

SCDO Laboratory · TECH PR · UNIT 03

Automated Contract Ledgers for AI Agents · Next-Gen Smart Payments

Public-Chain Atlas: Where Agents Land

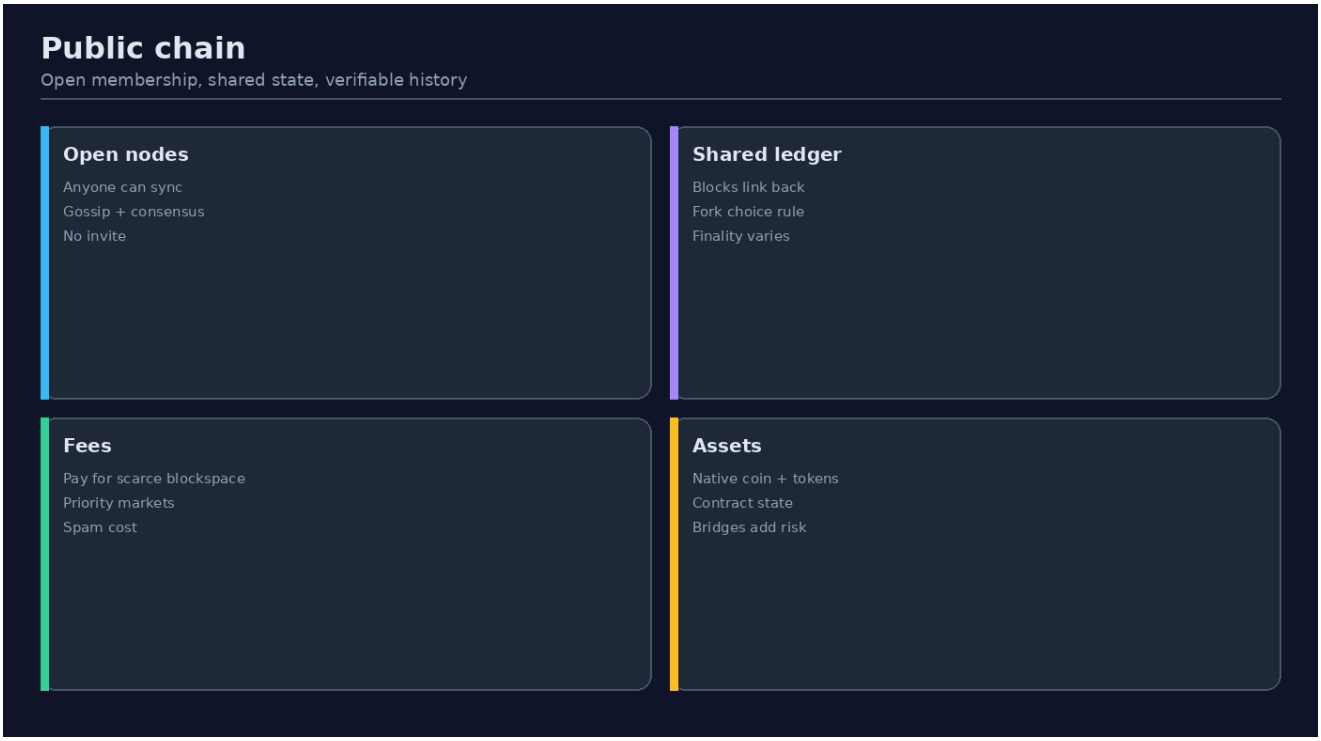
Mainnets, consensus, and ecosystem trade-offs · Ying Willam Zhang · 2026

Disclaimer: This primer is for education only — not investment, legal, or tax advice.

3. Public chains

SCDO Lab · Web3 Primer for Newcomers v2 · English edition
Chapter 2 framed Web3 as a “verify + self-custody” toolbox. This chapter covers the **operating system** under it: public blockchains.
Focus: what public chains are, why Ethereum matters, smart contracts and ERC standards, how the difficulty bomb became history, and how to read the 2026 L1 map.
No project promo lists. No multiple narratives. Not investment advice.

What is a public chain?



Plain definition

Public blockchain = an open network: in principle anyone can read the ledger, submit transactions, and run a node (or light client) under the rules—no single company may unilaterally say “today the ledger looks like this.”

