

Mesh™
by BAREMAX

TECHNICAL WHITEPAPER / DEVELOPMENT DRAFT

Understand what can hurt you.

Mesh: a Solana-native security sentinel.
Token and position analysis, evidence principles and the path toward wallet-aware security.

Nicco Dabrowski
Founder, BAREMAX

Draft v0.2

Revised November 7, 2026

Released beta, reviewed candidates and planned capabilities are distinguished.

This document describes scope and evidence; it does not guarantee safety.

Contents

Start with the founder’s introduction, or use the chapter links to go directly to the evidence model, current implementation or limitations. Page numbers refer to this edition.

Abstract 4

1. Introduction 5

2. Design Philosophy 8

3. System Architecture 13

4. BAREMAX Risk Domains 21

5. The BAREMAX Evidence Model 27

6. Token-2022 Security 34

7. Liquidity and Execution Risk 41

8. The BAREMAX Mesh 49

9. Security Against False Confidence 57

10. Current Product Experience 60

11. Future Wallet Integration 67

12. Threat Model 75

13. Limitations 81

14. Roadmap 89

15. Conclusion 97

Appendices 98

Appendix A - Evidence-State Semantics 99

Appendix B - Authority and Controller Analysis 100

Appendix C - Liquidity and Exit Modeling 101

Appendix D - Token-2022 Validation Model 103

Appendix E - Historical and Creator-Evidence Coverage 105

Appendix F - Ownership Evidence and Concentration 106

Appendix G - Program Upgradeability 107

Appendix H - Risk Interpretation and Consumer Labels 108

Appendix I - Methodology Versioning and Reproducibility 109

Appendix J - Scope of the Whitepaper 110

References 111

Abstract

What's changed since October 28. This paper keeps the October 28 release as its implementation baseline. Since then, the beta has added a risk-card summary with key stats; a holder count from CoinGecko market data and 24-hour buy and sell counts from DEX Screener, both shown as market context rather than risk inputs; a note when a token's main market trades on a venue Mesh does not yet model; evidence hardening for launch-buyer holdings, token age and bonding-curve read slots; Cloudflare Access verification enforced at the application origin; a plain "not a token mint" response for wallets, programs and missing accounts; and venue logos beside the modeled pool's name. Private testing is now open to anyone who signs in with an email address, and the public beta is still planned for December 27, 2026. Risk grades, thresholds and evidence rules are unchanged, and statements dated October 28 below describe that release.

Mesh is BAREMAX's Solana-native security and risk-analysis system. It helps a trader understand who can change a token's behavior, how supply is held, what a position could receive through supported liquidity, and which parts of the evidence remain unresolved. Its purpose is to make consequential onchain conditions understandable before a user commits to an interaction.

The system keeps observations, evidence quality and risk interpretation distinct. An active authority establishes a capability; it does not establish malicious intent. A missing lookup establishes a limitation; it does not establish safety. Each conclusion should remain traceable to the accounts, programs, amounts and assumptions that support it.

As of October 28, 2026, Mesh has a protected, read-only beta with six risk domains, a token-control console, position-sized exit estimates, and inspectable supporting evidence. Authenticated exit modeling is limited to supported PumpSwap and Raydium CPMM paths. Token metadata and market history are acquired separately and do not certify authenticity, executable prices or current security state. Section 10 identifies the released baseline and distinguishes it from unmerged work.

This paper also describes the longer-term architecture: wallet-aware security, persistent monitoring, supported pre-sign inspection and user-selected protection. Those capabilities require separate integration, validation and release gates. Mesh does not take custody, execute trades or guarantee that an asset can be sold. The aim is to reduce the work required to understand supported security risks while keeping uncertainty visible.

1. Introduction

Around Christmas in December 2024, I invested approximately $10,500 in a newly released token on the Solana blockchain. That money was my life savings, built through a variety of jobs throughout high school and college. I remember seeing a verification check on Pump.fun and endorsements from several prominent voices on Crypto Twitter. At the time, those signals gave me confidence in the project. Within about two hours, the position was effectively worthless.

Before buying, I had also used a token-checking service. I thought I had taken reasonable steps to understand the risk, yet I still did not have a clear picture of what could happen to the money I was putting in. Losing those savings that quickly was devastating. What stayed with me was the gap between the signals I trusted and the risks I had not understood.

That trade was part of a much longer learning process. Over several years, I traded hundreds of tokens through decentralized exchanges, mainly on Solana and, earlier, on BNB Chain. Across that period, my cumulative result was approximately negative $45,000. I made poor decisions, took speculative risks and experienced ordinary market losses. I take responsibility for that. It would be misleading to describe all of those losses as scams or technical failures.

What I kept returning to was the amount a trader is expected to understand before making a decision. Price and position size are visible. The authorities that can change a token, the relationships behind its supply, and the liquidity available for a particular exit are much harder to interpret. A quick check can surface useful facts without explaining how those facts fit together or what the tool could not verify.

That is why I started building BAREMAX and Mesh. I want someone researching a token to understand what can affect their position, why it matters and how much confidence the evidence supports. The first answer should be clear enough to use. The supporting detail should still be there for anyone who wants to inspect it.

Mesh begins with token and position analysis. It examines controls, creator observations, ownership, transfer conditions and supported exit liquidity, then makes the findings and their limits available together. The longer-term goal is to bring that context closer to the wallet, including supported monitoring and pre-sign inspection. Each step has to earn its place through evidence and validation.

I cannot promise that Mesh would have prevented my December loss. I can build around the question that experience left me with: what could hurt a user here, and what evidence do we actually have?

> **We don’t tell you what to buy.**
> **We tell you what can hurt you.**

Nicco Dabrowski · Founder, BAREMAX

About the account above. The amounts, timing, verification indicator and endorsements are described as I remember them. My understanding was that the December collapse involved a bundle, but I have not reconstructed the transaction history to establish the mechanism. I cannot reliably identify the separate token-checking service from memory. This account makes no allegation against a named platform.

1.1 Why Solana

Solana is the environment where I have spent most of my recent trading experience and where I want Mesh to become useful first. That focus is also an engineering decision. The account model, token programs, authority relationships, transaction structure and liquidity venues need to be understood on their own terms.

The original Token Program and Token-2022 support different kinds of behavior. Token extensions can introduce transfer fees, delegates, hooks and other controls that are easy to overlook in an ordinary asset view. Evaluating them properly requires more than attaching the same generic checks to another network.[7][8]

Solana’s low base transaction fees and active application ecosystem make it a practical place to begin.[1] Its uses also extend beyond speculative trading: the dated sources in this paper describe stablecoin settlement and tokenized investment activity.[2][3] Those developments support the case for clearer security information around programmable assets; they do not establish that Solana will dominate every financial use case.

The priority is depth within one network. Support for another chain would require its own evidence model, protocol validation and operational standards. It should follow demonstrated utility on Solana.

1.2 From a Scan to a Security Layer

A token scan is a useful starting point. Tools such as RugCheck and SolSniffer have made important onchain facts more accessible. Mesh should be judged by how accurately and clearly it explains the risks it supports, rather than by the number of indicators it can display.

The relationships between findings matter. An active mint authority and an active freeze authority describe different powers. If the same recorded address holds both, that shared control is another observable fact. It still does not identify a person or prove that either power has been used maliciously.

A similar distinction applies to liquidity. A pool can have a substantial headline balance while offering poor modeled proceeds for a particular position. A price chart can describe a market that Mesh does not yet support for authenticated exit modeling. The interface must make those boundaries understandable.

Over time, I want Mesh to retain useful context about a user-selected wallet and surface material changes without asking the user to reconstruct the same investigation repeatedly. Monitoring and pre-sign protection remain later stages, dependent on supported integrations and stronger validation. The current beta does not provide universal wallet protection.

1.3 What This Paper Covers

The paper moves from the product’s evidence principles to its architecture, risk domains and technical limits. Section 10 provides a dated account of the released product and work that remains outside it. Sections 11 and 14 describe future integration and planning targets. The appendices retain deeper methodological detail.

A reader should be able to distinguish three things throughout: what Mesh currently returns, what a reviewed development candidate adds, and what the architecture is intended to support later. A working prototype or green test suite does not, by itself, move a capability into the released product.

The same distinction applies to terminology. The current engine uses domain-specific labels including LOWER, ELEVATED, HIGH and UNKNOWN. Later passages discussing SAFE describe a proposed, scope-bound research vocabulary. This revision does not adopt SAFE as a replacement for LOWER or change the engine’s scoring policy.

2. Design Philosophy

Reading the architecture. Later sections retain proposed security semantics and roadmap targets. Current deployment scope is defined in Section 10; future capabilities and proposed SAFE terminology are not release claims.

Mesh is being designed around a security philosophy that prioritizes evidence quality, explicit uncertainty, explainability, and practical usefulness to the person controlling the wallet. These principles are not separate from the technical implementation. They determine how data is acquired, how failures are interpreted, how risk conclusions are produced, and how those conclusions should eventually influence wallet protection.

The central objective is to avoid two common failures in security analysis: overstating what the available evidence proves and overwhelming the user with technical information without translating it into a useful conclusion. Mesh is intended to operate between those extremes. It should preserve the complexity necessary to make a defensible security determination while presenting that determination at a level appropriate for the user.

2.1 Evidence Before Conclusions

Every security conclusion produced by Mesh should begin with the evidence supporting it. The analytical process can be reduced to three questions: what evidence was observed, how reliable and complete that evidence is, and what conclusion that evidence actually justifies. This ordering is deliberate. Mesh should not begin with a desired classification and then interpret incomplete observations in whatever way produces the cleanest result.

A failed lookup must not be treated as evidence that a risk is absent. Section 5 formalizes this distinction across authority state, historical coverage, liquidity, and program mutability; each requires a validated observation before a favorable conclusion is justified.

Mesh therefore separates observation from interpretation. Evidence must first be acquired and structurally validated before it is allowed to influence a security conclusion. This approach also reflects established secure-software-development principles. NIST's Secure Software Development Framework recommends integrating security practices throughout the software development lifecycle, reducing vulnerabilities before release, and addressing root causes so discovered weaknesses are less likely to recur.[6]

2.2 Uncertainty Is an Explicit Security Outcome

Security systems are often pressured to produce simple answers even when the available information does not justify them. Mesh is intentionally designed to preserve uncertainty when necessary rather than forcing every observation into a positive or negative conclusion.

Within that model, UNKNOWN is a legitimate security state. It does not mean that an asset or interaction is safe, and it does not mean that it is dangerous. It means that Mesh was unable to acquire or validate enough material evidence to support a stronger conclusion within the relevant domain. The reason for that uncertainty should also be available to the user, whether it originated from incomplete historical coverage, unsupported program behavior, unavailable account state, failed classification, or another unresolved condition.

This does not mean that Mesh should default toward excessive caution whenever any information is unavailable. Not every missing datapoint is material to every security decision. The system should instead determine whether the unresolved evidence could reasonably change the conclusion. The objective is to preserve meaningful uncertainty while progressively reducing unnecessary UNKNOWN states as evidence acquisition, protocol coverage, and validation improve.

2.3 SAFE Must Be Earned

Mesh is intended to be capable of returning a genuine SAFE result. However, SAFE should mean substantially more than the absence of an obvious warning. A positive security conclusion should require the material checks supported for the relevant interaction to have been resolved with sufficient evidence and no significant supported risk condition remaining.

The burden of proof is intentionally asymmetric: a validated dangerous condition may justify a warning while unrelated evidence remains unresolved, whereas SAFE requires sufficient resolution across the material supported checks. Section 5.3 develops the implications for overall interpretation.

SAFE therefore remains scope-bound. It represents a high-confidence conclusion within the evidence, capabilities, and conditions Mesh was able to validate at the time of analysis; it does not represent an absolute guarantee against exploitation, future state changes, unknown vulnerabilities, device compromise, or unsupported behavior. The proposed consumer vocabulary is SAFE, CAUTION, HIGH RISK, and UNKNOWN, with UNKNOWN representing insufficient evidence rather than an intermediate level of danger. Existing engine terms such as LOWER, ELEVATED, and HIGH remain distinct from that proposed presentation. In particular, LOWER must not be renamed SAFE without a separately validated evidence-eligibility policy; Appendix H explains this boundary.

2.4 Security Should Degrade Toward Uncertainty

A fundamental design requirement for Mesh is that losing material evidence about an otherwise unchanged state, under the same methodology, should not make an interaction appear safer. If a system initially observes an active authority and subsequently loses access to the information necessary to classify its controller, the resulting analysis should become less certain. It should not improve simply because part of the evidence disappeared.

This rule applies across domains and must remain distinct from a genuine improvement in blockchain state. A verified change can reduce risk; an acquisition failure cannot create reassurance by removing inconvenient evidence. Section 5.6 defines the freshness and state-change qualifications.

At the same time, uncertainty in one domain should not erase risk that has already been established elsewhere. Mesh should preserve positively identified findings while separately identifying unresolved evidence. This distinction allows the system to remain decisive where the evidence is strong without pretending that unrelated uncertainty does not exist.

2.5 Programmability Requires Verification

Solana's token architecture increasingly supports functionality that extends well beyond conventional token transfers. The Token Extensions Program, commonly referred to as Token-2022, provides optional mint and token-account extensions for capabilities including transfer hooks, permanent delegates, non-transferability, transfer fees, and other specialized behavior.[7]

BAREMAX does not treat the presence of programmable functionality as evidence of malicious intent. A transfer hook may support a legitimate application. A transfer fee may implement an intentional economic model. A permanent delegate may exist for a valid operational purpose. The relevant security questions are what those capabilities permit, who controls them, whether that control can change, how the capabilities interact with one another, and what they mean for the wallet attempting to use the asset.

The Mesh philosophy is therefore to increase verification as programmability increases. Additional functionality creates additional state that must be understood; it does not automatically make an asset dangerous. This principle is particularly important for Token-2022, where technically valid functionality can materially affect a user's expectations about transferability, control, fees, or execution.

2.6 Practical Security for Real Users

The intended user of Mesh is not necessarily a protocol engineer. A technically correct security system can still fail as a product if the information it produces cannot be understood at the moment a decision must be made. BAREMAX is therefore being designed around progressive disclosure rather than a single technical interface for every user.

The basic view should explain the conclusion, primary reasons, and material uncertainty. Expanded domain views and the underlying technical record should remain available to users who need them. Section 10 describes how that progression works without creating different analytical standards for beginners and experienced users.

The amount of technical information available should therefore increase with the depth requested by the user, without forcing every user to interpret the deepest layer by default. As Mesh becomes more integrated with wallets, this principle becomes even more important: the depth of security analysis should increase while the amount of manual security work required from the user decreases.

2.7 Explainability by Design

Mesh should not depend on an opaque score that users are expected to trust without understanding how it was produced. Every material conclusion should be explainable through the evidence that caused it.

A HIGH RISK authority conclusion should identify the relevant authority or concentration of privileged control. An elevated liquidity conclusion should explain the modeled execution conditions responsible for it. An UNKNOWN program-upgradeability result should explain which evidence could not be established. The same standard applies in the opposite direction: a SAFE conclusion should be supported by resolved checks rather than presented as an unexplained green indicator.

Numeric scoring may still be useful for internal normalization, product organization, or certain interfaces, but the number itself should not replace domain-level reasoning. The user should be able to move from the conclusion to the evidence supporting it. This is especially important for a product intended eventually to interrupt or warn against transactions, because increasingly consequential protective actions require increasingly clear explanations.

2.8 Objective Evidence, Personalized Protection

The facts observed by Mesh should remain independent of the user's personal risk tolerance. If a token retains an active freeze authority, that authority exists regardless of whether an individual user is comfortable accepting the condition. If a position is modeled to incur substantial exit impact, the modeled result should not change because the user has chosen a more aggressive risk profile.

What can change is the protective response. As Mesh develops, users should be able to determine how the system reacts to objective security conditions. One user may choose to receive a warning when an active freeze authority is detected, while another may configure Protection Mode to interrupt the interaction. One user may tolerate a modeled exit impact of 10%, while another may establish a lower threshold.

This separation between evidence and response is essential to the long-term architecture. Mesh should determine what is happening onchain and what security implications can be supported by the evidence. The user's protection policy can then determine what action the system takes in response. Personalization should alter the boundary at which Mesh warns or intervenes; it should not alter the underlying facts.

2.9 Development Discipline

These principles also influence how BAREMAX Labs develops the system itself. Security functionality should be introduced only when the underlying evidence handling is sufficiently reliable to support it. Expanding the number of protocols, extensions, venues, or risk signals is not useful if broader coverage weakens confidence in the conclusions produced.

Malformed inputs, conflicting observations, provider failures, incomplete historical coverage, unsupported behavior, and boundary conditions should be treated as expected engineering cases rather than exceptional circumstances. When a failure is discovered, the preferred response is not merely to patch the individual case but to understand why the system permitted the failure, add regression coverage, and strengthen the relevant evidence contract. This development approach is consistent with the

secure-development emphasis on reducing vulnerabilities, addressing their root causes, and incorporating lessons from discovered weaknesses into future software behavior.[6]

For BAREMAX Labs, this means that product expansion should remain constrained by the reliability of the underlying security engine. The system can become broader only where its ability to validate, explain, and safely degrade under failure expands with it.

2.10 Design Standard

Taken together, these principles define the standard Mesh is intended to follow. The system should remain conservative when material evidence is unresolved without becoming unnecessarily alarmist, decisive when dangerous conditions are established, capable of producing positive security conclusions when those conclusions have been earned, and transparent about the evidence supporting either outcome.

The objective is not to maximize certainty in the interface. It is to ensure that the level of certainty presented to the user is justified by the evidence underneath it. That distinction forms the foundation for the system architecture described in the following section.

3. System Architecture

BAREMAX Labs is developing Mesh as the security architecture underlying the BAREMAX product experience. BAREMAX is the product brand, and BAREMAX Labs is the working name of the development initiative led by Nicco Dabrowski; it is not a registered legal entity as of this draft. Mesh refers to the interconnected analytical and protective system being built around the wallet. These names describe the project and its intended architecture, not an existing corporate structure or a guarantee that every proposed capability has been released.

The architecture is based on the premise that onchain security cannot be reduced reliably to a collection of independent checks. A Solana interaction may simultaneously involve token authorities, executable programs, mutable account state, ownership concentration, liquidity conditions, Token-2022 extensions, historical creator behavior, transaction instructions, and the wallet's existing exposure. Solana's execution model reinforces this interconnected structure: network state is stored in accounts, programs operate on accounts through instructions, and one or more instructions are composed into an atomic transaction.[8]

Mesh is therefore organized as a sequence of analytical layers in which evidence is acquired, structurally validated, interpreted within specialized security domains, evaluated for completeness, and ultimately translated into a user-facing security conclusion. Each layer has a distinct responsibility so that incomplete or malformed information at one point in the system cannot silently become an unjustified conclusion later in the pipeline.

3.1 The Mesh Architecture

At a high level, Mesh follows the progression onchain evidence → structural validation → domain analysis → evidence quality → risk interpretation → user protection. The purpose of this structure is to keep evidence collection, technical validation, security interpretation, and protective action conceptually separate even though they operate as parts of the same system.

The acquisition layer determines what information is available. Structural validation determines whether that information can be trusted and interpreted according to the expected account, program, transaction, or token structure. Domain analysis determines the security meaning of the validated evidence. Evidence quality establishes whether the available information is sufficiently complete to justify the proposed conclusion. Risk interpretation then combines those findings into a result that can be presented to the user or, in later versions of Mesh, compared against the user's Protection Mode policies.

These distinctions are important because a successful RPC response is not automatically valid security evidence, and valid evidence is not automatically complete evidence. Likewise, a technically valid observation does not necessarily determine how the user should respond to it. Mesh is intended to preserve those distinctions throughout the analytical path rather than compressing them prematurely into a generalized score.

3.2 Domain Architecture

Mesh is structured around multiple specialized security domains rather than a single generalized risk engine. The analytical architecture includes authority risk, creator and deployment history, ownership concentration, liquidity and exit risk, transfer restrictions, transfer fees, program upgradeability, and Token-2022 behavior. These are analytical areas rather than eight interchangeable scored outputs: the released analytical baseline exposes six risk domains, with program upgradeability and Token-2022 evidence contributing across relevant domains.[9] Transaction-level and wallet-level analysis can later build on these domains rather than requiring an entirely separate security model.

Each domain retains enough independence to answer its own security question. Authority analysis determines what privileged capabilities remain and who controls them. Ownership analysis evaluates observed concentration and controller relationships. Liquidity analysis evaluates whether a position can be exited under modeled conditions. Program analysis evaluates mutability and the control structure surrounding executable logic. Token-2022 analysis interprets additional programmable behavior that may affect transfers, fees, delegation, or other token capabilities.

The Mesh layer becomes particularly important when relationships exist across domains. A single controller may hold several privileged authorities. A highly concentrated asset may simultaneously have weak exit liquidity. A transfer hook may introduce additional program logic whose behavior can itself be modified through an active upgrade authority. These relationships may materially change the security context even when each underlying observation is individually valid. Figure 1 summarizes the conceptual architecture and its relationship to the six-domain baseline.

BAREMAX / TECHNICAL WHITEPAPER 01

Mesh architecture

Conceptual design; implementation scope is defined in Section 10.1.

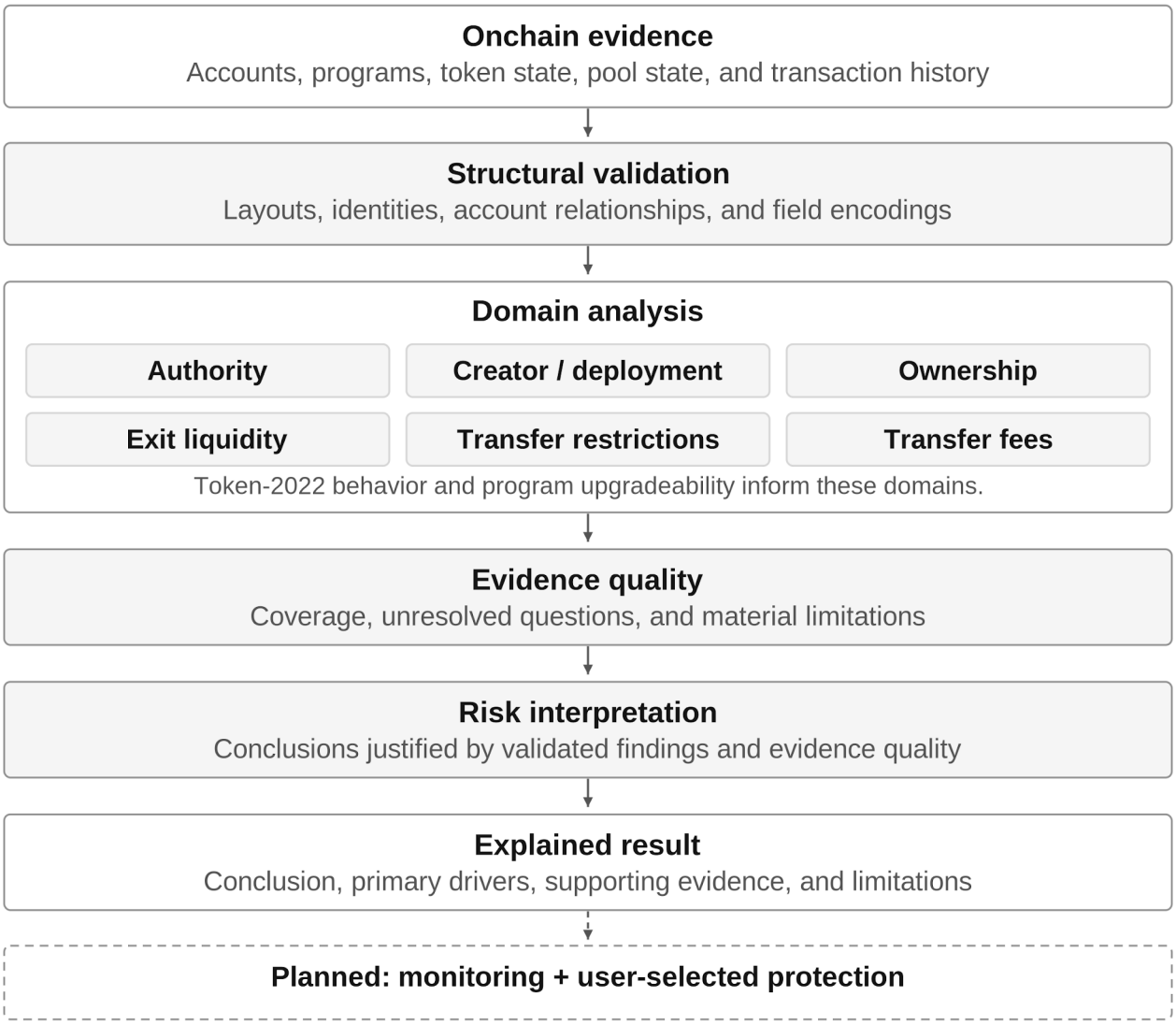


Figure 1. Mesh architecture. The conceptual pipeline combines the six risk domains with separate evidence-quality and interpretation stages. The dashed extension denotes planned monitoring and user-selected protection within supported integrations. Source: Sections 3.1-3.8 and 3.10-3.12; implementation scope: Section 10.1.
View full-size figure

3.3 Evidence Acquisition

The evidence-acquisition layer collects the raw onchain information required by the individual security domains. Depending on the analysis, this may include mint state, token-account balances, account ownership, authority addresses, executable programs, associated program state, Token-2022 extension data, transaction history, creator activity, liquidity pools, reserves, and the holdings of a connected wallet.

