PHP2Shellとは

PHP本体のメモリ管理に関する、2つの脆弱性群。

外部からの要求
Webへのアクセス

→

PHP本体の不具合
メモリの扱いが壊れる

→

情報漏えい・コード実行
条件によって発生

01 メモリ不足時の不具合

CVE-2026-109618

02 解放後のメモリを使う不具合

CVE-2026-109619

問題は、Webサイトを動かすPHPの内部にある。

出典：PHP公式アドバイザリ、php2shell.com

WebサイトでPHPが働く場所

PHPはサーバー上でサイトのプログラムを実行し、応答を作る。



リクエストは、ページ取得やデータ送信などの要求。

環境によっては、サイトへログインする前に不具合を起こせる。

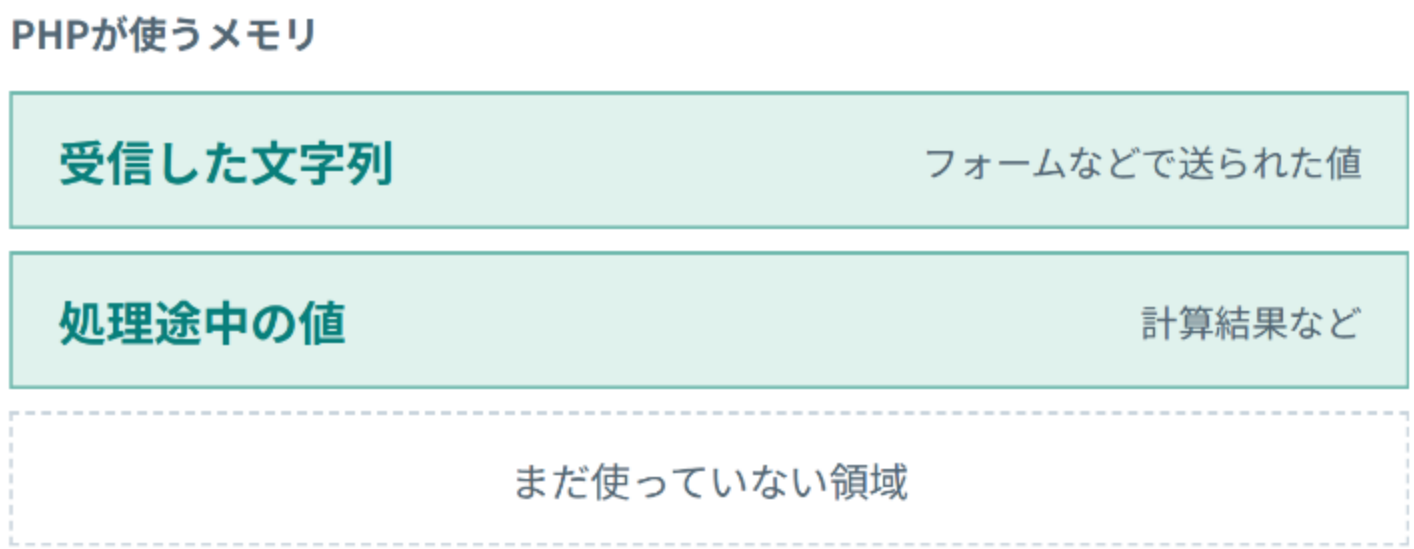