Three common shapes:

Type	Who participates	Typical use
Public chain	Open by default	Bitcoin, Ethereum, etc.
Consortium chain	Invited institutions	Enterprise settlement, supply-chain pilots

Type	Who participates	Typical use
Private chain	One organization	Internal experiments, permissioned ledgers

When newcomers say “go on-chain,” they almost always mean a **public chain** (or a Layer2 under a public chain’s security—next chapter).

Is a public chain like an “OS”?

Common metaphor: public chain ≈ Web3 operating system—accounts, verification, transaction storage, and (if present) program execution; wallets, DeFi, games, and social grow on top.

Useful—but do not mythologize:

- Real OSes rarely rewrite consensus weekly; chain upgrades need community coordination and sometimes **hard forks**.
- “Open” ≠ “free and delightful”: Gas, congestion, bridges, and cross-chain risk exist.
- Security comes from cryptography + economic incentives + sufficiently dispersed validators/miners—not from a white-paper sentence.

Checklist of usual properties

When reading a new chain, ask:

1. **Openness:** permissionless read/write? how hard is running a node?
2. **Consensus and security budget:** PoW hashpower or PoS stake? how are attackers punished?
3. **State and execution:** transfers only, or general smart contracts?
4. **Native asset:** what pays fees? are inflation/deflation rules clear?
5. **Client diversity:** one team’s client only? single-point risk?
6. **Ecosystem and exit:** many apps? can users migrate assets if they dislike the rules?

No perfect score. Bitcoin is extreme on simple/conservative/hard-to-change; Ethereum is extreme on programmable ecosystem; many new L1s push speed and fees harder—and often have shorter battle history on decentralization.

What is Ethereum?



From Bitcoin to a “programmable world computer”

Bitcoin showed scarce digital assets can move on a global ledger without a bank.
Its scripting is deliberately weak—more like a single-purpose calculator.

Ethereum’s core idea: a **distributed computer**—nodes jointly maintain state; anyone can deploy programs (smart contracts) that execute under rules. Native coin **ETH** pays Gas and, in the PoS era, is staked.

Teaching shorthand (not an official hierarchy):

- Bitcoin ≈ blockchain 1.0 (digital cash / settlement asset)
- Ethereum ≈ blockchain 2.0 (smart-contract platform)

Much value and innovation still sits on Ethereum and its Layer2s; other L1s compete on performance, languages, modularity, and more.

Vitalik and early story (minimal)

Ethereum was advanced by **Vitalik Buterin** and others: ideas and crowdfunding around 2013–2014; **2015** mainnet launch. Skip gossip. Remember: it aimed at “programmable” from day one—hence the later tree of ICOs, DeFi, NFTs, L2s (good and bad).

Versus Bitcoin (2026)

