GUIDE

# Social Engineering and Phishing Defence Guide

Breaking the targeted-attack kill chain before it reaches the ask

| | | | |
|---|---|---|---|
| Prepared for | Web3 teams | Classification | Public |
| Prepared by | Oak Security | Date | 2026-06-17 |

Oak Security GmbH

# 1. What This Guide Is For

This guide is for everyone on a Web3 team, technical or not. Engineers who run `npm install`, signers who press approve, operators who authorise wires, founders who answer recruiter DMs. The attacks below do not target your code. They target you.

The most expensive Web3 losses of 2024 and 2025 did not start with a contract bug. They started with a conversation. A recruiter on LinkedIn. A finance worker on a Zoom call. An engineer opening a take-home from a company that turned out never to have existed. Every technical control in your stack terminates at a human decision, and a human who has been talked into bypassing the control defeats it.

> **NOTE**
>
> **Baseline rule:** treat every unsolicited contact as hostile until verified out-of-band on a channel the other party did not introduce, and treat manufactured urgency as a red flag, not a reason to move fast.

The defences here are mostly procedural, not technical. They work because they slow the attack down at the point where the attacker needs you to act quickly. A culture in which "let me check and come back" is normal stops most of these attacks before any money moves.

# 2. The Kill Chain: Six Stages of a Targeted Attack

A modern targeted attack is not a single event. It is a sequence of stages, each one setting up the next. The model below is a simplified version of the Lockheed Martin Cyber Kill Chain and the MITRE ATT&CK taxonomy. The point of naming the stages is that you can break the chain at any one of them, and the earlier you break it, the cheaper the defence.

| Stage | What the attacker does | Where you break it |
|---|---|---|
| 1. OSINT | Reads your public footprint: LinkedIn role and tenure, conference talks, GitHub history, X handle, ENS records, conference photos showing your hardware-wallet model. | Publish less. Do not leak your tooling, your signer role, or your hardware-wallet model. |
| 2. Pretext | Builds a story shaped by Stage 1: a "stealth Layer-2" recruiter for the engineer who just open-sourced a library, a "CFO" with a confidential transaction for the finance officer. | Recognise that a story tailored to your public work is a signal, not a coincidence. |
| 3. Contact | First touch on a chosen channel: LinkedIn DM, Telegram, X, email, a Calendly link. Calibrated to look normal, not to close. | Treat unsolicited contact as hostile until verified out-of-band. This is the cheapest reliable break. |
| 4. Rapport | Two or three weeks of fluent, technically coherent exchange. A deck, a "co-founder", a GitHub org with real-looking commits. | Verification does not expire. A warm three-week conversation is still unverified. |
| 5. The ask | The first request requiring action: run this take-home, join this call, sign this message to verify your wallet, approve my screenshare. Always something you can plausibly do without | Slow down. Run unknown code only in a disposable VM. Refuse signing or approving outside the normal process. |

| | | |
|---|---|---|
| | alarm. | |
| 6. Payload | The action lands the attack: a `postinstall` script exfiltrating keys, a Zoom remote-control grant, an EIP-712 signature authorising a transfer. | By here the cheap defences are gone. Sandboxing or hardware verification is your last line. |

The defining property: the earliest break is the cheapest. A target who never publishes their hardware-wallet model never reaches Stage 1. A target who treats every recruiter DM as hostile breaks the chain at Stage 3. A target who runs unknown code only in a disposable VM survives a Stage-5 ask with no consequence.

# 3. Phishing Today, Not What It Was Five Years Ago

Five years ago, phishing meant misspelt "Nigerian prince" emails, pattern-matched in two seconds and filtered. The modern equivalent shares the name and almost nothing else.

Today's phishing arrives as a recruiter-style DM on LinkedIn or Telegram from a person whose profile is real, whose company website checks out, whose GitHub has two years of commits, and whose mutual connections overlap with yours by three or four people. The language is fluent. The technical references are correct. They have read your public work. Personalisation is automated: large language models produce convincing first-contact messages tuned to a target's published interests at scale.

The hook lands where you are weakest. Flattery for the engineer who just shipped something significant. Fear of missing out for the founder watching another protocol close a round. Urgency for the operations lead told "the CEO needs this signed before the wire window closes". The ask is always runnable: try my repo, do a quick screenshare, sign this message to verify your address, just open this PDF.