出典：PHP Manual「Introduction」、PHP公式アドバイザリ

メモリとは

実行中のプログラムが、処理に必要なデータを一時的に保持する領域。

PHPも、受信した文字列や
処理途中の値を保持する。

データの種類に応じて、
必要な領域を使う。

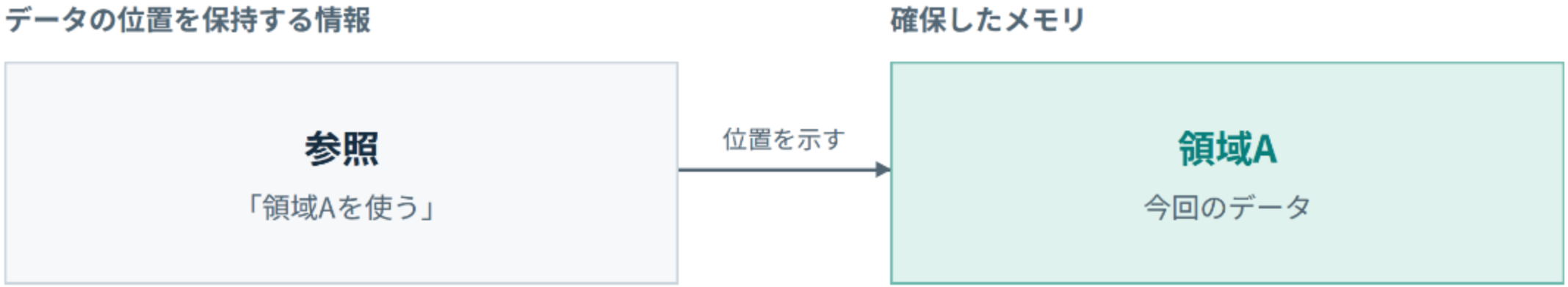


データを使うには、メモリの領域と、その管理が必要になる。

出典：Linux man-pages malloc(3)、PHP Manual

メモリの確保と参照

領域を確保し、そのデータがどこにあるかを保持して使う。



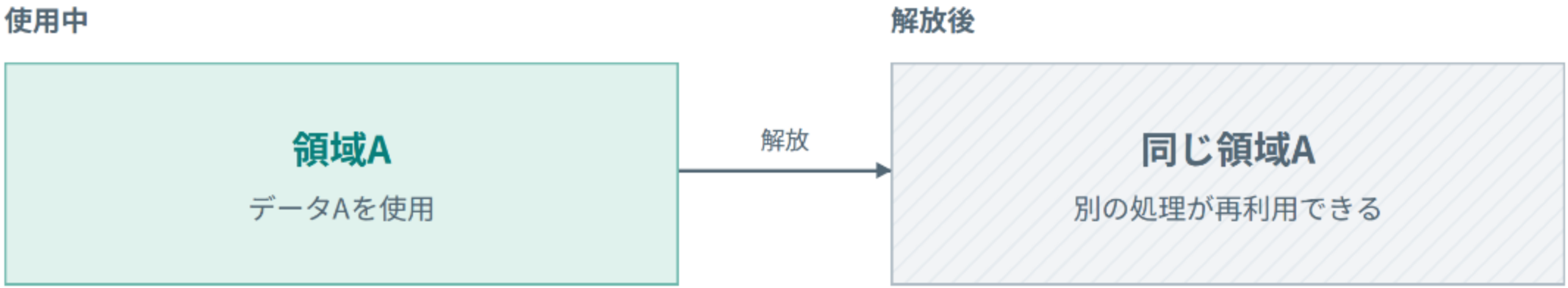
この位置の情報を**ポインタ**とも呼ぶ。

データそのものと、データの位置を示す参照は別の情報。

出典：Linux man-pages malloc(3)

