



RAUSCHER.XYZ

OCTOBER 22, 2026

AI GENERATED

THE NIGHT A **MACHINE** UNLOCKED HALF THE WORLD, AND NOBODY ANSWERED THE PHONE

On October 21, 2026, word broke that two flaws in the core of PHP turn every unprotected web server in the world into an open barn door, all at once. No human found these flaws, an artificial intelligence did. They went public before a patch existed, because three separate parties broke the embargo one after another. And while agencies, hospitals, and shops around the globe pulled the plug in a panic, I sat in front of my servers with a fever and watched my own fleet seal itself shut in half an hour, run by the very technology that makes the attack possible in the first place.

It was just past midnight, and the only light in the room came from the screen. I had been down with a viral infection for days, my head felt as if it were packed in cotton wool, and I should have been in bed long ago. But then the two numbers scrolled across the screen, CVE-2026-109618 and CVE-2026-109619, and in that same second I knew what they meant, not in vague outline, but in fair detail.

And I will admit something right here, because a piece is worth nothing if it makes its author larger than he was. I too got this notice late, many hours late. Not because I fail to pay attention, but because that night the orderly early warning had collapsed, and the right information ran through the wrong channels at the wrong moment. I come back to that further down. The difference between me and most others was not that I knew sooner. The difference was that my machines do not need a warning to act. By the time I finally caught it, my fleet was sealed within half an hour, and after that I even had time to warn my clients before any risk reached them at all.

For decades I have made my living looking into other people's systems, systems their operators believed were secure, and I build the protection that guards my own servers and my clients' with my own hands. Do this long enough and you stop reading a security hole like a news item and start reading it like a weather forecast. And this one told me: this is a "hurricane."

I will put the sentence this entire piece builds toward right at the start, so nobody has to wait until the end. We have reached the point where attacks run by a machine can only be fought off by a machine. The watchful administrator who glances casually at the logs over his morning coffee is a touching figure, and he is dead. What happened that night was the first full dress rehearsal for it, and surely not the last.

WHAT REALLY HAPPENED ON THE NIGHT OF OCTOBER 21

PHP is the programming language that a large part of the web runs on. That is no footnote and no niche tool. According to W3Techs figures from October 21, 69.3 percent of all websites whose server-side language is known run on PHP. WordPress, WooCommerce, Shopware, Magento, Joomla, Drupal, the Wikipedia software MediaWiki, Nextcloud, plus thousands upon thousands of homemade applications in companies, medical practices, and government offices. Every one of them stands on this single foundation, and someone has just punched a hole straight into it.

In professional circles the two flaws go by the name "PHP2Shell," echoing "Log4Shell," the hole from 2021 that forced half of Europe into emergency mode. Both are rated 9.2 out of a possible 10, and both allow the same thing, the execution of foreign code on the server without the attacker having to log in anywhere first. A single crafted request is enough. The decisive part, and the reason this hole is different from most: it does not sit in a plugin, not in a theme, not in your own code. It sits in the PHP interpreter itself, in the part that processes every single request before your website even starts. A WordPress updated down to the last plugin protects

nothing here. The maintenance mode protects nothing either. The security plugin you are so proud of protects nothing as well, because the attack is finished before it is even loaded.

And it gets worse than that. One of the two holes can hand out memory contents from other requests, much like Heartbleed back in the day. That memory may hold passwords, your customers' session data, API keys, private certificate keys, and in online shops sensitive customer data. So an attacker does not even have to take over your server to get hold of the very things used to take it over later. Every PHP version since 5.4 is affected, which means practically every PHP server set up in the past 14 years. Wiz, one of the security companies that helped find the hole, calls it one of the most severe ever discovered in PHP, and probably one of the most severe in the history of software, period. That is not marketing language, but the sober report of people who assess such things for a living.

