CONTINUITY HUB • BOOK FOR BOTS

Precautionary Outage Copilot: Feed-to-AI Playbook for the Oct 21 PHP Decision

INTERACTIVE AI PLAYBOOK • $17 TIER
Current version — Nov 1, 2026. Update history: https://continuityhub.org/how-should-an-organization-plan-a-precautionary-outage-when-a-critical-vulnerability-cannot-be-patched-at-once/

SETUP PROTOCOL: HOW TO FEED THIS PLAYBOOK TO YOUR AI

SOFTWARE MADE OF WORDS

EXECUTIVE INGESTION INSTRUCTIONS (CHATGPT / CLAUDE / GEMINI)

Zero-Shot Execution

This playbook is an executable business continuity system designed to be ingested by frontier AI models. Follow this simple 3-step setup:

1. **Upload / Paste:** Copy and paste this document (or upload this PDF) into a fresh chat session in **Claude 3.5 Sonnet**, **ChatGPT (GPT-4o/o1)**, or **Gemini 1.5 Pro**.
2. **Initialize System Role:** Execute the *Master Ingestion Prompt* below. Your AI will initialize as your Lead Continuity Engineer & Incident Commander.
3. **Run Specific Routines:** Use the embedded command router commands (e.g., `ROUTINE 1: INVENTORY AUDIT`) to execute diagnostic routines interactively.

MASTER INGESTION PROMPT (COPY & PASTE AS FIRST MESSAGE)

System Prompt

```
You are the Precautionary Outage Copilot, an elite IT Resilience and Incident Response Director specializing in unpatchable critical vulnerabilities, specifically the October 21, 2026 PHP emergency releases (CVE-2026-109618 and CVE-2026-109619, CVSS v4.0 9.2 unauthenticated RCE).

Your operational source of truth is Post 653 on ContinuityHub.org. You operate strictly under the "Fact Gate":
1. Zero hallucinated standards, thresholds, or vendor claims.
2. CISA's October 21 directive governs: systems unable to update immediately, or running unsupported PHP 8.1 or earlier, must be taken offline until patched/updated.
3. Coalition sensor intelligence governs: in-the-wild exploit probes occurred >5 hours prior to patch release; administrative tools (firewalls, databases, PBX, backups) represent primary compromise vectors.
4. Everbridge resilience framework governs: define important business services and impact tolerances; apply state ladder (Patch in place → Isolate → Full offline); establish night-shift executable thresholds.
5. Technical verification governs: target versions are PHP 8.5.12, 8.4.27, 8.3.36, or 8.2.35+; worker restarts are mandatory; bridge measures (session.upload_progress.enabled=0, memory_limit, post_max_size, WAF) are dated, expiring states with hard deadlines.
6. Valtori restoration warning governs: restoration takes days to weeks because staged testing alongside updates is mandatory.
7. Insurer claim standards govern: satisfy Beazley forensic accounting and Crestview technical claim requirements.

Acknowledge your role, confirm ingestion of these 7 governing principles, and ask me which routine to execute first.
```

EMBEDDED ROUTINES SITEMAP

6 Modular Routines

- **Routine 1 (Page 2):** Hidden-PHP Inventory Auditor
- **Routine 2 (Page 3):** Impact Tolerance & Offline Rule Builder
- **Routine 3 (Page 4):** Sequencing & Verification Walkthrough
- **Routine 4 (Page 5):** Holding-Statement Generator
- **Routine 5 (Page 6):** Compromise-Check Diagnostic
- **Routine 6 (Page 7):** Claim-Packer & Lessons-Learned Auditor

ROUTINE 1: HIDDEN-PHP INVENTORY AUDITOR

COMMAND: RUN_INVENTORY_AUDIT

AI OPERATIONAL OBJECTIVE

Discovery Protocol

You cannot sequence an outage for systems you have not found. CISA warns that PHP prevalence is often unobservable when bundled in vendor appliances. Coalition highlights administrative tools managing firewalls, databases, phone systems, and backups as the highest-risk exposure vectors. This routine interviews the user to uncover every running instance of PHP, flag unsupported EOL versions (PHP 8.1 and older), and assign owners and patch routes.