This structure corresponds directly to Solana's account model. Solana stores network state in accounts identified by 32-byte addresses, with account fields that include data, owner, executable state, and lamport balance. Programs contain executable logic, while mutable program state is generally maintained in separate data accounts supplied to program instructions.[8][10] Transactions contain the account references and compiled instructions necessary for execution.[8]

Acquisition is deliberately separated from interpretation because retrieved information may still be incomplete, malformed, stale, unsupported, or inconsistent with the structure Mesh expects. The fact that a provider successfully returned bytes or transaction history does not establish that those observations are valid or sufficient for a security conclusion.

3.4 Structural Validation

Before acquired evidence can influence risk, Mesh should determine whether the evidence is structurally valid. The exact validation requirements depend on the type of information being analyzed and may include confirming account ownership, expected data layouts, mint relationships, program relationships, liquidity-pool identity, authority encodings, Token-2022 extension framing, transaction identity, or historical-record consistency.

Program upgradeability provides a clear example of why this layer matters. Under Solana's loader-v3 deployment model, a deployed program can remain upgradeable while an upgrade authority is present, while revoking that authority makes the program immutable under that deployment model.[11] A security system cannot safely determine which state applies merely because one expected lookup is missing. It must first establish that the correct program and associated deployment state were identified and decoded successfully.

Token-2022 creates a similar requirement. Its extensions introduce additional state beyond the base token layout, with extension-specific data stored and interpreted according to the enabled extension types.[7] Malformed framing, incorrect lengths, conflicting entries, or improper field interpretation can therefore affect the meaning of neighboring evidence if the extension set is not validated before higher-level analysis begins.

3.5 Domain Analysis

Once evidence has passed the required structural checks, it can be interpreted by the relevant security domain. Domain analysis converts validated technical observations into security meaning without discarding the underlying evidence that produced them.

An authority address becomes relevant because of the capability it controls and the controller behind it. A holder balance becomes relevant because of its contribution to ownership concentration. A reserve measurement becomes relevant because of the execution conditions that can be modeled against it. A Token-2022 transfer hook becomes relevant because it introduces additional program logic into token transfers and therefore creates additional behavior that must be understood.

Keeping domain analysis separate from acquisition and structural validation also makes the architecture easier to extend. BAREMAX Labs can improve the interpretation of a specific risk domain or add support for new Solana functionality without changing the fundamental rule that security conclusions must be built from validated evidence.

3.6 Evidence Quality

A technically valid observation may still be insufficient to justify a complete security conclusion. Mesh therefore treats evidence quality as a separate dimension from the risk identified by that evidence.

For example, the system may successfully establish that a privileged authority remains active while possessing only partial creator history. Ownership data may represent a bounded sample rather than the complete holder population. A liquidity result may be valid across supported and authenticated venues while other markets remain outside current coverage. In each case, the established evidence remains useful, but the limits of the surrounding analysis should remain visible.

The evidence model is intended to preserve distinctions such as resolved, partial, unknown, failed, and not applicable, with the exact semantics developed in greater detail later in this whitepaper. Separating evidence quality from risk allows Mesh to distinguish between the security condition itself and how completely the system understands the surrounding state.

3.7 Risk Interpretation

Risk interpretation determines what conclusion is justified after domain findings and evidence quality have been established. This stage should not operate as a simple average of individual signals because different security conditions carry different meanings and different evidentiary requirements.

A confirmed dangerous condition may independently justify a strong warning even while unrelated evidence remains incomplete. Conversely, a collection of favorable observations should not automatically justify SAFE if a material security domain remains unresolved. A confirmed authority risk should not be diluted merely because liquidity appears healthy, and incomplete creator evidence should not erase a separately proven ownership or program risk.

The architecture therefore favors explicit security reasoning over opaque aggregation. An overall conclusion may still provide a simple summary for the user, but that conclusion should remain traceable to the domain findings and evidence conditions that produced it. This same separation eventually allows Mesh to distinguish objective security interpretation from user policy: the system establishes the security state first, while Protection Mode determines how a particular user chooses to respond.

3.8 Progressive User Experience

The interface should reflect the analytical hierarchy: an overall conclusion, its domain-level reasoning, and the supporting technical evidence. Material limitations belong with the conclusion rather than only in an advanced view. Section 10 develops this presentation model.

This progression changes the depth of presentation, not the evidence or analytical standard. It allows the same result to serve an immediate consumer decision and a more detailed technical review.

Simplicity should reduce the work required to understand a result without preventing a capable reader from checking why it was reached.

3.9 Initial Product Surface

The current beta uses a web application to make the analytical engine available through a read-only interface. Broader public access and deeper wallet integrations remain separate release decisions, governed by evidence quality, operating reliability and the scope the system can support.

The web application is expected to support direct asset analysis and, if development and final stability requirements are satisfied, wallet connection in the initial public v0. Wallet connection represents an important architectural transition because it allows Mesh to begin interpreting security relative to the user's actual holdings and position sizes rather than treating every asset as an isolated object.

The public beta is not intended to implement the entire long-term Mesh architecture immediately. Its purpose is to establish that the evidence pipeline, validation model, security domains, and user-facing conclusions are reliable enough to support progressively deeper integration with the wallet.

3.10 Pre-Sign Security

Once transaction-level analysis is sufficiently reliable, Mesh is intended to move closer to the point at which a transaction is authorized. A browser extension provides a practical intermediate environment for developing this capability before deeper integrations with wallet providers become available.

Solana transactions contain signatures and a message identifying the accounts, recent blockhash, and compiled instructions required for execution, while each instruction identifies the program and relevant accounts involved.[8] Those structures provide part of the technical foundation for pre-sign inspection, although determining what a transaction is actually capable of doing requires substantially more analysis than reading the transaction envelope alone.

Future pre-sign analysis may evaluate the programs invoked, assets entering or leaving the wallet, destination accounts, relevant authorities, Token-2022 behavior, expected wallet-state changes, existing wallet exposure, and modeled post-transaction liquidity conditions. The browser layer should be treated as an intermediate stage rather than the intended permanent interface. Where technically and commercially possible, the long-term direction is deeper integration with wallet infrastructure so that security analysis can occur as naturally as possible within the transaction flow.

3.11 Continuous Monitoring

As Mesh becomes wallet-aware, the architecture is intended to shift from isolated analysis toward persistent monitoring. The system should increasingly maintain an understanding of the security state surrounding assets already held by the wallet and identify material changes without requiring the user to initiate another analysis manually.

Relevant changes may include authority rotation, program upgrades, liquidity deterioration, worsening exit efficiency, ownership concentration changes, creator activity, or changes in programmable token behavior. Because many of these conditions can change after an asset enters the wallet, continuous monitoring is necessary if Mesh is to evolve from a research tool into a persistent security sentinel.

A future mobile application can extend this architecture through persistent access to the wallet's security state and real-time push notifications. BAREMAX Labs intends to pursue that surface only after the web, monitoring, and transaction-analysis systems are sufficiently stable to support it reliably.

3.12 Protection Mode

Protection Mode represents the action layer of the Mesh architecture. Once evidence has been acquired, validated, interpreted, and compared against the user's security preferences, the system may eventually respond with more than an informational result.

Within supported participating integrations, and depending on user configuration, Mesh could warn the user, require additional confirmation, or interrupt the local signing flow when an interaction violates a predefined security rule. Wallet connection alone does not establish that intervention capability; Section 11 defines the enforcement boundary. Examples may include exceeding a user-selected maximum exit-impact threshold, encountering an unacceptable authority configuration, proceeding with materially unresolved evidence, or attempting a transaction classified as HIGH RISK.

These controls are intended to remain optional and user-directed. BAREMAX should not take custody of the user's assets or exercise discretionary authority over the wallet. The purpose of Protection Mode is to allow users to translate their own security preferences into persistent rules while keeping the underlying evidence and security analysis independent of those preferences.

3.13 Consumer Access and Product Tiers

The shared architecture supports a useful Free analytical foundation, a substantially broader Pro service for one primary wallet, and a later Max environment for multiple wallets. Sections 8.8-8.10 define the intended feature boundaries.

Tier entitlements should govern additional analysis, retained context, monitoring capacity, and policy capabilities rather than alter the interpretation of identical evidence. This allows the service to expand commercially without introducing a different truth standard for paying users.

Product availability and service limits remain separate from the architecture. Each released tier must describe its actual supported capabilities and retain the evidence-transparency rules established in Section 8.

3.14 Architectural Direction

The long-term development path of Mesh can be understood as a progression from asset analysis to wallet-aware analysis, pre-sign security, continuous monitoring, and user-configured protection. Each stage moves the system closer to the point at which security analysis can help prevent loss rather than merely describe risk after an interaction has already occurred.

The architecture is deliberately modular so that additional security domains, Solana programs, liquidity venues, token functionality, and eventually other blockchain environments can be incorporated without replacing the evidence model underneath them. Expansion should occur only where Mesh can preserve the same standards for structural validation, explicit uncertainty, evidence quality, and explainable conclusions.

At maturity, Mesh is intended to operate between user intent and onchain execution, maintaining an evolving understanding of the wallet's exposure and applying the appropriate security analysis whenever that state changes or the user prepares to interact with the network.

4. BAREMAX Risk Domains

Mesh is designed around multiple security domains because onchain risk rarely originates from a single condition. An asset may have acceptable liquidity while retaining powerful administrative authorities, decentralized ownership while depending on upgradeable program logic, or apparently unrestricted transfers while containing Token-2022 functionality that materially changes execution behavior. Compressing these conditions too early into a single score can remove information that matters to the user.

BAREMAX therefore evaluates distinct risk domains independently before interpreting how they interact. Each domain is intended to answer a specific security question, preserve the evidence supporting its conclusion, and expose material uncertainty when the available evidence is incomplete. The domains can then be combined through the broader Mesh architecture without losing the meaning of the individual findings.

4.1 Authority Risk

Authority risk evaluates the privileged capabilities that remain associated with an asset and the entities that control them. Under the SPL Token model, a mint authority can create additional token units, while a freeze authority can freeze individual token accounts and prevent affected accounts from transferring, burning, or changing delegates.[12] Token-2022 introduces additional authority-bearing capabilities, expanding the number of control relationships that may need to be understood.[7]

Mesh should therefore evaluate more than whether a single authority exists. Relevant evidence may include mint authority, freeze authority, transfer-fee configuration authority, withheld-fee withdrawal authority, permanent delegate control, pause authority, metadata-related authorities, transfer-hook relationships, and upgrade authority associated with supporting programs. Some of these capabilities are legitimate operational tools, and their presence does not by itself establish malicious intent. The security significance depends on what the capability permits, who controls it, and whether the control can be changed.

Authority concentration is especially important. Several individually understandable capabilities can create a materially different security profile when controlled by the same address or controller. Mesh is therefore intended to identify shared control across privileged capabilities rather than displaying every authority as an isolated field. A user should be able to distinguish an asset with distributed or revoked authority from one in which a single controller retains broad influence over supply, transfers, fees, or supporting program logic.

4.2 Creator and Deployment Risk

Creator and deployment analysis examines the observable history surrounding the origin of an asset. Relevant evidence may include deployment transactions, creator or deployer addresses, prior launches associated with those addresses, historical activity, and the completeness of the history Mesh was able to recover.

The purpose of this domain is not to create a simplistic reputation score for developers. A creator with multiple prior deployments is not automatically malicious, and a newly observed creator is not automatically trustworthy. Historical activity is contextual evidence that may become meaningful when combined with other findings. Repeated short-lived launches, identifiable deployment patterns, or relationships to previously concerning assets may justify additional scrutiny, while ordinary development activity may not.

Historical coverage must remain explicit because onchain history acquisition can be bounded or incomplete. If Mesh observes one previous launch under partial historical coverage, it should report one observed launch rather than claiming that the creator has launched only one asset. Similarly, failure to recover prior activity cannot be interpreted as evidence that no prior activity exists. Creator risk should therefore reflect both what was observed and how completely the relevant history was acquired.

4.3 Ownership Risk

Ownership risk evaluates how token supply is distributed across observable token accounts and, where possible, the controlling entities behind those accounts. The objective is to determine whether a relatively small number of controllers hold enough supply to create meaningful concentration risk while distinguishing ordinary wallets from program-controlled or market-structure accounts.

This distinction matters because raw token-account rankings can be misleading. A large balance may belong to an automated market maker, treasury structure, program-controlled account, or another identifiable system rather than an ordinary holder. Conversely, one controller may hold tokens across several accounts, making account-level concentration appear lower than controller-level concentration. Mesh is therefore intended to combine token-account balances with owner aggregation and account classification where the evidence permits.

Ownership conclusions should remain bounded by the data actually observed. Early analysis may rely on a defined sample of the largest token accounts rather than complete beneficial ownership of the supply. Even when every onchain address is visible, the real-world entities controlling multiple addresses cannot always be established. Mesh should therefore describe this domain as onchain ownership and controller concentration rather than claiming knowledge of definitive beneficial ownership.

4.4 Liquidity and Exit Risk

Liquidity risk evaluates whether a position can realistically be exited under current supported market conditions. This is intentionally different from displaying a generic liquidity figure. Automated market makers price trades against token reserves, and execution changes as trade size becomes larger relative to those reserves.[13] As a result, the same asset can present very different exit conditions to two wallets holding different position sizes.

Mesh is therefore intended to model liquidity relative to the position being analyzed. Relevant evidence may include authenticated pools, reserve state, supported venues, modeled token input, expected quote output, estimated price impact, and exit efficiency. A position representing a small fraction of available liquidity may be relatively easy to exit, while a much larger position in the same asset may experience substantial execution loss.

This domain must also preserve liquidity provenance. A quote should remain attributable to the exact pool and venue that produced it, and an unsupported or unquoteable market should not be converted into a claim that no liquidity exists. Mesh should distinguish between confirmed poor liquidity, unavailable supported liquidity, and incomplete market coverage so that limitations in the analysis are not mistaken for characteristics of the asset itself.

4.5 Transfer Restriction Risk

Transfer restriction risk evaluates whether an asset contains functionality capable of limiting, modifying, or adding logic to token movement. These conditions may arise through the base token model or through Token-2022 extensions such as transfer hooks, non-transferability, default frozen state, or pausability.

A transfer hook is particularly important because it allows custom program logic to execute during transfers of a Token-2022 mint. Solana's Transfer Hook interface supports use cases including allowlists, blocklists, custom transfer logic, and other application-specific behavior.[14] The NonTransferable extension can prevent holders from transferring tokens between token accounts, while a pausable mint can reject transfers, minting, and burning while the mint is paused.[7]

The presence of these capabilities should not automatically be interpreted as malicious. They may support legitimate compliance, identity, application, or asset-design requirements. Mesh instead evaluates what restriction exists, how it operates, which authority or program governs it, whether that governing logic can change, and what the behavior means for the user attempting to hold or transfer the asset.

4.6 Transfer Fee Risk

Transfer-fee risk evaluates whether token movement can incur protocol-level fees and how those fees are controlled. The Token-2022 TransferFeeConfig extension allows a mint to apply a fee to transfers, maintain current and future fee configurations, and designate authorities responsible for changing the configuration and withdrawing withheld fees.[15]

A transfer fee is not inherently dangerous. It may be an intentional and transparent part of an asset's economic design. The security concern arises when users do not understand the effective fee, when the configuration can change materially, when powerful fee-related authorities remain active, or when the same controller holds several other privileged capabilities.

Mesh should therefore evaluate the effective fee configuration, applicable limits, configuration authority, withheld-fee withdrawal authority, and relationships between those authorities and other controllers associated with the asset. The objective is to explain the practical execution consequence rather than simply flagging the existence of a fee.

4.7 Program Upgradeability Risk

Program upgradeability evaluates whether executable logic relevant to an interaction can still be modified. Under Solana's loader-v3 deployment model, a program remains upgradeable while an upgrade authority is set; revoking that authority makes the program immutable under that deployment model.[11]

This creates an important trust distinction. A program may behave acceptably at the moment Mesh analyzes it while still retaining an authority capable of changing its executable logic later. That does not prove that the program will be modified maliciously, but it means the user is relying partly on the controller of that authority rather than only on the code currently deployed.

Mesh should therefore distinguish between proven mutable, proven immutable, and unresolved program state. Failure to retrieve or validate the relevant program evidence should not be interpreted as immutability. This principle becomes particularly important when the program participates in another risk domain, such as a transfer hook whose current behavior appears benign but whose underlying code remains upgradeable.

4.8 Token-2022 Extension Risk

Token-2022 deserves dedicated analysis because it expands the range of behavior that can be associated with a token mint or token account. Extensions are optional and can introduce functionality including transfer fees, permanent delegates, transfer hooks, non-transferability, pausing, scaled UI amounts, metadata structures, and other specialized capabilities.[7] Solana also documents compatibility constraints between certain extensions, meaning the extension set itself has structural rules that must be interpreted correctly.[7]

Mesh does not treat Token-2022 as an inherently higher-risk token standard. Instead, additional programmability increases the amount of security-relevant state that must be validated. A permanent delegate, for example, is a mint-level authority capable of authorizing transfers and burns for token accounts of that mint, and token-account owners cannot revoke that delegate themselves.[16] That capability may be legitimate, but it materially changes the control model of the asset and should be made visible to the user.

Token-2022 analysis therefore focuses on what extensions are enabled, whether their underlying structures are valid, what capabilities they introduce, which authorities control them, whether those authorities or associated programs remain mutable, and how extension combinations affect the broader security state. Section 6 develops this domain in substantially greater technical detail.

4.9 Evidence Quality Across Domains

Evidence quality is not itself another risk domain. It is a property that applies across every domain in Mesh.

Authority analysis can be complete or incomplete. Creator history can be fully resolved or partial. Ownership analysis can operate on a validated sample while remaining bounded in scope. Liquidity can be authenticated across supported venues while other markets remain outside coverage. Program upgradeability can be proven or unresolved. The risk conclusion and the completeness of the evidence therefore represent separate questions.

This distinction prevents missing information from being converted into a security state it does not justify. A HIGH authority result can remain HIGH even when creator evidence is incomplete, because the authority condition has already been established. Conversely, several apparently favorable domains may still be insufficient for an overall SAFE conclusion if a material part of the analysis remains unresolved. Section 5 formalizes this relationship through the BAREMAX Evidence Model.

4.10 Wallet-Drain Risk as an Emergent Outcome

Wallet-drain risk is broader than any single initial Mesh domain. A wallet can lose assets through malicious transaction execution, deceptive permissions, hostile programs, unexpected token mechanics, compromised wallet interactions, upgradeable logic, manipulated transaction paths, or combinations of conditions that become dangerous only when interpreted together.

For that reason, BAREMAX does not initially treat "wallet drainer" as one isolated checkbox. The long-term system must understand what a transaction, asset, program, or permission can actually do to the connected wallet. Transaction-level protection therefore depends on several parts of Mesh working together: program interpretation, authority analysis, account relationships, token behavior, expected balance changes, wallet exposure, and the evidence quality surrounding each of them.

As Mesh approaches pre-sign protection, wallet-drain analysis can become a higher-level security outcome built from those underlying domains. The long-term question is not merely whether a known drainer signature or program address appears in a transaction, but whether the proposed interaction can create an unexpected or materially dangerous change to the wallet.

4.11 Consumer Presentation

Domain breadth should not force every user into the same technical depth. The layered interface in Section 10 should keep the principal finding and its evidence limitations visible while allowing each domain to be inspected independently.

Each expanded domain should expose the evidence relevant to its own question, rather than repeat a generic score explanation. Authority records, historical coverage, and liquidity provenance remain different kinds of supporting evidence even when presented through a consistent interface.

This structure allows Mesh to maintain analytical depth without turning every security decision into a technical investigation. The goal is not to hide complexity but to organize it so that users can access as much of it as they need.

4.12 Domain Scoring and Overall Interpretation

BAREMAX may use domain-specific scoring internally or within certain product views, but the system should not rely on one generalized numeric score as a substitute for security reasoning. Different domains represent different types of risk, and the same numeric value would not necessarily carry the same meaning across authority control, liquidity, ownership, or program mutability.

The overall Mesh conclusion should therefore preserve important domain behavior. A confirmed HIGH condition should not disappear because several unrelated domains appear favorable. Material uncertainty should remain visible rather than being averaged away. Similarly, a positive SAFE conclusion should require sufficient resolution across the material domains relevant to the interaction rather than simply exceeding a numerical threshold.

Exact beta thresholds and scoring mechanics may continue to evolve as BAREMAX Labs tests the system against broader production data. The methodology can become more formally specified as the product matures, but the underlying principle should remain stable: scores may summarize evidence, but they should never replace the reasoning supported by that evidence.

5. The BAREMAX Evidence Model

The BAREMAX Evidence Model defines how Mesh determines whether the information available to it is strong enough to support a security conclusion. Risk and evidence quality are treated as related but distinct dimensions. A system can possess strong evidence of a dangerous condition while other parts of the analysis remain incomplete, just as it can observe several favorable conditions without possessing enough evidence to justify a SAFE result.

This distinction is central to Mesh because many security failures do not begin with an incorrect calculation. They begin when missing, malformed, or incomplete information is interpreted as though it were valid evidence. The purpose of the evidence model is therefore to preserve the difference between what Mesh has positively established, what it has only partially observed, what it cannot determine, and what falls outside the relevant scope of analysis.

5.1 Risk and Evidence Are Separate Dimensions

A security conclusion should describe what the validated evidence indicates, while evidence quality should describe how completely the relevant state has been established. Combining those questions into one value can create misleading results.

For example, Mesh may positively establish that a mint authority and freeze authority remain active while creator-history coverage is incomplete. The authority finding does not become weaker simply because creator evidence is partial. The appropriate result may therefore contain a strong authority warning alongside an explicit limitation in creator-history coverage.