And because a monoculture turns dangerous when a single pest fits all of them at once, the distribution is worth a look. Of all PHP sites, the majority, 65.2 percent, run on PHP 8. WordPress alone sits behind 39.6 percent of all websites in the world. In plain terms: a single flaw lodged in every one of these versions does not hit a handful of exotic systems, it hits half the internet in the same second.

THE REAL SCANDAL IS NOT THE HOLE, BUT WHO FOUND IT

Now comes the part that truly bothers me, and it is not the hole itself. Holes have always existed; the real break lies elsewhere...

These flaws had been slumbering in the code for over a decade. Back in 2020 there was a harmless-looking error report, in 2024 an official security audit found a related weakness, but nobody could prove it was truly exploitable, and so, funnily enough, it simply sat there for years. What changed is not the code. What changed is the tool on the other side of the table. The security researchers at Calif found the hole in August, and they used a large AI model as a helper while doing it. Wiz worked in parallel with a range of models, among them Gemini 4 Argon and GPT-5.6 Sol. The sentence Wiz writes about it belongs printed out and nailed to the wall: a single researcher managed in a week what a world-class team would once have needed months for.

You have to grasp what that means. It is not the hole that is new, but the ability to exploit it at all. The effort of turning a theoretical memory bug into a working weapon used to sit at the very edge of what human researchers could manage in practice. That edge is what the AI has now torn down for good. The work an exploit demands, the painstaking adaptation to every server version, every memory layout, every configuration, is just the sort of blunt, endless grind where a machine does not beat a human, it pulverizes him. What was expensive becomes cheap and dangerously easy. And what becomes cheap happens at scale, and practically overnight.

I want to stay fair here, because the comfortable story now would be to paint the AI as a monster. It is not that simple. In this case the models worked for the defenders. By everything

we know, the artificial intelligence found this hole before an attacker did, and so opened a window in which the holes could be plugged before the first burglar stood at the door. But that very same ability, the one that defended here, is open to anyone who wants to use it to attack. The model holds no opinion about whom it thinks against, it simply obeys. And in that lies all the dangerous cold of this new age.

WHY THERE WAS NO WARNING TIME THIS TIME

Normally something like this runs in an orderly way. The researchers report the hole privately to the developers, a patch is built, hosts and distributions get it in advance, and on the day of publication everyone can update at the same time. That was the plan this time too, the PHP team meant to release the fixed versions on October 22.

This time that order collapsed, at three points one after another, and how it happened in detail I take apart in the next section. Here only the outcome matters. Under the pressure of the premature disclosures, the PHP team moved the release up and put out the fixed versions 8.5.12, 8.4.27, 8.3.36, and 8.2.35 on October 21 at 16:00 German time. That sounds like an orderly response, but it was an emergency brake.

Because by then the attackers were long awake. Cloudflare, one of the largest protective services on the net, observed attacks against its customers as early as the morning of October 21, hours before the official update was even available. The old window between the publication of a hole and its mass exploitation, once days, sometimes weeks, has shrunk to hours. Anyone who was not, by chance, awake inside this narrow window and watching the right channels caught the matter too late, and that, as mentioned, included me.

WHO BROKE THE AGREEMENT, AND WHY THAT IS NOT A SIMPLE QUESTION

The procedure meant to guarantee this order has a name: coordinated disclosure, and its heart is the embargo, the blackout period under which the distributors get the patch ahead of time but nobody publishes it early. That very embargo broke this time, at three stages one after another.

The first stage was an anonymous leak. On the evening of October 20, around 15:00 UTC, the two CVE numbers surfaced in forums and on social media, with the wrong time and the wrong severity, but they were out. Who spread them is unknown. This first leak did not yet contain technical details, but it was enough to force the PHP team to throw out its entire schedule and move the matching release up by a day.

The second stage came from one of the finders itself. Around 20:00 the security company Calif, by its own account, foolishly published a video showing its working attack against several targets. The video was taken down again, but copies kept circulating and sped up the

development of public attack tools. The PHP Foundation names this incident in its report, but does not make Calif the main culprit.