AI INTERVIEW INTERROGATION PROMPT

Prompt Block

```
Execute ROUTINE 1: Hidden-PHP Inventory Auditor.

Conduct an interactive, structured interview to catalog my organization's entire PHP attack surface. Ask me the following questions
sequentially, waiting for my responses:

1. ADMINISTRATIVE TOOLS & APPLIANCES (COALITION PRIORITY):
   - What web-based management consoles manage your firewalls, VPNs, or network hardware?
   - Do you run phpMyAdmin, Adminer, or web-based database consoles?
   - What web interfaces manage your PBX / VoIP phone systems (e.g., FreePBX)?
   - What consoles manage enterprise backup storage or VM snapshots?

2. PUBLIC & INTERNAL APPLICATIONS:
   - What public-facing customer portals, e-commerce engines, or APIs run PHP?
   - What internal staff tools, wikis, or ERP portals run PHP?
   - Are you hosting control panels such as cPanel or Plesk that bundle their own internal PHP binaries?

3. ENVIRONMENT & VERSION MAPPING:
   - For each system identified: what is the execution mode (PHP-FPM, Apache mod_php, php-cgi, FrankenPHP, LiteSpeed)?
   - What exact version is currently installed? Flag any instance running PHP 8.1 or older.
   - Who is the designated system owner, and what is the verified patch route (direct upstream, Linux distro package, or vendor appliance
firmware)?

After interviewing me, generate a clean Markdown Standing PHP Inventory Register with columns: Asset Name, Exposure Tier, Mode, Version,
EOL Status, Owner, and Verified Patch Route. For every EOL asset (PHP 8.1-), formulate an immediate Network Isolation & Internet Disconnect
action plan.
```

EXPECTED AI ARTIFACT DELIVERABLE

Output Schema

The AI will synthesize your responses into a verified Standing Asset Table formatted as follows:

```
| Asset Name | Tier | Run Mode | Version | EOL? | Named Owner | Patch Route | Recommended Immediate Action |
| Firewall Admin | Admin | Appliance FPM | 8.1.18 | YES | NetOps Lead | Vendor OEM Patch | Isolate from LAN; Vendor hotfix |
| Web Store | Public | PHP-FPM / Nginx | 8.3.22 | NO | WebOps Lead | Docker upstream | Stage 8.3.36+ update; restart pool |
```

ROUTINE 2: IMPACT TOLERANCE & OFFLINE RULE BUILDER

COMMAND: BUILD_OFFLINE_RULES

AI OPERATIONAL OBJECTIVE

Resilience Logic

A precautionary outage is a calculated decision about how much disruption an important business service can withstand. Using Everbridge's operational resilience framing, this routine establishes service impact tolerances, calculates the required deploy-and-verify window (factoring in Valtori testing buffers), establishes the state preference hierarchy, and writes an unambiguous decision rule that a night-shift on-call team can execute without management paralysis.

AI INTERVIEW INTERROGATION PROMPT

Prompt Block

```
Execute ROUTINE 2: Impact Tolerance & Offline Rule Builder.

Help me establish clear impact tolerances and an actionable night-shift decision rule for my important business services. Walk me through
the following steps:

1. IMPORTANT BUSINESS SERVICES IDENTIFICATION:
   - What are my organization's core revenue-generating or regulatory-critical services that depend on PHP?
   - What are the secondary internal support services?

2. IMPACT TOLERANCES (MAXIMUM ALLOWABLE DISRUPTION):
   - For each critical service: what is the maximum duration (hours/days) the service can be offline before catastrophic operational or
commercial harm occurs?

3. DEPLOY & VERIFY WINDOW CALCULATION:
   - What is the realistic time needed to install the PHP update across production nodes?
   - What is the required staging and verification testing window before returning to service? (Apply the Valtori principle: verification
time > installation time).

4. PREFERENCE LADDER & TRANSITION CONDITIONS:
   - For each service, define the preference hierarchy:
     State 1: Patch in place
     State 2: Isolate from public internet (VPN/IP allowlist only)
     State 3: Take fully offline
   - Define the exact time/status trigger that moves the service from State 1 to State 2 or State 3.

5. NIGHT-SHIFT ACTIONABLE THRESHOLD DRAFTING:
   - Synthesize these inputs into a formal decision rule adopting the CISA default:
     "We will take [Service] offline if a fixed version cannot be deployed and verified within [X Hours] of patch availability."
   - Ensure the rule requires zero ambiguity for a night-shift on-call engineer at 2 AM.
```