The reverse is also important. Several resolved checks with favorable results do not necessarily justify an overall SAFE conclusion if another material domain remains unresolved. Evidence completeness therefore affects how strongly Mesh can support a positive conclusion without requiring every incomplete observation to erase security findings that have already been established.

5.2 Evidence States

Mesh is intended to preserve explicit distinctions in the evidence underlying its analysis. RESOLVED, PARTIAL, UNKNOWN, FAILED, and NOT APPLICABLE provide the conceptual vocabulary used in this whitepaper; they are not presented as one uniform enum already implemented across every domain. Acquisition outcome, historical or market coverage, controller classification, applicability, and eligibility for scoring remain separate properties. Appendix A relates this vocabulary to the inspected result interfaces.[9][17]

RESOLVED means the relevant evidence was successfully acquired and validated to the degree required by the supported analysis. PARTIAL means valid evidence exists, but the available coverage is incomplete in a way that may limit the scope of the conclusion. UNKNOWN means Mesh cannot currently

establish the relevant condition with sufficient confidence. FAILED indicates that an expected acquisition or validation path did not complete successfully and therefore cannot be used as clean evidence. NOT APPLICABLE indicates that a particular check does not apply to the asset, program, or interaction being analyzed.

These states are designed to preserve meaning rather than merely describe software status. A failed history request, for example, is not the same as a creator having no prior launches. An extension that does not exist is different from an extension whose state could not be decoded. A program that has been proven immutable is different from one for which upgradeability evidence could not be established.

5.3 Different Burdens of Proof for Risk and Safety

Mesh deliberately applies different evidentiary burdens to negative and positive security conclusions.

A confirmed dangerous condition can justify a strong warning without requiring the system to resolve every unrelated domain first. If a token is positively shown to retain a highly privileged authority, the existence of that authority is actionable evidence even if ownership or creator history remains incomplete. Requiring complete system-wide evidence before displaying a confirmed threat would reduce the usefulness of the security layer.

A SAFE conclusion requires a higher evidentiary threshold. Mesh should not infer safety merely because it failed to detect a threat. The material security domains supported for the interaction must be sufficiently resolved, and those resolved checks must not contain a significant supported risk condition.

This asymmetry reflects the practical consequences of the two error types. A false warning may create unnecessary friction, while unsupported reassurance may encourage a user to proceed with an interaction under an incorrect assumption about its security state.

5.4 What SAFE Should Mean

SAFE is a proposed consumer conclusion requiring explicit eligibility criteria; it is not the current engine's LOWER classification and does not denote a calibrated probability of safety. Within the intended Mesh model, it should represent a positive, evidence-qualified conclusion within the system's validated and supported scope. It should communicate that the material checks relevant to the interaction were successfully resolved and that no significant supported risk condition was identified at the time of analysis.

SAFE should not be interpreted as a guarantee that an asset cannot lose value, that an unknown vulnerability does not exist, that future onchain state will remain unchanged, or that the user's device and private keys are uncompromised. Those limitations are inherent to the distinction between a security analysis and an absolute guarantee.

A defensible interpretation of the label is therefore: BAREMAX found no material supported risk after successfully resolving the evidence required for this interaction. Figure 2 summarizes the evidentiary standard, which should remain consistent even if the interface uses simpler wording.

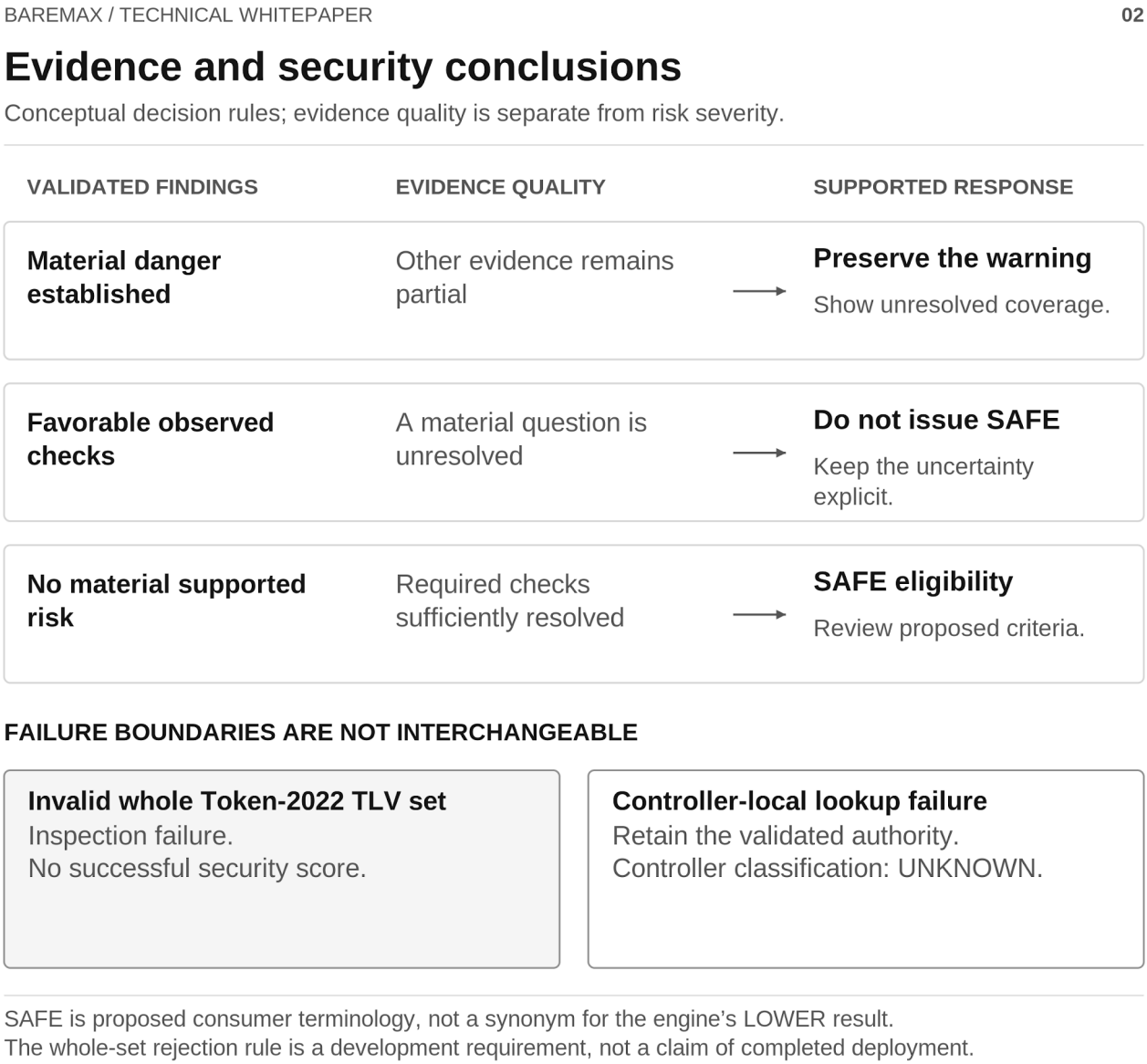


Figure 2. Evidence and security conclusions. Supported warnings, unresolved evidence, and proposed SAFE eligibility require different reasoning. Whole-Token-2022 validation failure is distinct from a controller-local lookup failure. Source: Sections 5.1-5.4 and 6.12; Appendices A, D, and H.
View full-size figure

5.5 Material and Immaterial Uncertainty

Not every unresolved datapoint should prevent Mesh from reaching a useful conclusion. The evidence model must distinguish between uncertainty that could reasonably change the security result and information that is immaterial to the decision being made.

If a nonessential metadata field cannot be retrieved, that failure may have no meaningful effect on the security conclusion. If the system cannot determine whether a program governing transfer behavior remains upgradeable, the missing evidence may be material because a different answer could substantially alter the user's exposure.

The relevant question is therefore not whether every possible datapoint is known. It is whether the unresolved evidence could reasonably change the conclusion Mesh is presenting. This allows the system to remain conservative without becoming unusably sensitive to every minor acquisition failure.

5.6 Evidence Should Be Monotonic

The evidence model is designed around a monotonic security principle: removing valid information should not create a stronger positive conclusion solely because the system now knows less.

If Mesh determines that ownership concentration is elevated and later loses the evidence required to reproduce part of that analysis, the correct direction is toward increased uncertainty rather than toward a lower concentration classification. If an active authority has already been established, a later failure to classify its controller should not transform that authority into a revoked or harmless state.

This principle is a constraint on information loss for a fixed underlying state and methodology, not a rule that risk can never decrease. A verified authority revocation, improved execution conditions, or another substantiated state change may legitimately reduce risk. When material evidence merely becomes unavailable, the result should retain independently supported findings or degrade toward uncertainty. Earlier observations must retain their timestamp or slot and applicable freshness limits; stale evidence should be described as last observed, not indefinitely presented as proof of the current state.

This approach is consistent with the broader secure-design principle of failing toward a safer state when a security control encounters an error rather than allowing an exception to produce a more permissive outcome.[18]

5.7 Conflicting Evidence

Mesh must also account for situations in which multiple observations disagree.

Conflicts may arise from different historical records, duplicate observations, non-atomic reads, inconsistent provider responses, or contradictory representations of the same underlying state. The presence of a conflict does not automatically reveal which observation is correct, and choosing whichever result produces the cleanest conclusion would create another form of false confidence.

When conflicting evidence is material and cannot be reconciled through structural validation, the conflict should reduce the completeness of the relevant evidence state. Mesh may still preserve individual findings that remain independently valid, but it should not represent the disputed portion of the analysis as fully resolved until the inconsistency has been explained.

This approach also preserves the auditability of the system. A technically advanced user should be able to determine that conflicting observations existed rather than receiving a final result from which the disagreement has silently disappeared.

5.8 Malformed Evidence

Malformed evidence should never be allowed to imitate a legitimate security state.

Examples may include truncated Token-2022 extension data, invalid authority encodings, malformed transaction identifiers, impossible numeric values, inconsistent account layouts, or incorrectly framed data structures. A malformed value must not become zero, an invalid authority must not become revoked, and a truncated structure must not be accepted simply because it can be coerced into a technically usable value.

The distinction between absent, revoked, unsupported, and malformed is especially important. These conditions may appear similar at the user-interface level if the system does not preserve the evidence semantics underneath them, but they represent fundamentally different security states.

Mesh should therefore validate the structural integrity of evidence before higher-level interpretation begins. Where validation fails materially, the appropriate outcome depends on the failed boundary. A controller-local lookup may leave a validated authority address intact while its classification becomes UNKNOWN; invalid whole-Token-2022 extension evidence must instead reject the inspection under the whole-set contract in Section 6.12 and Appendix D. Neither path should fabricate a successful clean result.

5.9 Provenance

Security evidence should remain traceable to its source.

For important findings, Mesh should be able to preserve enough provenance to explain which account, transaction, program, pool, historical record, or other onchain observation produced the result. A modeled liquidity quote should identify the pool responsible for the calculation. An authority conclusion should remain linked to the decoded authority state. A creator-history finding should remain connected to the historical transaction evidence supporting it.

Provenance becomes increasingly important as Mesh combines multiple providers, venues, historical acquisition strategies, and security domains. Without it, the system may be able to display a conclusion without being able to explain precisely where that conclusion came from.

The long-term objective is therefore not only that Mesh produce explainable conclusions, but that the evidence supporting those conclusions can be reconstructed when necessary for debugging, advanced user inspection, incident review, or future methodology changes.

5.10 Evidence Quality in the Interface

Evidence quality should be visible to the user without overwhelming the primary product experience. A basic view should be able to communicate both the security conclusion and whether material evidence remains unresolved, while more advanced views can expose exactly which domains are complete, partial, or unknown.

For example, a HIGH RISK conclusion may coexist with PARTIAL overall evidence because one dangerous condition has been positively established while another domain remains incomplete. Conversely, a set of favorable domain results may still produce an UNKNOWN overall conclusion when a material security question cannot yet be answered.

This two-dimensional presentation is important because risk severity and confidence are not interchangeable. A user should be able to understand whether Mesh is warning because it found something dangerous, exercising caution because important evidence is missing, or expressing confidence because the relevant checks were resolved successfully.

5.11 Personalized Risk Without Personalized Facts

As Mesh becomes wallet-aware, users should eventually be able to configure their own protection thresholds. A conservative user may want an interaction interrupted when evidence quality falls below a chosen threshold, while a more experienced user may accept certain unresolved conditions after reviewing the supporting information.

Personalization should not alter the evidence itself. If Mesh observes an active authority, an 8% modeled exit impact, or partial creator-history coverage, those facts should remain the same for every user analyzing the same state at the same time. What changes is the user's chosen response to those facts.

This separation ensures that Mesh can remain analytically consistent while supporting different security tolerances. The evidence model establishes what is known and what is not; the risk model interprets the security significance; Protection Mode determines how the individual wallet owner wishes to respond.

5.12 Evidence Integrity as a Development Requirement

The Evidence Model is not intended only for the user interface. It establishes requirements for how the system itself should be developed and tested.

When BAREMAX Labs discovers that malformed data, incomplete history, duplicated evidence, provider failure, or another edge case can produce an unjustified conclusion, the appropriate response is to correct the behavior and preserve the failure as regression coverage. NIST's Secure Software Development Framework similarly emphasizes addressing the root causes of vulnerabilities and incorporating practices that reduce the likelihood of recurrence throughout the software-development lifecycle.[6]

Over time, this creates a stronger evidence contract between the raw onchain state and the conclusion displayed to the user. Mesh should become more confident because its acquisition, validation, and interpretation improve, not because unresolved states are gradually classified away.

The Evidence Model therefore forms one of the core foundations of Mesh. The system should be decisive when danger is established, conservative when safety is uncertain, and transparent about the evidence supporting both outcomes.

6. Token-2022 Security

Token-2022 represents one of the clearest examples of why Solana security requires protocol-specific analysis. The Token Extensions Program expands the conventional token model through optional extensions that add specialized state and behavior to token mints and token accounts. These capabilities include transfer fees, permanent delegates, transfer hooks, non-transferability, pausability, scaled UI amounts, metadata structures, confidential functionality, and other extensions. Solana stores extension-specific state in type-length-value data appended to the base account state, meaning a Token-2022 asset can contain substantially more security-relevant information than a conventional token analysis would reveal.[7]

BAREMAX does not treat Token-2022 as inherently dangerous. Many extensions exist to support legitimate financial, compliance, payment, identity, and application use cases. The security problem is that additional programmability introduces additional assumptions that must be verified. Mesh is therefore being designed to determine which extensions are present, whether their underlying structures are valid, what capabilities they introduce, who controls those capabilities, whether that control can change, and how the resulting behavior affects the wallet.

This makes Token-2022 an important area of specialization for BAREMAX. The advantage is not simply detecting that an extension exists. The objective is to interpret the extension within the broader Mesh architecture and determine whether its behavior, authority structure, mutability, and evidence quality materially affect the security state of the asset.

6.1 From Token Balance to Programmable Asset

A conventional token inspection can often begin with relatively familiar properties such as supply, decimals, mint authority, freeze authority, and ownership distribution. Token-2022 expands that analytical surface because a mint or token account may contain additional extension state that changes how the asset behaves.

Solana's Token Extensions Program defines multiple extension types, and individual extensions can add state to either the mint or token accounts associated with it. Some extensions are also incompatible with one another, while most must be anticipated during account initialization rather than added freely afterward.[7] These characteristics mean that Mesh must understand the extension set as part of the asset's actual control and execution model rather than treating extensions as secondary metadata.

The presence of additional functionality can affect questions that matter directly to users: whether transfers can invoke external program logic, whether transfers incur fees, whether another authority can transfer or burn tokens, whether the mint can be paused, whether balances are displayed using a mutable multiplier, and where authoritative metadata is stored. Token-2022 analysis must therefore move beyond identifying the token program and into interpreting the programmable capabilities attached to the asset.

6.2 Extensions Are Capabilities, Not Automatic Warnings

The existence of an extension should not automatically produce a negative security conclusion. A transfer-fee configuration may be appropriate for a tokenized financial product. A permanent delegate may exist for regulated asset administration. A transfer hook may enforce legitimate application rules. A pausable mint may be intentionally designed for emergency or compliance controls.

Mesh should instead evaluate the capability created by each extension and the trust assumptions that follow from it. The relevant questions include whether the capability can affect the user's ability to transfer or exit, which authority controls it, whether that authority can be changed, whether supporting program logic remains mutable, and whether multiple sensitive capabilities are concentrated under the same controller.

This produces a more useful security principle for Token-2022: increased programmability should increase the amount of verification performed by Mesh, not automatically increase the severity assigned to the asset. The security conclusion should follow from the actual configuration and control structure rather than from the presence of advanced functionality alone.

6.3 Transfer Fees

The TransferFeeConfig extension allows a Token-2022 mint to impose protocol-level fees on transfers. The configuration includes fee parameters, a transfer-fee configuration authority, and an authority capable of withdrawing accumulated withheld fees. Solana maintains older and newer fee configurations, and updates made through SetTransferFee take effect beginning two epochs later.[15]

For Mesh, detecting that a transfer fee exists is only the beginning of the analysis. The system should understand the effective fee, applicable maximum, currently active and scheduled configuration, configuration authority, withheld-fee withdrawal authority, and the relationship between those authorities and other privileged controllers associated with the mint. A modest fixed fee controlled through a clearly understood authority structure represents a materially different condition from a configuration capable of changing substantially under an address that also controls other sensitive capabilities.

Transfer fees also interact with execution analysis. The amount displayed before a transfer may not equal the amount ultimately available after token-level fees and market execution are considered. As Mesh develops deeper transaction and exit modeling, transfer-fee behavior should therefore be incorporated into estimates of what actually enters or leaves the wallet rather than being presented only as a separate token property.

6.4 Permanent Delegates

The PermanentDelegate extension establishes a mint-level delegate capable of authorizing transfers and burns across token accounts associated with the mint. Individual token-account owners cannot revoke this delegate from their own accounts.[16] Authority changes must be evaluated against the applicable program version: in the inspected Token-2022 program@v10.0.0 implementation, SetAuthority with AuthorityType::PermanentDelegate validates the existing permanent-delegate authority. Possessing mint authority alone does not establish permission to rotate that distinct authority.[19]

This capability materially changes the conventional assumption that possession of tokens gives the holder exclusive authority over movement or destruction of those tokens. Its presence does not establish malicious intent, but it introduces an additional trust relationship that should be clearly visible to the user.

Mesh should therefore identify the permanent delegate, determine the controller where possible, evaluate whether that controller also holds other sensitive privileges, and explain the practical capability created by the extension. If the same controller acts as permanent delegate while also retaining mint, freeze, fee, pause, or program-level authority, the concentration of privileged control may be more important than any individual authority considered alone.

6.5 Transfer Hooks

The Transfer Hook extension allows custom program logic to execute during transfers for a Token-2022 mint. During a qualifying transfer, the Token Extensions Program invokes the configured transfer-hook program through a cross-program invocation. Solana documents potential uses including allowlists, blocklists, custom transfer fees, transfer events, and other application-specific logic.[14]

Transfer-hook analysis must distinguish three sources of change: the authority that can select a different hook program, the loader authority that can upgrade the currently selected program, and mutable state consulted by that program. The inspected Token-2022 implementation permits an active hook-configuration authority to update the configured program ID.[20] Proving that the current executable is immutable therefore does not prove that future transfer behavior is permanently fixed. Mesh should identify the configured program and preserve separate findings for configuration control, executable mutability, and any supported state dependencies.

The standard transfer-hook Execute interface applies read-only and non-signer restrictions to the original transfer accounts passed into the hook invocation. Read-only access and signer status are separate account privileges; one should not be described as causing the other.[14] These safeguards do not establish that arbitrary custom logic is harmless or that configuration and program state cannot change. Mesh should distinguish extension detection, validated control relationships, and supported behavioral interpretation, rather than claiming that identifying the hook program is equivalent to auditing its code.

6.6 Non-Transferable Tokens

The NonTransferable extension creates a token whose holders cannot move their tokens between token accounts through ordinary transfer instructions. Solana documents this pattern as useful for assets intended to remain associated with a particular wallet, while the tokens can still participate in certain other supported operations such as minting and burning under the applicable authorities.[21]

For security analysis, this behavior is important because ownership of an asset does not imply that the holder can freely transfer or sell it. Mesh should identify non-transferability explicitly and distinguish it from an ordinary transfer failure, insufficient liquidity, or an unsupported execution path.

The extension should not automatically be characterized as malicious. It may be entirely appropriate for identity credentials, compliance-oriented assets, or other intentionally non-transferable instruments. The relevant requirement is that the user should know the restriction before acquiring the asset under the assumption that it behaves like a conventional transferable token.

6.7 Pausable Mints

The PausableConfig extension gives a designated pause authority the ability to pause and resume activity for an entire mint. When the mint is paused, the Token Extensions Program rejects transfers, minting, and burns until activity is resumed.[22]

This creates a significant operational capability that may be legitimate while still affecting the security assumptions of the holder. A user evaluating an asset should be able to distinguish between a token whose activity cannot be globally paused and one whose transfers can be stopped by an active authority.

Mesh should therefore determine whether the extension is enabled, whether the mint is currently paused, which authority controls the pause function, and whether that controller overlaps with other sensitive authorities. As with other Token-2022 functionality, the presence of the capability and the concentration of control surrounding it are separate pieces of evidence that should both remain visible.

6.8 Scaled UI Amounts

The ScaledUiAmount extension allows a mint to apply a multiplier to the amount displayed to users without changing the raw token amount stored in individual token accounts. Solana describes applications such as stock splits, dividends, and yield representation. The configuration can include a current multiplier, a future multiplier, an effective timestamp, and an authority capable of updating the multiplier.[23]

This extension requires Mesh to distinguish between raw token amounts and user-interface amounts. A displayed balance may change because the multiplier changed even though the underlying amount stored in the token account did not. Solana also notes that conversion helpers for scaled UI amounts use floating-point arithmetic and are not guaranteed to round-trip exactly, which makes precision and display handling relevant when systems convert between raw and displayed units.[23]

Scaled UI behavior is not necessarily a direct security threat, but it is security-relevant because an analyzer that confuses displayed amounts with raw token amounts can produce incorrect ownership, position, or execution interpretations. Mesh should therefore preserve raw values for security calculations and treat UI scaling as an additional presentation and control layer rather than allowing displayed amounts to replace underlying token state.

6.9 Metadata and Pointer Extensions

Token-2022 can also represent metadata through the MetadataPointer and TokenMetadata extensions. The metadata pointer identifies the account where token metadata is stored and includes an authority capable of updating that pointer, while TokenMetadata can store fields such as the token name, symbol, URI, update authority, and additional metadata directly in the mint's extension data.[24]

Metadata does not normally determine whether a token can be transferred or whether a position can exit, so Mesh should avoid giving metadata-related findings the same weight as more consequential execution controls. However, metadata authorities and pointers remain part of the broader trust structure surrounding an asset. A mutable pointer or update authority can alter what users and applications are shown even when the underlying mint remains the same.

The security significance should therefore be proportional to the capability. Mesh should identify relevant metadata and pointer authorities, preserve them as part of the asset's control structure, and avoid allowing presentation-level metadata to substitute for more authoritative onchain evidence about the token's actual behavior.

6.10 Extension Combinations and Privileged Control

Individual extensions become more meaningful when analyzed together. A token may contain a transfer hook, transfer-fee configuration, metadata authority, and another privileged capability simultaneously. Even if none of those features is independently malicious, their combined control structure may create a substantial dependency on one controller.