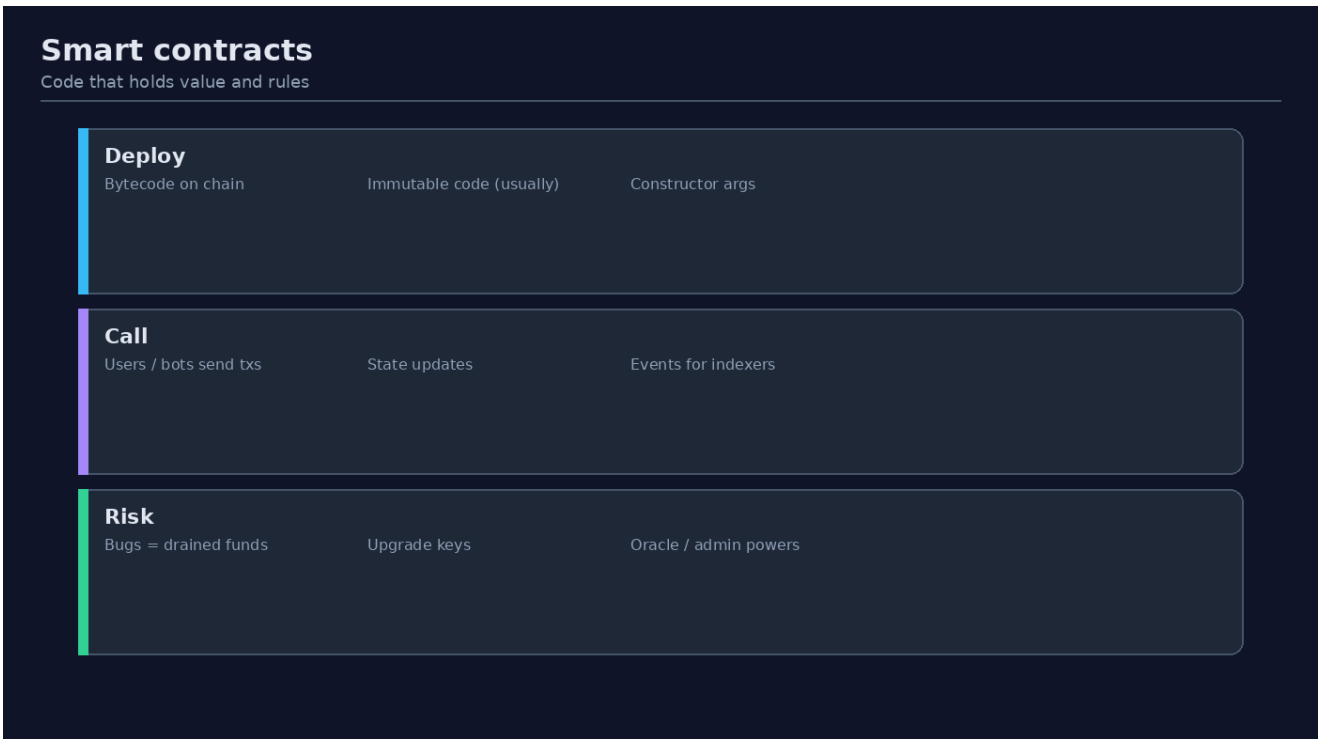
	Bitcoin	Ethereum (2026)
Main role	Robust digital scarcity and settlement	General contract platform + settlement layer
Consensus	Still **PoW** mining	**PoS after The Merge (2022-09)**
Programming	Limited script	EVM contracts (Turing-complete, Gas-bounded)
Accounts	UTXO	Account balances + contract accounts
Fees	BTC fees	ETH Gas (can be high; many users move to L2)
Upgrade pace	Extremely conservative	Roadmap exists; still needs broad consensus

Critical fact: Ethereum completed **The Merge**. Mainnet no longer mines ETH with GPUs/ASICs; validators stake ETH. Treat “Ethereum miners GPU-mine ETH” or “the difficulty bomb is forcing migration now” as outdated.

What is the EVM?

EVM (Ethereum Virtual Machine) executes contract bytecode. Many other chains claim “EVM compatibility”: similar tools/languages (e.g. Solidity), familiar addresses and wallets. Compatibility lowers migration cost; it does **not** automatically equal Ethereum-grade security or decentralization.

What are smart contracts?



Definition

A **smart contract** is an on-chain program: when a transaction triggers it and conditions hold, it updates state by code (transfers, minting, permissions).
“Smart” is not AI—it means **automatic, verifiable execution**. “Contract” does not automatically equal legal force—that depends on jurisdiction and extra arrangements.

Plain examples:

- Crowdfund: release funds only if the goal is met; else refund.
- Swap: exchange token A for B by formula (classic DeFi).
- Credential: mint a unique NFT as a ticket.

Once deployed, code is usually transparent (whether anyone reads it is another matter). Upgrade proxies and admin keys weaken “immutability”—always ask: **who can change the rules?**

Powers and pits

Powers: less dependence on a company manually executing the backend; strangers can coordinate under public rules.
Pits:

- Bugs get exploited (history is long);
- Bad oracle prices blow up systems;
- Wrong allowances hand over the keys;
- Expensive Gas or congestion makes great contracts unusable—why L2s rose.

Smart contracts turn public chains into app platforms—and why security needs its own chapter.

ERC-20 and ERC-721: why standards?



ERC (Ethereum Request for Comments) are community interface standards so wallets, exchanges, and explorers know how to show balances and transfer—without custom software per project.

ERC-20: fungible tokens

- Units are interchangeable: like one dollar bill, or game gold.
- Interfaces cover name, symbol, supply, transfer, allowance (`approve`/`transferFrom`), etc.
- Uses: governance tokens, stablecoins, points, LP receipts, and more.
- **Allowances are powerful and dangerous:** after you approve a contract, a malicious one may drain tokens. Manage allowances; never blind-sign.

