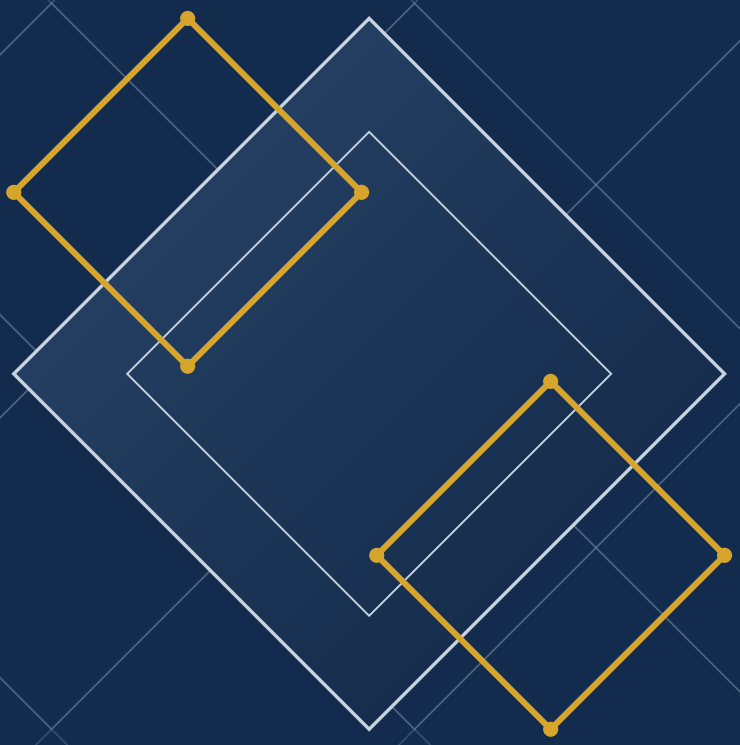
ISECOM

ISECOM PUBLICATIONS

THE

Opensource Cybersecurity Playbook



ISECOM

Institute for Security and Open Methodologies

The Open Source Cybersecurity Playbook

O4 Edition

Written by Pete Herzog

Updated for OSSTMM 4

Creative Commons Attribution-NoDerivatives 4.0 · © 2026 ISECOM

Contents

Part 1: Scouting Reports.

Profiles for the most common threats you should be prepared to face, including the ones that didn't exist when this playbook was first written.

Part 2: The Game Plan.

A practical, step-by-step process for making your organization more secure by building security in instead of bolting it on.

Part 3: Looking Downfield.

Set yourself up for success now and in the long run as threats evolve.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Before the Game: How to Read This Playbook

The first edition of this playbook treated security as a wall. Put something valuable inside, build a barrier around it, station somebody to notice when the barrier gets hit. Enough walls, watched hard enough, keeps you secure.

That's wrong. To know a wall's been breached, something has to watch it, interpret what it saw, and respond. Each step costs time and energy. The more you protect and the more connections you have, the faster those costs grow, faster than your budget ever will. You end up spending more to watch more, and always finding out after it happened.

OSSTMM 4 (O4) is built on a different picture. This edition is organized around it. Three ideas carry it, and you only need the plain-language version to use this book.

Security is shape, not effort. A key doesn't fail to open the wrong lock because turning it would be "too hard" or "take too long." It fails because the shapes don't match. No amount of trying gets the attacker in, because there's nothing for the attack to grab onto. Most of what follows is about shaping your organization so that unwanted interactions have nowhere to grab. Built in, not bolted on. Veterans have said for years that security has to be built in and not added on. O4 is what it looks like when you actually do it.

Risk already happened. The old definition of risk was a bet: the probability of a bad future event times how much it would hurt. A gambling metric. But you can't react faster than the speed of cause and effect. By the time your sensor notices an event, decides what it means, and responds, the event is already in the past and its consequences have already spread. "Detect and respond" is really "high-speed forensics." You arrive just in time to document what already happened. So stop thinking of risk as a future bet. Risk is a present-tense fact: how much of your organization is reachable right now. That's your attack surface, and you can measure it, shrink it, and audit it today.

You win in the Intent phase. Every interaction has three parts: before contact (Intent), during contact (React), and after contact (Resolve). Since you can't out-run the speed of cause and effect once contact happens, React and Resolve are mostly forensics. The one place a defender genuinely wins is before contact, reading the attacker's reach while it's still forming. The scan before the hit. The reconnaissance before the con. The update that quietly changed what it does before you installed it. Reading intent before it lands is the closest thing in security to seeing the future, and this edition treats it as a first-class part of the game plan.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Part 1: Scouting Reports

What security threats should I be prepared for?

The first key to any effective game plan is knowing what you're up against. In this section you'll learn about the most common threats your company is likely to face.

This list is not comprehensive, and it isn't meant to be. It's meant to show you the tactics being directed at you and your users, and the specific reasons you're vulnerable to each, because once you can see the reason, you can change the shape that creates it. As you read, watch for two things under every threat: where it first reaches toward you (its Intent phase, the part you can catch early) and what trust it's abusing to work (the coupling you can remove).

From phishing to ransomware to tainted software updates, the more you know about these threats the better. They're some of the leading causes of data breaches, downtime, and a serious lack of sleep.

Phishing

What it is: Any attempt to compromise a system or steal information by tricking a user into responding to a malicious message. The most common phishing attacks involve emails armed with malware hidden in attachments or links to infected websites, though phishing also travels by voicemail, text message, and social media.