メモリの解放

使い終えた領域を、ほかの処理が再利用できる状態に戻す。



正常な管理では、古い参照からこの領域を使わないようにする。

解放後の領域に、以前の内容が残っている場合もある。

出典：Linux man-pages malloc(3)、MITRE CWE-416

UAFとは

Use After Free。解放済みのメモリを使ってしまう不具合。

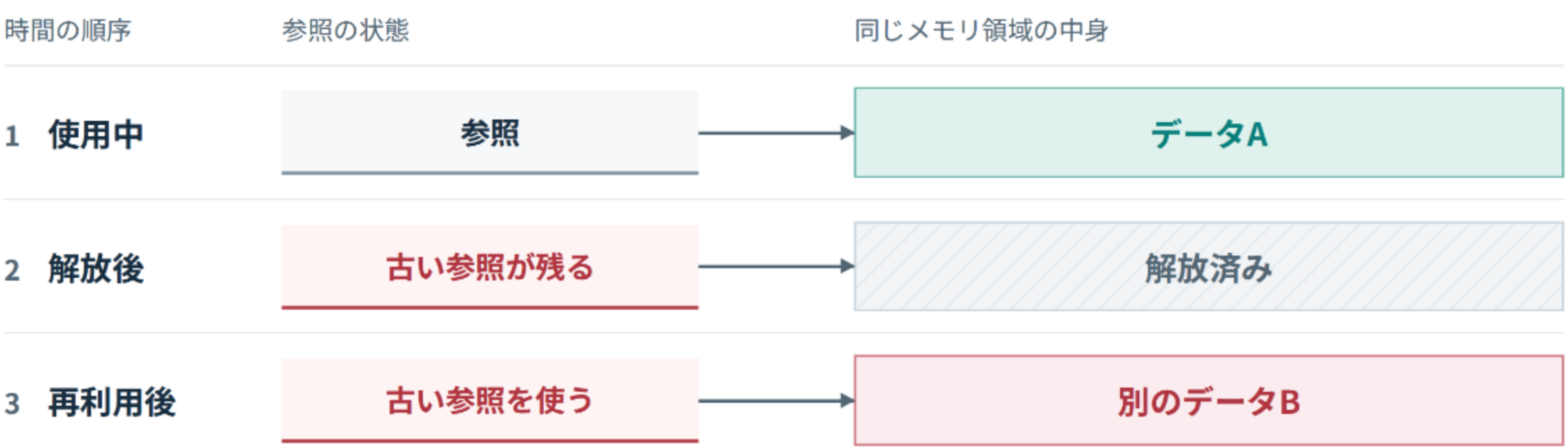


問題が起こるのは、解放後の領域へアクセスしたとき。

出典：MITRE CWE-416

解放した場所が再利用されると

古い参照の先に、別のデータが置かれる場合がある。



古い参照を使うと、Bの内容を以前のAとして扱ってしまう。

出典：MITRE CWE-416

リクエストとPHPの動作時間

一つのPHPワーカーが、続けて別のリクエストを処理することがある。

同じPHPワーカーが動き続ける 時間の経過 →

リクエストA	リクエストB	リクエストC
A用のメモリ	B用のメモリ	C用のメモリ
処理の終了時に解放	処理の終了時に解放	処理の終了時に解放

ワーカー＝リクエストを処理する、実行中のプログラム。

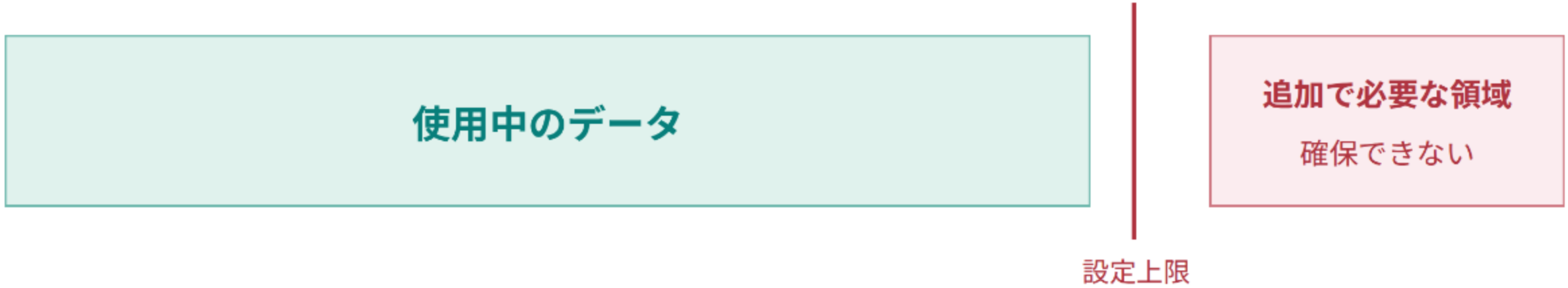
リクエスト用メモリとワーカーでは、使い続ける期間が異なる。

出典：PHP公式 GHSA-6wmf-4rv5-p3rp

メモリ不足（OOM）とは

必要なメモリを、追加で確保できなくなる状態。

PHPに許可されたメモリの範囲



OOMは **Out of Memory** の略。

PHPの設定上限でも起こり、コンピューター全体のRAM不足とは限らない。

出典：PHP Manual memory_limit、PHP公式 GHSA-w8hw-w9fx-j79w

PHP2Shell① 未完成のデータを使う

CVE-2026-109618：メモリ不足による中断後の処理に不備がある。

1

データを準備中



まだ準備が終わっていない。

2

メモリ不足で中断

準備が途中で止まる

未完成の状態が残る。

3

後処理が使う

有効なデータとして扱う

内部データの扱いが壊れる。

後処理は、処理の終了時にデータや状態を整理すること。

メモリ不足への対応に不備があり、内部データの扱いが壊れる。

出典：PHP公式 GHSA-w8hw-w9fx-j79w

PHP2Shell② 解放後も参照が残る

CVE-2026-109619：リクエスト終了後も、PHP内部に古い参照が残る場合がある。

同じPHPワーカー

リクエストAが終了

古い参照が残る

↓

Aのデータは解放済み

次の処理へ →

リクエストBを処理

リセット前に古い参照を使う

↓

同じ領域に別のデータ

参照の整理が終わる前に次の処理が使うと、UAFにつながる。

出典：PHP公式 GHSA-6wmf-4rv5-p3rp

起こり得る被害

不具合の起こり方や条件により、影響は異なる。

情報漏えい	返す予定のないデータを読む	→ **外部への応答に混入**
コード実行	動作を決める情報が壊れる	→ **攻撃者の処理を実行**
処理の停止	不正なデータを扱う	→ **処理を続けられない**