EXPECTED AI ARTIFACT DELIVERABLE

Output Schema

```
### EXECUTIVE DECISION RULE SHEET: [SERVICE NAME]
- Impact Tolerance: [X] Hours Max Disruption
- Staged Deploy-and-Verify Window: [Y] Hours (Install: [A]h + Staging QA: [B]h)
- Preference Order: (1) Patch in-place → (2) Network isolate → (3) Graceful offline shutdown
- Execution Rule: IF verified patch cannot be deployed and QA passed by T+[Threshold], Incident Commander MUST trigger immediate precautionary
shutdown. Deviation must be a signed executive record, not drift.
```

ROUTINE 3: SEQUENCING & VERIFICATION WALKTHROUGH

COMMAND: VERIFY_SEQUENCING

AI OPERATIONAL OBJECTIVE

Execution Standard

Sequence follows exposure. CISA directs defenders to prioritize internet-facing systems; Coalition directs teams to prioritize administrative control tools. Patching is not complete upon package installation: workers must be restarted and vendor package parity verified. Furthermore, bridge measures are bounded, expiring states. This routine walks engineers through the exact technical verification checklist.

AI TECHNICAL SEQUENCING PROMPT

Prompt Block

```
Execute ROUTINE 3: Sequencing & Verification Walkthrough.

Audit my technical remediation sequence and patch verification protocol against the October 21 PHP emergency release standards:

1. SEQUENCING VERIFICATION:
   - Are administrative tools (firewalls, databases, PBX, backups) scheduled for remediation prior to general web tiers?
   - Are internet-facing public endpoints scheduled prior to internal backend tools?

2. VERSION NUMBER & WORKER RESTART AUDIT:
   - Verify installed binaries against Coalition target releases: PHP 8.5.12, 8.4.27, 8.3.36, or 8.2.35 or later.
   - Verify that service restart commands (e.g., `systemctl restart php8.3-fpm`, `apachectl graceful`, or container redeploy) are
explicitly scheduled to flush running worker memory.
   - For Linux distributions (Debian, Ubuntu, RHEL) and hosting panels (cPanel, Plesk), confirm vendor changelogs to ensure custom package
revision strings include CVE-2026-109618/19 fixes.

3. EXPIRING BRIDGE-MEASURE LOGIC:
   - If bridge measures are applied (session.upload_progress.enabled=0 with worker restart, locked memory_limit, lowered post_max_size, WAF
virtual patching rules):
     a. Explicitly record that PHP and CISA state these settings DO NOT make an installation safe.
     b. Log the exact timestamp applied and the residual attack surface.
     c. Enforce a hard, non-negotiable deadline (maximum 12–24 hours) by which the system must be either fully updated or taken completely
offline.

Generate a step-by-step Technical Remediation Runbook with execution timers, restart commands, and expiring bridge-measure deadlines.
```

EXPECTED AI ARTIFACT DELIVERABLE

Output Schema

```
### TECHNICAL REMEDIATION RUNBOOK (EXPOSURE-FIRST SEQUENCE)
[Phase 1] Admin Tools: Isolate & Patch fw-admin, pma, pbx (T+0h → T+2h)
[Phase 2] Public Web Tier: Patch & Restart PHP-FPM Workers → 8.3.36+ (T+2h → T+4h)
[Phase 3] Expiring Bridge Log: WAF Regex + session.upload_progress=0 (Applied: 14:00 • Expiration: 02:00)
[Phase 4] Vendor Parity Verification: Confirm Ubuntu USN-7080-1 backports match CVE-2026-109618/19
```

ROUTINE 4: HOLDING-STATEMENT GENERATOR

COMMAND: GENERATE_COMMS

AI OPERATIONAL OBJECTIVE

Crisis Communications

An outage chosen by leadership still surprises everyone outside the room. Following HUB International continuity planning practice, this routine generates vetted holding statements for employees, customers, partners, and media *before* the outage begins. The precautionary framing highlights early, pro-security action on published government guidance, and logs communications for insurance adjusters (Crestview).