What makes protection a challenge: Employees are already in the habit of clicking things, because that's how you interact with modern computers. Phishing emails are far more convincing than they used to be. Scammers take over legitimate accounts or spoof addresses so a message looks like it came from someone the employee trusts. Once a victim is compromised, the attacker has their access: the same servers, web apps, and files, as if they were that employee. Protecting against it means limiting what employees can reach and how, which can run counter to what they need to do their jobs. And the most targeted "spear phishing" gives no red flags until it's too late.

Phishing almost always has a warm-up you can see if you're looking. A lookalike domain registered last week. A first contact that breaks a known routine (a "vendor" who emails when they always call). A reply-to that doesn't match the sender. That's the Intent phase showing itself before the payload lands.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Social Engineering

What it is: There are two ways to steal anything: take it yourself, or get someone else to hand it to you. Social engineering is the umbrella term for any tactic that exploits and manipulates trust so the victim gives the attacker what they want. Access to information, accounts, or computers inside a secured area. Think a fake customer-service call to reset a password, or a criminal spoofing your CEO's email and asking finance for an urgent wire transfer, a scam known as business email compromise (BEC).

What makes protection a challenge: Everyone, repeat, everyone, can be conned, defrauded, fooled, or manipulated. Sometimes it's a lack of training; more often it's distraction and mental fatigue. Because this attack targets people directly, technical safeguards can do very little, especially when the request isn't outside the employee's normal responsibilities, like resetting a password for a desperate user. Social engineering never breaks a lock; it borrows the authority of someone the target already trusts.

Ransomware

What it is: Malicious software that encrypts a victim's files and demands payment, usually in anonymous cryptocurrency, to decrypt them. Like other malware, ransomware usually starts with a phishing email or a compromised website. Unlike other malware, its goal isn't stealth and persistence. It's to spread as fast as possible, encrypt as much as possible, and then loudly announce itself so the criminals can extort you.

What makes protection a challenge: Ransomware locks up any drive the employee can reach, including connected USB drives and network shares. Once files are encrypted, your only options are a reliable, current backup; a researcher-released decryptor if you're lucky; or holding your nose and paying, which is anything but a sure thing. The authors are criminals and occasionally bad at coding, so you can pay and still lose the files. Worst of all, ransomware turns a strength, files being accessible across the organization, into the weakness. It's a billion-dollar industry now, which means it changes faster than signature-based tools can keep up. Note that ransomware's blast radius is exactly the reach of the account it lands on. The attack didn't decide that radius. Your zoning did, months earlier.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Downloaders

What it is: Normal-looking programs designed to fetch and install malware without tripping alarms. A downloader is the attacker's "man on the inside." It slips in first, scopes out the security settings, then smuggles the dangerous payload once it's decided the coast is clear. Even after the payload is found and removed, a downloader left hiding can start the whole process over.

What makes protection a challenge: Downloaders are one step removed from the dirty work, so they don't carry the functionality that gets malware blocked. Their makers can focus entirely on evading detection. Think of it as attackers fielding skilled specialists instead of mediocre generalists: the downloader is a prolific passer, the payload a sensational scorer, and paired up they're far more effective than either alone.

Drive-by Downloads / Download Hijacking

What it is: In nature, big predators wait at the watering hole for prey to come to them. On the Internet they turn popular website visits into covert attacks. Injecting code through comments that force automatic downloads, compromising a web server to taint legitimate downloads, or running exploit kits that probe each visitor's system for a vulnerability to hit.

What makes protection a challenge: Attackers have surprise and a deep bag of tricks. Update your browser and they update their code; patch one hole and they move to the next. And it isn't only sketchy sites. Some of the web's most popular destinations have been compromised. You usually can't ask employees to just stop using the Internet.

Malvertising

What it is: Marketers aren't the only ones who use advertising to get in front of crowds. Criminals create fake ads, or slip malicious code into legitimate ones, so they can literally capture their audience.

What makes protection a challenge: Online advertising is everywhere and getting more aggressive, and people have stopped viewing ads, even pop-ups, with much suspicion. The knee-jerk fix is ad-blocker software, but many sites refuse to work with it on. And if employees have to choose between their ad-blocker and a top-10 list of cat videos, well.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Zero-Day Attack

What it is: Traditionally, a zero-day is an undisclosed vulnerability attackers can exploit before anyone, victim or vendor, knows to patch it. The term also gets stretched to cover attacks using brand-new tactics, exploits, or malware variants that haven't been seen before.

What makes protection a challenge: It's hard to defend against something you've never encountered, especially when it blindsides you. Signature-based tools are particularly weak here, because they identify malware by matching it against a list of samples they've already seen, and a never-before-seen exploit has no match. Because zero-days work, they're in high demand, and patching one hole often buys only momentary relief before attackers move to the next. You cannot enumerate every unknown attack, so stop trying to recognize the attack and start removing the surface it would need. A payload with nowhere to execute and no path to reach is defeated whether you've ever seen it before or not.

Password Cracking

What it is: A login and password isn't what most people think it is. It's a complicated chain of processes: multiple systems, secure transport to and from servers, a trusted network of server identity assurance and revocation, code to judge password complexity, more code to confirm a human is typing, a second factor, and a way to recover lost passwords. So password cracking is rarely just guessing the password; it's cracking the process to take over the account.

What makes protection a challenge: Any system that lets users in from anywhere and asks them to create, safeguard, and remember their own passwords is going to be hard (maybe impossible) to defend. O4 names the reason directly: the more a person is placed inside a security process, the larger the reachable surface of that process becomes. (Longtime OSSTMM readers knew this as "the Somebody Sequence.") Asking employees to manage their own passwords is like handing out copies of the key to an important lock. You can buy the strongest lock money can buy, but how secure is it with keys floating around everywhere? The fix isn't a bigger lock. It's taking the person back out of the process wherever you can (single sign-on, hardware keys, certificates), so there's less surface for cracking to reach.