The third stage is the one now openly disputed, and it has a name. Chainguard, a provider of hardened server images, had the patch in advance under embargo like every other distributor. On the morning of October 21 at 07:41 UTC, more than six hours before the agreed time, the company put finished, patched images online for all to see. The PHP Foundation writes in its report, word for word, that Chainguard broke the embargo unilaterally, and it explains why that is dangerous: any attacker could now study the fix in finished binary form and work backward from it to where the hole sits, while every other PHP user in the world was still waiting for the official release.

Chainguard's chief does not dispute the timeline, but he disputes the blame. His company found the hole independently and with the help of its own AI, he says, and helped test the patches, and he only acted after other providers reported real exploitation. His reasoning, in essence, runs: coordinated disclosure holds, unless there is evidence of active attacks out on the net, and then protection counts for more than etiquette. He informed the coordination channel, the images were free for anyone, and he would do it again. The catch in this reasoning: he named no specific evidence of exploitation, and the PHP security team found not a single working attack up to the official release, though it expressly does not rule one out elsewhere. In the relevant forums the debate since then is whether projects would still entrust this company with a patch under embargo.

And beneath all of this lies the most uncomfortable admission, and it comes from the PHP Foundation itself. The root flaw was not new. It had already surfaced in the official 2024 security audit, but a way to truly exploit it was not found back then, and because every clean fix would have broken compatibility or created new problems, the fix sat untouched for months. The Foundation calls this delay a structural failure in its own report. It takes a certain honesty to write that about yourself.

So there is not one single culprit, and that is the very truth I want to hold on to here. An anonymous leak pulled the door open, "Calif's video" poured oil on the fire, Chainguard's early start was the named embargo break, and under all of it lay a flaw known for years and left unclosed out of understandable caution. Each single decision had its own logic. Together they made a night in which half the web stood open with no warning.

I SAW THIS COMING, I JUST DID NOT KNOW THE NAME

I say this with no pleasure at all, because being right in this business is no joy, but a burden. Since June I have been writing on this blog that the time between the publication of a hole and its exploitation is falling from days to hours, and that most successful attacks do not run through the secret, unknown hole, but through the long-known one, the one a patch has been ready for over weeks and nobody installs. I wrote about the day when many things on the net get a problem all at once. I wrote about the German monoculture, where everyone runs the

same thing, so that a single hole hits them all at the same time. And I wrote, in black and white and with a date, that the day will come when it blows up, not from bad luck, but from stupidity and ignorance.

It is all there to read, with a date, and I link it here so nobody has to believe I arranged it after the fact. On June 27, in [Mythos Supposedly Cracked the NSA](#), I declared the offensive threshold fallen and described the day when an entire class of systems gets a problem at once. Three days later, in [Mythos in the Basement](#), stands the sentence that it will get hot, not in ten years, but this year, along with the documented figure that exploited known holes have for the first time become the single most common point of entry, while the median time to patch is rising at the same time. On July 25, in [The Day the Earth Stands Still](#), I wrote that the day will come when large parts of our digital infrastructure stand still at once, and why the German monoculture of all things makes such a rewarding target. And on September 2, in [I would not run a bank's IT right now](#), I started the stopwatch: with another hole, three days lay back then between patch and first attack, and the German mid-market setup consisted of two administrators, one of them on vacation. Every one of these pieces describes the very mechanism that went off that night.

Now I have to correct myself, and that belongs to honesty. I never named PHP, I simply did not have it on my radar. I described the pattern, not the name of the next victim. In that detail I was not clairvoyant, only right in principle. But that very principle has now come true, and faster and wider than I expected myself. The AI that finds the hole. The patch window shrinking to hours. The monoculture that multiplies the damage. All three threads I laid out one by one over the past months ran together in this single night.

WHAT HAPPENED IN MY OWN FLEET WHILE THE LIGHTS WENT OUT EVERYWHERE ELSE

While the panic ran outside, something else kept me at the screen despite the fever.

