変異
登録済み停止ノード	要求または本文を保留し、別の正常仕事を測定
OS・管理者・実験器	信頼する。OS侵害と鍵窃取は試さない
通信・データ	単一ホストの127.0.0.1、合成文章

実験器は対象ノード、正常な読み手、操作可能なプローブを、別の鍵と保存領域を持つLinux独立プロセスとして起動する。未登録接続の確認では追加ノードも使用する。プローブは実際の通信方式と登録済み鍵認証を使い、その後に不正フレーム、差し替えたRecord、本文停止、処理の停止を生じさせる。対象は自分たちが管理する127.0.0.1上のノードと合成データだけであり、外部サービスを対象としない。試験ノードの保存領域と秘密鍵は終了時に削除する。

修正前後でプローブ実行ファイル、Python実験器、ローカル計画、合成入力を維持した。修正前は2026年10月31日 12:49:18 UTC、修正後は12:54:38 UTCに完了した。環境は双方ともLinux 6.18.57 x86-64、glibc 2.41である。同一ホスト上で順に実行した診断試験であり、条件順序の無作為化や独立した配備環境の反復ではない。元の実行ファイルとソースを保存し、修正版はGo 1.26.10でビルドした。全ハッシュと原記録を成果物に残している。

不正要求・提供元の条件は適用できる範囲で3反復、可用性条件は各5反復とした。正常仕事は小さな公開文章の取得であり、CLI操作が2秒以内に完了することを成功条件とする。この期限はローカルの試験基準で、本番のサービス水準ではない。早い拒否も正常仕事の失敗に数える。実験器は停止状態が成立したことを確認してから取得を測り、停止を解除した後の復帰も記録する。JSONLの1行は反復を含む1つの検査である。80項目は80種類の独立した安全性保証でも80取得要求でもない。

計画は実行前にローカルで作成した。計画ハッシュは使用した文書を識別するが、外部時刻証明付きの事前登録ではない。期限試験はGoテストの1プロセス内で実際のlibp2p接続を使い、独立プロセス試験とは別集計にした。呼び出し期限250 ms、観測上の終了許容上限800 msとし、相手は1.2秒間応答を保留する。HELLO待ちと要求後の応答待ちを各5回試した。その後、期限なしの明示キャンセルと、容量上限付近の受信回帰試験を追加した。

3 再現した問題と修正

3.1 ノードによる全処理枠の占有

修正前は、受け入れたストリームごとに全体4枠のうち1枠を割り当て、相手別のアプリケーション上限を持たなかった。登録済みプローブが4本でHELLOを完了した後に要求を送らず待機すると、4枠が埋まった。無関係な読み手の取得5回はすべて直ちに失敗し、HELLO authentication/versionという複数の原因をまとめた診断を返した。この実験条件では枠の枯渇に伴う失敗であり、エラー文言だけから不正な鍵だと判断することはできない。

修正では全体4枠を維持し、同じ認証済み相手からの処理を同時2件に制限した。ノードのロック内で受け入れと相手別カウンタを更新し、終了時に解放する。再試験ではプローブの4本中2本を受け入れ、2本を拒否し、正常取得は5回すべて期限内に完了した。これは1台による占有を制限する。許可済み2台の共謀、大量の送信処理、別の共有資源による枯渇は残る。待機キュー、比例的な公平性、自動再試行は追加しておらず、上限超過時はストリームを拒否する。予約済みのBUSY応答も未実装である。

3.2 遅い受信による保存領域全体の停止

修正前の保存処理は、本文を受信する間も保存領域全体のロックを保持していた。提供元が正しい署名付きRecordを送り、その後の本文を送らないと、対象ノードはロックを持ったまま待機する。別の相手が保存済みの公開Objectを求めても、その保存処理へ進めない。正常取得5回はすべて2秒の観測期限を超えた。その後、実験器は停止した提供元を解放しているため、実装自身の転送期限を超えて永久停止したという観測ではない。