The red flags survive even when the language and backstory are flawless:

| Red flag | What to check |
|---|---|
| Sender domain | Does the address match the real domain exactly? Look-alikes and homoglyphs hide here. |
| Link target | Does the actual destination match the displayed text? Hover or inspect before clicking. |
| Out-of-band channel | Can you verify this person on a channel they did not introduce? If not, do not trust it. |
| Time pressure | Is there a deadline, a window, a "before someone notices"? Urgency is the attacker's tool. |
| Local action | Does the ask require you to run, sign, install, or approve something on your machine? |

The structural defences are procedural. Phishing-resistant MFA everywhere: hardware keys or passkeys, not SMS, not TOTP for production access. Magic links and one-time passwords are phishable by design, because the user types the OTP into the wrong site and the attacker relays it within seconds. Type URLs by hand or use bookmarks for every service the team uses. Treat unsolicited contact as hostile until verified out-of-band. And the single strongest defence against a manufactured-urgency ask is a culture in which slowing down is encouraged rather than punished.

# 4. The Fake Job Offer: The Modern DPRK Classic

The most common shape of an active payload in 2024 and 2025 is the fake job offer, also called the fake recruiter or coding-exercise attack. Threat-intelligence vendors attribute the bulk of the activity to North Korean state actors under names such as Lazarus and BlueNoroff, though the technique is widely copied by financially-motivated groups.

The shape is consistent. An unsolicited recruiter DM lands, usually on LinkedIn or Telegram, occasionally over email. Compensation is above market. The company story is polished: a stealth Layer-2 with a recognisable VC backer, an L1 moving from research to engineering, a market-maker hiring quants. After two or three exchanges comes the "small coding exercise" or "take-home". The target receives a `takehome.zip`, an npm package name, or a URL to a forked GitHub repository.

The payload lives in the hook. Sometimes a `postinstall` script in `package.json` that runs the moment the developer types `npm install`. Sometimes a compiled binary disguised as a build artefact. Sometimes a malicious VS Code extension that auto-installs when the project opens. The targets are predictable: wallet files, SSH keys, browser session tokens, GitHub personal access tokens, AWS credentials in `~/.aws`, any private-key file on disk.

This works on senior engineers, and the headline incidents almost always involve senior engineers, because they are confident they would spot it. The repository looks real. The commits are real. The code compiles. The malice sits in a post-install hook, a compiled binary, or a few lines of obfuscated JavaScript in a build dependency, not in the obvious source. Confidence is the failure mode the attacker is exploiting.

> **NOTE**
>
> **Case study: Axie/Ronin, March 2022, approximately $625 million.** The initial vector was a fake job offer delivered through LinkedIn to a senior engineer at Sky Mavis. The "interview" culminated in a malicious PDF. Opening it on the work laptop let the attacker compromise validator keys and ultimately the bridge's signing quorum. A textbook Stage-6 payload, with free Stage-1 reconnaissance: the engineer's role and seniority were public.

The defences are simple to state and hard to enforce. Never run unknown code on your work laptop. Use a disposable VM, a Firecracker microVM, or a cloud sandbox for any take-home, prototype, or "quick test". When suspicion is non-trivial, decline. A recruiter who pushes back on "I will look at this in a sandbox" has just told you more than the dossier did.

# 5. Email, DKIM Replay, and Sender Spoofing

Email remains a primary delivery channel because its authentication model is bolted on rather than designed in. The protocol was specified without authentication: anyone can send `From: vitalik@ethereum.org` and a vanilla receiver cannot tell. SPF, DKIM, and DMARC are the three layers that, deployed together, close most of the gap.

| Layer | What it does | Limit |
|---|---|---|
| SPF | DNS TXT record listing IPs/hosts authorised to send for the domain. Receiver checks the envelope `MAIL FROM`. | Does not survive forwarding. Does not authenticate the visible `From:` header the user sees. |
| DKIM | Sending server signs each message with a private key; public key in DNS. Receiver re-verifies the signature. | Survives forwarding, but vulnerable to replay (see below). |
| DMARC | Ties SPF and DKIM together, requires the visible `From:` to align with an authenticated identity, sets policy: `p=none`, `p=quarantine`, or `p=reject`. | Without `p=reject` you have email aspirations, not email authentication. |

The DKIM replay attack matters because it bypasses naive deployments. A legitimately-signed message (a Google sign-in notification, a transactional email from a SaaS vendor) is captured by the attacker and re-sent, unmodified, to a different inbox. The signature still validates. The sender domain still looks authentic. The content was never meant for the new recipient. Replay is increasingly used to slip phishing past filters that trust DKIM-passing mail, especially when the replayed message resembles something the target was expecting.