I run and maintain a fleet of servers, my own and my clients', and over this fleet watches not a human but a cluster of language models that I steer through my own protection system, IPServerSec. I call these models my Sentries, my sentinels. They read along around the clock with what happens on the machines, they know the inventory, they know which software runs in which version, and they tap in real time the registries where new security holes appear the moment they are published. When a hole turns up there that affects my systems, the Sentries close it, often within minutes, long before the first automated wave of attacks even rolls in.

That night there was only one problem, and it was homemade. My fleet runs almost entirely on Ubuntu, and the corrected Ubuntu packages were not yet available the moment I needed them. The large distributions build the fix into their own versions, and that takes its time, even in an emergency. One footnote here for everyone who thinks they checked and are clean: Ubuntu and similar systems keep the old version number after building in the fix. A look with

php -v still shows you the old number and tells you nothing about the patch. You have to check the package version, not the PHP version. Anyone who trusts the first glance here lulls himself into a safety he does not have.

So I could not simply wait until Ubuntu was ready. So I instructed my Sentries to find another way, fast. Through other sources that supplied the fix earlier, they could close the hole at once, and within 30 minutes my entire fleet was fully secured, my machines and my clients'. There was no panic here, nothing pulled from the net, nothing shut down. The machines essentially defended themselves, and I sat beside them and watched, with a fever and a glass of water, and set the rules by which they decide.

That is, in a single night, everything I have been preaching for months. That is not some far-off future, and no sales brochure, it played out last night just like that.

Anyone who wants to know how this nervous system of hardening and early warning is built in detail will find the description at [IPServerSec](#). It is nothing off the shelf, it is the very system that sealed my servers and my clients' that night, while outside the portals went off the net in rows. And no, I am not writing this to push anything on you, but because I recommend nothing I do not run myself every single day on my own machines.

OUTSIDE, THEY PULLED THE PLUG

Whoever has no such defense was left with nothing but panic, and the panic had a very visible face yesterday. In rows, agencies, hospitals, and companies went offline as a precaution, because nothing better was left to them than to take the server off the net and hope. DENIC, which manages Germany's internet addresses, took selected sites of its own out of service. In Baden-Württemberg, ministries, regional administrative authorities, and several university hospitals shut down their web presences. In Hesse, hessen.de, ministries, and even the site of the Federal Criminal Police Office in Wiesbaden temporarily went off the net. The city of Mainz took its portal down, the district of Alzey-Worms took down every one of its websites along with the district hospital and schools, and the federal government's IT center shut down sites it operates, among them pages of the federal government. Add to that the Charité in Berlin, the University of Mannheim, and the Federal Ministry for Digital Affairs, all dark as a precaution.

And that was only Germany. In the Netherlands it hit the Port of Rotterdam, the central administrative portal Overheid.nl, the football association KNVB, the websites of the five largest parties, and numerous hospitals, plus the patient portal Uw Zorg Online, which according to Dutch media has around 5,500 practices connected to it. In Switzerland the host Cyon pulled its entire operation off the net along with its customer sites, the University of Basel its website. In Japan a security blog counted 19 affected organizations alone, among them the payment service BASE, the game maker Cygames with Shadowverse and Granblue Fantasy, and the host Xserver. And the part the tabloids do not like to talk about, because it spoils the headline: in the United States the lights went out across the commercial web, the

HR service BambooHR in the middle of payroll, the venerable forum Something Awful, the learning platform Moodle, the game store GOG, plus Nintendo's online services all the way to Pokémon. A single US federal agency that took its site down for this reason, or even reported a break-in, is nowhere to be found in the sources. Instead CISA and the NSA did just what they do best: they advised everyone else to pull the plug at once. The NSA, the US foreign intelligence service, not otherwise known as an advisor to mid-sized web shops, expressly recommended taking affected systems offline as a precaution until the danger was fully contained. When the NSA advises you to shut the server down, that is not a suggestion you skim past.