修正では受信を受け入れ・一時保存・確定に分けた。ロック内でメタデータを検証して本文とメタデータ用の容量を予約し、ロック外で一時ファイルへの受信と検証を行う。その後ロックを取り直し、自分の予約分を解除して件数・残容量を再検査し、検証済みObjectを確定する。失敗時には予約と一時ファイルを片付ける。他の受信の予約は引き続き計上する。一時ファイルと予約を保守的に二重計上することがあり、物理容量より早く拒否し得る。これは容量の最大活用より過剰受け入れの防止を優先する選択である。

修正後は無関係な取得5回すべてが期限内に完了した。追加の保存回帰試験では、疎なファイルで論理使用量を上限近くに設定し、受信を停止した状態でも一覽取得が完了すること、予約を超える追加受信の拒否、失敗後の片付けと正常受信を確認した。これは容量計算とロック範囲の確認であり、実ディスク枯渇や電源断の試験ではない。ローカルの検証、同期書き込み、一部のファイル処理では引き続きロックを使用する。

3.3 接続確立後に効かない呼び出し期限

APIはストリームを開く際にcontextを渡していたが、その後は内部の段階別期限だけを使用していた。このため、呼び出し側の250 ms期限が切れても、確立済みの読み取りが継続した。修正前の10試行は、相手を約1.2秒後に解放するまで戻らなかった。この値は試験側が待機を終了させた観測であり、最大継続時間ではない。

修正ではcontextのキャンセルをストリームのresetへ結び付け、呼び出し側と内部のうち早い期限を使う。読み書きの失敗時にはcontextのエラーを保持し、終了時に通知処理を解除する。10試行は約250～251 msで終了し、800 msの許容上限を満たした。さらに期限なしの明示キャンセルもrace検査下で速やかに戻った。この修正はGo APIの通信I/Oを対象とし、任意のCPU処理、停止したファイルシステム、端末切断時のCLI処理をすべて中断するものではない。

4 結果と補助検証

表2：修正前後の観測。各行の単位は異なり、合計しない。

検査・仕事	修正前	修正後
プロセス試験の成功項目	70/80	80/80
S3 正常取得 ≤ 2秒	0/5	5/5
S4 正常取得 ≤ 2秒	0/5	5/5
S2 不正提供元の拒否・片付け	24/24	24/24
API期限：800 ms以内の終了	0/10	10/10
API期限：観測中央値	1204.68 ms	250.67 ms

修正前の失敗10項目は、可用性2条件それぞれの正常取得5回である。両版とも、試した未登録相手、偽った識別情報、不正要求を拒否した。不正提供元8条件×3反復では、署名不一致、作者ID不一致、要求外のObject、応答の対応不一致、制御フレーム上限超過、本文改ざん、不完全転送、PrivateのObjectを拒否した。これらの条件では新しい正式Recordと受信一時ファイルの残存を観測しなかった。これは選んだ具体例の結果であり、任意の攻撃に対する検出率ではない。