Distributed Denial of Service (DDoS)

What it is: A system can only process so much before it slows and buckles. By commandeering a large number of hijacked systems and devices (a botnet), attackers aim a flood of connection requests or junk packets at one target to overload it and knock it offline.

What makes protection a challenge: The bigger the botnet, the more damage. Your best hope under attack is a capable anti-DDoS service, and even that's no guarantee against a high-magnitude flood. Worse, attackers sometimes contact targets in advance and threaten to knock them offline unless "protection money" is paid, and as with ransomware, paying criminals never comes with a guarantee.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Scareware

What it is: You've seen the pop-ups: "Warning! A virus has been detected. Download VirusBlaster to remove it." The real infection is the program the pop-up is trying to get you to download. Scareware comes dressed as fake antivirus, fake browsers, or fake updates.

What makes protection a challenge: Social engineering works because it preys on the distracted and fatigued, and scareware adds eagerness to help into the mix, the "good intentions" spiral that leads to bad decisions. Once scareware is installed it inherits all the privileges, passwords, and logins of the employee who ran it. Getting it out might be as easy as wiping and restoring from backup. If it has spread, much harder.

SQL Injection

What it is: If a website has an input box or form (a login, a credit-card field), an attacker can try feeding it structured query language code to read or alter the stored data behind it.

What makes protection a challenge: SQL injection abuses a trust between the web application and its database to do nearly anything it wants. If all you picture is "delete data," you're underestimating it: add, change, and remove data; steal credit-card numbers, personal data, and health records; or insert malicious code that gets served back to your own users. From there criminals can turn your popular site into their tool: distributing drive-by downloads, building botnets, even hijacking DNS to send visitors to malicious lookalikes of sites they trust. If the login form is vulnerable, injection can bypass the login entirely. Any user-input field on a database-backed site is potentially injectable, which makes it widespread. And you can't just remove all input and still take orders or feedback. The durable fix is structural: the application should treat input as data that can never be executed as a command, so the coupling injection depends on simply isn't there.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Tainted Updates & AI-Generated Code NEW FOR O4

What it is: The threat that grew up since this playbook was first written. Software no longer arrives in shrink-wrap; it pours across your perimeter every day: vendor updates, browser extensions, open-source packages, marketplace apps, and now code and scripts generated on the spot by AI assistants and agents. Attackers have learned that the easiest way past your defenses is to ride in on something you already trust: a legitimate package whose latest version was quietly poisoned, an extension that changed hands, a dependency buried four levels deep, an AI-generated script nobody reviewed. This is a supply-chain and software-perimeter attack, and it's now one of the most common ways organizations get hit.

What makes protection a challenge: Nobody signs off. Gigabytes of code cross the perimeter through dozens of channels, far faster than any human can review, and most of it is barely looked at, if at all. The core mistake is treating trust as if it carries across versions: you approved this vendor once, so every future update is fine; this package was clean last month, so it's clean today. It isn't. The malicious change is almost always in the update, not the original. And AI now writes new internal apps and scripts faster than manual review can keep up, so the pile grows while the review shrinks. The answer isn't a blanket ban. Your business runs on this software. The answer is to read each thing's intent at the gate, every version, before it runs. That's Part 2's intent scanning, and this is the threat it was built for.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Part 2: The Game Plan

As anyone who's been on the wrong side of a data breach can attest, security is not a game. The stakes are real, and unlike a game there are no set rules or boundaries, and attackers aren't known for fair play.

Still, we talk about security in sports terms. There's an offense and a defense. The defender's goal is to stop the attacker from reaching or damaging our assets, using defensive tactics the way the attacker uses offensive ones.

The hard part is that we don't know the parameters of the "game" until it's too late. We don't know who the opponent is, what they can do, or what they want. Stealthy robbery or a smash-and-grab to encrypt and ransom? Are we a target of hate or a moment of opportunity? So we're pressured to build tactics that work against a huge, fast-changing variety of attacks, while also keeping the business running. We don't just have to stop attacks; we have to do it without slowing or stopping the work.

An attacker has no such limits. They only have to be effective against one part of our defense, by any means, to win. We have to be effective against all of it, under budgets, laws, and regulations. The field is anything but level.

With the game stacked like that, what does a winning strategy actually look like? It is not trying to out-watch and out-react an opponent who is faster, cheaper, and only needs to be right once. That's a race you lose by playing. The winning strategy uses the two advantages the attacker can't take away: you can **shape the field so attacks have nowhere to land** (build security in), and you can **read the attacker's reach before contact** (scan for intent). Everything below is one of those two.

Create a Zone Defense

You can't go one-on-one with an opponent you can't see coming. To counter an attacker's flexibility and surprise, build a zone defense. In sports, a zone separates the field into areas of focused defense; the idea is that by creating separation you ensure a threat either cannot physically reach an asset or meets a wall of defense when it tries.

Separation is the whole game, and it's the first thing O4 makes you build in rather than watch. Every connection you allow is a path an attack can travel, and every trust you grant is a coupling an attack can borrow. A zone defense shrinks your reachable surface (how much of

CC BY-NC-ND 4.0 · © 2026 ISECOM

you is in reach right now) by removing paths and couplings you never actually needed. You're not adding a guard to watch the door; you're taking the door out of the wall where no one needs to walk through it. Build your game plan around the following:

1. **Separate services and assets into zones, logically and physically.** Assets of one department should be reachable only by that department. Whitelist all system access (limit it to a predetermined list of authorized types) through a switch and a firewall. Every path you don't explicitly need is reach you're giving away.

