THE CALCULATIONS · VERSION 1.2 · 29 OCTOBER 2026

How we came up with 2× safer

About 2× safer than a traditional server. Here is how we worked that out, every number with its source, so you can check it.

Dinanath Gupt · Omnira Inc. · with the arithmetic shown

Abstract. The same job is **about 2× safer on Omnira than on a traditional server**: it has about 2× fewer of the ways real breaches begin through the machine it runs on (1.6× to 3.3× across three published datasets). If an attacker already holds a machine, the job is exposed to it **N× less**, where N is the number of machines it can run on (about 10× on omnira.dev today). Omnira closes the machine's own doors: 0 inbound ports opened by Omnira, 0 logins to deploy or run, 0 job keys or copies on its disk. The doors in your application, your accounts, your storage and your people stay as they are. **It is not a million times.** Every number below has its source, so you can redo the arithmetic.

CONTENTS

1 · Context

2 · Scope — the two deployments compared

3 · Definitions — what "N×" measures

4 · Data — every input with its source

5 · Results

6 · Sensitivity analysis

7 · Limitations — what we do not say

8 · Method

References

About these calculations

1 · Context

Buyers ask for one number: is the same job 2×, 5×, 10× or a million times more secure on Omnira than on a traditional server? "More secure" is not a quantity, so we define two that are, compute both from published breach data and from what Omnira is proven or designed to do, and show the arithmetic. When we say the same job is about 2× safer on Omnira, we mean the first of the two: about 2× fewer of the ways real breaches begin through the machine it runs on. We do not report a number we cannot derive.

2 · Scope — the two deployments compared

A traditional server — a virtual machine on a public cloud, or a machine on premises — running one workload. It has a fixed address on a network. It opens ports so clients and operators can reach it. The code, the image, the keys in its configuration, its temporary files, logs, snapshots and backups sit on its disk. Operators log in to deploy and run. Whoever has root on it can copy any of it. Cloud providers close some of these doors by default and leave others to the customer; on premises, every one of them is the customer's.

The same job on Omnira. The job is streamed into the memory of a machine chosen for it. Nothing of the job is installed on or copied to that machine's disk. The job's keys are fetched for the run, scoped and short-lived, and never rest on the machine. Omnira opens 0 inbound ports on the machine for the job; the machine dials out to the platform. The job has 0 published addresses of its own. The next request may land on another machine. The job's data rests in the customer's own storage, not on the serving machine. Each of these is a claim with a status in Section 4.3 — proven live, by design, or a target — and the arithmetic uses only the first two, saying where a residual remains.

3 · Definitions — what "N×" measures

3.1 Measure one — the reachable surface

Every intrusion begins somewhere: a port left open, a stolen login, a key in a file, a copy on a disk, a person. Published breach reports (Section 4.1) say which beginnings are common. Call the share of breaches that begin through class d $\mathbf{w_d}$. Call $\mathbf{o_d}$ how open that class is in a deployment: 1 for the traditional server (its baseline), and for Omnira a residual between 0 (closed for the serving machine) and 1 (unchanged by Omnira).

The reachable surface is $\mathbf{S = \Sigma_d\, w_d \cdot o_d}$, and the multiple is $\mathbf{R_{surface} = S_{server} / S_{omnira}}$.

Two scopes are reported, because they answer two different questions:

- **Like for like — the server's own doors.** Only the classes that reach the job through the machine it runs on: exposed services and ports, remote-access credentials, secrets on the machine, copies and snapshots of its disk, insiders on the box. This is the buyer's question — *the same job on a traditional server*.
- **All beginnings.** Everything the reports count, including what Omnira does not change: a phished employee, a compromised supplier, a bug in the customer's own code. This is the upper bound on what any infrastructure change can do.

3.2 Measure two — exposure of the job to a machine an attacker holds

Suppose the attacker already holds a machine. On a traditional server, holding *the* server means holding the job and everything it ever wrote: $\mathbf{E_{server} = 1}$, for as long as the foothold lasts (median dwell times in Section 4.2). On Omnira a held machine sees the job only while the job runs there: $\mathbf{E_{omnira}}$ is the share of the job's requests that land on that machine, which over N eligible machines is of the order $\mathbf{1/N}$, and nothing is kept on its disk between requests. The

multiple is $\mathbf{R_{exposure} = E_{server} / E_{omnira} \approx N}$, with *N* the number of eligible machines in the placement pool — stated for the pool a customer actually has.