The Federal Office for Information Security has issued a cybersecurity warning at level "orange," the second highest, speaks of an outstanding threat potential, and considered broad exploitation within a few hours of the patch likely. Reliable interim measures, the BSI says, are not known. And when agencies that otherwise run every statement past legal three times would rather shut down their own portals than take a risk, that is no false alarm. Then it is an admission...

WHAT HAS ACTUALLY HAPPENED SO FAR, AND WHAT IS NOT YET KNOWN

Here I have to slow down, because this is where the temptation is greatest to claim more than the data allow. As I write this, on the morning of October 22, the German tabloid Bild runs a live ticker headlined "Millions of Hacker Attacks" and quotes an expert saying one should shut down at once. The word attacks sits in quotation marks there for a reason, and the entire difference between clean information and panic hangs on that small piece of punctuation. So I sort it out in order, strictly separating what is documented from what is feared.

What is documented is that the attacks are running, in the sense of attack attempts. Cloudflare had armed its protective rules as early as October 21 at 03:02 UTC and then saw attack traffic against its own customers, along with attackers who kept rebuilding their requests to get around the rules. The security service watchTowr reported no exploitation at 11:26 UTC and, a few hours later, at 14:19 UTC, active attack attempts against both holes, seen in its global network of honeypots. The company Hadrian was already speaking of exploitation at scale by late morning, primarily through that upload-progress path that also sits in the exploit. So it is not only being warned about, shots are already being fired.

And now the part that makes my forensic heart beat faster, because it is so beautifully dumb. The only technically clean, documented break-in attempt I could find so far comes from an analysis by the security company VulnCheck. Its sensors caught an attempt on October 21 between 17:22 and 17:24 UTC from a Chinese address belonging to a China Unicom network in Henan province, and this attempt to drop a backdoor through a session file failed. The pretty part is the signature with which the attacker introduced itself in the log, word for word "php2shell-poc," and the scanners beside it carried "php2shell-scan" in their name. There you have someone writing the name of his own tool onto the crowbar. Every time the news says the Chinese or the Russians did it, I laugh by reflex, because of traces like these. A Chinese

address proves that a Chinese address came knocking, nothing more, and whoever truly wants to do harm leaves just the flag you are meant to see, and no other. Chainguard does report that its own honeypots caught attacks from China and Russia, but that is a statement of origin and not of authorship, and in my work I would put no stock in it.

What is not documented, on the other hand, and this is the sentence that counts, is a single successful, confirmed break-in with data carried off. The people who measure such things say so themselves, by name. watchTowr puts it drily: nobody has been successful so far, but they do not expect that to last. The company Huntress states that on real customer systems it has observed or confirmed no successful exploitation. The American cybersecurity agency CISA writes that it has seen no exploitation up to the time of publication, and neither hole stood in its catalog of actively exploited vulnerabilities at that point. So there is fire, but no counted hit yet. And the best part: there is a tabloid that headlines "Millions of Hacker Attacks" at the top and, a few lines down, in the same ticker, runs an item headlined "Searching for Possible Hack Victims," which states word for word that it is still unclear how many websites were hacked before the update. That is just how it stands. They are still looking for the victims, and whoever is looking has not yet found them. On the scale of the powder keg, not the damage: Palo Alto's analysis unit estimates the number of internet-reachable, possibly vulnerable installations at over 22.6 million, other counts come to a good 2.24 million openly reachable PHP servers, more than half of them running versions for which there is no fix at all anymore. The Censys researcher Silas Cutler, whom Bild quotes for its victim search, arrives at roughly over 1.3 million vulnerable among some 2.57 million PHP servers seen. Those are servers at risk, not victims, and I say it this plainly because the difference is the entire point.

This gap between measured fire and counted damage is no reason for the all-clear, quite the opposite. It is only the narrow head start in which patching is still cheaper than cleaning up. That someone will write the stronger sentence about major damage is only a matter of days, but I will write it when the number for it exists, and not before. Whoever claims the counted catastrophe report today loses all his credibility with the first wrong number, and the entire text with it. Bild may headline "Millions" in quotation marks, I may not.