Beyond authentication, the operational red flags repeat because they still work:

- **Look-alike domains.** `oaksecurityy.io` with a doubled letter, `oak-security.com` with a hyphen, Unicode homoglyphs substituting Cyrillic for visually-identical Latin letters. The eye skims; the header tells the truth.
- **Reply-to swaps.** The `From:` reads as the CEO; the `Reply-To:` points to an attacker inbox. Most clients do not surface the difference.
- **Display-name spoofing.** The visible name is "CFO Maria Chen"; the address is `marja.chen@gmail.com`. The client renders the friendly bits and hides the diagnostic bits.

The structural defence: email must never be the channel that authorises a high-value action. A treasury wire, a multisig signature, a DNS change. None of these belong on an unsigned email. The Oak baseline for outbound mail is SPF ending in `-all`, DKIM with a 2048-bit key rotated annually, DMARC ramped from `p=none` for thirty days while reports are read, then `p=quarantine`, then `p=reject`. Any sensitive action verified through email is verified again on a second channel: a call to a known number, a Signal message to a verified safety number, a signed transaction in a tool with its own authentication.

# 6. Live-Session Social Engineering: Zoom, Calls, and Deepfakes

The Zoom-call era introduced a category older social-engineering literature did not cover: live-session manipulation, where the attacker is on a video call with the target in real time. Three sub-patterns are now common.

- **The Zoom screenshare-control ask.** An attendee asks the host to approve them to share their screen, then requests remote-control permission of the host's screen. One click hands the attacker keyboard-and-mouse access. This is not a vulnerability; it is a feature working as designed. The defence is procedural: never approve screenshare or remote-control for a person you do not personally know, on a meeting you did not personally schedule.
- **The fake-exec voice call (vishing).** Voice-phishing. The attacker calls a finance officer, claims to be the CFO, authorises an urgent transfer. Voice cloning convincing across a thirty-second sample is now a commodity service, and real-time synthesis can hold a conversation, not just play a recording. The SMS cousin is smishing.
- **The fake-exec video call (deepfake).** Real-time deepfake video is operational, not theoretical. A live deepfake renders a moving, talking face on the operator's video feed in real time, with a voice clone. Cameras-on, the canonical proof of identity for remote meetings for the past fifteen years, no longer proves the person on screen is who you think. It proves only that somebody put effort into the call.

The counter-play is procedural, not visual:

| Control | How it works |
|---|---|
| Pre-agreed challenge phrases | A question whose answer is known only to the two people involved, agreed in a different channel before the call. |
| Out-of-band callback | Call back on a number stored in advance, never the number shown on screen. |
| Dual signoff | Any unusual transfer requires a second signer, reached through a different channel from the first. |

None of these are new. They are the procedural opsec used by intelligence agencies and high-net-worth families for decades. What changed is that they now apply to a finance team in a remote-first company.

# 7. Case Study: Arup Deepfake, February 2024, $25 Million

The Arup case is the inflection point. In February 2024, a finance employee at the British engineering firm Arup was invited to a video call about a "confidential transaction". The original lure was a phishing email from the firm's "CFO" requesting discretion. The employee was suspicious, until the video call. On the call were the CFO and several recognisable colleagues. The faces moved. The voices matched. The colleagues spoke to each other as well as to the employee. Reassured, the employee authorised approximately $25 million across fifteen wire transactions over the course of the meeting.

Every face on the call was a live deepfake. The "CFO" was a synthesised face on an attacker-controlled video feed; so were the colleagues. The voices were cloned from publicly-available audio: interviews, panel recordings, internal-comms videos leaked into search results. The call was an entirely synthetic environment, populated by avatars, addressing the one real human in the room. The transfers cleared before any out-of-band verification was attempted.