3.3 Putting them together

For a class *d* that lands on a compute machine, the chance that an intrusion through *d* reaches the job's data is $\mathbf{o_d \cdot e_d}$, with $\mathbf{e_d = 1}$ on the server and $\mathbf{e_d = E_{omnira}}$ on Omnira. For a class that reaches the data where it rests (the customer's own storage), $\mathbf{e_d = 1}$ on both, because Omnira does not change where the data rests. Then

$$\mathbf{R = \Sigma_d\, w_d \,/\, \Sigma_d\, w_d \cdot o_d \cdot e_d}$$

is the number at the top, for each scope, with every w_d, o_d and e_d listed in Section 4 with its source.

3.4 What the model does not do

It does not claim Omnira is unbreakable. It does not drop the hub — the one published door the platform keeps, hardened and operated once instead of on every server — from the count (Section 4.3 keeps it). It does not change the customer's own storage, people or code. And it does not count data in use — a job's memory on the machine while it runs — as closed; that residual is what E_{omnira} measures.

4 · Data — every input with its source

4.1 Where breaches begin (w_d)

Three independent datasets, each with its own denominator, so we compute the multiple on each and report the spread rather than blending them. All figures were read from the publisher's own page or report text on 26 October 2026. Bracketed numbers point into the references.

class *d*	Verizon DBIR 2025 — 12,195 breaches, Nov 2023 to Oct 2024; shares of breaches other than error and misuse [1]	Mandiant M-Trends 2025 — initial infection vector, 2024 investigations [3]	Sophos Active Adversary 2025 — 413 incident-response cases, 2024, root cause [4]
exploitation of an exposed service or vulnerability (open ports, a public address, edge devices)	**20%** (up a third on the year; edge devices and VPNs 22% of it)	**33%** (the first vector for the fifth year)	**22%** (external remote services present in 71% of all cases)

class *d*	Verizon DBIR 2025 — 12,195 breaches, Nov 2023 to Oct 2024; shares of breaches other than error and misuse [1]	Mandiant M-Trends 2025 — initial infection vector, 2024 investigations [3]	Sophos Active Adversary 2025 — 413 incident-response cases, 2024, root cause [4]
remote-access credentials stolen, brute-forced or reused (SSH, RDP, VPN, console)	**22%** (31% in the 2024 report [2])	**16%** stolen credentials, plus 8% prior compromise	**41%** compromised credentials, plus **21%** brute force
secrets left in code, configuration or images	no breach share is published; prevalence: 23.8 million secrets leaked on public GitHub in 2024 and 35% of private repositories hold one [8]; 14% of the 10,000 most-pulled public container images carry a hard-coded credential [10]; 8.5% of 337,000 images [11]	—	—
misconfigured or exposed copies: snapshots, backups, buckets, images	**12%** “cloud misconfiguration” [6]	—	—
insider or privilege misuse on the machine	**6.2%** (757 privilege-misuse breaches of 12,195; nine in ten by internal actors)	—	—
phishing and social engineering of people (not the server)	**about 15%**	**14%** e-mail phishing, plus 9% web compromise	18 to 19% malicious e-mail or phishing [5]
supply chain, third parties, the customer’s own code	third-party involvement doubled to 30% of breaches (overlapping)	—	15% third party or supply chain [7]

What the datasets do not say: no source gives a share of breaches that began from temporary files or logs on a disk, from installed code as such, or from a stolen snapshot as the first step. Those classes are counted here inside “exposed copies” and “insider”, and the model adds no share for them. Ransomware surveys (exploited vulnerability 32%, compromised credentials 23% [5]) are self-reported root causes and are shown for context only. Insurance claims and red-team data point the same way: stolen credentials began 47% of claims and software exploits 29% [12]; valid accounts gave 41% of successful initial access in US government red-team assessments [13].