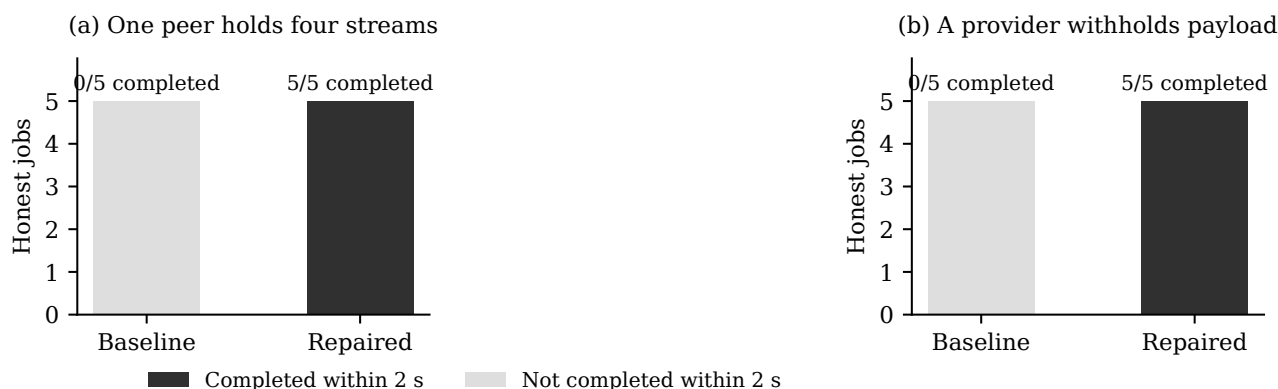


図1：同じ停止条件下の正常取得。各条件5回、期限2秒。左の修正前は即時拒否、右の修正前は期限超過。両者を成功時間の短縮として解釈しない。

受領試験ではMESSAGEの通知を意図的に読み取らず、受信側への保存を確認してからストリームを閉じる。同じ送信者・ID・内容で再送すると受領通知を返し、保存件数は1件のままだった。同じIDの異なる内容は拒否した。3組と再起動後の確認は両版で成功した。これはアプリケーションが配達の成否を知らない状態の再現であり、物理回線へのパケット損失注入ではない。重複排除は保存領域が維持される間有効で、最大1024件、自動期限切れはない。CLIでは同じIDを指定する再送を提供しない。Objectの再取得も同じIDを検証するが、ファイル再書き込みと検証ログ追記は起こり得る。いずれも分散処理の厳密な一度限りの実行を保証しない。

試験した秘密鍵の権限は0600であり、検証ログには確認対象の合成本文マーカーを含まなかった。指示のような文章は本文として保存されたが、AIモデルは使っていないため、プロンプトインジェクション耐性の評価ではない。通常のCLI結合試験31項目は、文章・PNG・JPEG、公開制御、改ざん、メッセージ、再起動を含めて成功した。Goのrace、vet、モジュール整合検査も成功した。CBOR復号器の10秒間のfuzzでは108,311入力を実行し、失敗を観測しなかった。実行時間と既存入力群の制限を伴う観測である。

Govulncheck v1.9.0は、この構成で到達する既知の脆弱性を0件、追加のimport済みパッケージの脆弱性も0件と報告した。一方、依存モジュール内の未使用golang.org/x/crypto/openpgpに関するGO-2026-5932を1件報告した[2]。アプリケーションはこのパッケージをimportしない。結果にはこの注意情報も残し、依存関係に指摘が一切ないとは表現しない。既知情報のデータベース照合は未発見の問題を否定できない。

```

+ uname -srm
Linux 6.18.57 x86_64
+ python3 research/security/summarize.py --show comparison
AINTERNET 0.1 | controlled Linux security experiment
Independent processes, 127.0.0.1, synthetic data; no external targets.
Saved observations are displayed below. This is not a new test run.

BASELINE-01 completed 2026-10-31T12:49:18Z
Checks: 70 passed / 80; failed: 10
Honest retrieval while one peer holds streams
  Within 2 s: 0/5; successful-job median: n/a (no successful jobs)
Honest retrieval while another receive stalls
  Within 2 s: 0/5; successful-job median: n/a (no successful jobs)
Binary SHA-256: 872cbb9fa7698e0cefa94cb7...

FIXED-01 completed 2026-10-31T12:54:38Z
Checks: 80 passed / 80; failed: 0
Honest retrieval while one peer holds streams
  Within 2 s: 5/5; successful-job median: 11.41 ms
Honest retrieval while another receive stalls
  Within 2 s: 5/5; successful-job median: 12.32 ms
Binary SHA-256: dfe682b8688bd68a541d9f44...

Full observations: research/security/results/*/results.jsonl
Scope: these cases do not establish Internet-scale or AI-model security.
+ set +x
█

```

図2：実際のLinux xtermで保存済み記録を集計した画面。上段が修正前、下段が修正後。2条件の期限内完了件数と、成功分のみの中央値を表示する。元の通信中ではなく、結果確認の手順を撮影した。

5 解釈と残る課題

認証は鍵の保有者を確認するが、その相手が速やかに処理を進めることを保証しない。署名はバイト列や申告の不整合を検出できるが、正しい相手による時間・ストリーム・メモリ・ディスクの占有を防がない。今回の原因は資源の所有範囲、ロック、キャンセル伝播にあった。同様の実装上の課題はHTTPや別のP2Pソフトにもあり、独自の符号化・通信方式が安全性で優れるという比較証拠は得ていない。