For two decades, "I saw them on camera" was sufficient proof of identity for almost any decision a remote organisation made. Arup retired that assumption overnight. The opsec failures are textbook: no out-of-band callback to a known number, no dual-authorisation requirement on unusual transfers, the camera-on signal treated as proof of identity, and the "confidential transaction" urgency framing allowed to short-circuit the standard authorisation flow.

The lessons generalise to Web3 immediately. A Zoom call with somebody who looks and sounds like your CFO, or a portfolio-company founder, is not identity. Pre-agreed challenge phrases verified before any consequential action. Out-of-band callback on a known number for any urgent transfer. Dual signoff on any unusual movement of funds.

# 8. Case Study: Bybit Safe{Wallet}, February 2025, $1.5 Billion

The Bybit heist, at $1.5 billion the largest crypto loss recorded, is treated in detail elsewhere as a multisig and supply-chain incident. Its human-factor dimension belongs here too.

What the Bybit signers experienced was not user error. They were seasoned operators. They opened the Safe{Wallet} interface, read the transaction description, checked the operation type the UI displayed, and signed what they were shown. The Safe{Wallet} UI had been compromised through a developer-machine compromise at the Safe team: a Lazarus operation whose multi-stage kill chain began with social engineering against a Safe developer. The compromise was scoped to activate only when the signing address matched Bybit's, so other Safe{Wallet} users at the same moment saw a clean UI. The Bybit signers saw a UI engineered to lie to them specifically.

The human failure was not ignorance. It was trust in the tooling. The signers had no independent way to verify that the calldata they were signing matched the displayed description, because their hardware wallets did not decode the operation independently of the UI, and no second tool re-rendered the calldata against an expected template. One compromise point, the UI provider, controlled what every signer saw.

> **NOTE**
>
> The Bybit lesson: even the most careful human, doing exactly what the procedure asks, loses when the tooling that mediates the decision has been compromised. Verify calldata against a known-good decoded template before signing. Prefer hardware wallets that decode and display the actual operation rather than a UI-supplied summary. Assume the signing UI may itself be the attacker.

# 9. Verification Protocols

The protocols below are the operational core of this guide. They convert ″be careful″ into specific steps. Each defeats a specific attack from the sections above.

# Out-of-Band Confirmation

Any sensitive request received on one channel is confirmed on a second, independent channel before action. The second channel must be one the requester did not introduce.

1. A request arrives (wire, signature, credential reset, urgent favour from an "exec").
2. Do not act on the channel it arrived on.
3. Contact the requester through a separately-sourced channel: a phone number from your own records, a Signal contact with a verified safety number, an in-person check.
4. Confirm the request matches. If you cannot reach the requester, the request does not proceed.

# Callback Verification

For voice and video requests that look like a known person:

5. End the call, or hold it.
6. Call back on a number you already have stored, never the number that called you and never one supplied during the call.
7. If the person who answers has no knowledge of the request, you have caught a vishing or deepfake attempt.

# Code Words and Challenge Phrases

For teams handling treasury, exec, or signing decisions, agree challenge phrases in advance, out of band:

8. Each pair or group agrees a phrase or a question-and-answer known only to them.
9. The phrase is never sent over the channel it protects. It lives in a password manager's secure note or is memorised.
10. On any high-stakes live call, either party can issue the challenge. A correct response is required before proceeding.
11. A deepfake or voice clone trained on public audio does not know the phrase.

# Treating Urgency as a Red Flag

Urgency is the attacker's primary tool because urgency suppresses verification. Make it a trigger, not a pressure.

12. When a request carries a deadline, a "do not tell anyone", or a "before the window closes", escalate scrutiny rather than speed.
13. State plainly: "I will verify this and come back." A legitimate counterparty accepts this. An attacker pushes back, and the pushback is the signal.
14. Never let a manufactured deadline override a standard authorisation flow.