4.2 Exposure in time

quantity	value	source
time to the first login attempt on a fresh cloud virtual machine with a public address	**52 seconds** (the fastest of ten regions; the slowest 1 h 45 min); every one of ten remote-desktop honeypots was attacked within its first day	Sophos, 2019 [14] [15]
login attempts against one exposed SSH port	**about 13 a minute, about 757 an hour** — about 18,000 a day; 5.4 million in 30 days across ten hosts, 95.6 percent of them as the root user	Sophos, 2019 [14]
exposed remote desktop: share brute-forced to compromise	**0.08%** of about 45,000 machines with a public RDP port and failed sign-ins; an attack lasts two to three days on average	Microsoft, 2019 [16]
how fast the internet scans a new exposure	scans for a newly disclosed vulnerability begin within **48 hours** (peer-reviewed, 2014) and, in one vendor's case, within **15 minutes** of a CVE (2022); about 20 sweeps of more than half the IPv4 space a day in 2014	Durumeric et al., 2014 [17]; Unit 42, 2022 [18]
cloud compromises that begin with weak or absent credentials, or misconfiguration	**47% / 29%** (first half of 2025); **27% / 21%**, with third-party software at 44.5% (second half of 2025) — one provider's customers, "a subset of observed activity"	Google Cloud Threat Horizons [19] [20]
median attacker dwell time	**10 days** (2023) · **11 days** (2024) · 14 days (2025)	Mandiant M-Trends 2024 to 2026 [3]
time to identify and contain a breach that began with stolen credentials	**292 days**	IBM, 2024 [6]
what a cloud virtual machine exposes by default	AWS: an instance in a default subnet gets a public IPv4 address; a disk snapshot can be shared publicly ("you are giving others access to all of the data on the snapshot"); the older instance-metadata service is reachable through a request-forgery bug, the mechanism of the 2019 Capital One breach (about 100 million people). Google Cloud: the default network ships rules that allow SSH (port 22) and RDP (port 3389) from anywhere, and a new instance gets an external address	AWS documentation [21] [22] [23]; US Department of Justice [24]; Google Cloud documentation [25]
moving-target defence, peer-reviewed results	address shuffling against k probes over n addresses: static $P = k/n$; perfect shuffling $P = 1 - ((n - v)/n)^k$, which for one vulnerable host and $k = n$ caps at $1 - e^{-1} \approx 0.63$ — a **37% reduction, no more**, when the attacker can keep probing. **Non-persistence**: the steady-state compromised fraction of a fleet is $\boldsymbol{\lambda_c / (\lambda_c + \lambda_r)}$ (the compromise rate over compromise-plus-reset), so resets every 10 minutes instead of every 2 hours take an adversary's share from 0.60 to 0.11 in the authors' example	Carroll et al., 2014 [26]; Donovan et al., MIT Lincoln Laboratory, 2015 [27]

quantity	value	source
why “one chance in millions per guess” is not a security bound	16 bits of address randomisation fell to **32,768 probes on average, 216 seconds** for a remote shell, because probes were cheap; periodic re-randomisation adds no more than one bit against brute force	Shacham et al., 2004 [28]

4.3 What Omnira closes (o_d, e_d)

Read from the platform’s own record of what it is built to do, what it has proven and what it has not, as of 26 October 2026. Status words: **proven live** = measured on a real machine with a record; **by design** = the code works this way and is tested, not separately measured in the field; **target** = a stated goal not yet met. The residual $\mathbf{r_d = o_d \cdot e_d}$ is what the arithmetic uses; the pessimistic value is used for the abstract and the optimistic one in Section 6.

class *d*	what Omnira does on the serving machine	status	residual r_d, pessimistic → optimistic	why not zero
exposed service or vulnerability	Omnira opens 0 inbound ports on the machine for the job; the machine dials out to the hub over TLS and verifies the hub; the only thing the runner listens on is the machine’s own loopback address, and it refuses any other; the job has 0 published addresses — the platform’s gateway is the one public door, and it relays bytes	by design (the loopback-only listener is tested; the checks that a device proves its identity on every connection are live in audit mode and not yet enforced)	**0.78 → 0.6**	the part of this class that is **the application’s own bug** — the main vector in the DBIR — still reaches the application through the gateway, wherever it runs; only the host-level part is gone (edge devices and VPNs, 22% of exploitation in DBIR 2025; unpatched operating-system services; brute-forceable daemons). A customer’s own native program may still open a port of its own on Linux today; the figure is for Omnira’s own surface
remote-access credentials	0 logins are needed to deploy or run; nothing on the serving machine accepts a login for Omnira; the owner publishes through the console	by design	**0.3 → 0.2**	the operator’s own account — console session, API key — can still be stolen or phished and used to republish, the same door as a cloud console; the machine-level logins (SSH, RDP, VPN to the box, brute force) are gone