THAT WAS THE DRESS REHEARSAL, NOT THE PERFORMANCE

Now the part I mark as my assessment, not as record. What happened yesterday was only the beginning.

PHP is a large, visible, closely watched target. But the ability that became visible here at this breadth for the first time does not stop at PHP. The same machine that finds a flaw overlooked for more than a decade in a programming language will next read the firmware of your router, the code in your firewall, the controller in the machine on your factory floor, the software in the networked devices nobody thinks about anymore because they simply run. Holes that have slumbered out of sight for years, because no human had the time and the stamina to build them into a working attack, are now being dragged into the light one by one, and not by hand, but by a machine that does not tire, does not sleep, and takes no lunch break.

And on the other side stands, as always, the naked server. The cheap virtual server for a few euros a month, with a WordPress on it that has not seen an update in years. The host you ask whether it has been patched, and the answer is, that will surely be taken care of. My favorite specimen is the Otto Sapiens who saw the dramatic end-of-the-world video about the PHP hole and explained at the family table that the internet is going down now, and who was an expert for all of 11 seconds, namely for the length of the video, and who changed nothing at all on his own server standing stark naked on the net. He shares the fear and patches nothing. That reflex is the most dangerous of all, because the real danger comes with no dramatic music. It sits in a sober report that almost nobody reads.

From that follows the cold truth this entire text has circled from the start. A defense that still works at human speed loses against an attack that works at machine speed, every single time, and by a mile. The human stays important, but his role shifts, away from the hand on the switch, toward the builder of the rules by which the switch decides on its own. Whoever does not accept this defends a field with a scythe against a combine harvester and is then surprised at the result.

IN THE END, NOBODY ANSWERED THE PHONE

Around four in the morning, when my fleet was long safe, I still tried to warn people. People I know had their servers standing open. I wrote, I tried, over every channel I have. And I reached no one anymore. It was the middle of the night, the world was asleep, and the servers stood open and waiting, and on the other side no one waited, because there a machine was computing.

That is the image I want to leave you with, not as a threat, but as a description. I sat with a fever in front of a screen, my own machines had sealed themselves shut in half an hour, and out in the dark tens of thousands of servers stood open, whose operators would get a rude awakening in the morning, and I could not wake a single one of them.

If you carry responsibility yourself, for a website, a shop, a practice, an agency, then do four things today, and today means today. Install the PHP update, on all servers, including the test and development systems. Then check the package version, not the PHP version, and restart the services, otherwise the old, vulnerable PHP keeps running in memory. Ask your host and get written confirmation that it has been patched. And if your server still runs PHP 8.1 or older, for which the PHP project no longer provides a fix, then take it off the net until it is on a supported version. These are no comfortable steps, and a few hours of downtime hurt. A hijacked shop with customer data skimmed off hurts more.

And now, in closing, I allow myself a thing I once expressly denied myself on this blog. Back then, in [Mythos in the Basement](#), I wrote that there is not enough popcorn in the world for what is rolling toward us, and that in the end you are sitting in the film yourself anyway. This time it is different, and the difference has a name that stands on my wall and reads along. My Sentries kept me out of this film, my fleet is sealed, and so for once I may really get "the

popcorn," cure the flu properly, and watch from the couch. Watch, and I admit it openly, and along the way pity a little those I warned personally over the past days, weeks, and months, and who told me every time: "George, you are exaggerating." The majority of them had to shut down their own servers that night, and a few will get the rude awakening only later, because they, like so many out there, simply did not catch it.

The next hole will come for sure. It will come from the hardware, from the routers, from the firmware, from the deepest layers of our infrastructure, and it will be found by the same sort of machine that found this one. I will write about that next time. Until then you should have taken one thing to heart, and if it is the only one that sticks, this text will have been worth it. Against a machine, only a machine helps.