Mesh should therefore evaluate privilege concentration across extensions. If the same address controls transfer-fee configuration, withdrawal of withheld fees, a transfer hook, metadata updates, or other sensitive capabilities, the user should be able to see that concentration explicitly rather than interpreting each extension as a separate isolated field.

The extension set itself must also be structurally coherent. Solana documents compatibility constraints between certain extensions; for example, NonTransferable and TransferFeeConfig cannot be enabled together because their behaviors conflict.[7] Mesh should respect protocol-level compatibility rules as part of validation rather than assuming that any collection of decoded extension identifiers represents a valid mint configuration.

6.11 Structural Integrity Before Interpretation

Token-2022 places extension-specific state in TLV data following the base account structure. That data must be deserialized according to the extension types present in the account.[7] This makes decoding integrity part of the security boundary rather than a purely technical implementation concern.

If an analyzer accepts truncated payloads, impossible lengths, malformed authority fields, conflicting extension representations, or otherwise invalid structures, higher-level analysis can become unreliable even when the parser returns a value. A malformed authority could appear to be a legitimate address, a restriction could be missed, or neighboring extension state could be interpreted incorrectly. Mesh should therefore establish structural validity before extension-specific values are permitted to affect controller analysis, privilege grouping, or risk.

The relevant distinction is between an extension being absent and the system being unable to validate the extension data. Those states should never collapse into one another. Absence can be legitimate evidence. Malformation or failed validation indicates that the system cannot safely determine the corresponding state.

6.12 Whole-Set Validation

Because extension records occupy a shared type-length-value (TLV) region, the approved Mesh validation contract requires whole-set validation before extension-specific values influence a successful inspection. Required checks include bounded framing, declared lengths, supported payload layouts, field encodings, unique extension types, and applicable compatibility rules. This is an implementation requirement, not a claim that every aspect is already integrated into the baseline identified in Section 10.1.

Whole-set validation is intended to prevent a malformed record from undermining the interpretation of neighboring records. The validator must reject truncated headers or payloads, lengths outside the available region, invalid supported encodings, and every duplicate extension type, including duplicates with identical payloads. These concrete checks establish the boundaries the decoder is allowed to trust; they do not imply that the intended meaning of arbitrary opaque bytes can be recovered.

An invalid whole extension set must produce an inspection failure, not a partial successful security score assembled from whichever fields happened to decode. Diagnostic facts may be retained for investigation, but they must not be published as a completed InspectionResult. This differs from a controller-local acquisition failure after an authority field has already been validated: that narrower failure can preserve the authority identity while marking its controller classification UNKNOWN. Appendix D states the same contract for implementation and regression testing.

6.13 Unknown and Unsupported Extensions

Structural framing and semantic support are separate questions. An unknown extension record may have a readable type and a payload length bounded by the available TLV region while its payload layout, compatibility requirements, and security meaning remain outside Mesh's current understanding.

A bounded unknown record should remain opaque rather than be automatically classified as malformed, absent, or harmless. Validating its outer framing does not establish semantic validity. Mesh should state which checks were possible and which were unsupported, and material unsupported behavior must prevent an unjustified positive conclusion. This preserves the distinction between invalid evidence and a limitation in product support.

The same principle applies as the Token Extensions Program evolves. BAREMAX Labs should be able to add semantic support for new extension types without weakening the structural rules governing existing ones. This allows the Token-2022 security layer to expand over time while preserving the evidence discipline underneath it.

6.14 Token-2022 as a Solana-Native Security Advantage

Token-2022 illustrates why BAREMAX is beginning with deep Solana specialization rather than treating every blockchain through a generic token-security abstraction. Correct analysis requires understanding not only that an extension exists, but how Solana encodes it, which authorities govern it, which programs it can invoke, how extension combinations interact, and what evidence is required before a security conclusion is justified.

For Mesh, the relevant analytical sequence is therefore broader than extension detection. The system must determine which functionality is enabled, whether the underlying extension state is structurally valid, what capability that functionality creates, who controls it, whether the control or associated executable logic can change, whether multiple privileges are concentrated under the same controller, how the functionality affects execution, and how complete the evidence supporting those conclusions actually is.

Token-2022 should become one of the clearest demonstrations of BAREMAX's broader engineering philosophy. Programmable assets do not need to be treated with automatic suspicion, but their behavior must be understood before the system can responsibly characterize them as safe. As Solana's token infrastructure becomes more capable, the role of Mesh is to make the additional security assumptions created by that programmability visible and verifiable.

7. Liquidity and Execution Risk

Liquidity is frequently presented to users as a static property of an asset. A wallet interface or token scanner may display a pool balance, total liquidity figure, or estimated market depth and leave the user to infer whether the position can be exited efficiently. For security analysis, that information is incomplete. The existence of liquidity does not establish that sufficient liquidity exists for a particular wallet, position size, execution path, or point in time.

Mesh therefore treats liquidity as a position-dependent execution property rather than a single token-level metric. The relevant question is not simply whether a market exists, but what the wallet can realistically recover if it attempts to exit through the supported liquidity available to it. This requires BAREMAX to authenticate liquidity sources, model execution at defined position sizes, preserve the provenance of the resulting quotes, and distinguish current execution conditions from broader predictions about future market value.

This approach is grounded in the mechanics of automated market makers themselves. Solana documentation defines an AMM as an onchain exchange model in which users trade against reserves governed by a pricing formula, with price movement and slippage increasing as trade size becomes larger relative to available reserves.[13] Pump.fun similarly documents its bonding curve as a constant-product AMM in which larger trades receive progressively worse execution as the reserves change.[25]

7.1 Liquidity Is Relative to Position Size

The same market can represent materially different levels of execution risk for different wallets. A relatively small holder may be able to sell a position with little price impact, while a larger holder of the same asset may move substantially through the available liquidity and receive considerably less value than the position's displayed mark-to-market value suggests.

This relationship follows directly from AMM mechanics. In reserve-based markets, each trade changes the relationship between the assets in the pool, and larger trades consume a greater share of the available liquidity. Deeper liquidity generally reduces slippage, while lower depth increases the sensitivity of execution to trade size.[13] Concentrated-liquidity markets add another consideration because execution depends on liquidity active within the price ranges traversed by the trade.[26] This is protocol background, not a claim that the inspected BAREMAX baseline models CLMM execution.

Mesh should therefore evaluate liquidity relative to the position being modeled. A generic statement that an asset has “good liquidity” is less useful than determining how a specific quantity of tokens is expected to execute against the authenticated markets currently available to the system.

7.2 Exit Efficiency

One of the core concepts in the Mesh liquidity model is exit efficiency. Exit efficiency describes the relationship between the position's modeled reference value and the amount that can be recovered through the supported execution path under the conditions observed at the time of analysis.

For an illustrative existing position valued at 10.00 SOL using a stated spot-reference basis, a modeled net exit of 9.45 SOL represents 94.5% exit efficiency and a 0.55 SOL modeled shortfall against that reference. These are illustrative values, not product thresholds or a guaranteed execution result. The purpose is to distinguish reference valuation from modeled realizable output, not predict the future price of the token.

The calculation must identify a positive reference denominator, the raw token quantity, the quote asset, and the fees included in the modeled net output. Pool fees and supported Token-2022 transfer fees may affect that output.[15] Network fees, priority fees, account-creation costs, and unsupported transfer behavior are not implicitly included. The released analytical baseline defines its exit-impact metric as net-output shortfall against the selected venue's spot reference, including modeled fees; this is distinct from the change in the pool's marginal spot price and from deviation between a quote and later execution. Appendix C specifies these distinctions.[27]

7.3 Personalized Exit Analysis

As Mesh becomes wallet-aware, liquidity analysis should become personalized automatically. A connected wallet can provide the position information necessary to evaluate the user's actual exposure rather than requiring manual position entry for every asset.

The system can combine wallet holdings with validated liquidity evidence to determine the raw position size, supported execution venues, estimated output, modeled price impact, and resulting exit efficiency. A user holding 8.2 million units of an asset should therefore receive an analysis of the approximately 8.2 million-token position rather than a generic liquidity score derived from an arbitrary trade size.

This personalization is important because the economic risk created by limited liquidity is specific to the amount the wallet is attempting to move. Mesh should ultimately be able to tell a user that a position's displayed value differs materially from the amount that current supported liquidity is expected to return, while preserving the market and execution assumptions responsible for that estimate.

7.4 Pre-Trade Exit Modeling

The same liquidity model can eventually operate before a position is entered. A conventional swap interface primarily answers how much of the purchased asset the user is expected to receive. Mesh should also evaluate what the resulting position would look like from an exit perspective immediately after the proposed purchase.

For a hypothetical purchase of 10 SOL, a future pre-sign model must first estimate the purchased quantity and the purchase's effects on the relevant pool or route state. It must then model the proposed exit against that resulting state, using the same defined quantity and accounting for supported fees and token behavior. Two independent quotes against unchanged pre-purchase reserves are not a round-trip simulation. Existing reference-based SOL position sizing should not be represented as this future purchase-and-exit capability.[27]

The purpose is to expose supported execution costs and restrictions before a position is entered, not to assume that ordinary curve movement is itself an entry trap. Under the ideal constant-product invariant, reversing an exact purchase against the resulting reserves returns the original input when fees, rounding, and intervening changes are excluded; this follows algebraically from the invariant.[28] Any modeled shortfall must therefore be attributable to identified assumptions such as fees, token restrictions, route differences, existing holdings, or subsequent state changes. The user need not intend an immediate sale for this analysis to be informative.

7.5 Authenticated Liquidity

Liquidity should not influence a security conclusion merely because Mesh discovered an account that appears to contain token reserves. Before a pool is used for execution modeling, the system should establish that the account structure is consistent with the venue and pool type it claims to represent.

Depending on the protocol, authentication may require validating program ownership, pool identity, paired mints, vault relationships, authority structure, reserve accounts, token-program relationships, and other venue-specific invariants. Raydium, for example, operates several distinct onchain liquidity designs, including constant-product and concentrated-liquidity programs, each with its own accounts and execution behavior.[29]

This is an important application of the broader BAREMAX evidence model. A discovered pool candidate is not automatically an authenticated liquidity source. The account relationships supporting the pool must first satisfy the structural requirements necessary for Mesh to trust its reserves and execution model.

7.6 Authentication and Quoteability

A liquidity pool can be structurally authentic without being usable for a particular modeled execution. Mesh should therefore distinguish authentication from quoteability.

Authentication establishes that the system has identified and validated the pool it believes it is analyzing. Quoteability establishes that Mesh has enough valid state and supported execution logic to model the requested trade through that pool. A pool may fail the second condition because of unavailable reserve evidence, unsupported token behavior, invalid execution state, insufficient depth, or another condition that prevents the system from producing a trustworthy quote.

These outcomes should remain separate. An authentic but currently unquoteable pool should not be treated as fraudulent, while a pool that cannot be authenticated should not contribute reserves or quotes to the security conclusion simply because it appears economically attractive.

7.7 Multiple Pools and Per-Size Selection

An asset may trade through several pools, protocols, or market structures simultaneously. The best supported execution path for one position size may not be the best path for another.

A small exit may receive its strongest result from one pool, while a larger position may execute more efficiently through another venue with greater depth. Concentrated-liquidity pools can further complicate this relationship because active depth depends on the price ranges traversed by execution.[26] Such pools remain outside the released analytical baseline's supported exit models.

Mesh should compare equivalent raw inputs and quote-asset units independently for each modeled size. In the released analytical baseline, this means comparing the existing selected candidate from PumpSwap with the existing selected candidate from Raydium CPMM where both paths are supported. Selection prioritizes exact net proceeds, then exact impact, then deterministic venue precedence; the highest-proceeds quote need not have the lowest percentage impact.[27] Broader within-venue candidate evaluation is a development objective, not evidence that the baseline searched every pool. Best supported exit means best among the evaluated eligible single-venue candidates, not globally optimal or split-route execution.

7.8 Execution Provenance

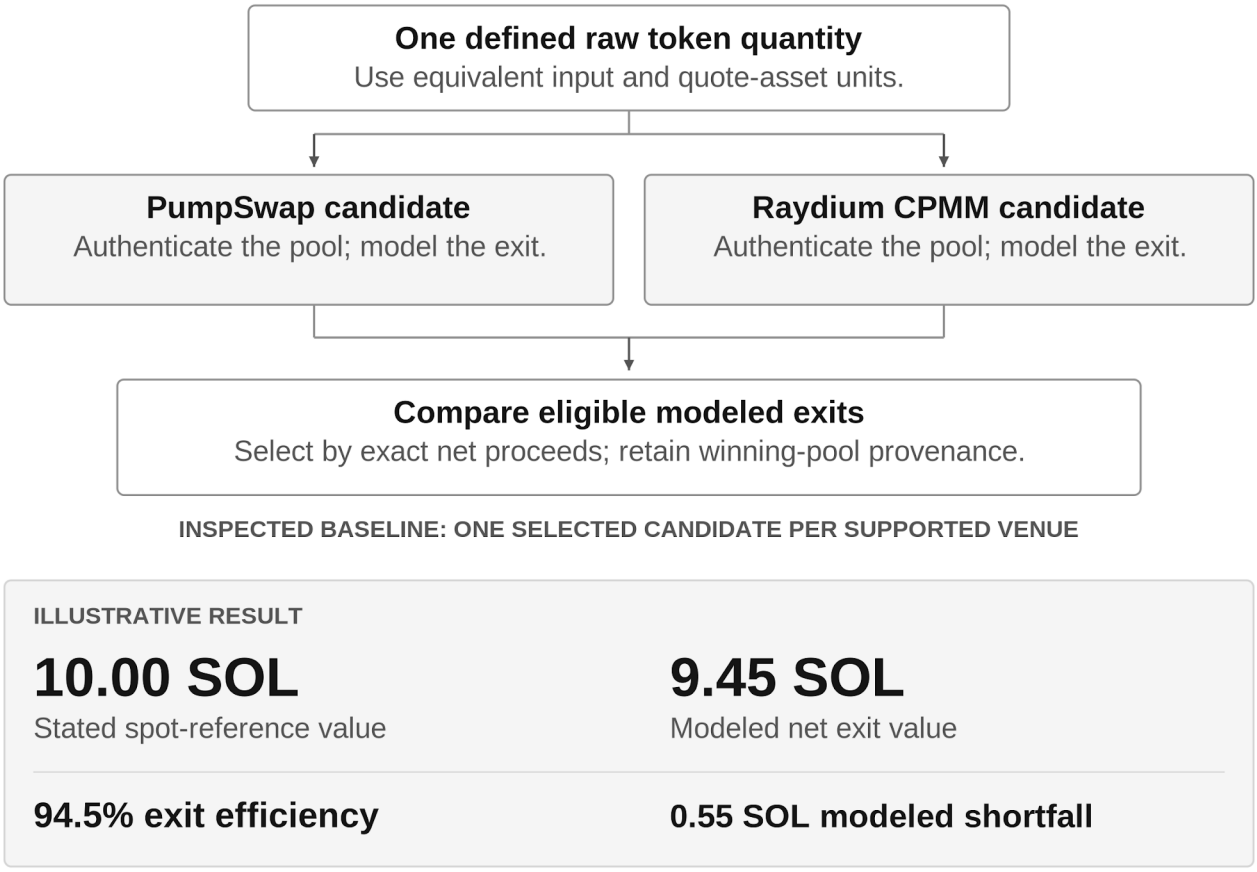
Every material liquidity conclusion should remain attributable to the evidence that produced it. If Mesh reports a modeled exit, the system should retain the venue, pool address, input amount, relevant reserve state, estimated output, applicable supported fees, and other information necessary to reconstruct the result.

Provenance becomes especially important when different position sizes use different winning pools. A 0.1% supply exit may be modeled through one market while a 5% exit uses another. The interface should not display the reserves of one pool while presenting an execution estimate produced by another, because doing so creates a technically precise result with incorrect supporting evidence.

The same standard applies when liquidity analysis later becomes part of pre-sign protection. If Mesh warns that a proposed purchase would result in poor exit efficiency, a user or technical reviewer should be able to determine which supported market conditions generated that conclusion. Figure 3 illustrates position-specific exit analysis and the provenance required to interpret its output.

Position-specific exit analysis

Illustrative existing position; not a live quote or a purchase-and-resale simulation.



Exit efficiency (%) = 100 × modeled net exit value ÷ stated reference value.
Only modeled fees are included. Candidate coverage is bounded; actual execution may differ.

Figure 3. Position-specific exit analysis. The 10.00 SOL reference, 9.45 SOL modeled exit, and 94.5% efficiency reproduce the illustrative existing-position example, not live market data. Comparisons remain bounded to evaluated eligible candidates. Source: Sections 7.2 and 7.5-7.8; Appendix C.
View full-size figure

7.9 Liquidity Risk Is Not Price Prediction

Mesh does not attempt to predict whether the market price of an asset will rise or fall. Liquidity and market direction are separate questions.

A token may subsequently appreciate while remaining difficult to exit at size. Another asset may decline substantially while retaining deep, efficient liquidity. The purpose of the liquidity domain is therefore to estimate the current ability to execute under observed conditions, not to forecast investment performance.

This distinction is essential to BAREMAX's broader positioning. A poor exit-efficiency result is a statement about current execution conditions within the supported model. It is not a prediction that the token is a poor investment. Similarly, favorable liquidity does not imply that the asset will retain its value or that the user should purchase it.

7.10 Reference-Dependent Analysis

Position modeling requires a stated reference. A token-quantity or wallet-derived position establishes a raw token input directly. A SOL-denominated input in the released analytical baseline instead follows an existing PumpSwap reference-sizing path; it is not a simulated purchase, and the corresponding Raydium SOL-sizing path remains unavailable.[27] Future support for a fully modeled purchase must distinguish that operation from reference conversion.

Once a reference position has been established, competing exit candidates should be evaluated against the same raw token input. Allowing each pool to redefine the position size according to its own price would make the resulting execution comparisons inconsistent and could cause apparently superior results to reflect different underlying quantities.

Mesh should therefore preserve the relationship between the reference used to establish the position and the raw amount used for exit modeling. This allows the system to compare execution paths without silently changing the object being compared.

7.11 Poor Liquidity as a Security Signal

Liquidity conditions can become severe enough to justify a security warning even when a technically functioning market exists. Extremely high modeled impact, low exit efficiency, rapidly deteriorating reserve depth, or the absence of any supported quoteable exit may materially affect the user's ability to recover value from a position.

Mesh should communicate those conditions directly while remaining precise about the boundaries of the conclusion. There is an important difference between establishing that no liquidity exists and establishing that no supported, authenticated, and quoteable liquidity source was available to Mesh. Unless the system can justify the broader statement, it should use the narrower one.

This distinction preserves the Evidence Model while still allowing serious execution problems to influence the security result. Incomplete market coverage should create an explicit limitation, while confirmed poor execution through validated markets should remain a positive risk finding.

7.12 Protection Mode and User-Defined Thresholds

Liquidity analysis becomes increasingly useful when it can be compared against security parameters selected by the wallet owner. Mesh should provide sensible default thresholds for ordinary users while eventually allowing more experienced users to define their own acceptable execution boundaries.

As illustrative policy choices rather than published defaults, a user might require additional confirmation when modeled exit impact exceeds 8%, interrupt a supported purchase flow when modeled exit efficiency falls below 90%, or establish stricter limits for particularly illiquid positions. Another user may deliberately tolerate substantially higher impact because of a different strategy or risk tolerance.

The underlying calculation should remain unchanged regardless of the user's preference. If Mesh models 12% impact, the evidence should remain 12% for every user analyzing the same position and state. Personalization determines whether that observation results in an informational message, warning, additional confirmation, or policy-based interruption.

7.13 Continuous Liquidity Monitoring

Execution risk can change after a position has entered the wallet. Liquidity providers may remove capital, active concentrated liquidity may shift, new pools may become available, fees may change, or the wallet's own position size may change relative to the supported market depth.

Continuous Mesh monitoring should therefore be capable of recalculating relevant exit conditions over time. Rather than repeatedly showing the same static liquidity figure, the system can focus attention on material changes in the wallet's realizable position.

A future alert may indicate that the modeled exit efficiency of a position has deteriorated materially since the previous observation or that the liquidity source previously responsible for the strongest supported exit is no longer available. This turns liquidity analysis from a one-time research calculation into an ongoing security signal tied directly to the wallet's exposure.

7.14 Liquidity Within the Mesh

Liquidity should remain a distinct security domain while contributing to the broader Mesh interpretation. Deep liquidity does not eliminate authority, ownership, program, creator, or transfer-behavior risk, just as a technically decentralized authority structure does not guarantee that a large position can be exited efficiently.

The value of the Mesh architecture is that these conditions can be interpreted together without losing their individual meaning. An asset with concentrated ownership and shallow exit liquidity may present a different security environment from one with the same ownership concentration but substantially deeper markets. Similarly, an asset with strong current liquidity may still require caution if an active authority or mutable program can materially change its behavior.

The liquidity domain is therefore designed to answer a specific question within the larger security system: given this wallet, this position, the authenticated markets Mesh currently supports, and the observed execution state, how much value can the user reasonably expect to recover if the position must be exited? That question turns liquidity from a generic market statistic into a wallet-specific security property.

8. The BAREMAX Mesh

Mesh is the core security architecture being developed by BAREMAX Labs. The name describes both the technical structure of the system and the long-term product model: multiple specialized security signals operating as an interconnected layer around the wallet rather than as isolated checks performed only when the user requests them.

Conceptually, Mesh can be understood as a web extending through and around the wallet. Individual nodes represent validated pieces of security evidence, while the connections between them represent relationships that may alter the security significance of that evidence. Authority control, ownership, liquidity, program behavior, creator history, Token-2022 functionality, transaction execution, and wallet exposure are analyzed separately where necessary, but Mesh is intended to interpret how those conditions interact. The long-term objective is for this structure to remain active continuously, allowing BAREMAX to move from reactive analysis toward an autonomous security sentinel.

8.1 A Web of Security Signals

A wallet does not interact with risk through one isolated object. On Solana, transactions invoke programs through instructions that reference accounts containing network state, while token assets may introduce their own authorities, extensions, liquidity dependencies, and supporting programs.[8] The security conditions relevant to a wallet can therefore exist across several layers of the same interaction rather than within the token mint alone.

Mesh connects the analytical areas defined in Section 4 without replacing their individual evidence requirements. Transaction-level and wallet-level interpretation should build on those findings as the corresponding integrations become available.

An active authority, for example, may warrant additional attention when the same controller also governs transfer fees or programmable transfer behavior. Concentrated ownership may become more consequential when available exit liquidity deteriorates. A transfer hook may appear operationally acceptable while depending on a program whose executable logic remains upgradeable. Mesh is intended to preserve the individual findings while identifying these relationships across the broader security state.

8.2 Nodes and Relationships

Within the Mesh model, a node represents a security-relevant observation that has survived the validation required for its domain. Examples may include a mint authority address, a program upgrade authority, an observed ownership concentration measurement, a liquidity-pool reserve, a transfer-hook program, a creator-history transaction, or a modeled exit quote.

Relationships express how those observations affect one another: shared authority control, program dependencies, historical deployment links, or the liquidity supporting a particular position. They must be supported by evidence rather than inferred merely because two observations appear together.

The objective is not to construct complexity for its own sake. The web is a conceptual model for relationships whose security significance can change together. It should not be read as evidence that a persistent graph database, autonomous wallet-wide correlation engine, or every proposed cross-domain rule is already deployed. The implementation boundary in Section 10.1 distinguishes existing address grouping and domain analysis from that longer-term architecture.

8.3 The Wallet as the Center of the Mesh

The current development process begins largely with assets and the evidence surrounding them, but the long-term architecture places the wallet at the center of Mesh. This is an important shift in how the security problem is defined.