class *d*	what Omnira does on the serving machine	status	residual r_d, pessimistic → optimistic	why not zero
secrets on the machine	no job keys or passwords rest on the machine: fetched for the run, scoped, short-lived; devices hold no credential for the customer's storage	by design (a fleet-wide fetch credential that every device used to receive was withdrawn on 10 October 2026)	**0.05 → 0**	no published breach share exists for this class, so it is folded into the credentials row; the residual is the copy of a secret that lives in the customer's own repository or configuration, which Omnira does not touch
copies, snapshots, backups	nothing installed, no image copy on the machine, no logs or temporary files of the job on its disk, nothing to snapshot; proven on Linux (0 of 134,236 files changed; 0 bytes of the job written) and refused at the kernel on the two platform machines that enforce it	proven live on Linux, on supported and verified paths; the hub machines themselves do not yet carry the kernel-level file wall	**0.7 → 0.5**	most "cloud misconfiguration" breaches are the customer's **storage** left open (public buckets); Omnira does not change where the data rests, only that no copy rests on the compute machine (public snapshots, images with secrets, logs on disk)
insider on the machine	nothing on the disk to copy; the hub chooses the machine for each request, so a held machine sees only the requests sent to it; **data in use — the job's memory — remains readable to the machine's administrator**, as the platform's security page says	by design (placement) · target (the platform's own operators still handle code and traffic in memory on the hub; locking them out is a stated goal, not a property today)	**1.0 → 0.1**	pessimistic: the platform's operators are insiders who see what the hub routes; optimistic: counting only the serving machine, a held machine sees about 1/N of the job's requests (Section 4.4)
phishing and people	unchanged	—	**1**	a phished operator is a phished operator
supply chain, the customer's own code	unchanged	—	**1**	the job's code is the customer's; a bug or a poisoned dependency runs wherever the job runs

The one published door. On a regular fleet every server has an address and ports; on Omnira only the gateway has them (the hub's dispatcher also answers on each hub's own cloud address today). It is counted as a host-level exposure that remains, once per platform rather than once per server; we add no share for it beyond the residual kept in the first row. **The one**

Omnira write. The runner saves its own sealed state on the machine; it is the one file Omnira writes, and it holds none of the job.

4.4 The placement pool (N) and how the job moves

What the code does, as of 26 October 2026: every request is placed by the hub over the devices that pass the service's requirements, in strict priority tiers, then **round-robin through a shared counter with a weighting for locality** — a device that holds the program warm in memory gets two slots to a cold device's one. It is **not random**: it is unpredictable only to someone who cannot read the counter, and we never call it random. A job therefore moves **per request**, not on a timer — more often than every second under load, and not at all while idle. Three exceptions, all counted against Omnira in the arithmetic: a service that asks for **one copy** stays on its holder; **hedged reads** are raced to up to five devices at once, so one read touches several machines; and the **program stays resident in memory** on a warm device between requests — on no disk, but in RAM — until evicted.

So for an attacker who holds one machine of a pool of N equally weighted machines, the share of a job's requests that machine sees is $\mathbf{E_{omnira} \approx 1/N}$ (up to 2/N if the held machine is warm and the others cold; up to 5/N for hedged reads). **N today:** omnira.dev's own fleet was between eleven and sixteen devices at the time of writing, so N ≈ 10; a Sovereign Omnira fleet is the customer's own count; the home page's **950 compute servers** is an illustration, labelled so on the page. The two figures on that page follow from it: a held machine sees about **1 request in 950** of each job (1/N with N = 950); and to *guess* the right machine at the right second for a one-second job is **1 in 82,080,000** — 950 × 86,400 seconds in a day — a probability under a model (uniform, independent, a guess rather than a hold), not a measured bound. We report it as such.

Persistence. On a traditional server a foothold persists: median attacker dwell time 11 days [3], 292 days to identify and contain a credential-based breach [6]. On Omnira 0 bytes of the job persist on the serving machine's disk between runs; what a held machine keeps is the program and its data in use, in memory, while resident. This is counted in E (a share of requests), not as a separate multiplier.