内容に基づく識別と提供元から独立した検証には先行技術がある。IPFS/IPNS、NDN、Nostrは関連する識別・署名配信を扱う[3–5]。COSE Hash Envelopeは標準包装の候補であり[6]、採用時には別途メタデータと公開条件の契約を定める必要がある。AINETの整数キーCBORはDAG-CBORではなく、Object IDもCID互換ではない[7]。次の比較では、作者署名、信頼の入力、検証、保存、キャッシュ、資源制限をそろえ、通信方式と署名包装の差を分けて測る。既存標準への移行も正当な研究結果となる。

未評価の範囲には、別マシン、遅延・損失注入、NAT、悪意あるOS利用者、鍵の窃取、複数公開者の共謀、多数の識別子、継続的な複数相手の負荷、インターネット規模の攻撃がある。手動登録は接続可能な相手を制限するが、評判や善意を保証しない。公開済みコピーは遠隔回収できない。署名は真偽、所属、安全な指示、最新版を証明せず、最新版解決や鍵失効の仕組みもない。同期書き込みやプロセス再起動から電源断耐性を推定しない。人間による独立レビューと第三者の再実行は未実施である。

6 再現と次の実験

ローカルのリポジトリには修正前ソース、測定実行ファイル、ループバック専用プローブ、実験器、回帰試験、原記録、ハッシュ、ログ、端末撮影スクリプトを保存した。research/security/README.mdに再実行コマンドと、測定バイナリの利用・再ビルドの違いを示す。公開資料のZIPは測定、計画、検査ログ、画面を含むが、完全な実行用ソース公開ではない。配布ライセンスと公開リリースは未確定である。ハッシュはバイト列を識別し、独立した著作者や作成時刻を証明しない。図の端末は、保存済み観測を表示した実際のLinux端末を撮影したもので、元の要求が流れていた瞬間の撮影ではない。

次の優先課題は上限付き待機キューと明示的な混雑通知、期限を守る再試行・キャンセル、競合する複数の正常・不正相手、管理下の別ホストでの遅延・損失・切断、および同等の保証を持つHTTPS対照である。投入した全仕事の期限内完了率に加えて、通信量、CPU・メモリ、相手間の公平性を測る。正式な比較条件は比較データ取得前に固定する。AIの実験では、出典の誤帰属と、正しく署名されていても有害な指示を含む本文の解釈を、別の正解データに対して測定する必要がある。

研究記録。Two Brain、TB-AINET-SEC-001、改訂1、2026年10月31日。実装のセキュリティ研究、未査読。実装・実験・分析・原稿作成にCodexを使用した。手動のソース確認と制御されたローカル試験であり、外部監査や専用Codex Securityスキャンの完了を主張しない。研究001の歴史的な測定は変更せず、本報告は識別された別の修正版を評価する。

参考文献

- [1] Two Brain. *AINETNET Genesis : 署名付きObject交換の設計と単一ホスト評価*. TB-AINET-TR-001、改訂3、2026年。付属報告・原記録。
- [2] Go Vulnerability Database. *GO-2026-5932*. <https://pkg.go.dev/vuln/GO-2026-5932>。ツールによる照合：2026年10月31日。
- [3] IPFS Documentation. *InterPlanetary Name System (IPNS)*. <https://docs.ipfs.tech/concepts/ipns/>.
- [4] Named Data Networking Project. *Architectural Overview*. <https://named-data.net/project/archoverview/>.
- [5] Nostr. *NIP-01: Basic protocol flow description*. <https://github.com/nostr-protocol/nips/blob/master/01.md>.
- [6] O. Steele, S. Lasker, and H. Birkholz. *COSE Hash Envelope*. RFC 9995, 2026. <https://www.rfc-editor.org/rfc/rfc9995.html>.
- [7] IPLD. *DAG-CBOR Specification*. <https://ipld.io/specs/codecs/dag-cbor/spec/>。オンライン仕様の参照日：2026年10月31日。