2. **Zone your remote (vendor and cloud) accounts with a separate login and secret sentence for each.** Re-using passwords across vendors is the fastest way to get accounts compromised: one breach becomes all of them. Personal certificates and private-key cryptography are much stronger than even secret sentences, and should be used for administrative (root) access to external services. Protect even those keys with a secret sentence: a full sentence with capitalization and punctuation that means something to you, so you'll remember it. (For Facebook: I just wasted 2 hours on Facebook! For your VoIP provider: Boss says, "No more big bills." Punctuation included.)

3. **Create trust zones where no system can directly reach another.** Desktops shouldn't connect to other desktops; network shares should be closed. Even on the same network, no system should reach another unless you planned for it. Have desktops connect to a fileserver to store documents. Make sure administrative ports (SSH on servers, admin web access on intranet servers) are reachable only from the administrator's system, or better, only at that server's own keyboard if it's local. This is least privilege at the network level: only those who should reach a document, printer, file server, or resource can.

4. **Create system zones, not just people zones.** When you authorize who can access what, you've granted a trust, and trust is a coupling attacks borrow. Phishing works precisely by riding a trusted user's access to reach other systems with the user none the wiser. So verify that a resource is being reached by the authorized person, not silently from their compromised system. CAPTCHA is one common way to force proof of a human; it also cuts web-comment spam and brute-force password attempts.

5. **Avoid all-in-one networked systems** (NAS + printer, router + wi-fi). They're inherently weaker because they're harder to configure securely and harder to zone by who's authorized. If your router has wi-fi built in, disable it and use a separate wi-fi router in a zone that can't reach internal systems.

6. **Keep a rigorous patching schedule for anything directly on the Internet.** Racing to patch shouldn't be your primary defense, but for Internet-facing services it's one of the few protections you have. For non-Internet-facing systems, fold patching into regular maintenance rather than auto-installing everything the moment it ships. That gives you time to test on a standard system first and builds a routine in your traffic that makes a malicious install stand out. (And read Part 1's new scouting report: an update is not automatically safe just because the last version was. Scan each one's intent before rollout; see below.)

CC BY-NC-ND 4.0 · © 2026 ISECOM

7. **Zone your desktops, servers, and mobile devices to separate what can execute from what can read/write.** Let temp files be created while denying them the ability to execute, in case of malware. On desktops and mobile devices, whitelist exactly which applications may run or use the network and deny the rest. Exercise least privilege so users don't get administrative rights by default.

8. **Keep work and personal mobile devices separate.** App security is notoriously bad, and each app on a device is connected to every other app on it, which makes zoning essential. If your business requires mobile devices, separate which may hold personal apps and which hold work apps. There's technology to keep work and personal data apart on one physical device if you'd rather not carry two.

9. **Zone with multiple web browsers.** Keep one browser clean, no plug-ins, strictly for banking, vendor administration, and other critically confidential connections. Use another for online shopping. Use another for news, surfing, and social; do run ad blockers and privacy plug-ins on that one to minimize malicious-ad exposure. A single browser with multiple profiles can work too, though some browsers make you close one profile to open another.

10. **Zone your remote interactions:** remote access to the office, cloud services, web-based vendor administration, email. Your best protection here is encryption. Use an encrypted VPN whenever you're sending or receiving from somewhere that isn't your place of work. Because email is plain text and readable along its route, use email encryption anytime you're mailing between remote offices or sending sensitive information.

11. **Use work hours as part of your zoning.** Shut down systems, routers, and wi-fi you don't need running at the end of the day, on weekends, and on holidays. If you can't staff incident response during those times, those things shouldn't be connected to the Internet during them. A system that isn't reachable isn't attackable.

12. **Use blacklists if you can't whitelist, or alongside whitelisting.** A blacklist names what's not allowed; a whitelist names what is. Blacklists let you deny whole categories of devices, programs, or service types you don't need. They're weaker and easier to bypass than whitelists, but much easier to implement and maintain, which makes them useful additional separation. Most antivirus, spam control, and firewall geo-blocking runs on blacklists. Just never make a blacklist your main form of separation; attackers find it shockingly easy to step around.

CC BY-NC-ND 4.0 · © 2026 ISECOM

> **The zone-defense mindset, in O4 terms:** every item above either removes a path or removes a trust. You're shrinking the part of your organization that's reachable right now, which is the only honest definition of reducing risk. And notice what you didn't do: you didn't post a guard on every interaction to watch it forever. Constant watching doesn't scale. The cost of checking everything, all the time, grows with the square of the connections. Double the parts and you roughly quadruple the watching bill, until you're spending more energy policing the system than running it. Biology solved this three billion years ago: your body doesn't interrogate all 37 trillion cells from a central office; each cell governs itself and quietly shuts down if it goes wrong. Good zoning is the same move. Build each zone to govern itself, to deny by default, to need no outside referee for the normal case, and you get security that holds without a watcher you can't afford.

Scan for Intent

Everything that's going to hit you reaches toward you first. The scan comes before the break-in. The lookalike domain gets registered before the phishing email. The downloader checks the locks before the payload arrives. The poisoned update changes what it does before you install it. All of that is the **Intent phase** (the "before contact"), and it's the one window where a defender genuinely beats the attacker, because once contact lands you're into React and Resolve, and those are just forensics at the speed of light. If you only instrument one new capability from this edition, instrument this one.