5 · Results

5.1 Measure one — the reachable surface

$R_{surface} = \Sigma\, w_d \,/\, \Sigma\, w_d \cdot r_d$, from the tables in Sections 4.1 and 4.3; the script is in Section 8.

dataset (denominator)	scope	server Σw	Omnira Σw·r, pessimistic → optimistic	$R_{surface}$
Verizon DBIR 2025 (12,195 breaches)	like for like: the server's own doors (exploit 20 · credentials 22 · copies 12 · insider 6.2)	0.602	0.368 → 0.230	**1.6× → 2.6×**

dataset (denominator)	scope	server Σw	Omnira Σw·r, pessimistic → optimistic	$R_{surface}$
Verizon DBIR 2025	all beginnings (plus phishing 15, third party 15)	0.902	0.668 → 0.530	1.35× → 1.7×
Mandiant M-Trends 2025 (initial infection vector)	like for like (exploit 33 · credentials 16 + 8)	0.570	0.329 → 0.246	**1.7× → 2.3×**
Mandiant M-Trends 2025	all beginnings (plus phishing 14, web 9)	0.800	0.559 → 0.476	1.4× → 1.7×
Sophos Active Adversary 2025 (413 cases, root cause)	like for like (exploit 22 · credentials 41 + brute force 21)	0.840	0.358 → 0.256	**2.3× → 3.3×**
Sophos Active Adversary 2025	all beginnings (plus malicious e-mail 18)	1.020	0.538 → 0.436	1.9× → 2.3×

Sophos root causes sum past one hundred percent because a case can have more than one; the ratio is unaffected.

Reading. Across three datasets with different denominators, a workload on Omnira presents **1.6× to 3.3× fewer of the ways real breaches begin through the machine it runs on — about 2×** — and 1.35× to 2.3× fewer of all beginnings. The spread is the datasets, not the model: the more a dataset counts remote-access and credential abuse (Sophos's incident caseload), the more Omnira removes; the more it counts application exploitation and people (the DBIR), the less.

What is closed outright, and what is not. Of the server-side beginnings in DBIR 2025, about half — 29 of 60 points — are the machine's own doors: exposed logins and brute force, host-level and edge-device exploits, copies and snapshots of its disk, an insider copying from it. Omnira closes those on the serving machine: 0 inbound ports opened by Omnira, 0 logins needed to deploy or run, 0 job keys on the machine, nothing on its disk to copy or snapshot. The other half is the application's own bug reached through the front door, the operator's own account, the customer's own storage left open, and the platform's own operators until they too are locked out — and none of that moves when the machine does. That is why the number is about 2×, not 10× and not a million.

5.2 Measure two — exposure of the job to a machine an attacker holds

$R_{exposure} \approx N$, against the traditional server whose one machine holds the job always ($E_{server} = 1$).

placement pool N	a held machine's share of the job's requests (E_{omnira})	with the held machine warm (2 slots)	hedged reads (up to 5 machines)	$R_{exposure}$
10 (omnira.dev's own fleet at the time of writing: 11 to 16 devices)	1/10	1/5	1/2	**about 10×**
100 (a mid-size private fleet)	1/100	1/50	1/20	**about 100×**
950 (the home page's illustration)	1/950	1/475	1/190	**about 950×**
10,000	1/10,000	1/5,000	1/2,000	about 10,000×

This is the home page's "1 in 950", with its assumptions written out: an even spread of requests over the whole pool (the code is weighted round-robin, two to one toward warm machines, in strict priority tiers), the attacker holding one machine, a service that is not pinned to one copy, and "keeps none of it" meaning the disk — the program and its data in use stay in that machine's memory while it is resident.

The third figure on the home page. To *guess* the right machine at the right second for a one-second job: 950 × 86,400 = 82,080,000, "1 in 82 million per guess". We report it with its model (uniform, independent, one guess per machine-second) and its limit: a probability per guess is not a cost to the attacker. Address randomisation with 16 bits fell to 32,768 probes and 216 seconds when probes were cheap [28], and an attacker who *holds* a machine is not guessing at all — that is the 1-in-N case above. The figure says how little a blind guess is worth, not how long a determined attacker needs.