ERC-721: non-fungible tokens (NFTs)

- Each tokenId is **unique**: like a title number, a concert seat, a collectible.
- Uses: art/avatars, tickets, memberships, game items, achievements...
- NFT ≠ “must be a valuable picture”; it is an **on-chain unique credential**. Value depends on rights, community, and liquidity—most go to zero or ignore; that distribution is normal.

Quick contrast

	ERC-20	ERC-721
Fungibility	Homogeneous	Each distinct
Balance meaning	Quantity	Which IDs you hold

	ERC-20	ERC-721
Typical pitfalls	Inflation, allowance drains	Fake floors, phishing mints

ERC-1155 mixes fungible and non-fungible in one contract; beginners can hold 20 and 721 first. Other chains have analogous standards under different names.

Appendix: Ethereum’s “difficulty bomb” — finished history

Some old articles still write the bomb in present tense as if The Merge never happened. **Treat this as history class.**

What it was

The **Difficulty Bomb** in PoW-era Ethereum made mining difficulty rise exponentially after certain epochs, slowing blocks until PoW mining became uneconomic. One goal: **force migration to PoS**, reduce endless delay, and lower the appeal of stubbornly staying on the old chain.

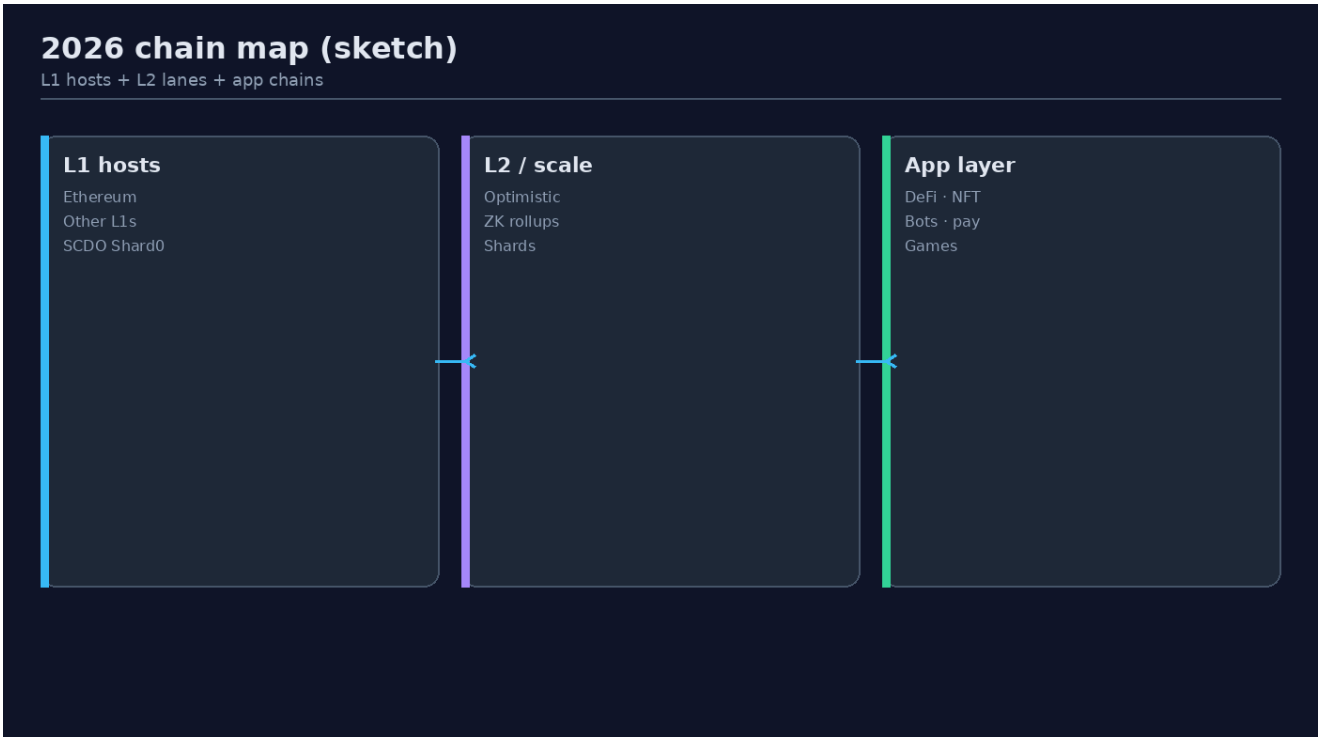
In practice the bomb was **delayed** many times to match the roadmap; hard-fork parameter tweaks were normal—not a sci-fi “detonate and destroy the network” plot.