A token scanner primarily asks what risks exist around a particular asset. A wallet-centered security layer can ask which risks actually matter to the wallet currently being protected. The distinction allows Mesh to incorporate the user's holdings, position sizes, interacting programs, available exit liquidity, relevant token authorities, and proposed transactions into the analysis rather than evaluating every asset in isolation.

Solana's consumer wallet ecosystem already treats the wallet as the point through which users hold assets, approve transactions, and connect to applications, and Solana supports standardized wallet connection patterns for application developers.[30] Mesh is intended to add a persistent security layer around those interactions without taking custody of the wallet or its signing keys.

8.4 From Reactive Analysis to Persistent Monitoring

The earliest versions of Mesh will necessarily be more reactive. A user may provide a token address, connect a wallet, or explicitly request an analysis. Over time, BAREMAX Labs intends to reduce the amount of manual intervention required by shifting toward continuous observation of the wallet's security state.

Persistent monitoring should identify material changes in the supported state affecting assets already held and determine which wallet exposures need reassessment. Sections 11.7-11.9 describe the observation, change-detection, and alert-delivery boundaries; this is more than scheduling identical scans without retaining context.

This transition is fundamental to the product direction because persistent monitoring reduces the need for users to repeatedly reconstruct an asset's security context. The longer-term sentinel experience depends on maintaining that context and identifying material changes, rather than only returning another point-in-time scan.

8.5 Autonomous Sentinel

BAREMAX Labs intends for the mature form of Mesh to operate as an autonomous security sentinel. Autonomy refers to observation, analysis, alerts, and user-configured controls within supported signing paths, not discretionary management of the user's assets. Automatic sales, fund transfers, and changes to onchain permissions are excluded from the 2027 product scope. Any asset-moving or permission-changing transaction presented through Mesh would require the user's explicit approval of that transaction in the wallet; enabling monitoring or Protection Mode would not grant standing authority to execute it.

The sentinel model consists of several functions operating together. Mesh observes relevant onchain state, validates the resulting evidence, interprets material changes, compares those findings against the user's protection preferences, and determines whether an alert or intervention is warranted. A user might therefore be notified because the upgradeability state of a relevant program changed, an authority was rotated, or the modeled exit efficiency of an existing position deteriorated materially.

Paid tiers are intended to extend this model through deeper acquisition, higher-frequency monitoring, configurable policies, and additional supported protection capabilities as they are validated. Subscription status should not alter the interpretation of identical evidence. Additional analysis can legitimately change a conclusion when it establishes new material evidence, but its coverage and observation time must remain visible. Material findings already detected for a user must not be concealed or deliberately delayed to create an upgrade incentive.

8.6 Pre-Sign Mesh

The planned pre-sign stage places Mesh before execution, when the user still has the opportunity to reconsider a proposed interaction. Solana transactions contain messages and instructions that specify the programs and accounts involved in execution, providing part of the onchain structure necessary for transaction-level inspection.[8] Applications within the Solana ecosystem already deliver transactions to wallets for preview and signing across web and wallet contexts, demonstrating the importance of the signing boundary as a point of interaction between applications and users.[31]

Future pre-sign analysis should relate supported transaction effects to the wallet's existing exposure. Section 11 specifies the required transaction binding, program and account interpretation, expected-state analysis, and failure handling. The broader vision does not imply that every transaction type will be supported by the first pre-sign beta.

This moves the security question from whether an asset contains an isolated warning to whether the proposed transaction can produce an outcome inconsistent with the user's expectations or protection policy. Pre-sign analysis therefore represents the point at which the broader Mesh architecture begins to function as preventative security rather than only analytical security.

8.7 Protection Mode

Protection Mode is intended to convert validated Mesh findings into configurable protective behavior. The underlying evidence remains objective, but the response to that evidence can reflect the user's own security preferences.

A user might configure Mesh to require additional confirmation when material evidence remains UNKNOWN, interrupt transactions classified as HIGH RISK, enforce a maximum modeled exit-impact threshold, or react to specific authority configurations. Another user may deliberately accept a broader range of conditions. Mesh should report the same underlying security facts to both users while allowing their individual policies to determine the degree of friction introduced before execution.

This architecture preserves self-custody and user control. The wallet owner defines the conditions under which Mesh should warn, interrupt a supported signing flow, or require additional confirmation. These controls do not give BAREMAX discretionary signing authority, do not apply universally to transactions signed elsewhere, and do not authorize the automatic asset movement excluded in Section 8.5. Deeper integration should reduce friction without changing those boundaries.

8.8 Mesh Free

Mesh Free is intended to provide useful on-demand Solana asset analysis and baseline security context for one wallet as wallet-aware functionality becomes available. The foundation includes supported domain findings, material evidence limitations, plain-language explanations, and the supporting evidence already acquired for a result. Free is the entry-level analytical experience, not the complete autonomous service with only a lower request allowance.

As monitoring is validated, Free should add baseline observation of a limited, disclosed set of supported holdings, essential change alerts, and standard warning settings. Advanced historical investigation, retained security timelines, expanded execution scenarios, and configurable transaction-protection policies are intended paid capabilities. Published limits will identify analysis allowances, monitored assets, observation cadence, and acquisition depth; one-wallet access does not imply exhaustive or unlimited coverage.

When a material threat has already been detected for a Free user, its warning and essential explanation must remain available without upgrading. Findings must not be replaced with an undisclosed paywall, and a monitoring limit or unavailable check must not be presented as a clean result. Paid acquisition may establish additional evidence that Free did not retrieve; the product must distinguish that coverage difference from withholding an existing finding.

8.9 Mesh Pro

Mesh Pro is intended to be the primary paid security service for active traders managing one wallet. It combines the Free foundation with a substantially broader set of analytical, monitoring, and policy tools. The upgrade should change what users can investigate and automate, not merely how often they can run the same scan. Pro remains part of the same Mesh architecture: evidence standards, result explanations, and self-custody do not change with the subscription. The capabilities below are development objectives to be introduced as validated, not a claim that the full Pro service is available at the initial public beta.

Persistent monitoring is intended to cover a larger supported set of the wallet's holdings at a higher observation frequency, automatically reassessing material authority, program, token-configuration, ownership, and liquidity changes. Retained security timelines and comparisons should show what changed, when it was observed, and why the conclusion changed. Deeper bounded creator and deployment-history acquisition should support investigations beyond the Free result, while preserving coverage limits rather than treating a larger search budget as complete knowledge.

A dedicated position-analysis workspace should allow users to compare supported exit scenarios at different sale sizes, inspect the eligible venues and assumptions behind each result, and save or export analysis records. Advanced pre-sign views should connect interpretable asset movements and permission changes to relevant wallet exposure within documented participating flows. Configurable Protection Mode policies should let users set evidence requirements, authority restrictions, and execution thresholds, with warnings or local signing-flow interruption only where the relevant integration and control have been validated.

Pro should also provide configurable alert rules, severity-based delivery preferences, and retained alert context so that repeated observations can be grouped without concealing material changes. Higher analysis allowances and monitoring capacity support these tools rather than substitute for them. Additional checks may establish evidence outside Free coverage, but findings and explanations already delivered to Free users remain accessible under Section 8.8. Pro's commercial value is the additional investigation, continuity, and user-directed automation it performs.

Pro pricing will be set against the usefulness of the released feature set and measured service costs, with the objective of a fair subscription and a healthy, sustainable operating margin. The cost model must support infrastructure, user support, security review, continued development, and meaningful Free access. The published specification should state the features actually available, usage and retention limits, observation cadence, supported integrations, and exclusions. Neither unlimited monitoring nor unvalidated future functionality should be used to justify the subscription.

8.10 Mesh Max

Mesh Max is a later-stage subscription intended to extend Pro across multiple user-selected wallets and add analytical capabilities that become useful at that scale. Late 2027 is an earliest planning possibility, conditional on stable multi-wallet infrastructure, a proven Pro experience, and validated additional detection features. Max is not a prerequisite for the public beta or the first Pro release, and it should be deferred if those conditions are not met.

The intended multi-wallet foundation includes consolidated security state, per-wallet and aggregate exposure views, reusable protection policies with wallet-specific overrides, retained analysis history, and position-specific exit analysis. Shared dependency analysis should identify when several selected wallets are exposed to the same supported asset, authority, program, or liquidity source without confusing a technical relationship with proof of common beneficial ownership.

Additional Max objectives include automatic reassessment of affected positions after a material shared-state change, correlation of supported warning combinations across the selected wallets, and consolidated incident timelines that reduce duplicated alerts while preserving the underlying evidence. These are proposed autonomous detection and analysis capabilities, not promises to predict every attack or move assets in response. Each requires its own validation, coverage description, and operational limits before being offered.

Max should extend rather than hollow out Pro: advanced single-wallet protection remains a reason to subscribe to Pro, while Max adds coordinated multi-wallet analysis and a validated suite of additional detection tools. Wallet counts, retained history, and aggregate workloads must be disclosed. All tiers share the same evidence-integrity standard and the obligation to reveal material findings already detected for the user. The prohibition on automatic asset movement in Section 8.5 applies equally to Max.

8.11 Consumer-First Development

BAREMAX Labs intends to remain consumer-focused during the first major phase of Mesh development, beginning with active retail Solana memecoin traders and less-experienced users participating in the same environment. That initial audience defines the first workflows and explanations to validate; it does not limit the eventual relevance of Mesh to other Solana wallet users. Before the architecture is offered as infrastructure to other organizations, it should demonstrate useful and understandable analysis in its own consumer product.

Consumer use creates a particularly valuable feedback environment for a security product. Real users can reveal where warnings are confusing, where alerts become excessive, where unsupported behavior appears, which risk conditions actually alter decisions, and which parts of the system produce false positives or false reassurance. Those observations can then become improvements to evidence handling, protocol support, regression coverage, and product presentation.

This development sequence also keeps the immediate objective clear. BAREMAX Labs is initially building Mesh as a product users choose to protect their own wallets, not as a generic security API searching for downstream applications.

8.12 Institutional Expansion

If Mesh demonstrates sufficient reliability and utility at the consumer level, selected capabilities may later become useful to other organizations operating within the Solana ecosystem. Potential integrations could include wallets, funds, trading platforms, fintech applications, payment systems, programs, or other infrastructure providers.

In those environments, BAREMAX Labs may eventually expose parts of the Mesh through an API, SDK, or dedicated institutional integration. Another product could potentially request the evidence or security interpretation associated with an asset, program, wallet interaction, or proposed transaction without reproducing the entire BAREMAX analytical pipeline internally.

This remains a later-stage direction rather than the primary initial business model. The objective is to establish the consumer security system first, learn from its operation under real conditions, and consider infrastructure partnerships only once the underlying Mesh is mature enough to justify external dependence on its conclusions.

8.13 Visual and Product Identity

The visual identity of Mesh is intended to reflect the architecture itself. The wallet sits at the center of an interconnected web, with security nodes extending throughout and around it. Those nodes correspond to the authorities, programs, assets, liquidity conditions, historical evidence, transactions, and other signals relevant to the wallet's security state.

The web metaphor is useful because it does not imply that protection depends on one checkpoint. A disturbance at one point can affect several connected relationships. An authority change may alter the security state of an asset; that asset may represent a substantial wallet position; the position may simultaneously depend on deteriorating liquidity; and the combined condition may require a stronger response than any observation would justify independently.

The visual design should represent this intended network of validated relationships, not imply that every connection is known or that the full autonomous architecture is already deployed.

8.14 Long-Term Mesh State

At maturity, the user's relationship with BAREMAX should be substantially different from using a conventional scanner. The system should already know what the connected wallet holds, the position sizes involved, the principal authorities and programs affecting those assets, the supported liquidity available to them, and whether material security conditions have changed since the previous observation.

When the wallet prepares to interact with the network, Mesh should be able to incorporate that existing context into the proposed transaction analysis rather than beginning from zero. After execution, the resulting exposure becomes part of the monitored security state. Over time, the system can therefore maintain continuity across analysis, execution, and post-transaction monitoring.

The long-term architecture can be summarized as a progression from isolated asset analysis to persistent wallet awareness, pre-sign interpretation, continuous monitoring, and user-configured protection. The objective is for security to become an ongoing property of the wallet experience rather than a separate research activity performed only when the user remembers to initiate it.

9. Security Against False Confidence

Mesh treats false confidence as a security failure: a result can mislead users not only by overlooking a threat, but by presenting incomplete or invalid evidence as a completed clean analysis. Section 5 establishes the evidence rules; this section describes how those rules should govern product behavior, incident handling, and user trust.

The objective is not to eliminate every false positive or false negative, but to prevent failure paths from silently becoming reassuring output. This follows the secure-design principle that an error in a security control should not create a more permissive outcome.[18]

9.1 The Cost of a False SAFE

A false warning can delay an interaction or discourage a legitimate transaction. Unsupported reassurance can instead encourage a user to proceed under the mistaken belief that material checks were completed. The asymmetric burden of proof in Section 5.3 therefore matters at the decision point: the interface must distinguish a supported favorable finding from a check that could not be completed. A positive label must not conceal the latter.

9.2 Prefer Uncertainty to Unsupported Confidence

When missing evidence could materially change the conclusion, an explicit UNKNOWN result with an explanation is preferable to a green result the system cannot defend. This is not a reason to escalate every missing datapoint or unusual configuration. Section 5.5 defines the materiality distinction, and the development objective remains to reduce unnecessary uncertainty through better acquisition, validation, and protocol coverage rather than lower the standard for confidence.

9.3 Malformed Evidence Is Not Negative Evidence

The failure-handling rules in Sections 5.8 and 6.12 must survive every layer between acquisition and the interface. Invalid authority bytes must not become revocation, an unusable balance must not become zero, and malformed extension data must not become an absent extension. Whole-TLV inspection failure remains distinct from a controller-local UNKNOWN classification that preserves a validated authority address. Default values or partially decoded output must not obscure which boundary failed.

The fail-secure principle applies to the conclusion as well as the software path: an evidence-processing exception must not produce reassurance that the same analysis could not justify with valid input.[18]

9.4 Preserve Proven Risk

A material finding supported for the observation being reported should remain visible when an unrelated domain is unresolved. The overall result must not average away a severe condition or discard it merely because another lookup failed. Where a finding is retained from an earlier observation, its timestamp and freshness limits remain part of the evidence; a verified state change can supersede it. Section 5.6 establishes this distinction between information loss and a genuine improvement in the underlying state.

9.5 Conflicting Evidence

When observations disagree, Mesh should attempt reconciliation using the provenance and validation rules described in Section 5.7. Unresolved material disagreement must remain visible rather than being removed by selecting the cleaner result. Independently valid observations can be retained without representing the disputed condition as fully resolved. The record should allow a reviewer to understand both the conflict and its effect on the conclusion.

9.6 False Positives and Alert Discipline

Avoiding false reassurance does not justify excessive warnings. A security layer that routinely treats ordinary behavior as dangerous risks training users to ignore its output. An active authority, programmable extension, upgradeable program, or concentrated holder can be relevant without establishing malicious intent.

Mesh should distinguish those conditions through protocol-specific evidence and context. Alert severity should reflect the supported finding, while repeated observations should be grouped without suppressing a material change. The objective is to reduce both missed threats and unnecessary friction, not maximize the number of warnings displayed. User feedback on misunderstood or excessive alerts should inform the same failure-analysis process as reports of missed risks.

9.7 Transparent Failures and Postmortems

BAREMAX Labs intends to examine material security mistakes as engineering events. Review should establish what evidence was available, why the conclusion was incorrect, which assumption or implementation path failed, and what corrective action and regression coverage are required.

Where appropriate, a public postmortem should describe the failure, impact, technical cause, remediation, and tests added afterward. Disclosure must protect user privacy and coordinate with remediation rather than unnecessarily enabling exploitation before mitigations are available. The purpose is to turn discovered failures into more reliable future behavior, consistent with NIST's emphasis on addressing root causes and reducing recurrence.[6]

9.8 Trust Through Verifiability

The user should be able to move from a conclusion to the principal evidence supporting it, then to the relevant technical record as needed. The progressive interface described in Section 10 organizes that access without requiring every user to inspect raw accounts, programs, or transaction records.

This standard applies to favorable results as well as warnings. Proposed SAFE eligibility must be inspectable, with the resolved checks, observation context, and material scope limitations available rather than replaced by an unexplained green indicator.

9.9 Security Against False Confidence as a System Property

A disclaimer cannot repair a result that lost the distinction between missing and valid evidence earlier in the pipeline. Acquisition, structural validation, domain interpretation, aggregation, and presentation must preserve the same evidence meaning. Release review should therefore evaluate the complete path to the user-facing result, including failure paths, rather than checking only individual calculations. The standard is not that Mesh must always provide a definitive answer, but that every answer remains justified by the evidence it presents.

10. Current Product Experience

The current BAREMAX development effort is focused on turning the underlying Mesh security architecture into a product that can communicate technically complex Solana risk without requiring the user to interpret raw account data, program state, transaction history, or liquidity structures independently. The analysis engine already operates across several of the domains described earlier in this whitepaper, while the public-facing product is intended to organize those findings into a progressively understandable security experience.

The first public version of Mesh is planned as a web application. Its purpose is not to expose every internal calculation immediately, but to present the most important security conclusion first, explain the evidence responsible for that conclusion, and allow the user to inspect deeper technical detail when desired. Because Solana applications operate through accounts, programs, instructions, and transactions, and because token behavior can extend through both the original Token Program and Token-2022, the product must translate several layers of protocol-specific information into a coherent result.[7][8]

10.1 Released Analytical Foundation

This revision uses the protected beta released on October 28, 2026 as its implementation baseline. The deployed application source is commit `4cf69b52daf12b7c5e91eed0d2052ee8d035ee7b`; the accompanying release and CI documentation is retained at main commit `bfa472a1e2e9e16237408ddc498fc1ef89507882`. These identifiers describe different records of the same release, rather than interchangeable evidence-observation times.[34]

The beta combines SPL Token and Token-2022 inspection with six risk domains: authority, creator, ownership, exit liquidity, transfer restriction and transfer fee. Program-upgradeability and extension evidence inform those domains. It also provides bounded relationship, lifecycle and liquidity-history evidence, position-sized models and a read-only interface with inspectable explanations.[9][17][32]

Authenticated exit modeling covers selected supported PumpSwap and Raydium CPMM candidates paired with SOL/wSOL. It does not search every pool, construct split routes or simulate an executed sale. A bounded transfer preflight examines a one-raw-unit transfer scenario, not a trade from the user’s account. The separate token-account conditions candidate remains local/mock-only.

Release verification established authenticated analyses and selected live metadata, image and chart paths. Provider throttling, stale or conflicting market records and unavailable evidence remain operational limitations. A refreshed chart does not refresh the security assessment. This editorial review does not claim to have rerun the release’s entire test suite or reverified the hosted backend.

Table 1. Development status as of October 28, 2026

State	Scope	Boundary
Released protected beta	Read-only analysis, control console, evidence investigation, supported position-exit models, separate metadata and market history.	Access-controlled beta. Limited protocol coverage and provider-dependent observations.
Reviewed backend candidate	PR #62, commit 732c8ff: exact sampled LP balances, recorded authorities, accounting-supply gaps and historical burn references.	Unmerged at this revision’s review checkpoint. No recognized locker, unlock condition or verified protected percentage is claimed.
Isolated candidates	Further account conditions, recovery/provenance work and landing-page refinements.	Candidate validation is separate from release integration. Local/mock-only paths are not hosted capabilities.
Planned architecture	Persistent wallet monitoring, pre-sign inspection, user-selected protection and deeper wallet integration.	Requires separately supported integrations, appropriate review and release readiness.

10.2 The Protected Beta Experience

The beta begins with a Solana mint address. A user can optionally supply a supported position input, such as a token quantity, SOL-reference amount or public wallet address. A public address can provide read-only position context; it is not a connected signing session.

The overview presents token identity, a capability console and expandable findings, followed by supporting market context and modeled exits. Overall assessment and evidence coverage remain distinct. The full investigation exposes addresses, conditions, history and Proof without requiring every technical detail to occupy the initial dashboard.

The interface consumes the backend’s analytical result. It does not create a second risk model. Metadata and price history are separate descriptive sources: a recognizable name or image does not authenticate a token, and a market chart does not establish an executable quote.

10.3 Layered Security Results

The product should present information in layers rather than forcing every user into either an oversimplified score or a raw technical report. At the highest level, the user sees the overall Mesh conclusion, primary risk drivers, and evidence coverage. Beneath that summary, each security domain can be expanded independently.

An illustrative domain summary might show Authority Risk as HIGH, Creator Risk as UNKNOWN, Ownership Risk as ELEVATED, and Exit-Liquidity Risk as LOWER. Coverage should be displayed

separately using the actual versioned contract for the result: a conceptual PARTIAL label must not be substituted for a different implemented global or domain-specific field. LOWER is not equivalent to the proposed consumer SAFE label. These distinctions preserve the fact that a strong finding can coexist with incomplete evidence without presenting an aspirational interface as a current schema.[9][17]

This layered model also avoids implying that all HIGH results are equivalent. A HIGH conclusion caused by a concentrated set of privileged authorities is meaningfully different from one driven by severe execution conditions or another risk category. Users should be able to identify that distinction immediately and inspect the relevant evidence if they want to understand it further.

10.4 Evidence Coverage in the Interface

Evidence quality should remain visible alongside the security conclusion. Section 5's five terms describe the intended shared vocabulary, while the inspected implementation uses domain-specific data alongside global evidenceCoverage and unknownSignals fields.[9][17] The interface should preserve those actual distinctions. In particular, a non-UNKNOWN risk label does not by itself demonstrate complete evidence: ownership sample resolution, historical acquisition, and exit-model coverage require separate interpretation.

The distinction should remain understandable without requiring knowledge of the internal pipeline. If creator history is partial, for example, the product can explain that the observed historical activity is valid but may not represent the creator's complete deployment history. If program upgradeability is UNKNOWN, the interface should identify the unresolved program evidence rather than implying immutability simply because an upgrade authority was not successfully recovered.

This allows the product to expose uncertainty without making the experience unnecessarily technical. Users should understand when a limitation matters even if they never inspect the raw acquisition or validation state underneath it.

10.5 Authority Analysis

Authority analysis should present privileged control in terms of what the controller can actually do. The SPL Token model includes mint and freeze authorities, while Token-2022 adds additional authority-bearing capabilities through extensions such as transfer fees, permanent delegates, pausability, metadata controls, and other programmable functionality.[7] The product should therefore communicate both the individual authorities and any important concentration of control across them.

A basic user may see that an active freeze authority exists and that the same controller retains several sensitive capabilities. A deeper view can expose the controller address, account classification, associated extension or program, mutability information, and the evidence used to establish the relationship. This converts what would otherwise be a list of addresses into an explanation of the asset's control structure.

The same principle applies when an authority is absent or revoked. The positive conclusion should appear only when the relevant state has been decoded and validated successfully rather than because a lookup returned no usable value.

10.6 Ownership and Creator Analysis

Ownership analysis should translate token-account balances into a more useful picture of observable control. Raw holder rankings may include automated market makers, program-controlled accounts, or several accounts belonging to the same controller. Mesh is therefore intended to distinguish recognized market infrastructure from ordinary wallet exposure where the evidence permits, while clearly describing the boundaries of the sample being analyzed.

Creator analysis follows a similar evidence-aware presentation. If Mesh observes one historical launch under PARTIAL coverage, the user should see that one launch was observed, not that the creator has conclusively launched only one token. A technical view may expose the supporting deployment transactions, creator address, coverage state, and acquisition method, while the consumer view explains only the part of that information necessary to understand the conclusion.