5.3 A third lens — persistence, with a peer-reviewed formula

Donovan et al. [27] give the steady-state share of a fleet an adversary holds when hosts are compromised at rate λ_c and reset at rate λ_r: $\boldsymbol{\lambda_c / (\lambda_c + \lambda_r)}$. Their example, λ_c = 0.75 compromises per host-hour: resetting every two hours leaves the adversary 0.60 of the fleet; every ten minutes, 0.11. On a traditional server the job is never reset — a foothold lasts the median dwell time of 11 days [3], or 292 days when it began with stolen credentials [6]. On Omnira the job's **data** is on a machine for the length of a request, and its **program** for as long as the device stays warm. With the same λ_c — an assumption borrowed from the paper, not a measurement of Omnira — a program resident for ten minutes gives 0.11 (**5.4× less** than the two-hour reset); data present for one second gives 0.0002 (**about 2,900× less**). Reported as what it is: a published model applied with stated inputs.

5.4 Knocks per machine per day

A traditional server with a public address and SSH open receives about **18,000 login attempts a day** [14] — about 13 a minute, the first within 52 seconds — and a cloud virtual machine in a default network has that port open to the world [21] [25]. On Omnira the serving machine has

no listener for the job: those attempts have nowhere to land on it; the platform's gateway takes them, once for everyone, as the one published door. **18,000 → 0 per machine, per day**, for that class of attack. (A customer's own native program may still open a port of its own on Linux today; the figure is for Omnira's own surface.)

5.5 Conclusion

- **Reachable surface: about 2× fewer** of the ways real breaches begin through the machine (1.6× to 3.3× across the three datasets); 1.35× to 2.3× of all beginnings.
- **Exposure to a held machine: N× less**, N the placement pool — about 10× on omnira.dev's own fleet today, about 100× on a hundred-machine fleet, 950× in the home page's illustration.
- **The machine's own doors — ports, logins, keys on disk, copies, snapshots, an insider at the box — closed on the serving machine**; the application's, the account's, the storage's and the people's doors unchanged.
- **Not a million times.** No published breach data and no model in the literature supports it. We show where the big numbers come from — guessing games and N-sized pools — and what they do and do not mean.

6 · Sensitivity analysis

what moves	effect on the like-for-like R (DBIR 2025)
exploit residual 0.78 → 0.6 (more of the class counted as host-level)	1.64× → 1.81× (pessimistic insider)
credentials residual 0.3 → 0.2	1.64× → 1.74×
copies residual 0.7 → 0.5	1.64× → 1.75×
insider residual 1.0 (the platform's operators count) → 0.1 (only the serving machine counts)	1.64× → 1.93×
all four at their optimistic values	**2.6×**
the dataset: M-Trends or Sophos instead of the DBIR	1.7 to 2.3× / 2.3 to 3.3×
the pool halved (N = 5 on a small fleet)	$R_{exposure}$ 10× → 5×; one-copy services 1× (they do not move)
hedged reads on	a held machine sees up to 5/N of reads, not 1/N
device-identity enforcement switched on (in audit mode today)	no change to the arithmetic — admission is not a class above; it removes the residual of an unproven device joining the pool
the platform's operators locked out of what the hub handles	the insider residual's hub part falls away: the pessimistic column becomes the optimistic one

The result does not change under any single move; it moves from “about 2×” to “about 3×” only when every residual is taken at its most favourable value and the dataset that counts the most credential abuse is used.

7 · Limitations — what we do not say

- that Omnira makes a workload impossible to break into, or that any door is closed that the platform itself still marks a target;
- that the customer’s own storage, people, code or suppliers are safer because the job moved;
- that a figure here is a measurement of Omnira in the field — it is arithmetic over published breach data and the platform’s documented behaviour, and it says so wherever it is quoted.

8 · Method

- **Sources were fetched on 26 October 2026** from the publisher’s own page or report. Where a figure exists only as a chart in the primary and was read from a secondary summary, the table says so (IBM’s 12% cloud-misconfiguration share; the DBIR 2024 credential share, which summaries dispute — the 2025 report’s own restatement, 31%, is used). The breach-share datasets have different denominators (DBIR: breaches other than error and misuse; Mandiant: its own investigations; Sophos: incident-response root causes, more than one per case) and are never blended; each gives its own multiple.
- **The residuals r_d** are our judgement from the platform’s own record (Section 4.3), stated with their reason, with a pessimistic and an optimistic value; the abstract uses the pessimistic.
- **The arithmetic**, reproducible in a few lines:

```
w = {"exploit": 0.20, "creds": 0.22, "copies": 0.12, "insider": 0.062, "phi
r = {"exploit": (0.78, 0.6), "creds": (0.3, 0.2), "copies": (0.7, 0.5), "in
like = ["exploit", "creds", "copies", "insider"]
R = sum(w[c] for c in like) / sum(w[c] * r[c][0] for c in like)   # 1.64 (p
E = 1 / N                                                         # a held
Pc = lc / (lc + lr)                                               # Donova
```