# 10. Defensive Checklist

# For Individuals

| Control | Why |
|---|---|
| Phishing-resistant MFA (hardware keys or passkeys) on every critical account. | SMS and TOTP are relayable; OTP phishing happens in seconds. |
| Type URLs or use bookmarks; never click links in unsolicited mail or DMs. | Defeats look-alike domains, homoglyphs, and link-target spoofs. |
| Treat every unsolicited recruiter or "exec" contact as hostile until verified out-of-band. | Breaks the kill chain at Stage 3, the cheapest point. |
| Run unknown code only in a disposable VM, microVM, or cloud sandbox. | Survives a fake-job-offer payload at Stage 6 with no consequence. |
| Never approve Zoom screenshare or remote-control for someone you do not personally know. | The remote-control grant is a one-click full compromise. |
| Verify calldata on your own hardware screen before signing; disable blind signing. | The signing UI may be the attacker, as in Bybit. |
| Do not publish your hardware-wallet model, signer role, or tooling. | Denies Stage-1 reconnaissance. |
| Treat urgency as a reason to slow down, not speed up. | Manufactured urgency is the attacker's main lever. |

# For Teams

| Control | Why |
|---|---|
| No high-value action (wire, signature, DNS change) is ever authorised by email alone. | Email authentication does not prove intent or identity of the request. |
| Outbound mail: SPF `–all`, DKIM 2048-bit rotated annually, DMARC ramped to `p=reject`. | Closes spoofing and most replay-based phishing against your domain. |
| Out-of-band callback on a known number required for any urgent transfer. | The Arup counter-measure; defeats vishing and deepfake exec impersonation. |
| Dual signoff on any unusual movement of funds, second signer on a different channel. | One compromised person or one synthetic call cannot move funds. |
| Pre-agreed challenge phrases for high-stakes live calls. | Defeats real-time deepfakes trained on public audio. |
| Interviews camera-on and recorded; unscripted questions about the candidate's public work; live ID checks; voice reference calls. | Raises the bar against deepfake candidates and DPRK insider placement. |
| Principle of least privilege from day one; two-person rule on production actions; commit rights separated from deploy rights. | Limits the blast radius of a phished or rogue identity. |
| Offboarding as a security event: rotate every secret the departing person touched (PATs, npm tokens, cloud IAM, SSH, multisig seats, VPN, Slack). | A former employee with stale publish rights is a live supply-chain risk. |
| A culture in which "let me check and come back" is encouraged, not punished. | Stops most attacks at Stage 5, the ask. |

# 11. Channel Triage

No single channel is correct for everything. Match the channel to the sensitivity.

| Channel | Use for | Never use for |
|---|---|---|
| Signal (verified safety numbers) | Sensitive operations, incident comms, high-trust one-to-one and small groups. | Anything before safety numbers are verified. |
| Slack / Teams (enterprise) | Team collaboration, internal records. | Secrets, signing material, treasury coordination in broadcast channels. |
| Email with SPF/DKIM/DMARC | Contracts, formal record. Layer S/MIME or PGP for sensitive bodies. | Authorising any high-value action. |
| Discord | Community engagement, public channels. | Anything signing-related. |
| Telegram | Casual chat only. | Keys, secrets, transaction signing, admin decisions. |

Sensitive decisions live in a named channel with verified members. The OpSec circle of trust is small and bounded: every additional person in the loop is an additional channel that can leak and an additional attack surface for the kill chain to cross.

# 12. Key Takeaways

- The kill chain has six stages (OSINT, pretext, contact, rapport, the ask, payload) and the earliest break is the cheapest. Treating unsolicited contact as hostile breaks it at Stage 3.
- Modern phishing is a multi-week rapport build, not a typo-ridden email. The language is fluent and the personalisation is automated.
- The fake job offer is the modern DPRK classic. Run unknown code only in disposable VMs; decline when in doubt. Axie/Ronin (approximately $625 million) started this way.
- Email authentication is SPF + DKIM + DMARC ramped to `p=reject`. DKIM replay bypasses naive deployments by re-sending legitimately-signed messages to new recipients.
- Cameras-on no longer proves identity. The Arup deepfake (February 2024, $25 million) retired that assumption. Challenge phrases, out-of-band callbacks, and dual signoff are the counter-play.
- Bybit (February 2025, $1.5 billion) shows that even careful humans lose when the tooling lies. Verify calldata independently; assume the signing UI may be the attacker.
- Slowing down is the single strongest defence. Attackers manufacture urgency because urgency wins.