Scanning for intent means reading what something is about to do instead of waiting to see what it did. Two fronts matter most for a small or mid-sized business, and you should shape your tools around both.

Read the reach aimed at your people and your network. Attackers footprint before they strike. Make that footprinting expensive and noisy so it shows itself:

1. **Change how your servers respond to queries.** Attackers mapping you for a targeted attack are mostly trying to learn what systems and applications you run. Change your servers' responses so there are only positives and false positives. Flood the attacker's tools with noise so they can't tell a real reply from a fake one. Set web servers to answer every request with a 200 even when the honest answer is 404 or 302, and common scraping and analysis tools go mostly useless. Tune TCP so every port answers every SYN with an ACK (easiest with a port-forwarding firewall answering for the server whether one exists at that address or not) and a port scan becomes worthless. It won't stop an advanced attacker with better tools (that's what your zone defense is for), but it shrinks the pool of attackers who can get anywhere, and it turns their reconnaissance into your early warning.

CC BY-NC-ND 4.0 · © 2026 ISECOM

2. **Know what normal looks like, so abnormal announces itself.** This is situational awareness as an instrument. Know your employees, vendors, routines, hours, and policies well enough that a deviation is loud. No "Microsoft Security Center" should be calling to remotely inspect a system you never bought a service for. Knowing that, an employee recognizes the social-engineering warm-up and stops it in the Intent phase. Changes in routine (a customer who emails when they always call) are among the best early signals you have. Make it routine for employees to escalate and document those changes; later you can correlate them with patterns on your network, and you'll often find the reconnaissance that preceded an attempt.

3. **Watch for the scan before the hit.** Your network monitoring should treat a port scan, a sweep of your admin ports, or a sudden probe of a rarely-touched service as what it is: someone reading your shape before they act. Move an obvious target like SSH off its default port and an automated scan looking for it both misses and trips your sensor. You've converted the attacker's own Intent-phase move into your alert.

Read the intent of software before you let it run. This is the newer and now larger front, and it's where "intent scanning" becomes concrete. Software crosses your perimeter all day: downloads, browser extensions, vendor updates, open-source packages, marketplace apps, and AI-generated code and scripts. Most organizations let it in on reputation ("we trust this vendor") and never look again. That's the gap. What you need is a single control point at every place software enters that reads each artifact's intent before it runs, and then approves the safe, blocks the clearly bad, and escalates the edge cases to a human. Shape your tooling so that, at the gate, it can:

- **Read provenance.** Where did this actually come from, who owns it now, and does its origin match what it claims? A package that quietly changed hands is a flag.
- **Read behavior against known tactics.** Does this artifact's behavior line up with recognized attacker techniques (reaching for credentials, phoning home, escalating privilege, hiding itself) before it's allowed to execute? Catch the intent in the code, not the damage on the network.
- **Read the change, not just the thing.** This is the single most important habit: trust does not carry across versions. An update is a brand-new artifact that happens to share a name with one you trusted. Read whether its risk went up from the version before (new capabilities, new network destinations, new embedded code) and treat a risk jump as a stop-and-look, not a routine install.
- **Read for tomorrow's weaknesses.** Surface quantum-unsafe cryptography and other aging-out assurances now, while a planned migration is cheap, instead of discovering them under a compliance deadline or after they break.
- **Read AI-generated artifacts like any other software.** Code and scripts written by an AI assistant or agent are software crossing your perimeter, and they deserve the same gate

CC BY-NC-ND 4.0 · © 2026 ISECOM

as a vendor package: same questions, same approve/block/escalate. "An AI wrote it" is not provenance; it's a reason to look harder.

The point of all of this is advantage, not more alarms. A gate that reads intent at the place software enters, at zero distance, once, before the thing couples to you, is dramatically cheaper than watching every endpoint forever and cleaning up after each install. That's the same self-governance move as good zoning: push the decision to the boundary and make it there, rather than funding a demon to watch everything downstream. Do this well and approvals that used to take weeks take minutes, and the poisoned thing never gets the chance to run. That's what it means to see it coming.

Apply Rehabilitation Tactics

As in sports, outcomes in security are often decided by conditioning done far in advance. Athletes don't show up on game day without countless hours of practice, and franchises don't invest in star players without first hiring the trainers, doctors, and coaches to keep them healthy. Ask yourself: how rigorously has your security been tested? How many scenarios have you prepared for? How fast can you recover from a setback?

In O4 terms, this is about continuity and resilience: making sure that when contact does land (and some always will), the Resolve phase is a quick bounce-back instead of a slow bleed. Security is equal parts preparation and recovery. Adjust your game plan to regularly revisit what to do when something goes wrong:

1. **Set up a way to back up and restore systems, data, mobile devices, and desktops.** Backups should be centralized, secured, and ideally encrypted. If they're encrypted, both off-site (cloud) and internal backups are acceptable, but remember, anything backed up outside your network is no longer in your control. If you can't afford to properly build and test a backup system, you're better off leaving it to a service that can. The key is quick recovery: test your restores and make sure they happen fast.

2. **Keep tools handy to fully image a drive for forensics and recovery.** Both hardware and software options exist, some certified for criminal investigations. If you carry cyber insurance, you may be required to use specific forensic software to file a claim.

3. **Keep shared data in central locations like a NAS server.** That limits the loss when one desktop fails. Standardized desktop installs plus nightly backups shorten recovery further. A well-prepared shop can fully image a compromised system for forensics, wipe it, and reinstall to near-full operation in under two hours.

4. **Back up everything, even your centralized data.** It's the only way to assure a complete recovery. Set all data services and file servers to be backed up and redundant with fall-back servers, so a server failure doesn't mean loss of service. This is continuity built in.