These views are designed to prevent precise-looking numbers from implying greater knowledge than the system actually possesses.

10.7 Liquidity and Position Analysis

Liquidity should be presented from the perspective of the wallet rather than as a generic pool statistic. When a position is known, Mesh can estimate the amount currently represented by that position, the supported exit output, modeled execution loss, price impact, and exit efficiency.

For illustration, an existing position valued at 10 SOL using a stated spot-reference basis might have an evaluated net exit of 9.45 SOL: 94.5% exit efficiency and a 0.55 SOL modeled shortfall. The expanded result should identify the selected venue, pool, raw input, reference basis, observation context, and modeled fees. This example is not a product threshold, a guaranteed fill, or proof that a 10 SOL purchase would immediately lose 0.55 SOL. A future purchase-and-exit model must use the post-purchase state described in Section 7.4.

The interface should also allow several position sizes to be compared when useful. This makes it possible to demonstrate that execution risk grows as the modeled position becomes large relative to available liquidity, rather than giving the user a single liquidity number that appears equally applicable to every wallet.

10.8 Token-2022 Presentation

When an asset uses Token-2022, Mesh should surface the programmable behavior that materially affects the user's security assumptions. The Token Extensions Program allows optional functionality to be added to token mints and token accounts, including transfer fees, permanent delegates, transfer hooks, pausability, non-transferability, and other extensions.[7]

The product should avoid presenting those extensions as an undifferentiated list of warnings. A Transfer Hook result, for example, should indicate that custom transfer logic is present, identify the associated program where possible, and explain whether Mesh has established the program's mutability. A Permanent Delegate result should explain that the mint-level delegate can authorize transfers or burns across token accounts for the mint and that individual account owners cannot revoke the delegate themselves.[16]

At the technical level, the user should also be able to inspect the authority, program, extension state, controller classification, and evidence quality responsible for the interpretation. This makes Token-2022 analysis one of the clearest examples of progressive disclosure within the product.

10.9 Progressive Explanations

The same security condition should be understandable at several levels of expertise. A basic explanation may state that an asset can currently be frozen by an active authority. An intermediate explanation can identify that the freeze authority is controlled by a standard wallet that also controls another privileged capability. A technical view can then expose the controller address, decoded mint state, account classification, related authorities, and evidence provenance.

This approach allows BAREMAX to remain technically rigorous without forcing advanced terminology onto every user. The security engine does not change according to experience level; only the amount and form of information presented changes.

Progressive explanation also supports trust. A user who initially relies on the simplified result can inspect increasingly detailed evidence when a transaction is important enough to justify additional review, while an experienced user can move directly toward the underlying technical state.

10.10 Wallet Context and Connection

The released beta can use a supplied public wallet address to derive a supported position for a specified mint. This is read-only context. It does not connect to a wallet, request a signature, discover a complete portfolio or continuously monitor holdings.

A future connected-wallet experience may reduce repeated input and support a broader view of exposure. Connection permissions, transaction interpretation and signing controls require their own supported integration and validation. No connection feature should be presented as permission for Mesh to take custody or move assets.

10.11 Beta Scope and Product Restraint

The first public release does not need to contain every capability described in this whitepaper. The v0 objective is to make the existing analytical foundation usable, understandable, and reliable enough for real users while preserving the evidence standards developed throughout the system.

The expected scope includes web-based Solana asset analysis, authority risk, creator history, ownership analysis, exit-liquidity modeling, Token-2022 analysis, explicit evidence states, and explainable domain-level conclusions. Wallet connection is also a current v0 objective, subject to final reliability and security requirements before release.

Continuous monitoring, comprehensive pre-sign analysis, transaction interruption, native mobile protection, multi-wallet Mesh, broad institutional integrations, and complete coverage of every Solana program or liquidity venue are later stages. Keeping those boundaries explicit is important because the public beta should demonstrate a reliable subset of the architecture rather than imply that the entire long-term Mesh already exists.

10.12 Feedback as Part of Security Development

The public beta is also intended to expose Mesh to a wider range of real-world assets, programs, token configurations, market structures, and evidence failures than can be reproduced through controlled internal testing alone. User feedback should therefore be treated as part of the security-development process rather than only as product feedback.

A false positive, unexplained UNKNOWN state, unsupported token structure, incorrect pool interpretation, malformed evidence case, or confusing explanation can reveal a weakness in the analytical or presentation layer. Where the issue is technically meaningful, it should become a reproducible test case, a correction to the evidence pipeline or domain methodology, and regression coverage designed to prevent recurrence.

This approach is consistent with the broader BAREMAX development philosophy described earlier: the system should become more reliable by learning from concrete failures and strengthening the conditions under which conclusions are allowed to reach the user.

10.13 What the Public Beta Must Establish

The first public beta does not need to prove that Mesh can protect against every onchain threat. It needs to establish that the underlying security model provides information users can understand and use, that material risk can be surfaced without hiding the supporting evidence, and that explicit uncertainty improves the integrity of the result rather than making the product unusable.

The beta should also test whether wallet-specific analysis materially improves the experience. If users find value in understanding their actual exit conditions, relevant authorities, programmable token

behavior, and security-state changes rather than viewing generic token information, that provides evidence for the transition toward continuous wallet protection.

The first release is therefore best understood as the public validation of the Mesh security model. The product should present a simple conclusion first, explain the most important reasons for that conclusion immediately afterward, and allow the user to inspect the underlying evidence as deeply as necessary.

11. Future Wallet Integration

The long-term direction of Mesh is to move from analyzing assets and wallet state on demand toward providing persistent security around the wallet itself. The web application establishes the analytical foundation, but the intended product is not one that requires users to repeatedly return to BAREMAX, enter an address, and request another scan. As the system becomes sufficiently reliable, security analysis should move progressively closer to the transactions, assets, and programs with which the wallet is actually interacting.

This progression must remain non-custodial. BAREMAX Labs does not intend to hold user assets, private keys, or seed phrases, and Mesh should not require custody in order to provide analysis or protection. Solana applications can connect to compatible wallets through standardized wallet interfaces while the wallet retains signing authority, providing a technical foundation for BAREMAX to observe relevant public state and request user-authorized interactions without assuming control of the underlying keys.[35]

11.1 Non-Custodial by Design

The wallet owner should remain the ultimate authority over the wallet at every stage of the Mesh roadmap. BAREMAX should never require a seed phrase or private key for ordinary analysis, monitoring, or transaction protection, and those credentials should never be requested as part of the consumer product experience.

A connected public address already provides access to substantial onchain information relevant to Mesh, including balances, token accounts, program relationships, transaction history, and other observable state. When a transaction requires authorization, the wallet remains responsible for producing the required signature. On Solana, a signature authorizes the serialized transaction message presented to the signer; changing the message, accounts, instructions, blockhash, or other signed content requires a new signature.[36] This signing boundary is important because it allows Mesh to develop protection around the authorization process without becoming the custodian of the user's signing key.

Non-custodial design does not eliminate every security risk, particularly if the wallet application or user's device is itself compromised, but it prevents BAREMAX from introducing a new centralized key-custody dependency into a product whose purpose is to improve wallet security.

11.2 Connected-Wallet Analysis

The first major step beyond standalone asset analysis is persistent wallet context, which must be distinguished from an always-open wallet session. Wallet connection identifies an address and enables supported application requests while authorization remains with the wallet.[35][37] Public-address monitoring can continue through configured backend observation after the browser or wallet session closes; it does not require signing credentials or imply permission to sign transactions.

Once connected, Mesh can shift from asking what risks exist around a particular token to determining which security conditions are relevant to the actual wallet. The system can identify supported holdings, position sizes, associated authorities and programs, position-specific liquidity conditions, and other exposures that would otherwise require the user to enter assets manually.

This context also improves prioritization. A material change affecting an asset that represents a significant wallet position may deserve more immediate attention than the same change affecting an asset the wallet does not hold. Wallet connection therefore provides more than convenience; it becomes an important input into personalized security analysis.

11.3 Browser-Based Protection as an Intermediate Layer

A browser extension is the planned intermediate step between the web application and deeper wallet integration. It can place Mesh closer to proposed interactions, but transaction visibility must be established through an explicit supported integration rather than assumed from installation or wallet connection. Only transactions actually delivered to the Mesh analysis path can receive its pre-sign assessment; an extension is not a network-wide interception mechanism.

The extension can eventually provide transaction context, risk warnings, asset analysis, program information, position-specific liquidity analysis, and browser notifications before the user completes an interaction. This environment also gives BAREMAX Labs greater control over development and testing than would be possible if early pre-sign functionality depended immediately on integrations with third-party wallet providers.

The browser extension should not, however, become the permanent center of the BAREMAX experience. It is better understood as a development stage through which Mesh can prove reliable transaction-level protection before the system moves toward more seamless wallet and mobile integration.

11.4 Deeper Wallet Integration

Where technical access and commercial relationships permit, BAREMAX Labs ultimately intends to pursue deeper integration with major Solana wallet infrastructure. The higher the degree of reliable integration, the less security work the user should need to perform manually.

A deeper wallet integration could allow Mesh analysis to appear naturally within the transaction flow rather than requiring the user to move between a decentralized application, a BAREMAX interface, and a separate wallet window. Security information could be available at the point where authorization is requested, with the transaction's programs, accounts, assets, authorities, expected state changes, and relevant wallet context already incorporated into the analysis.

Any integration with specific wallet providers would depend on future technical capabilities and partnerships and should therefore be treated as a product direction rather than a current commitment. BAREMAX should maintain the ability to provide independent protection through its own interfaces while seeking integrations that reduce friction and improve how closely Mesh can operate to the signing boundary.

11.5 Pre-Sign Transaction Analysis

Pre-sign analysis is one of the most important transitions in the Mesh roadmap because it moves the system from describing existing risk toward evaluating an interaction while the user still has an opportunity to stop it. Solana transactions contain a message identifying the relevant accounts, recent blockhash, and compiled instructions, and individual instructions identify the target program, participating accounts, and instruction data.[8] Signatures authorize the transaction message itself rather than a human-readable description of the user's intent.[36]

The long-term pre-sign system should therefore analyze substantially more than the existence of a valid transaction. Relevant inputs may include the programs being invoked, accounts receiving writable or signer privileges, assets expected to enter or leave the wallet, destination accounts, Token-2022 behavior, authority relationships, program upgradeability, expected balance changes, current wallet exposure, and the modeled liquidity conditions that would surround a newly created position.

A technically valid transaction can still produce an outcome the user did not understand or intend. Future Mesh decisions should therefore be bound to the exact transaction message and analysis context. A changed account, instruction, permission, blockhash, or other signed field requires renewed validation rather than reuse of an approval for different bytes.[36] Simulation may contribute evidence without broadcasting the transaction, but its output depends on the queried state and is not a guarantee of settlement.[38] Material unsupported behavior, stale context, or analysis failure must remain visible.

11.6 Expected Wallet-State Analysis

A mature pre-sign system should increasingly reason about the expected state of the wallet after execution. Rather than examining instructions only as isolated pieces of transaction data, Mesh should attempt to determine what assets, permissions, balances, and exposures are expected to change if the transaction succeeds.

Solana executes the intended instruction effects sequentially and atomically: an execution failure prevents a partial successful application of those effects.[8][39] This does not mean the fee payer experiences no cost; transaction fees are charged even when execution fails.[40] Predicting a successful

transaction's effects still requires supported program interpretation, and a future purchase-and-exit estimate must use the post-purchase state described in Section 7.4.

For a proposed purchase, expected-state analysis could combine the anticipated token position with authority analysis, Token-2022 behavior, program dependencies, and immediate exit modeling. For other interactions, it may focus on assets leaving the wallet, new account permissions, or other state changes relevant to security. The objective is to move closer to answering whether the wallet's expected post-transaction state matches what the user believes they are authorizing.

11.7 Post-Transaction Monitoring

Pre-sign analysis addresses risk before execution, but the security state of an asset can change after it enters the wallet. Authorities can be rotated, programs can be upgraded, liquidity can deteriorate, ownership concentration can change, and programmable token configuration can become materially different from the state observed at acquisition.

Mesh is therefore intended to maintain monitoring after execution rather than treating a successful transaction as the end of the security process. The resulting position becomes part of the wallet's monitored state, allowing later observations to be compared with the evidence that existed previously.

This creates continuity across the product lifecycle. Mesh can analyze an interaction before execution, incorporate the resulting position into wallet exposure after execution, and then monitor the relevant security conditions for subsequent changes. That continuity is necessary for the system to function as a persistent sentinel rather than a collection of independent scans.

11.8 Security-State Change Detection

Continuous monitoring should emphasize meaningful changes rather than repeatedly presenting the user with the same static analysis. The value of an autonomous security layer comes partly from recognizing when previously understood conditions have changed enough to require attention.

Examples may include a new authority controller, a program upgrade, a substantial decline in modeled exit efficiency, a previously unavailable restriction becoming detectable, a change in Token-2022 configuration, or another material alteration in the evidence surrounding an asset already held by the wallet. The system should preserve enough prior state to determine whether a new observation represents a meaningful security transition rather than merely another reading of the same condition.

This approach also helps control alert fatigue. Users should not need constant confirmation that nothing has changed. Mesh should prioritize changes that materially affect the security interpretation or cross a user-defined protection threshold.

11.9 Alert Delivery

The initial monitoring experience should rely primarily on browser notifications because they can be integrated naturally with the web and browser-based product stages. This gives BAREMAX Labs a relatively direct way to test alert quality, severity classification, and user response before introducing a dedicated mobile notification system.

Once the mobile application is sufficiently mature, push notifications should provide a delivery channel while the user is away from the browser. A mobile dashboard and notification service are separate from mobile transaction-signing integration. Solana provides Expo-based development examples,[41] while the Mobile Wallet Adapter platform table checked for this draft supports Android and Chrome on Android, not iOS applications or browsers.[42] An iOS Mesh application is not excluded by that limitation, but its wallet-signing route would require an appropriate separately supported integration. Notification delivery remains subject to device permissions, connectivity, and service availability.

Longer term, alert delivery should become user-selectable. Browser notifications and mobile push can remain primary channels, while email, Telegram, SMS, or other supported services may be offered where appropriate. Notification urgency should also reflect severity so that material security events receive immediate attention while lower-priority observations remain available inside the application without generating unnecessary interruption.

11.10 Personalized Protection Policies

Wallet integration becomes substantially more useful when Mesh can respond according to security parameters selected by the wallet owner. The evidence itself should remain objective, but users should eventually be able to configure how the system reacts to particular conditions.

A conservative user might require additional confirmation whenever a material domain remains UNKNOWN, establish a maximum acceptable exit-impact threshold, or prevent interactions involving specific active authorities. A more aggressive user may choose higher thresholds or allow the same conditions after receiving a warning. In either case, Mesh should report the same underlying evidence; only the protective response changes.

These policies can eventually form the basis of Protection Mode. The system establishes the onchain facts, interprets their supported security significance, and compares the resulting state against rules the user has already chosen. This creates personalized protection without turning BAREMAX into a discretionary decision-maker over the user's assets.

11.11 Transaction Interruption and User Control

As integration becomes deeper and the transaction-analysis engine becomes sufficiently reliable, Protection Mode may eventually be able to interrupt transactions that violate the user's own security policy. This is a materially more consequential capability than displaying a warning and should therefore be introduced only after the supporting evidence and transaction interpretation have reached a high standard of reliability.

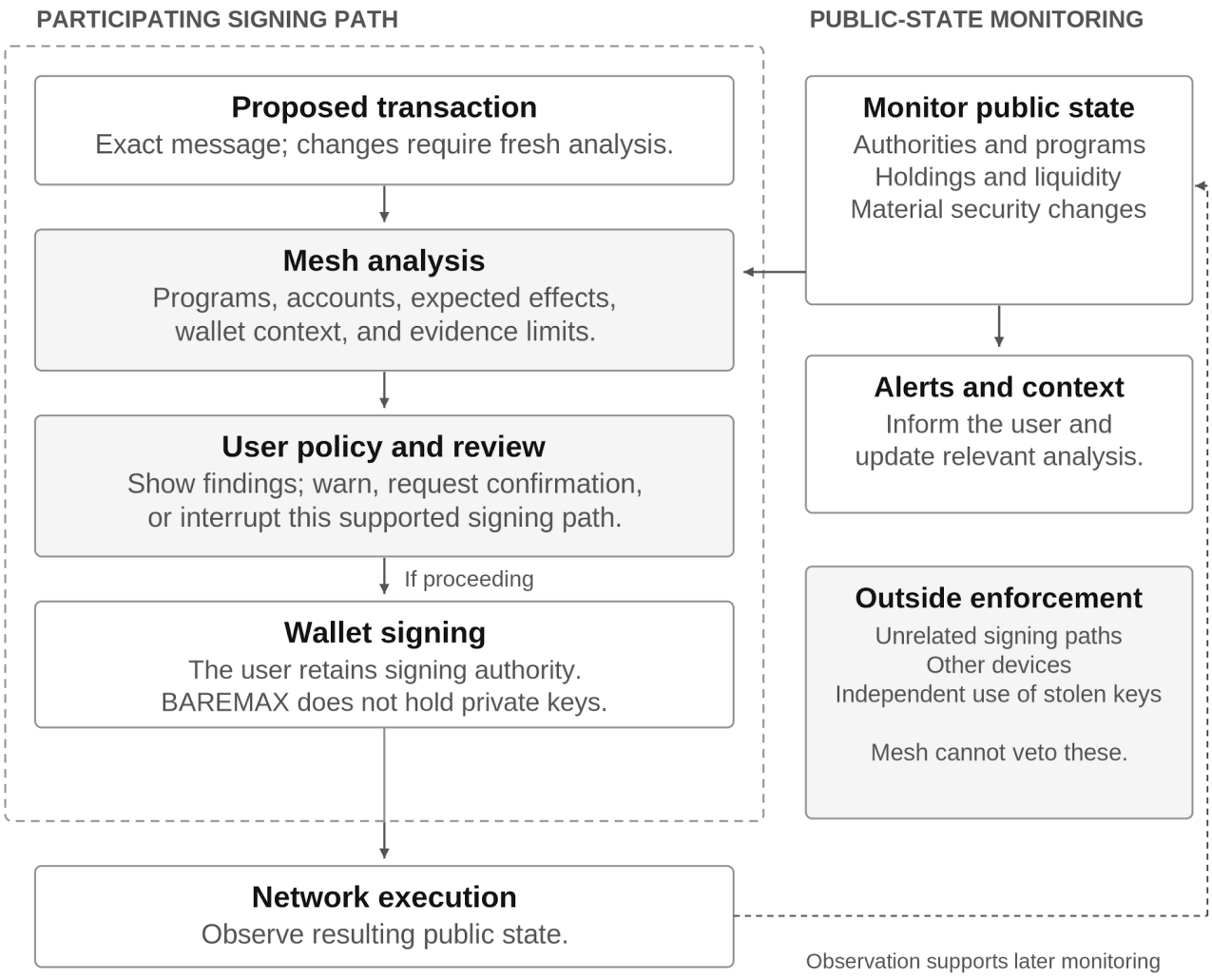
An interruption may be triggered because Mesh identifies a HIGH RISK interaction, because the transaction would create exit conditions beyond the user's configured threshold, because a material security domain remains unresolved, or because another explicit policy rule has been violated. Depending on the configuration, the user may then be able to review the evidence, provide additional confirmation, or abandon the interaction.

Protection Mode implements rules chosen by the wallet owner within supported participating signing paths. It cannot veto transactions signed through unrelated applications or devices, or transactions independently authorized by an attacker with stolen keys. Any supported override must be explicit and accompanied by the unresolved evidence or violated policy. Outages and stale analyses must not silently become approvals: the interface should identify unavailable protection and follow a documented, user-visible failure policy. BAREMAX does not acquire discretionary signing authority through these controls. The 2027 scope excludes automatic sales, transfers, and permission changes; enabling a protection policy is not authorization for those transactions. Figure 4 separates the participating signing path from public-state monitoring and transactions outside that enforcement boundary.

BAREMAX / TECHNICAL WHITEPAPER 04

Pre-sign protection and monitoring

Planned architecture; applies within supported participating integrations.



Simulation is evidence, not guaranteed settlement. Unavailable protection must not appear as approval.

Figure 4. Pre-sign protection and monitoring. Planned intervention operates only in supported participating signing paths; public-state monitoring is separate from signing authority. A changed transaction message requires fresh analysis. Source: Sections 11.2-11.11.
View full-size figure

11.12 Increasing Integration Over Time

Deeper integration should follow demonstrated reliability because an incorrect result has greater consequences near authorization. The stages and target dates are consolidated in Section 14; this section establishes the technical boundaries that each participating interface must preserve.

This sequence should not be interpreted as a requirement that every stage remain a permanent product surface. The browser extension, in particular, is a useful bridge rather than the intended long-term center of Mesh. As direct integrations become technically possible and the security engine becomes mature enough to support them, the product should move toward the lowest-friction interfaces available.

The degree of integration should increase only as the underlying system becomes capable of supporting it safely. A security layer operating directly in front of a transaction signature requires a higher reliability standard than an informational scanner because an incorrect conclusion can directly influence whether the user proceeds.

11.13 Intended Wallet Experience

At maturity, wallet integration should make security feel persistent rather than procedural. The user should be able to connect a wallet, establish protection preferences, and allow Mesh to maintain an evolving understanding of the assets, programs, authorities, liquidity conditions, and transactions relevant to that wallet without repeatedly reconstructing the same context manually.

Before execution, Mesh should evaluate the proposed interaction using that existing wallet context. After execution, the resulting exposure should enter continuous monitoring. When material state changes, the system should alert the user according to the selected notification and protection settings. The user's private keys remain under the user's control throughout that process.

The long-term objective is therefore not merely to place BAREMAX beside the wallet. It is to integrate Mesh deeply enough into the wallet experience that sophisticated onchain security analysis occurs naturally around the user's activity while remaining non-custodial, explainable, and subject to the wallet owner's authority.

12. Threat Model

The BAREMAX threat model defines the classes of onchain conditions Mesh is intended to detect, interpret, and eventually help users respond to. The scope is deliberately broader than malicious tokens or known scam signatures because wallet losses can result from interactions among programs, authorities, token mechanics, ownership structures, liquidity, transaction execution, and changing onchain state.

Mesh is not designed around a fixed list of attacks that can be considered permanently complete. Solana programs execute instructions against network accounts, transactions can compose multiple instructions atomically, Token-2022 expands the behavior available to token mints and accounts, and upgradeable programs can change after deployment while an upgrade authority remains active.[7][8][11] The threat surface therefore evolves as the network and the applications built on it evolve. BAREMAX Labs intends for the threat model to expand accordingly while preserving its primary focus on evidence that can be established from blockchain state.

12.1 Malicious Transactions and Wallet Drainers

One of the most important long-term threat classes for Mesh is the transaction that is technically valid but produces an outcome the user did not understand or intend. A user may knowingly authorize a signature request without understanding every program, account, instruction, permission, or asset movement embedded in the transaction. On Solana, a transaction contains the signatures and message required for execution, while individual instructions identify programs, accounts, and opaque instruction data.[8] The fact that a wallet is capable of signing the transaction does not establish that the user understands its consequences.

Future transaction-level Mesh analysis is intended to evaluate the interaction in the context of the wallet rather than rely solely on known malicious addresses or previously identified attack signatures. Relevant evidence may include programs invoked, writable and signer accounts, expected asset movement, token behavior, destination accounts, authority relationships, and expected changes to the wallet's state. Known malicious infrastructure may provide useful evidence where it can be established onchain, but the broader objective is behavioral: determine what the proposed interaction can do to the wallet.