How it ended

September 2022, The Merge: execution layer joined the beacon chain; consensus became **PoS**. Mainnet PoW mining ended; the bomb’s job as a migration stick was essentially over. PoW chains that remain are **other networks** (e.g. certain fork narratives)—do not confuse them with today’s Ethereum mainnet.

When you read 2017–2021 pieces about “defusing the bomb” or “Ice Age,” file them as archives—**not current ops guides**.

Innovation directions and a 2026 map (selective)



We do **not** reprint "watch sixty new chains / 100× bull lists." Keep **map coordinates**, not a recommendation list.

Ethereum vs new L1s: competing trade-offs, not one fill-in-the-blank

- **Ethereum path:** L1 for security and settlement; much activity on **Layer2**; deep ecosystem, mature tools, long battle testing.
- **Independent L1 path:** new VMs, consensus, higher throughput or lower fees; must cold-start liquidity and security reputation.
- **EVM-compatible chains:** lower migration cost; differentiate via niches—judge security and decentralization case by case.

There is no single "next Ethereum." More often: **multi-chain coexistence + bridges and intent routers**—treat bridge risk as high by default.

Layer 1 / 2 / 3 (intuition)

- **Layer1:** base chain with its own consensus and security root (Bitcoin, Ethereum, other independent public chains).
- **Layer2:** scales under L1 data/security (especially Ethereum Rollups)—next chapter.
- **Layer3:** more application-specific or recursive scaling talk; definitions still evolve—know "further layers exist."

Technical routes (name-level)

1. **High-performance / parallel L1s:** low latency, high TPS; often higher hardware/node requirements.

2. **Move and asset-oriented languages:** resource-type safety to reduce some asset bugs (language is not a silver bullet).
3. **Modular stacks:** split execution, settlement, data availability—flexible; complexity and assumptions rise.
4. **Vertical app chains:** games, DeFi, privacy—liquidity and cold-start remain hard.
5. **Bitcoin side ecosystems:** inscriptions, L2 experiments—different exam from “smart-contract L1s,” same broad universe.

How to look in 2026 (not a 60-name list)

Ask questions instead of memorizing lists:

- Where do money and users mainly settle?
- Are new apps more often on L1 or L2?
- Are clients and validator/miner distributions healthy?
- Whom do you trust when bridging?
- Do regulation and stablecoin rails affect real use?

SCDO Lab, lightly: the endgame is less “one chain left” and more **verifiable records as default infrastructure**—winners fit security, fees, developer experience, and real business. We care whether actions are auditable and permissions constrained (Agent Ledger-class questions), not which narrative is loudest this year.

A ten-minute hands-on with a public chain

Theory sticks better after one touch (use tiny amounts you can lose, or official testnets):

1. Open the chain’s **block explorer**; find one of your txs: success/fail, Gas, from/to.
2. Open a popular token contract page: holders, verified source, upgrade proxy?
3. Ask: am I on **mainnet L1** or some **L2/sidechain**? what unit prices fees?
4. If bridging: assume the bridge can fail; understand lock/mint before large amounts.

Finish those four and you are closer to a real user than someone who only forwards “chain ranking” screenshots.

Chapter wrap

1. Public chains are open shared ledgers and (often) programmable execution—the Web3 base.
2. Ethereum mainstreamed smart contracts; **it is PoS**; much activity is on L2.
3. ERC-20 / ERC-721 let wallets and markets share token semantics; allowances and speculation risks remain.
4. The difficulty bomb is a PoW→PoS historical device, not 2026 news.

5. Do not memorize mega lists; read maps with openness, security, execution, and ecosystem questions.

Next: **Layer2**—why Ethereum stacks another layer, sidechain vs Rollup, and short cards for Optimism / Arbitrum / ZK families in 2026—instead of an encyclopedia.

(End of Chapter 3 · v2 English edition · SCDO Lab)