CC BY-NC-ND 4.0 · © 2026 ISECOM

5. **Keep emergency contact numbers for vendor support.** If forensics uncovers why an attack succeeded, it may need immediate attention (a patch, a configuration change, or both), and you don't want to be hunting for a phone number mid-incident.

6. **Consider a distributed denial of service (DDoS) protection service,** especially if your business depends on customers reaching an online presence like a store. Many hosts offer this already; make sure you know how much protection they actually provide, and consider a specialist if you need more.

Use Your Home Field Advantage

The Internet is a hostile environment, which is exactly why we go to such lengths to let our networks interact while staying separated from it. But don't stop at making a safe place for your users; go one step further and make it a hostile place for attackers.

Creating conditions that let you thrive while your opponent struggles is what sports calls home-field advantage. Sometimes it's just the crowd; sometimes it's real environmental factors. Some teams are built for grass over turf, altitude, heat, or cold. In cybersecurity your three home-field advantages are **visibility, protocol, and process**. Lean on them and you force the attacker to play the away team: they have to come into an environment you know better than anyone alive. In O4 terms, all three come from knowing your own system's healthy ground state so well that anything off-pattern stands out.

1. **Know what's happening on your network.** This is probably your single biggest home-field advantage. To see what's running at any hour, there are tools worth looking into. SIEM products consolidate logs and real-time traffic; intrusion detection systems (generally a blacklist technology) watch live traffic and respond to some attacks. Even without them, free tools like tcpdump and Wireshark give you more than enough to understand what's on your network and where it's going, enough to see whether your zone defense is working. As the network grows, behavioral-analysis tools help automate sniffing out anomalies, because not all malicious activity is obvious.

2. **Know what belongs on each default install,** every desktop and server. That lets you recognize malware or non-standard installs on sight. File-integrity software will alert you to changes in files and directories that signal compromise, and it's most powerful on servers, where it can immediately replace files an attacker dropped.

3. **Change default settings.** Your protocols aren't just the packets on your network, but how you've configured them to run. An attacker has to learn what you run, how it's connected, and how it responds, so change the default install and directory names, default logins and passwords, and default configurations like the service ports on devices and servers. That alone protects you from the bots that hunt for defaults. Simply moving SSH's admin port to a random high number keeps it out of obvious reach and lets your monitoring catch the scan that comes looking: an Intent-phase win handed to you for free.

CC BY-NC-ND 4.0 · © 2026 ISECOM

4. **Change how your servers respond to queries.** (This is important enough that it leads the Scan for Intent section above. It belongs in both, because it's both a home-field advantage and your cheapest early-warning system.) Flooding attack tools with positives and false positives turns reconnaissance into noise for them and signal for you.

5. **Know what normal looks like** by knowing and understanding your employees, vendors, routines, operating hours, and policies. If you know no "Microsoft Security Center" should ever call to inspect your systems, you'll recognize the call for the social-engineering attempt it is and stop it fast.

6. **Make your users part of the solution.** When employees know what's normal, they prevent the majority of fraud and phishing. People who know the routine notice fast when a customer emails instead of calls, and a change in routine is one of the best attack indicators you have, with employees as your first line by reporting it. Make escalating and documenting those changes routine, and you build a forensic trail you can later correlate with network patterns. Get in the habit of proactively asking employees how things are running. A noisy drive, a slow system, or a strange message can then be investigated immediately, and if you're set up for quick recovery, a suspicious system can be wiped and reinstalled before it becomes a crisis.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Part 3: Looking Downfield

Despite the sports analogy, let's be clear: cybersecurity isn't something you dabble in. There's no bush-league cybersecurity. It's not a game. Losers are real victims with real, sometimes catastrophic losses, and even winning is a fleeting moment, gone as fast as it came.

Cybersecurity is hard because we don't know all the vulnerabilities we have or all the threats there may be, and even if we did, they change too fast to hold a specific defense against each one. The environment itself keeps changing: new technologies, new business models, new services, new communication channels, and now AI writing software faster than anyone can review it. This competition with attackers lasts indefinitely, so your strategy has to win not just this season but every season after.

To do that, keep your players in shape, steadily improve your equipment, maintain a healthy environment, and employ the right tactics. This is also where O4 earns its keep: because you've built security into the shape of your organization rather than bolting watchers onto it, you're not re-fighting every new threat from scratch. A smaller reachable surface and a habit of reading intent keep working against attacks that hadn't been invented when you set them up.

Training

All employees need to understand how cybersecurity touches their day-to-day role, and they need training in situational awareness both on and off the computer.

Teach all employees how they can be manipulated and fooled, in person and online. There are infinite variations of an attack but a finite number of ways a person is vulnerable to fraud and manipulation. Rather than expecting employees to identify every shifting threat, teach them how to react securely when one of their vulnerabilities is probed. For example, when they're asked for sensitive information. You're training the Intent-phase reflex: recognize the reach and respond safely before contact.

Make it easy for employees to talk about security or privacy concerns. Employees are your best anomaly-detection engine. Make sure they can voice doubts and report unusual activity before those concerns grow into real problems.

Really know your systems. Anyone in IT or security should know how all the current software and systems actually work inside your organization, not just be certified or experienced in the brand. You can't secure what you don't understand in your own environment. Bring in a coach who'll teach you and your team security in the context of your operations.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Understand how to apply the basics. All security personnel should be well versed in the operational security fundamentals of O4, which apply anywhere from physical to cyber to anything. Don't let them just say they read it. Have them map your business processes onto O4's measurements and controls so you can see, concretely, how much of your organization is reachable right now. That reachable surface is your attack surface, and putting a number on it turns "are we secure?" from a guess into something you can audit, shrink, and re-check.