This approach is important because new malicious programs or transaction patterns may not yet possess a known reputation. A mature sentinel should therefore attempt to identify suspicious or dangerous onchain effects even when the exact attacker or implementation has not previously been catalogued.

12.2 Privileged Authority Abuse

Administrative authority is not inherently malicious, but it creates a dependency on whoever controls the relevant capability. The base Solana token model can include mint and freeze authorities, while Token-2022 extends the set of possible privileged capabilities through features such as transfer-fee configuration, permanent delegates, pausability, metadata controls, and transfer-hook relationships.[7]

The threat model therefore includes both direct authority abuse and excessive concentration of privileged control. A controller capable of minting additional supply creates a different exposure from one that can freeze accounts, alter fee configuration, or influence programmable transfer behavior. When several capabilities are held by the same controller, the combined trust dependency may be more important than any individual authority considered independently.

Mesh is intended to identify these capabilities, resolve the controlling addresses where possible, detect privilege concentration, and preserve uncertainty when controller evidence cannot be established. The purpose is not to infer malicious intent from the existence of administrative control, but to make the power structure surrounding the asset visible before the user relies on it.

12.3 Liquidity and Exit Traps

An asset can retain a displayed market value while providing materially worse conditions when the user attempts to realize that value. Shallow reserves, concentrated liquidity, insufficient market depth, high execution impact, or a position that is large relative to available liquidity can create significant divergence between a portfolio valuation and the amount that can actually be recovered.

Automated market makers execute trades against available reserves, and execution quality changes as trade size becomes large relative to those reserves.[13] Mesh therefore treats poor exit conditions as a security-relevant threat rather than merely a trading inconvenience. The concern is particularly important when a user can enter a position more easily than they can later exit it under comparable conditions.

The threat model includes authenticated but severely illiquid markets, deteriorating liquidity after purchase, the disappearance of previously available supported liquidity, and situations in which Mesh cannot establish a trustworthy exit path. These states must remain distinct: confirmed poor liquidity is a positive risk finding, while an inability to identify supported liquidity represents a limitation in evidence rather than proof that no market exists.

12.4 Programmable Token Behavior

Token-2022 significantly expands the range of behavior that can exist within a Solana token. Extensions can introduce transfer fees, permanent delegates, transfer hooks, non-transferability, pausability, scaled UI amounts, and other specialized capabilities.[7] These features can support legitimate applications, but they can also alter assumptions a user might otherwise make about ownership, transferability, fees, or execution.

The threat model therefore includes both intentionally restrictive behavior and misunderstood programmable behavior. A token does not need to contain an exploit to create financial risk if the holder incorrectly assumes it can be transferred, sold, or controlled in the same manner as a conventional token.

Mesh is intended to determine which programmable capabilities exist, whether their underlying state is structurally valid, which authorities or programs control them, and whether those controls remain mutable. The security concern is not Token-2022 itself, but the gap between what the token can actually do and what the user believes it can do.

12.5 Mutable Program Logic

A program can behave acceptably at the moment of analysis while remaining technically capable of changing later. Solana programs deployed under the loader-v3 model can be upgraded when an upgrade authority remains set; removing that authority makes the program immutable under that deployment model.[11]

Mesh therefore treats program mutability as a trust condition. This is particularly important when a program participates directly in another risk domain. A transfer-hook program, for example, may currently implement behavior that appears acceptable while retaining an upgrade authority capable of changing that logic later.

The threat model includes malicious upgrades, compromised or misused upgrade authority, unexpected changes in executable behavior, and incorrect claims of immutability caused by incomplete program evidence. Mesh should distinguish proven mutability, proven immutability, and unresolved upgradeability rather than treating a missing authority lookup as proof that a program cannot change.

12.6 Creator and Deployment Behavior

Creator and deployment history can reveal context that is not visible from the current mint state alone. Prior launches, deployment relationships, recurring addresses, and historical transaction patterns may provide evidence relevant to the security interpretation of a newly encountered asset.

The threat model includes repeated deployment patterns that may be associated with abusive launches, rapid creation and abandonment of assets, or relationships between a current asset and historically concerning activity. At the same time, creator history should not become a simplistic reputation oracle. Legitimate developers may deploy many assets, and a newly observed address may have incomplete rather than genuinely clean history.

Mesh must therefore pair creator findings with explicit historical coverage. Observed behavior can contribute to risk, while missing or bounded history should reduce the certainty of the conclusion rather than fabricate an absence of prior activity.

12.7 Ownership and Control Concentration

Ownership concentration can create material economic risk even when the token and supporting programs behave exactly as designed. A small group of controllers holding a large fraction of the observable supply may possess disproportionate capacity to create sell pressure, influence an illiquid market, or materially alter the execution conditions available to other holders.

The threat model therefore includes severe observed concentration, controller aggregation across several token accounts, and concentration that becomes more consequential when combined with weak liquidity. Mesh should also separate recognized automated market maker or program-controlled holdings from ordinary wallet balances where the evidence permits, because raw account rankings can otherwise produce misleading interpretations.

The limitations established earlier remain important. Onchain addresses do not always reveal real-world beneficial ownership, and bounded holder samples do not establish complete ownership of the supply. Mesh should identify the concentration it can support from observable evidence without claiming identity relationships it cannot prove.

12.8 Evidence Manipulation and Analytical Failure

The security engine itself is part of the threat model. A dangerous interaction can be missed because the analyzer misinterprets evidence, or because the integrity of its own software and delivery path is undermined. Mesh's design must account for forged verdicts, compromised backend responses, unsafe update channels, and misleading alerts, with integrity controls and release validation appropriate to each interface. These are requirements for the protection service, not claims that it replaces general endpoint security.

Relevant failure modes include malformed account data, truncated Token-2022 extensions, impossible numeric values, conflicting duplicate evidence, malformed transaction structures, incorrect pool identification, incomplete historical acquisition, and provider responses that are technically successful but semantically insufficient. If these conditions are interpreted incorrectly, the analyzer itself can create false reassurance.

This is why Mesh treats structural validation and evidence quality as security controls rather than implementation details. A malformed authority must not become a revoked authority, an invalid balance must not silently become zero, and incomplete history must not become clean history. Established secure-development guidance similarly emphasizes reducing vulnerabilities, mitigating defects that remain undetected, and addressing their root causes so that they are less likely to recur.[6]

12.9 Changing Onchain State

Security analysis has a temporal dimension. A conclusion that was justified when the asset entered the wallet may no longer describe the same security environment several hours, days, or months later.

Relevant changes can include authority rotation, program upgrades, liquidity withdrawal, ownership concentration, altered token configuration, new creator activity, or other onchain state transitions. In each case, the risk is not necessarily that the original analysis was incorrect; the underlying evidence itself may have changed.

The threat model therefore extends beyond initial detection toward state-change monitoring. A mature Mesh should compare new observations against previously established security state and identify changes that materially affect the wallet's exposure. Continuous monitoring is necessary if the product is to remain useful after the initial transaction rather than functioning only as a point-in-time scanner.

12.10 Compound and Cross-Domain Threats

Some of the most consequential risks may emerge from combinations of conditions rather than from one individually extreme signal. Concentrated ownership becomes more significant when liquidity is shallow. A transfer hook becomes more consequential when its associated program is upgradeable. A privileged authority becomes more concerning when the same controller possesses several additional capabilities. A transaction involving a technically valid asset may become dangerous because of how that asset, a program, and the wallet's existing exposure interact.

Mesh is specifically designed to support this type of cross-domain interpretation. Individual risk domains retain their own evidence and conclusions, but the Mesh layer can identify relationships between them and determine when the combined condition warrants greater attention.

This prevents the threat model from becoming a checklist in which an interaction is considered acceptable merely because no single checkbox independently crosses a predefined threshold. Onchain security can be relational, and the architecture should be capable of representing that fact.

12.11 Scope and Expansion of the Threat Model

The current threat model should be understood as a foundation rather than a permanent limit on what BAREMAX can analyze. BAREMAX Labs intends to continue developing the security engine over months and years, adding support for additional Solana programs, liquidity venues, token functionality, transaction patterns, historical evidence, wallet behaviors, and newly observed attack classes as the system matures.

Expansion should remain disciplined. New threat coverage is useful only when Mesh can acquire and validate the evidence necessary to support it reliably. A broad list of nominally supported threats would be less valuable than narrower coverage in which the system can explain exactly what it observed and why the evidence justifies the conclusion.

The threat model should therefore evolve through concrete engineering support rather than marketing claims. Newly identified failure modes should become acquisition rules, validation requirements, analytical logic, and regression coverage before they are represented as conditions Mesh can reliably detect.

12.12 What Mesh Cannot Guarantee

Mesh is intended to reduce onchain risk, not eliminate every possible path to financial loss. No security layer can guarantee detection of every vulnerability, malicious program, previously unknown exploit, or future state change.

Device, wallet, and signing-key compromise can bypass Mesh's participating transaction paths. The exclusions in Sections 13.10-13.12 distinguish onchain analysis from protocol auditing, endpoint protection, and key management. Detecting resulting public-state changes would not mean that Mesh could prevent the original compromise.

Unknown vulnerabilities and future state changes also remain outside any universal assurance. Proposed SAFE conclusions are limited by the supported evidence, observation time, and eligibility rules specified in Sections 5.4 and 13.13.

12.13 Onchain-First Scope

The core scope of Mesh is onchain risk and blockchain-derived evidence, not a requirement that every part of the service execute onchain. Its analysis, monitoring orchestration, subscription management, policy configuration, and notification delivery may run outside the blockchain. Settled account and transaction state must remain distinguishable from a pending signing request, a modeled or simulated outcome, and user-provided intent. Simulations depend on observed state and do not themselves establish what a later transaction will execute.[38]

BAREMAX is not currently being designed as a generalized cybersecurity platform for malicious websites, operating-system malware, email phishing, social engineering, or endpoint compromise. Those risks are important, but attempting to absorb them into the core product would expand the threat surface far beyond the area in which Mesh is intended to develop deep technical specialization.

Any later network expansion must rebuild the relevant evidence and threat model for that network's execution, token, authority, and liquidity structures. Section 14.10 sets out the research direction; the onchain-first boundary is not permission to substitute a generic multichain score for network-specific validation.

Taken as a whole, the BAREMAX threat model is intentionally broad within its onchain boundary. Mesh is intended to continue learning new ways in which assets, programs, transactions, market structure, and changing state can place a wallet at risk while remaining disciplined about what the available evidence can and cannot prove.

13. Limitations

BAREMAX Labs is developing Mesh as a security layer intended to reduce onchain risk, improve the quality of information available to wallet users, and identify conditions that may otherwise be difficult to inspect manually. It is not designed to eliminate every possible source of financial loss or provide an absolute guarantee that an asset, program, transaction, or wallet interaction is safe.

These limitations are not separate from the BAREMAX security model. They define the boundary within which Mesh conclusions should be interpreted. A security system becomes less trustworthy when its stated capabilities exceed what its evidence, protocol coverage, and analytical methods can actually support. BAREMAX Labs therefore intends to make important limitations visible in both the product and its supporting documentation rather than allowing users to infer broader protection than the system can provide.

13.1 Supported Scope

Mesh can only analyze behavior that its current implementation is designed to acquire, validate, and interpret. During early versions of the product, that supported scope will necessarily be narrower than the full Solana ecosystem.

Limitations may exist across supported liquidity venues, Token-2022 extensions, program types, transaction patterns, historical acquisition methods, wallet integrations, and pre-sign analysis. Solana itself supports a broad execution environment in which programs can interact through cross-program invocations and operate on arbitrary account state, while Token-2022 continues to expose a substantial set of specialized token capabilities.[7][8] Complete semantic understanding of every possible interaction cannot reasonably exist in the first public version of Mesh.

Unsupported behavior should therefore remain explicit. A program, extension, market, or transaction pattern that Mesh does not yet understand should not be treated as safe merely because the system failed to identify a known problem within it. BAREMAX Labs intends to broaden support progressively, but additional coverage should be introduced only when the system can validate and interpret the relevant evidence with sufficient reliability.

13.2 Incomplete Onchain Evidence

Blockchain state is publicly observable, but public availability does not mean that every security-relevant fact can always be retrieved, reconstructed, or interpreted completely. Mesh may encounter incomplete transaction history, unavailable account data, unsupported account layouts, failed lookups, malformed responses, conflicting observations, or evidence that is valid but insufficient to establish the broader condition being analyzed.

The Evidence Model described in Section 5 is intended to prevent these limitations from silently becoming clean results. Partial evidence should remain partial, and an acquisition failure should remain distinguishable from valid evidence that a condition is absent.

This distinction is particularly important because Mesh combines several types of evidence that can have different acquisition characteristics. Current account state, historical activity, liquidity conditions, and controller relationships may each require different retrieval and validation paths. A complete response from one part of the system does not imply that every other part of the security state has been resolved.

13.3 Non-Atomic Observations

Many Mesh analyses require information from multiple accounts, programs, pools, transactions, and historical records. Those observations may not always originate from exactly the same slot or moment in network state.

Solana transactions execute atomically, but an external analytical system querying multiple pieces of blockchain state does not automatically receive a single atomic snapshot of every account and historical observation it requests.[8] In rapidly changing environments, one account may therefore change between the retrieval of two related pieces of evidence.

BAREMAX Labs should minimize this problem through acquisition design, slot awareness where appropriate, consistency checks, and bounded observation windows. It cannot guarantee, however, that every multi-source analysis represents one perfectly synchronized global state. Where timing differences could materially affect a conclusion, the system should preserve that limitation rather than imply precision that the observation process cannot support.

13.4 Provider and Infrastructure Dependence

Mesh depends on access to Solana infrastructure to acquire live evidence. RPC providers may differ in historical availability, latency, indexing behavior, reliability, throughput, and rate limits. Even Solana's public RPC endpoints are explicitly rate-limited and are not recommended as production infrastructure for high-traffic applications.[43]

BAREMAX Labs can reduce infrastructure risk through dedicated providers, redundancy, validation, fallback logic, bounded acquisition strategies, and careful handling of provider failures. It cannot guarantee that external infrastructure will always return complete, timely, or usable information.

A provider failure should therefore affect evidence availability rather than alter the meaning of the underlying blockchain state. If Mesh cannot retrieve evidence necessary to establish a condition, it should represent the resulting uncertainty rather than infer a favorable answer from the failed request.

13.5 Historical Coverage

Creator, deployment, wallet, and transaction-history analysis can require reconstruction of historical activity. Depending on infrastructure, account activity, and the acquisition strategy being used, Mesh may operate on bounded history rather than an exhaustive record of every transaction ever associated with an address.

This limitation affects the wording of historical conclusions. If Mesh observes one prior deployment within partial coverage, the defensible claim is that one deployment was observed. It is not that only one deployment has ever existed. Likewise, failure to recover earlier activity does not establish that earlier activity never occurred.

BAREMAX Labs intends to improve historical acquisition over time, but the underlying principle should remain stable: observed history and complete history are not interchangeable. The product should communicate the difference wherever that distinction could materially change the user's interpretation.

13.6 Ownership Analysis

Ownership analysis is limited by observable account relationships. An account's owner program is distinct from the token-account owner authority stored inside token-account data; neither field establishes the identity of a real-world beneficial owner.[10][44] Mesh should distinguish these technical roles before aggregating balances. The released analytical baseline uses a bounded ten-account sample and existing technical classifications, not exhaustive wallet clustering or complete beneficial-ownership attribution.[45]

A large balance may belong to an ordinary wallet, automated market maker, treasury, custodian, multisig, program-controlled account, or another identifiable structure. One real-world actor may also control several unrelated addresses without leaving sufficient evidence for Mesh to establish that relationship. Early ownership analysis may additionally focus on a bounded sample of the largest accounts rather than attempting exhaustive beneficial-ownership attribution across the entire supply.

Mesh should therefore describe ownership conclusions as observable onchain concentration and controller analysis. It should not imply knowledge of real-world beneficial ownership where the network evidence does not support that claim.

13.7 Liquidity Coverage

Mesh can model only liquidity structures it knows how to authenticate and quote reliably. An asset may trade through a venue, pool type, routing system, or market mechanism that is not yet supported by BAREMAX.

This creates an important distinction between no liquidity exists and no supported, authenticated, and quoteable liquidity was established by Mesh. The second statement describes the limits of the analytical system; the first is a considerably broader claim about the market itself.

BAREMAX Labs intends to expand venue support over time, but the objective should remain reliable execution modeling rather than nominal coverage of every available market. Unsupported liquidity may reduce the completeness of the result without justifying assumptions about the liquidity that exists outside Mesh's current analytical scope.

13.8 Execution Modeling

Exit-efficiency, price-impact, and transaction-output calculations are models of observed conditions rather than guaranteed execution outcomes. Actual transactions may differ because relevant state changes between analysis and execution.

Factors may include liquidity movement, competing transactions, transaction ordering, slippage parameters, priority fees, transfer fees, program behavior, newly updated state, or execution paths that differ from those available when Mesh produced its estimate. Solana transactions are executed against the network state available when they are processed, meaning a model generated before submission cannot guarantee that identical conditions will remain available when the transaction reaches execution.[8]

Mesh should therefore present modeled execution as an estimate tied to an observation point and supported market assumptions. Greater integration may reduce the time between analysis and signing, but it cannot convert an execution model into a guaranteed future settlement result.

13.9 Token-2022 Coverage

Token-2022 substantially expands the state and functionality that can exist within token mints and token accounts. Extensions are represented through additional state, and Solana documents both extension-specific behavior and compatibility constraints among certain extension types.[7]

Mesh may not immediately support every existing extension, every combination of extensions, or functionality introduced by future versions of the Token Extensions Program. An extension may therefore have bounded outer framing without its payload semantics being validated by the current BAREMAX implementation.

This limitation remains separate from malformed evidence. A bounded but semantically unsupported record is not proven valid, dangerous, or harmless. Where its behavior could materially affect the result, Mesh must disclose the gap and withhold an unjustified positive conclusion, as described in Section 6.13.

13.10 Program Security and Code-Level Vulnerabilities

Mesh can evaluate program-related evidence including ownership, upgradeability, authority relationships, transaction involvement, and other supported structural properties. It is not intended to replace a complete source-code audit, formal verification process, or specialized protocol-security review.

Solana programs contain executable sBPF code, while mutable program state exists in separate accounts supplied during execution. Programs deployed under loader-v3 may also remain upgradeable while an upgrade authority exists.[11] These properties allow Mesh to establish important trust and mutability conditions, but structural analysis alone cannot prove that all executable logic is free from vulnerabilities.

A program may satisfy every condition Mesh currently knows how to inspect while still containing an undiscovered logic error, economic exploit, unsafe assumption, or implementation vulnerability. Protocol audits and other specialized security practices therefore remain complementary to Mesh rather than being replaced by it.

13.11 Device, Wallet, and Key Compromise

The core Mesh threat model is onchain. It is not intended to function as antivirus software, endpoint protection, operating-system security, or a private-key recovery system.

If a seed phrase or private key is stolen, a device is compromised, a wallet application is malicious, or an attacker otherwise gains valid signing authority, the attacker may be capable of authorizing transactions regardless of the security state Mesh observes around an individual asset. A future sentinel may identify resulting transactions or suspicious changes under certain conditions, but this does not make BAREMAX a substitute for secure key storage, device security, or responsible wallet management.

This boundary is also one reason BAREMAX is intended to remain non-custodial. Mesh should improve the security environment around the wallet without requiring the user to transfer private-key custody to BAREMAX Labs.

13.12 Zero-Day and Previously Unknown Threats

No security system can guarantee detection of vulnerabilities or attack techniques that have not yet been identified, modeled, or understood. A previously unknown program vulnerability, novel transaction pattern, unexpected Token-2022 interaction, or new class of economic attack may initially fall outside the supported Mesh threat model.

The architecture is intended to accommodate continued expansion as those conditions become understood, but the possibility of unknown threats cannot be engineered away entirely. This is particularly important for a security product whose operating environment continues to evolve.

BAREMAX Labs should therefore treat the threat model as continuously developing. New failure cases should become validation rules, domain logic, tests, and monitoring capabilities where appropriate, but the existence of an extensive threat model should never be represented as proof that all future attack classes are already covered.

13.13 SAFE Is Scope-Bound

A SAFE result should always be interpreted within the scope and time of the analysis. It means that the material checks supported for the interaction were sufficiently resolved and that Mesh did not identify a significant supported risk condition from the validated evidence available at that time.

It does not mean that the asset cannot lose value, that an unknown vulnerability does not exist, that unsupported program behavior is harmless, that future state cannot change, or that the user's device and signing keys are uncompromised. Similarly, a SAFE result does not represent insurance against loss or a guarantee that a future transaction involving the same asset will produce the same conclusion.

This qualification does not make SAFE meaningless. It establishes the standard required for the label to remain technically defensible. Mesh should be able to provide meaningful positive security conclusions while remaining precise about what those conclusions do and do not establish.

13.14 False Positives and False Negatives

Even with conservative evidence handling, Mesh can produce incorrect conclusions. A false positive may classify an interaction as more concerning than subsequent evidence justifies, while a false negative may fail to identify a material threat.

BAREMAX Labs considers false reassurance particularly serious because users may rely on the system when deciding whether to proceed. At the same time, excessive false positives can degrade the product by creating alert fatigue and reducing user trust. The engineering objective is therefore to reduce both classes of error rather than simply maximize caution.

Improvements should come through stronger evidence acquisition, structural validation, protocol-specific interpretation, regression testing, production feedback, and analysis of discovered failures. NIST's Secure Software Development Framework similarly emphasizes reducing vulnerabilities, mitigating undetected defects, and addressing root causes so that weaknesses are less likely to recur.[6]

13.15 Financial and Market Risk

Mesh is a security product, not an investment-prediction system. A token can satisfy the security checks supported by BAREMAX and still decline substantially in market value because of speculation, broader market conditions, changing demand, economic design, project failure, or other ordinary financial risks.

Likewise, strong liquidity at the time of analysis does not guarantee future liquidity, and a technically decentralized authority structure does not imply that an asset has economic value. Security analysis and investment analysis overlap in limited areas, but they are not interchangeable.

BAREMAX should therefore describe structural, execution, and onchain security conditions without representing those findings as predictions of profitability, expected return, or future price performance.

13.16 User Responsibility and Self-Custody

Mesh is intended to improve the information and protection available to the wallet owner, not replace the wallet owner's authority. Users remain responsible for protecting signing credentials, reviewing material warnings, determining whether to proceed with transactions, and selecting security policies appropriate to their own use.

Future Protection Mode functionality may introduce stronger intervention, but those controls are intended to be configured by the user rather than imposed through discretionary control by BAREMAX Labs. Even where a user chooses strict protection parameters, the underlying wallet should remain non-custodial and under the owner's authority.

This distinction should remain clear in both product design and documentation. BAREMAX can help users enforce rules they choose; it should not present itself as assuming responsibility for every financial decision made through the wallet.

13.17 Beta Software and Product Evolution

The first public versions of Mesh will be beta software. Protocol support, evidence acquisition, terminology, thresholds, user-interface behavior, and individual analytical methods may change materially as the system encounters broader real-world usage.

This is particularly relevant to a product whose security engine is expected to develop continuously. A methodology that is appropriate for the initial public release may later be replaced by a more precise model as BAREMAX Labs gains better infrastructure, broader protocol support, stronger historical acquisition, additional production data, or more reliable transaction interpretation.

The whitepaper should therefore define the architecture, principles, and intended direction of Mesh without implying that every implementation detail described at the time of publication is permanent. Material methodological changes should be documented where they affect how users should interpret security conclusions.

13.18 Legal and Product Disclosures

This whitepaper describes the intended architecture, methodology, current development direction, and limitations of BAREMAX Mesh. It should not be treated as a guarantee of protection, insurance against financial loss, or a substitute for the contractual and consumer disclosures required for a public product.