- **Omnira’s own behaviour** is described from the platform’s documented invariants, its published proofs and the code as of 26 October 2026. The security page on omnira.dev describes the shipped links and what is planned but not yet done.
- **What we refuse to say**, by the platform’s own rules: that a serving machine has no ports at all (a native program may open one); that a service cannot read a machine’s files (not true on Linux today); that the platform’s operators can never see a job (a target, not met); the name of a trust model worn as a badge; “impossible to break into”; “100 percent secure”; “picked at random” (the placement is weighted round-robin); “0 bytes on the device, period” (the runner’s own sealed state remains); and any multiple without its measure and its assumptions beside it.

References

1. Verizon, *2025 Data Breach Investigations Report*. https://www.verizon.com/business/resources/reports/2025-dbir-data-breach-investigations-report.pdf — summary: https://www.verizon.com/about/news/2025-data-breach-investigations-report
2. Verizon, *2024 Data Breach Investigations Report*. https://www.verizon.com/business/resources/T6c3/reports/2024-dbir-data-breach-investigations-report.pdf
3. Google Cloud / Mandiant, *M-Trends 2025*. https://cloud.google.com/blog/topics/threat-intelligence/m-trends-2025/ — executive edition: https://services.google.com/fh/files/misc/m-trends-2025-executive-edition-en.pdf
4. Sophos, *Active Adversary Report 2025*. https://www.sophos.com/en-us/blog/2025-sophos-active-adversary-report
5. Sophos, *The State of Ransomware 2025*. https://www.sophos.com/en-us/blog/the-state-of-ransomware-2025
6. IBM, *Cost of a Data Breach Report 2024*. https://www.ibm.com/downloads/documents/us-en/107a02e94948f4ec — the 12% cloud-misconfiguration share is a chart value, read from https://10guards.com/en/blog/2024/07/31/ibm-average-breach-costs-hit-record-4-88m-in-2024-up-10-from-last-year — and https://newsroom.ibm.com/2024-07-30-IBM-Report-Escalating-Data-Breach-Disruption-Pushes-Costs-to-New-Highs
7. IBM, *Cost of a Data Breach Report 2025*. https://www.bakerdonelson.com/webfiles/Publications/20250822_Cost-of-a-Data-Breach-Report-2025.pdf
8. GitGuardian, *The State of Secrets Sprawl 2025*. https://blog.gitguardian.com/the-state-of-secrets-sprawl-2025/
9. GitGuardian, *Fresh from the Docks: uncovering 100,000 valid secrets in Docker Hub*, 2025. https://blog.gitguardian.com/fresh-from-the-docks-uncovering-100-000-valid-secrets-in-dockerhub/
10. Truffle Security, *Scanning for Secrets in Docker Images: A Deep Dive*, 2026 (served from a trufflesecurity.com subdomain at the time of writing). https://6ca3965f-6a12-417a-86b7-9a34473c43d1.trufflesecurity.com/research/secret-detection/scanning-secrets-docker-images-deep-dive — reported by https://www.techzine.eu/news/security/137233/docker-hub-contains-treasure-trove-of-10000-images-with-exposed-secrets
11. Dahlmanns, Sander, Decker, Wehrle, *Secrets Revealed in Container Images: An Internet-wide Study on Occurrence and Impact*, ACM ASIA CCS 2023. https://www.comsys.rwth-aachen.de/publication/2023/2023_dahlmanns_secrets-revealed-in-container/
12. Coalition, *Cyber Threat Index 2025*. https://www.coalitioninc.com/announcements/cyber-threat-index-2025
13. CISA, *FY23 Risk and Vulnerability Assessments: Analysis*. https://cisa.gov/sites/default/files/2024-09/FY23_RVA_Analysis_508.pdf
14. Sophos, *Exposed: Cyberattacks on Cloud Honeypots*, 2019. https://assets.sophos.com/X24WTUEQ/at/rgbjvgnx6qwwj7wvx764rmbn/sophos-exposed-