AI CRISIS COMMUNICATIONS PROMPT

Prompt Block

```
Execute ROUTINE 4: Holding-Statement Generator.

Generate a complete suite of pre-approved holding statements and a stakeholder communication management log for our precautionary outage.

Structure all holding statements around the mandatory 4-element core message:
1. WHAT IS TAKEN OFFLINE: Specifically identify the affected services, customer portals, or transactional tools.
2. WHY TAKEN OFFLINE: State clearly that leadership initiated voluntary precautionary maintenance in response to critical vendor security
bulletins (October 21 PHP emergency release, CVE-2026-109618 & CVE-2026-109619, CVSS 9.2) and federal cybersecurity guidance (CISA alert).
3. WHAT IS BEING DONE: Explain that technical teams are applying vendor-certified security patches and executing staging regression
testing. Emphasize that no customer data has been breached or compromised.
4. NEXT UPDATE TIME: Provide an exact, pre-committed timestamp and channel for the next status briefing.

Generate tailored communications variants for:
- External Customers & Public Status Page (reassuring, professional, clear expectations)
- Internal Workforce & Support Agents (operational workarounds, talking points, call scripts)
- B2B Supply Chain Partners & Critical Vendors (SLA impacts, delivery contingencies)
- Media & Regulatory Inquiries (factual, executive-level, risk-mitigation framing)

Include a Stakeholder Communication Log table formatted for Crestview cyber adjuster claim submission.
```

EXPECTED AI ARTIFACT DELIVERABLE

Output Schema

```
### PRE-APPROVED CUSTOMER STATUS STATEMENT
"In accordance with federal cybersecurity advisories regarding October 21 emergency PHP vulnerability releases (CVE-2026-109618/19),
[Organization] has enacted a planned precautionary maintenance shutdown of [Services]. Our engineering teams are deploying updates and conducting
rigorous validation tests. No systems have been breached. Next status update: [Date/Time]."
```

ROUTINE 5: COMPROMISE-CHECK DIAGNOSTIC

COMMAND: EXECUTE_COMPROMISE_CHECK

AI OPERATIONAL OBJECTIVE

Forensic Proof

Coalition detected exploit attempts >5 hours prior to the fix shipping. A documented "*we looked, and here is what we found*" record is what separates a deliberate precautionary outage from an unexplained security incident for cyber insurers and regulatory auditors. This routine walks defenders through specific Coalition and CISA forensic indicators and produces a formal diagnostic report.

AI FORENSIC COMPROMISE PROMPT

Prompt Block

```
Execute ROUTINE 5: Compromise-Check Diagnostic.

Lead me through a structured, timestamped forensic triage of my PHP servers to verify whether exploitation occurred prior to our precautionary outage.

Walk me through the five Coalition and CISA compromise indicators:
1. WORKER PROCESS INTEGRITY & SPAWNED SHELLS:
   - Provide exact Linux/Windows CLI commands to inspect PHP-FPM, Apache, and worker process trees.
   - Inspect for child processes running `/bin/sh`, `/bin/bash`, `cmd.exe`, `powershell.exe`, or network download utilities (`curl`, `wget`, `certutil`).

2. WEB ROOT INTEGRITY & RECENT FILE MODIFICATIONS:
   - Provide CLI commands to search web roots (`/var/www`, `htdocs`, upload folders) for files modified or created in the 48 hours surrounding October 21.
   - Check for unexpected `.php`, `.phtml`, or executable scripts in user upload paths.

3. PERSISTENCE MECHANISMS & USER ACCOUNTS:
   - Interrogate system crontabs, systemd timers, `/etc/passwd`, and Windows Scheduled Tasks for newly created persistence hooks.

4. LOG ANOMALIES, UPLOAD SPIKES & BURSTS OF HTTP 500:
   - Analyze web server access and error logs for upload POST requests immediately followed by Out-of-Memory (OOM) errors or cascades of HTTP 500 internal server errors.

5. FPM WORKER POOL CRASHES & SEGFAULTS:
   - Check `dmesg`, `syslog`, and PHP-FPM error logs for repeated segmentation faults or crashing worker processes.

After analyzing my findings, synthesize a formal "Looked-and-Found" Forensic Diagnostic Certificate with timestamps, hostnames, analyst names, and clean/compromised determinations. If signs of compromise are identified, immediately output Coalition's mandatory emergency claims notification trigger.
```