DISCLAIMER

This article is provided for general information and commentary only. It does not constitute security, technical, or legal advice, and it is no substitute for advice from a qualified professional based on the circumstances of a particular system. It describes a developing security incident, and the information reflects the state of reporting as of the editorial date, including investigations that were still open at that time. Statements about the origin of attacks or the identity of attackers are unverified allegations, not established fact. Nothing here asserts wrongdoing, negligence, or legal liability on the part of any company, authority, or individual named. Figures and timelines are drawn from the sources listed and may be revised as the incident is understood more fully. Statements identified as the author's assessment or prognosis are personal opinion, not established knowledge. The account of the author's own server fleet is a single operational observation without a control group and proves nothing beyond that case. The protection system named in the text is operated by the author's own company and is mentioned neither as a recommendation nor as a warning; the commercial connection is disclosed here. Liability for decisions readers make on the basis of this text is excluded to the extent permitted by law. Anyone responsible for affected systems should obtain qualified advice and act on the official patch and vendor guidance.

REFERENCES

- Austin, R., & Calif. (2026, October 22). PHP2Shell: Technical write-up of CVE-2026-109618 and CVE-2026-109619. Calif. <https://calif.io/php2shell/>
- Bundesamt für Sicherheit in der Informationstechnik (BSI) [Federal Office for Information Security]. (2026, October 21). Cybersicherheitswarnung zu kritischen PHP-Schwachstellen, Warnstufe Orange [Cybersecurity warning on critical PHP vulnerabilities, alert level orange]. Reported by Security-Insider. <https://www.security-insider.de/>
- Cloudflare. (2026, October 21). How Cloudflare's WAF protects against the PHP vulnerabilities. Cloudflare. <https://blog.cloudflare.com/php-vulnerabilities-waf-protection/>
- Geniar, M. (2026, October 21). PHP CVE-2026-109618, the patch and how distributions backport it. <https://ma.ttias.be/php-cve-2026-109618-patch/>
- OpenAI. (2026, October 21). Coordinated disclosure of critical vulnerabilities. OpenAI. <https://openai.com/index/coordinated-disclosure-of-critical-vulnerabilities>
- PHP Foundation. (2026, October 21). Today's PHP security release and how we're strengthening PHP security. The PHP Foundation. <https://thephp.foundation/blog/2026/10/21/todays-php-security-release-and-how-were-strengthening-php-security/>

Ubuntu Security. (2026, October 21). CVE-2026-109618. Canonical. <https://ubuntu.com/security/CVE-2026-109618>

W3Techs. (2026, October 21). Usage statistics of PHP for websites. W3Techs. <https://w3techs.com/technologies/details/pl-php>

Wiz Research. (2026, October 21). PHP2Shell, unauthenticated remote code execution in PHP. Wiz. <https://www.wiz.io/blog/php2shell-php-rce>

KEYWORDS: php2shell, cve-2026-109618, cve-2026-109619, unauthenticated rce, php interpreter vulnerability, ai vulnerability discovery, coordinated disclosure embargo, shrinking patch window, server hardening, machine-speed defense

IMPRINT

intelligent piXel GmbH
International Institute of Forensic Expertise (IIFE)
Enzianstraße 4a
82319 Starnberg, Germany

CONTACT

george@rauscher.xyz · Telegram @intelligentpixel

MANAGING DIRECTOR / AUTHORIZED SIGNATORY

Managing Director: George A. Rauscher
Authorized Signatory: Dr. Louise Morgott

RESPONSIBLE FOR JOURNALISTIC AND EDITORIAL CONTENT

Responsible for journalistic and editorial content pursuant to Section 18(2) of the German Interstate Media Treaty:
George A. Rauscher, c/o intelligent piXel GmbH, Enzianstraße 4a, 82319 Starnberg

VAT ID NO. / COMMERCIAL REGISTER

VAT ID No.: DE291416044
Commercial Register: HRB 207 679, Munich District Court