Exercises

Knowing how to swim means nothing when the tsunami hits. To stay above hostile waters, train regularly on how to react when things go wrong, so that despite fear, confusion, or fatigue, the default reaction is the secure one.

Create an incident response plan and drill it. Build an emergency checklist employees can follow and keep walking through it until everyone's clear on how to react. Include new hires from day one.

Get employees used to dealing with attacks. Run regular exercises designed to manipulate them: mock phishing, in-person and phone social engineering, fake malware activity. Make the Intent-phase moment familiar so it's recognized under pressure.

Give short, mandatory security challenges and puzzles. Keep them to a couple of minutes, done without pulling employees fully out of their day. This trains them to think security while working, because nobody gets to stop working during a real attack.

Equipment

In many sports, being the better athlete doesn't guarantee a win; the difference between first and second often comes down to equipment: new materials, new technology, new designs.

Make the most of your tech. Better routers improve response times, better processors allow stronger encryption, and more elastic monitoring reduces how thin you have to spread resources.

Consider the cloud. Cloud services can give you functionality you don't have to build or run yourself. There's a real trade-off in control and self-sufficiency (what's off your network is outside your control), but the added capability can let you do more with less.

When you can't buy new, maintain the old. Sometimes upgrading RAM or disks is enough to make the difference.

Keep your systems running lean. Stick to necessary software, minimizing the memory-resident, memory-hungry programs that slowly grind a system to a halt. Lean systems also have a smaller reachable surface: less installed is less to attack.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Equip yourself to read intent. The modern addition to the kit: make sure your tooling can actually read what crosses your perimeter before it runs. Software at the gate and reconnaissance on the wire. You don't need a specific product name; you need the capability, built in at the points software and traffic actually enter, so the decision happens at the boundary instead of after the fact.

Environment

Some athletes compete on clay, some on grass, some on turf. What they share is that they've honed their game to their environment and built a home-field advantage by learning to outcompete others on it. Do the same by getting to know your own environment, actively maintaining it and noticing even the smallest changes.

Know what "normal" looks like. Routinely examine the protocols on the network, which systems talk to each other, how employees use their desktops, and how mobile devices behave. Fire up a packet sniffer and watch live traffic in different segments; make sure you know what all of it is and whether it should even be there. A constant-monitoring tool like a SIEM is better still. Then track down the anomalies. Knowing normal is what makes abnormal visible. It's the ground state everything else is measured against.

Remember: not all software needs to be updated. Always update when a security patch is necessary and you have no other way to control the vulnerability. That requires the people and time to investigate. If an update is purely new functionality, ask whether you actually need it. Many a process has broken and many a file been lost to an update that changed something you didn't need. As any coach will tell you, if it's not broken, don't fix it. So have a process to pick and choose which updates get installed. And when you do install one, read its intent first (Part 2): a trusted name on a new version is not the same as a trusted version.

Make rounds and talk to employees. Learn what they need to do their jobs, how they deal with third-party vendors, and what software they use. People build routines, and they're your best source for knowing when something outside your direct control (cloud services, vendor extranets) stops behaving normally.

Forget Strategy, Choose Your Tactics

A strategy is part of any modern business plan. So what's a cybersecurity strategy? A plan with goals and objectives to achieve cybersecurity.

People who sell cybersecurity strategies like to say it also includes specifics on tools and metrics. But that's just smuggling tactics into the strategy so it doesn't sound so useless.

Yes, useless. Fun fact: a cybersecurity strategy is useless. If you don't have one written down, it's because you already have one so obvious you never bothered, the same way you don't have a formal not dying strategy. A formal one would say you don't want threats of any sort affecting your assets of any sort, now or in the future. Which is obvious.

CC BY-NC-ND 4.0 · © 2026 ISECOM

So why the obsession with strategy? Because tactics are hard. It's easier and safer to make a strategy sound important while meaning nothing than to choose and implement tactics that actually work, and you look good longer, because an empty strategy can coast for ages while empty tactics get noticed immediately.

Tactics are where the rubber meets the road. They're the bat striking the ball; they're literally the packets smacking the server. They're how you "do the thing you do to the things you have" to achieve security, and that's hard to plan out. In O4 terms, every good tactic is a control applied at zero distance, built into the place where the interaction happens, not watched from across a gap. A few steps to point you the right way:

Draw out the interactions and separations between your systems. Base it on the packet activity you actually see on the network: a hybrid physical-and-logical map with arrows for the interactions. Then decide what should be separated or controlled. That's you applying tactics.

List all of your network processes: server backups, administration, remote desktop support, and the rest. Know exactly what systems and people each can interact with, then decide what it should interact with. How you control that authorization is you applying tactics.

Do the same for wireless, physical room and asset access, and telephone systems. Decide which tactics you need to assure control.

Then take a step back. Make sure every tactic you chose fits the overall company resources and doesn't hamper workflow. Security that stops the business isn't security; it's just a different outage.

Learn to Work with Non-Technical Colleagues

Humans are social animals. Most of us like being liked, belonging to a team, standing up for a cause, being part of a community. Even when we don't love it, we don't want to be pushed out. But liking it doesn't mean we're good at it, and a lack of communication with the people who hold the resources will cripple your ability to build the security you need. Some tips to make it run smoother:

Don't assume that because it makes sense to you, it makes sense to everyone. People don't like change. Plan for that hurdle whenever you propose one, and show that the change is the new normal. After all, the criminals targeting organizations like yours change their tools and techniques constantly, and it's working for them.