cyberattacks-on-cloud-honeypots-wp.pdf

15. Sophos, *RDP Exposed*, 2019, as reported: https://securitybrief.co.nz/story/sophos-highlights-dangers-of-remote-desktop-protocol-attacks
16. Microsoft Defender research, *Data science for cybersecurity: a probabilistic time series model for detecting RDP inbound brute force attacks*, 2019. https://www.microsoft.com/en-us/security/blog/2019/12/18/data-science-for-cybersecurity-a-probabilistic-time-series-model-for-detecting-rdp-inbound-brute-force-attacks/
17. Durumeric, Bailey, Halderman, *An Internet-Wide View of Internet-Wide Scanning*, USENIX Security 2014. https://www.usenix.org/system/files/conference/usenixsecurity14/sec14-paper-durumeric.pdf
18. Palo Alto Networks Unit 42, 2022, as reported: https://www.theregister.com/2022/07/27/palo_alto_unit_42/
19. Google Cloud, *Threat Horizons Report, H2 2025*. https://services.google.com/fh/files/misc/cloud_threat_horizons_report_h22025.pdf
20. Google Cloud, *Threat Horizons Report, H1 2026*. https://services.google.com/fh/files/misc/cloud_threat_horizons_report_h12026.pdf
21. AWS, *Default VPCs and default subnets*. https://docs.aws.amazon.com/vpc/latest/userguide/default-subnet.html
22. AWS, *Share an Amazon EBS snapshot*. https://docs.aws.amazon.com/ebs/latest/userguide/ebs-modifying-snapshot-permissions.html
23. AWS Security Blog, *Defense in depth: open firewalls, reverse proxies, SSRF vulnerabilities and the EC2 instance metadata service*. https://aws.amazon.com/blogs/security/defense-in-depth-open-firewalls-reverse-proxies-ssrf-vulnerabilities-ec2-instance-metadata-service/
24. US Department of Justice, *United States v. Paige Thompson*. https://www.justice.gov/usao-wdwa/united-states-v-paige-thompson
25. Google Cloud, *VPC firewall rules* (the default network's rules). https://docs.cloud.google.com/vpc/docs/firewalls
26. Carroll, Crouse, Fulp, Berenhaut, *Analysis of Network Address Shuffling as a Moving Target Defense*, IEEE ICC 2014. https://zenodo.org/records/1269979
27. Donovan, McLamb, Okhravi, Riordan, Wright (MIT Lincoln Laboratory), *Quantifying the Effectiveness of Non-Persistence*, IEEE HST 2015. https://web.mit.edu/ha22286/www/papers/HST15.pdf
28. Shacham, Page, Pfaff, Goh, Modadugu, Boneh, *On the Effectiveness of Address-Space Randomization*, ACM CCS 2004. https://hovav.net/ucsd/dist/asrandom.pdf
29. Manadhata, Wing, *An Attack Surface Metric*, IEEE Transactions on Software Engineering, 2011.
30. NIST SP 800-160 Volume 2, Revision 1, *Developing Cyber-Resilient Systems*.

About these calculations

Version 1.2, 29 October 2026. Dinanath Gupt, Omnira Inc., Fremont, California, US. Corrections and questions: support@omnira.dev. Report a vulnerability in Omnira to security@omnira.dev. The platform's own security page, which describes the shipped code link by link and what is planned but not yet done, is at omnira.dev/security/. These calculations will be revised as the platform's record changes; each version is kept beside the next. Version 1.2 (29 October 2026) presents them as what they are: how we came up with the number, laid out with an abstract and numbered sections. The words change; the arithmetic and every number do not. Version 1.1 (28 October 2026) changed only the wording of the opening: it says the same job is about 2× safer on Omnira than on a traditional server, and defines "safer" in the same sentence as the first measure. The arithmetic and every number are those of version 1.0 (26 October 2026), Both earlier PDFs stay on this page.

CITE THE CALCULATIONS

```
Gupt, D. (2026). How we came up with 2× safer: the calculations,
version 1.2. Omnira Inc. omnira.dev/whitepaper/how-much-safer/
```