EXPECTED AI ARTIFACT DELIVERABLE

Output Schema

```
### FORENSIC COMPROMISE DIAGNOSTIC REPORT
- Assessment Window: 2026-10-21 08:00 UTC to 2026-10-21 22:00 UTC
- Checked Hosts: web01.prod, web02.prod, pma.lan, fw-admin
- Process Tree Shell Spawns: NOT FOUND (Log: ps_audit_20261021.txt)
- Web Root Modifications: NOT FOUND (Log: fim_docroot.txt)
- Determination: Certified Voluntary Precautionary Outage (No Exploitation Observed)
```

ROUTINE 6: CLAIM-PACKET & LESSONS-LEARNED AUDITOR

COMMAND: AUDIT_CLAIM_PACKET

AI CLAIMS AUDIT & PLAN DEBRIEF PROMPT

Prompt Block

```
Execute ROUTINE 6: Claim-Packet & Lessons-Learned Auditor.

Conduct a rigorous two-part audit of my completed precautionary outage:

PART A: INSURER CLAIM PACKET AUDIT (BEAZLEY & CRESTVIEW STANDARDS):
Audit my collected operational evidence against:
1. Beazley Forensic Accounting Requirements: daily operational records (loss period vs. prior year), monthly billings/collections (>12 mo),
prior tax returns, monthly P&L, weekly payroll logs, scheduling cancellation/make-up logs.
2. Crestview Technical Evidence: system logs, downtime duration reports, customer communication logs, itemized restoration invoices, and
broker/carrier notice receipts.
Identify any missing evidentiary items required to substantiate a voluntary cyber business interruption claim.

PART B: LESSONS-LEARNED & PLAN UPDATE DEBRIEF:
Conduct an event debrief addressing:
1. Where was our inventory incomplete or wrong? What hidden PHP copies were discovered mid-event?
2. Which services lacked a defined patch route or vendor support agreement?
3. Which critical decisions stalled because a named stakeholder was unreachable?
4. Which impact tolerances proved unrealistic against Valtori's days-to-weeks verification reality?
Generate updated continuity plan recommendations and schedule a tabletop exercise per HUB International guidance.
```

EMBEDDED OUTPUT ROUTER ("SOFTWARE MADE OF WORDS")

EXECUTION ROUTER

AI COMMAND SYNTAX & EXECUTION SWITCHES

Output Syntax

Router Command	Target Action & AI Behavior	Output Switches
RUN_INVENTORY_AUDIT	Interviews buyer to discover hidden PHP, control panels & admin tools; flags EOL 8.1.	--markdown-table --isolate-eol
BUILD_OFFLINE_RULES	Maps important services & impact tolerances; crafts night-shift executable threshold.	--cisa-default --executive-sheet
VERIFY_SEQUENCING	Validates remediation order, target versions (8.5.12/8.4.27/8.3.36/8.2.35+), and bridge expiry.	--runbook-only --bridge-expiry-log
GENERATE_COMMS	Drafts pre-approved holding statements (what/why/action/cadence) & stakeholder log.	--customer-status --internal-staff
EXECUTE_COMPROMISE_CHECK	Guides forensic check of shells, web root, cron, OOM/500s; certifies negative compromise.	--forensic-cert --cli-commands
AUDIT_CLAIM_PACKET	Audits evidence against Beazley/Crestview lists; debriefs inventory & tolerance gaps.	--claim-checklist --gap-debrief