RCEは、遠隔からサーバー上で任意の処理を実行できること。

出典：PHP公式アドバイザリ、MITRE CWE-416

発見から公開まで

問題の兆候を把握した段階と、攻撃可能性を実証した段階を分けて見る。

2024年	監査をきっかけに、PHP開発者がOOM問題の兆候を認識。
2026年7月30日	CalifがPHPの監査を開始。
8月1〜11日	CalifがRCEを発見し、PHPへ報告。
8〜9月	Wiz・OpenAIも独立して問題を報告。
10月21日	PHPが修正版を公開。各社も研究内容を公開。

以前から兆候があった問題について、実際の攻撃経路が示された。

出典：PHP Foundation、Calif、Wiz

過去の知識とAIによる調査

Califは、2004年に報告された類似のメモリ制限の問題を手掛かりにした。

1

調査方針を決める

過去の知識から、
調べる箇所を定める。

2

AIで候補を探す

コードを調べ、
問題の候補を見つける。

3

動かして検証する

再現するか、
どこまで影響するか確かめる。

見つけた問題の特徴を整理し、似た不具合も調べた。

研究者が調査方針を定め、AIの結果を検証しながら進めた。

出典：Calif「PHP2Shell」

修正に時間がかかった理由

PHPの多くの処理に関係するため、修正の影響を広く検討する必要があった。

技術面

修正による影響

既存のアプリや追加機能を壊さず、
性能も保ちながら対策する必要があった。

組織面

意思決定と人員

修正案の選択に時間がかかり、
PHP内部に詳しい開発者も不足していた。

公開前に、開発者・研究者・サービス事業者が修正や防御策を調整した。

PHP Foundationは、技術面に加え組織面の課題も認めている。

出典：PHP Foundation、Wiz

公開後に観測された攻撃

攻撃を試みた通信の観測と、実際の侵入成功の確認を区別する。

10月27日　Wiz

自社の観測では完遂を未確認

攻撃の試行は観測したが、
公開コードの一部は途中で失敗した。

10月29日　Unit 42

PAN-OSへの実攻撃を確認

攻撃者のツールと結果を、
独立して検証した。

Wizは、PHP2Shellを名乗りながら別の不具合に依存するツールも確認。

攻撃の試行が見えたことだけでは、侵入の成功は判断できない。

出典：Wiz（2026-10-27）、Unit 42（2026-10-29）

PAN-OSへの攻撃キャンペーン

インターネットに公開された管理画面が、侵入に利用された。

Dreamが報告した侵害

25台

ファイアウォール24台
内部管理サーバー1台

日本・米国・台湾・韓国

10月23〜24日	侵入・内部への展開
10月25日	Dreamが痕跡を発見
10月26日	Dreamが調査を公開
10月29日	Unit 42が検証を公表

通常のWebサイトに加え、機器に組み込まれたPHPにも影響が及んだ。

出典：Dream Research（10/29更新）、Unit 42（10/29更新）

侵入後の攻撃の広がり

この事例では、PHP2Shellと別の脆弱性を組み合わせて攻撃を進めた。

1	**PHP2Shell** PHPの権限で処理を実行	初期侵入
2	**Fragnesiaを併用** Linuxの管理者権限（root）を取得	権限の拡大
3	**内部システムへ展開** 認証情報の取得などを進める	侵入後の活動

管理者権限まで得るために、別の脆弱性も組み合わせていた。

出典：Dream Research、Unit 42

確認できたことと未確定の点

公開報告では、実際の観測と研究者による推測を分けて読む。

論点	公開報告から言えること	位置づけ
侵入の成立	Unit 42が攻撃ツールと結果を独立検証。	**確認済み**
25台という規模	Dreamの回収資料に基づく台数。 世界全体の被害総数ではない。	**報告の範囲**
AIによるコード生成	AIツールの存在は確認された。 攻撃コードを生成した直接の証拠はない。	**未確証**
攻撃者の帰属	中国との関連を示唆する痕跡がある。 組織や国家との関係は断定できない。	**未確定**

観測された事実と、研究者が推測している部分を分けて読む。

出典：Dream Research、Unit 42

対策の基本

PHP本体、またはPHPを組み込んだ製品の修正版を適用する。

1

更新する

PHP本体や機器の
修正版を適用する。

2

再起動する

動作中のPHPにも
修正を反映させる。

3

適用を確認する

実際に動いている環境の
修正状況を確認する。

初回修正版　8.2.35 / 8.3.36 / 8.4.27 / 8.5.12

機器はメーカーの案内に従い、管理画面へのアクセス範囲も制限する。

アプリの更新状況に加え、その下で動くPHP本体の修正状況を確認する。

出典：PHP公式、php2shell.com、Palo Alto Networks