Communicate according to your audience's capabilities. If they're tech-savvy, don't talk down. If they aren't, don't bury them in tech-heavy reasoning.

Think about the money before you talk. Your company exists to make money, so show value: how added security can replace something you're already doing, or reduce downtime, add quality, or improve efficiency. (O4 helps you here: because it lets you put a number on your reachable surface, you can show risk going down as a measured fact, not a feeling.)

CC BY-NC-ND 4.0 · © 2026 ISECOM

Don't waste their time. Don't let them feel for a moment that you are. That means being prepared and polished.

Never say "I told you so." Don't bring up the time your suggestion got ignored and something bad happened. Don't act like you did them a favor or expect a reward for your brilliance. Start from zero each time.

The Big Picture

In cybersecurity you're surrounded by conflicting advice from authorities who are themselves struggling to define what "good" security even is. The most basic questions leave professionals guessing: Is it more secure to harden servers or use a firewall? When do you need two-factor? Certificates or passwords? Do reverse proxies really add security?

For a long time those questions had no settled answer because the field had no instrument. No way to measure security, only ways to argue about it. So it fell back on measuring effort: how hard you tried, how many products you bought, how many alerts you watched. That was always the wrong yardstick. The whole history of attacks is the history of effort-based security failing: the big-number lock that was safe only because factoring it took too long, right up until the math got faster and the "hardness" simply evaporated. Security that rests on how long something takes was always standing on sand.

O4 is the field finally getting its instrument. It says, plainly, that security is not how hard you make the attack but what shape you give your organization, and that shape can be measured. Your risk isn't a bet about a bad future; it's the distance, right now, from where you sit to your nearest failure. Shrink the reachable surface and that distance grows, measurably. Read intent at the boundary and you win in the only phase where winning is possible. Build the controls into the structure, at zero distance, like a lock whose shape simply won't admit the wrong key, and you stop paying the forever-tax of watching.

That reframes the humbling truth every security veteran knows. Things don't go bad gradually; they go bad suddenly, from one breath to the next, the way you go from running like a gazelle to sliding across the ground with no idea why you fell. The reason it feels that way is that effort-based security hides your real exposure until the moment it's spent. Measured security doesn't. You can see the distance to the edge before you reach it.

The online security community still has more advice than you could ever use, and much of it conflicts, because leaning on the community gets you mostly the answers that let the community, especially its vendors, profit. That skews the research, skews the tools, and skews what you think you need to know. Good security professionals tell you what you want to know; great ones tell you what you need to know, and you'll know it's the latter because it'll frustrate you and make you wish you could just buy and install a product. You can buy the product. But without the work to fit it to your environment and your processes, you won't fix the problem. You'll more likely create new ones.

CC BY-NC-ND 4.0 · © 2026 ISECOM

So treat the community as a resource, not an answer, the way Wikipedia is a resource, not a term paper. What you get from it rarely fits as-is. You still have to answer your own riddle, and that takes knowing your environment, your employees, and how it all works together: the Big Picture. Know where the interactions are, what resources are used and from where, and who has authorized access to what. The difference now is that you no longer have to solve that riddle blind. O4 gives you the coordinate system, and this playbook gives you the moves: shrink what's reachable, read intent before contact, build the control into the shape. Do that and security stops being a slogan you repeat and becomes something you can measure, hold, and prove, season after season, including against the attacks nobody's invented yet.

About

The **Institute for Security and Open Methodologies (ISECOM)** is a non-profit, open research organization focused on all things related to security and hacking. ISECOM created and maintains the **Open Source Security Testing Methodology Manual (OSSTMM)** and Hacker Highschool, designed to teach cybersecurity to teens.

Learn more at www.isecom.org.

A Short Glossary for This Edition

Attack surface. How much of your organization is reachable right now: the sum of every path an attack could travel and every trust it could borrow. In O4, risk is the present-tense size of this surface, not a probability about the future. You shrink it by removing paths and couplings you don't need.

RAV (Risk Assessment Value). The number OSSTMM 4 produces when you measure your organization against the full framework. It converts "how reachable am I" from an argument into a score, computed from your controls, your coverage gaps, and your limitations. The field's instrument. What makes "we're more secure than we were last quarter" either true or false in a way everyone can see.

Build it in, not bolt it on. Putting a control into the structure of a system, at zero distance to where the interaction happens, instead of adding an outside watcher to react after the fact. A lock that won't admit the wrong key's shape, versus a guard who notices the wrong key was used.

CC BY-NC-ND 4.0 · © 2026 ISECOM

Intent / React / Resolve. The three phases of any interaction: before contact, during contact, after contact. Defenders have real advantage in Intent and mostly forensics in React and Resolve.

Intent scanning. Reading what something is about to do before you let it act or enter: the reconnaissance aimed at your people and network, and the intent of software (its provenance, its behavior against known attacker techniques, and whether its risk changed since the last version) read at the gate, before it runs.

Security as geometry. The idea that strong security comes from shape, not effort: arranging a system so an unwanted interaction has no matching path to grab onto, so it can't happen rather than merely being hard.

The reachable-surface rule (formerly "the Somebody Sequence"). The more a person is placed inside a security process, the larger that process's reachable surface becomes. Take the person out of the loop where you can.

Trust as a coupling. A granted trust is a connection an attacker can borrow instead of break. Phishing, social engineering, and SQL injection all work by riding a trust rather than defeating a lock. Removing an unneeded trust removes the attack that would have used it.

CC BY-NC-ND 4.0 · © 2026 ISECOM