Before broad release, BAREMAX Labs should maintain separate Terms of Service, risk disclosures, privacy documentation, security policies, and other materials appropriate to the jurisdictions and product functions in which it operates. Those documents should accurately describe the capabilities and limitations of the product as it exists at launch rather than relying on the whitepaper alone. Privacy design should also distinguish public blockchain data from the new associations created when wallet addresses are linked to subscribers, notification destinations, preferences, or pending signing requests. Data

minimization, access controls, retention, and consent choices require explicit product decisions; public address visibility alone is not a reason to treat those associations as non-sensitive.

Qualified legal counsel should review those materials before public deployment, particularly as Mesh develops wallet monitoring, paid subscriptions, Protection Mode, transaction intervention, institutional integrations, or other functionality that may introduce additional contractual, regulatory, privacy, or liability considerations. The whitepaper can establish technical expectations, but it is not the project's sole legal risk-management mechanism. BAREMAX Labs is currently a working development name, not a registered entity; entity formation, operating arrangements, and launch obligations require separate review before the relevant activity begins. This document does not determine that incorporation or other obligations can safely be deferred until a finalized release.

Taken together, these limitations establish the boundary within which Mesh should be evaluated. The purpose of the system is to improve the user's security position by making onchain evidence more understandable, persistent, and actionable while remaining explicit about conditions it cannot observe, behavior it does not yet support, and risks no onchain analytical system can eliminate.

14. Roadmap

The BAREMAX roadmap is designed around progressive increases in both capability and responsibility. Each stage moves Mesh closer to the wallet and closer to the point of transaction execution, but each increase in integration also raises the reliability standard required of the underlying security engine. A system that analyzes an asset after a user requests a scan can tolerate limitations that would be less acceptable in a system capable of interrupting a transaction immediately before signature.

For that reason, the roadmap prioritizes reliability over feature count. BAREMAX Labs intends to expand Mesh only when the evidence acquisition, structural validation, domain analysis, and user-facing interpretation required by the next capability are sufficiently mature. The dates below represent current development targets rather than immutable commitments, and later milestones may be adjusted if additional validation is required. Figure 5 summarizes these staged targets and the conditions governing progression.

BAREMAX / TECHNICAL WHITEPAPER 05

Mesh development roadmap

Planning targets from Section 14; release readiness determines the pace.

Q4 2026 — **Public v0 beta**
December target; web analysis and validated scope.

Q1 2027 — **Wallet-aware security**
Holdings, position context, and beta hardening.

MID-2027 — **Pre-sign development**
June–July target; browser layer as an intermediate step.

Q3 2027 — **Continuous monitoring**
Material security-state changes and useful alerts.

Q3–Q4 2027 — **Protection Mode + integrations**
User-defined policies in supported signing paths.

Q4 2027 — **Mobile Mesh**
Monitoring and push; signing support is platform-specific.

2028+ — **Explore institutional/API expansion**
Potential multichain research; Solana remains the priority.

Targets are not release commitments.
Evidence integrity, validation, and integration readiness govern progression.

Figure 5. Mesh development roadmap. Dates are planning targets, not release commitments. Mobile signing and deeper wallet integration require separately supported implementations. Section 14.7 describes the conditional late-2027 Max expansion; institutional/API expansion and multichain research remain later possibilities. Source: Sections 14.1-14.11.
View full-size figure

14.1 Q4 2026 - Mesh v0 Public Beta

October update. A protected, read-only beta is now released. Broader public access remains a separate readiness decision. The calendar below retains the founder's planning targets; this revision does not convert them into delivery commitments.

The first public version of Mesh is targeted for Q4 2026, with December 2026 as the working release target, subject to the readiness requirements below. Before public access, BAREMAX intends to run an invited testing stage with a small group of prospective users and selected technical reviewers. This stage should test the read-only analysis workflow, clarity of warnings, handling of unsupported evidence, and consistency of results without requiring participants to place trades or put funds at risk.

Invited testing is preparation for the public beta, not a substitute for it or evidence of an independent security audit. Participation and feedback should not be described as endorsements, partnerships, or formal review unless those roles have been explicitly agreed. The v0 objective remains a public web-based security-analysis experience grounded in a documented, validated subset of the Mesh architecture.

The expected v0 scope includes authority-risk analysis, creator and deployment history, observed ownership concentration, liquidity and exit modeling, transfer restrictions, transfer fees, Token-2022 analysis, evidence-quality states, and explainable domain-level security conclusions. Wallet connection is also a current v0 objective, provided it satisfies the same stability and security requirements as the rest of the release. Solana's current wallet tooling supports application connections through Wallet Standard discovery and compatible frontend libraries, providing an established technical foundation for non-custodial wallet connection.[35]

Development progress should be reassessed in approximately mid-November 2026 against the integrated build, known material defects, user-testing findings, and operational readiness. Additional functionality may be included only after the core release requirements are satisfied. If those requirements are not met, the public release should be narrowed or delayed with the change explained openly; invited access must not be relabeled as completion of the public milestone. A smaller v0 with reliable evidence semantics is preferable to a broader release whose conclusions BAREMAX cannot defend.

14.2 Q1 2027 - Wallet-Aware Security

Following stabilization of the public beta, development should increasingly shift from isolated asset analysis toward persistent wallet-aware security. A connected wallet allows Mesh to determine which assets are actually relevant to the user, the position sizes associated with those assets, and the security conditions surrounding the wallet's existing exposure.

This phase is expected to deepen automatic holdings analysis, position-specific liquidity modeling, wallet-level security summaries, persistent security state, and the infrastructure necessary for later monitoring. The principal change is conceptual as much as technical: Mesh begins moving from a system

that analyzes objects selected by the user toward one that maintains context about the wallet being protected.

The Q1 2027 period should also be used to evaluate public-beta failure cases and harden the analytical engine before transaction-level protection becomes a primary development focus. False positives, unresolved protocol behavior, unsupported markets, and evidence failures discovered in production should become regression cases and improvements to the underlying evidence contracts.

14.3 Mid-2027 - Pre-Sign Security Development

The next major milestone is a usable, limited pre-sign beta, targeted for June to July 2027. Completion means that users can inspect proposed transactions before signing within a clearly documented set of participating integrations and supported transaction flows. It does not mean universal wallet interception, comprehensive understanding of every program, or availability of every planned Protection Mode control. The target remains conditional on the preceding analytical foundation, integration access, and required validation.

A browser extension is the most logical initial environment for developing this capability because it can place Mesh closer to decentralized applications and wallet interactions while allowing BAREMAX Labs to control the security experience directly. The extension should be treated as an intermediate product layer rather than the intended permanent center of BAREMAX.

The first supported flows should expose the programs and accounts involved, interpretable asset movements, material authority or token-behavior findings, and unresolved conditions relevant to the proposed transaction. Expected wallet-state and post-purchase exit analysis may be added where their specific models are validated; the broader list of possible checks is not a promise that all will ship together. Release evidence should demonstrate that the assessment applies to the transaction actually presented for signature, that changed messages receive renewed analysis, and that unsupported behavior or service failure cannot silently become approval. Broader coverage and stronger policy intervention should follow this limited beta rather than be assumed from it.

14.4 Q3 2027 - Continuous Monitoring

By mid-to-late 2027, Mesh is intended to begin transitioning from primarily request-driven analysis toward continuous monitoring of connected wallet exposure. This represents one of the most important stages in the evolution from scanner to sentinel.

Monitoring may include authority rotation, program upgrades, liquidity deterioration, worsening exit efficiency, changes in ownership concentration, creator or deployer activity, and changes in programmable token state. Rather than repeatedly rerunning complete analyses without reason, Mesh should increasingly detect differences between previously established and newly observed security states and determine whether those differences are material.

This stage will also require greater attention to notification quality. The usefulness of continuous monitoring depends not only on detecting changes but on distinguishing security-relevant changes from ordinary network activity. Alert frequency, severity, evidence quality, and user response should therefore become part of the product's security design rather than being treated only as interface decisions.

14.5 Q3-Q4 2027 - Deeper Wallet Integration and Protection Mode

Once pre-sign analysis and continuous monitoring are sufficiently reliable, BAREMAX Labs intends to pursue deeper integration with wallet infrastructure where technically and commercially possible. The browser extension provides an effective proving environment, but the long-term objective is to reduce the number of separate interfaces a user must navigate in order to benefit from Mesh.

This phase also creates the appropriate foundation for the first meaningful version of Protection Mode. Users may eventually define rules governing conditions such as maximum modeled exit impact, acceptable evidence quality, active privileged authorities, unsupported programs, or HIGH RISK transaction classifications. The security facts themselves remain objective, while the wallet owner's selected policy determines whether Mesh displays a warning, requires additional confirmation, or interrupts the interaction.

Protection Mode should not be rushed into an early release. A system that can influence whether a user signs a transaction requires substantially stronger reliability than a system that only presents analysis. The depth of intervention should therefore increase only as BAREMAX can justify increasing confidence in transaction interpretation.

14.6 Q4 2027 - Mobile Mesh

A dedicated mobile application is targeted for the later part of 2027, but only after the web, wallet-aware, monitoring, and transaction-analysis layers are sufficiently stable. The mobile application should not be developed simply to replicate the web interface on another screen. Its primary purpose is to make Mesh persistent when the user is away from the browser.

Solana's Expo-based examples provide a starting point for mobile application development,[41] but wallet-signing support must be assessed by platform. At the review date, Mobile Wallet Adapter supports Android and Chrome on Android rather than iOS.[42] Dashboard and push-notification functionality can therefore be planned separately from each platform's signing integration. The mobile target does not imply that one adapter already provides identical protection across all mobile environments.

A mature mobile Mesh should prioritize push notifications, continuous connected-wallet monitoring, security-state change alerts, position-specific liquidity warnings, Protection Mode controls, and accessible portfolio-level security information. Mobile development should therefore follow the security engine rather than lead it.

14.7 Consumer Product Development

The commercial sequence begins with Mesh Free, followed by Pro when a substantially broader, validated feature set provides a recurring paid benefit. Pro is intended to combine persistent monitoring, deeper investigation, retained security context, position-analysis tools, and configurable protection rather than simply increase Free's scan limits. Sections 8.8-8.10 define the feature boundaries; the roadmap does not require all tiers to be sold on the first day of beta.

Max may be introduced in late 2027 if multi-wallet infrastructure is operationally viable and its additional autonomous detection capabilities have earned release. The target is a coordinated security environment with shared-dependency analysis, cross-wallet risk context, and automatic reassessment of affected positions, not an early promise to protect an unlimited number of wallets. This remains conditional expansion after Pro; its scope and timing may move without weakening the core single-wallet product.

Commercial release requires published feature, coverage, and usage specifications and sustainable service economics. Pricing should leave a healthy operating margin after the costs identified in Section 8.9, rather than rely on unlimited-use assumptions. The shared rules on evidence transparency and already-detected warnings apply to every tier. No price, retention outcome, or profitability result is established by this whitepaper.

14.8 End-of-2027 Objective

The principal objective for the end of 2027 is for Mesh to have evolved from a web-based security analyzer into a persistent wallet-protection system.

By that point, the target state is for Mesh to maintain context about a user-selected Solana wallet, understand supported holdings and positions, monitor material security-state changes, and analyze supported transactions before signing through participating integrations. It should evaluate wallet-specific exit conditions, deliver relevant alerts, and apply the owner's selected protection settings without taking custody or automatically moving assets. Each capability remains subject to the coverage, reliability, and integration boundaries described in this paper.

Product success should also be demonstrated through users choosing to renew paid subscriptions and credible, specific feedback from the people and communities using Mesh. Pro renewal should be assessed among subscribers who have actually reached a renewal decision; Max should be evaluated on the same basis only after that tier has launched and had time to establish a renewal history. Small samples, billing periods, and cancellation reasons should remain visible rather than being hidden behind an aggregate retention claim.

Feedback is most useful when it identifies a warning that was understood, an exposure the user had overlooked, an explanation that changed a decision, or a failure that requires correction. Community discussion and individual testimony can establish perceived usefulness, but social reach and positive comments are not substitutes for technical validation or independent security review. BAREMAX does

not set an unsupported user-count target or infer verified losses prevented from subscription renewals or testimonials alone.

14.9 Early 2028 - Institutional and API Expansion

BAREMAX Labs intends to remain consumer-focused through the first major development cycle. Institutional integrations should follow demonstrated consumer utility rather than precede it.

If Mesh reaches sufficient reliability and proves useful under real wallet activity, early 2028 may become an appropriate period to begin exploring selected institutional applications. Potential directions include security APIs, SDKs, wallet-provider integrations, fund tooling, trading-platform integrations, fintech infrastructure, and direct partnerships with Solana programs or other organizations that would benefit from Mesh analysis.

The objective would not be to convert BAREMAX immediately into a generic API provider. Institutional access should expose proven parts of the same security architecture that have already demonstrated value in the consumer product, while allowing BAREMAX Labs to preserve the quality and evidence standards established within its own interfaces.

14.10 Potential Multichain Expansion

Solana remains the development priority throughout the roadmap described in this whitepaper. BAREMAX Labs does not intend to dilute early development by pursuing broad multichain coverage before the Solana implementation has reached substantial maturity.

If Mesh succeeds on Solana, research into additional blockchain environments may begin in 2028 or later. Any expansion should remain onchain-first and network-specific. A security model built around Solana's account architecture, token programs, authority structures, program deployment model, liquidity environment, and transaction semantics cannot simply be transferred unchanged to another blockchain.

Multichain expansion should therefore reproduce the philosophy of the original project rather than merely increase the number of supported networks: understand one execution environment deeply, establish what evidence can be trusted within it, and build security conclusions around that evidence before exposing the system to users.

14.11 Roadmap Standard

The roadmap represents a deliberate progression from public analysis to wallet awareness, pre-sign security, continuous monitoring, personalized protection, and deeper integration. Each stage increases the degree to which users can depend on Mesh, which means each stage must also increase the standard applied to validation, testing, evidence integrity, and failure handling.

Development is planned to remain primarily founder-led through the initial launch, with selective collaboration or hiring considered as revenue and product needs justify it. The roadmap does not assume

an already-funded team or secured wallet partnerships. Specialist review can be obtained through a defined engagement without creating a permanent team; hiring plans and the need for independent validation are separate decisions. Before consequential pre-sign or policy-intervention capabilities are released, the relevant security work must receive appropriate independent review. If the necessary expertise, integration access, or operating capacity cannot be secured, the affected capability should be narrowed or deferred rather than released without those safeguards.

Release records should identify the integrated commit, supported scope, regression and integration evidence, unresolved material defects, and operational failure behavior. Independent review and resolution of material security findings remain requirements for consequential signing controls, not claims about the current build. Readiness must govern release, and material changes to roadmap targets should be communicated openly.

15. Conclusion

BAREMAX Labs is developing Mesh so that security analysis can become a persistent part of the wallet experience rather than a separate task users must repeatedly reconstruct. The starting point is practical: programmable assets, privileged authorities, mutable programs, and position-dependent execution conditions can expose retail users to risks that are difficult to assess consistently through disconnected tools.

Mesh brings those conditions into one evidence-based architecture while preserving the meaning of each analytical domain. A conclusion should be no stronger than the evidence supporting it. Validated material warnings must remain visible, and missing or malformed evidence must not become reassurance. Positive consumer conclusions require the evidence eligibility described in Section 5, rather than a change of label applied to the existing engine's LOWER classification. Users should be able to understand the principal finding and inspect its supporting record at the level of detail they need.

The initial public product is a web-based analytical foundation, not the completed sentinel described by the long-term architecture. Wallet-aware analysis, a limited pre-sign beta, continuous monitoring, and configurable protection are successive development objectives. Each depends on validated coverage and reliable failure handling. Pre-sign intervention applies only within participating integrations, and the 2027 scope excludes automatic sales, transfers, and permission changes. The wallet owner retains signing authority throughout.

The consumer model follows the same progression. Free should provide useful core analysis and reveal material findings already detected for its users. Pro should offer a substantially broader single-wallet service through deeper investigation, retained security context, position-analysis tools, persistent monitoring, and configurable protection as those capabilities are validated. A later Max tier can extend that foundation across multiple wallets and add coordinated detection and reassessment. Its possible late-2027 introduction depends on operational viability and demonstrated usefulness, not the calendar alone.

BAREMAX is beginning with active retail Solana memecoin traders, including less-experienced users, and prioritizing depth within Solana over superficial multichain breadth. Continued development should be informed by concrete failures, regression evidence, and credible feedback from people using the product. Paid renewals can demonstrate recurring perceived value, but neither testimonials nor commercial success replace technical validation and appropriate independent review.

Mesh does not promise to eliminate financial loss, identify every vulnerability, or protect signing credentials outside its supported boundary. The standard is disciplined evidence handling, transparent limitations, explainable conclusions, and corrective action when the system fails. If that standard is maintained as coverage expands, Mesh can reduce the technical work required from the user while providing a more continuous view of the wallet's exposure. That is the transition BAREMAX is working toward: from isolated asset analysis to persistent, user-controlled security around onchain activity.

Appendices

The appendices provide additional technical detail for readers who want to understand how Mesh represents evidence, evaluates specific security domains, and preserves analytical boundaries without expanding the main body of the whitepaper into an implementation specification.

These appendices describe the current design direction of BAREMAX Mesh. Certain thresholds, internal representations, supported protocols, and implementation details may evolve as the system is tested against broader production data. Where that occurs, BAREMAX Labs should preserve the underlying principles established throughout this whitepaper: validated evidence before interpretation, explicit uncertainty, provenance, and clear separation between what the system observed and what it is justified in concluding.

Appendix A - Evidence-State Semantics

The BAREMAX Evidence Model separates the security condition being evaluated from the quality of the evidence available to evaluate it. This distinction allows Mesh to preserve validated risk while remaining explicit about unresolved information.

The conceptual vocabulary used throughout this whitepaper is RESOLVED, PARTIAL, UNKNOWN, FAILED, and NOT APPLICABLE. It describes evidentiary meaning rather than risk severity and is not a claim that every inspected domain already implements one five-state enum. The baseline's resultVersion and nested summary.schemaVersion are both 1; the actual interfaces preserve separate domain structures, global coverage, and unknown-signal fields.[9][17]

Evidence State	Meaning
RESOLVED	The evidence required for the supported analysis was acquired, structurally validated, and interpreted successfully.
PARTIAL	Valid evidence exists, but coverage is incomplete in a way that limits the scope or certainty of the conclusion.
UNKNOWN	Mesh cannot currently establish the relevant condition with sufficient confidence.
FAILED	An expected acquisition or validation path failed and cannot be interpreted as valid negative evidence.
NOT APPLICABLE	The check does not apply to the asset, program, transaction, or configuration being evaluated.

These states should not be collapsed into one another. A failed lookup is not equivalent to valid evidence that a capability is absent. Partial historical coverage is not equivalent to complete historical coverage. An unsupported extension is not equivalent to a malformed extension.

The distinction becomes particularly important for positive security conclusions. A dangerous condition may be established under otherwise partial evidence if the relevant finding itself is independently validated. SAFE requires a broader evidentiary burden because Mesh is asserting that the material supported checks were sufficiently resolved and did not reveal a significant supported risk.

A.1 Evidence Monotonicity

Mesh is intended to preserve a monotonic relationship between information quality and positive security conclusions. Removing validated evidence should not make the system more confident solely because less information remains.

For a fixed underlying state and methodology, loss of material ownership evidence should move the conclusion toward uncertainty or preserve an independently supported finding, not produce reassurance. A verified change in holdings or authority state can legitimately lower risk. Evidence retained from an earlier observation must remain timestamped and subject to freshness limits, rather than being represented indefinitely as current proof.

The same requirement applies to authority state, creator history, liquidity, Token-2022 configuration, and program upgradeability. This behavior is consistent with the secure-design principle that failures in security controls should not create more permissive outcomes.[18]

A.2 Conflicts and Malformed Evidence

Conflicting and malformed evidence are separate from ordinary incompleteness.

Conflicting evidence exists when two or more otherwise meaningful observations disagree about the same security condition. Mesh should attempt to reconcile the disagreement through provenance, ordering, structural validation, or other domain-specific rules. If the conflict cannot be resolved, the affected evidence should not be represented as fully resolved.

Malformed evidence fails a structural requirement necessary for safe interpretation. Examples include truncated Token-2022 extension payloads, invalid field encodings, impossible numeric values, malformed transaction records, or data structures whose declared boundaries do not match their contents.

Malformed values should not be coerced into legitimate security states. An invalid authority should not become NONE, an invalid balance should not become zero, and an invalid extension record should not be interpreted as though the extension were absent.

Appendix B - Authority and Controller Analysis

Authority analysis distinguishes between the existence of a privileged capability and the controller responsible for that capability.

Under the SPL Token model, mint and freeze authorities can retain meaningful control over supply and token-account behavior. Token-2022 introduces additional capabilities through extensions including transfer-fee authorities, permanent delegates, pause authorities, metadata controls, and transfer-hook relationships.[7][12]

Mesh therefore evaluates authority risk across two related layers:

- Capability identification - what privileged action is technically possible.
- Controller analysis - which address, program, or other observable entity controls that capability.

A capability can remain security-relevant even when controller classification is unresolved. Mesh should preserve the decoded authority address and the capability it controls rather than discarding the finding simply because the controller cannot yet be categorized.

B.1 Privilege Concentration

Privilege concentration exists when one controller holds several security-sensitive capabilities.

For example, the same address may control the mint authority, freeze authority, transfer-fee configuration, or another privileged mechanism. Each capability can be displayed individually, but Mesh should also identify the aggregate relationship because the combined control may represent a materially different security condition.

The concentration analysis should rely on established address or controller relationships rather than assumptions about real-world identity. If Mesh cannot prove that two addresses belong to the same real-world entity, it should not merge them solely because their behavior appears related.

B.2 Program-Controlled Authorities

An authority controlled by a program requires additional interpretation because the controlling program may itself be mutable. In those situations, authority analysis and program-upgradeability analysis become connected domains.

A program-controlled authority should not automatically be treated as decentralized or immutable. Account owner-program identification, token-account authority, program-derived-address attribution, and beneficial ownership are distinct claims.[10][44] A technical classification or an address shared across capabilities does not establish real-world identity. Mesh should validate the relevant program relationship and applicable upgrade mechanism before using it to reduce perceived control risk.

Appendix C - Liquidity and Exit Modeling

Mesh treats liquidity as a relationship between a specific position and the supported markets available to execute that position. The objective is to estimate practical exit conditions rather than rely on a static pool-liquidity figure.

AMM execution depends on reserve conditions and trade size, while concentrated-liquidity execution additionally depends on active price ranges.[26][28] The inspected BAREMAX baseline models selected PumpSwap and Raydium CPMM token-to-SOL paths, not CLMM, universal routing, or every liquidity venue.[27][32]

C.1 Reference Value

A modeled position begins with a defined reference value. Depending on the analysis, this may be derived from the position's current supported execution reference, a proposed purchase amount, or another explicitly identified pricing basis.

The reference should remain consistent throughout the comparison. If several pools are evaluated as potential exits, each should receive the same raw token quantity rather than independently redefining the size of the position according to its own price.

This prevents the system from comparing apparently superior outputs that actually represent different quantities of the underlying asset.

C.2 Exit Efficiency

Exit efficiency is intended to express the relationship between the position's modeled reference value and the net value Mesh estimates could be recovered through the selected supported exit path.

For a valid positive reference denominator, the percentage is defined as:

Exit efficiency (%) = 100 × modeled net exit value ÷ stated reference value

A position modeled at 10 SOL that returns 9.45 SOL through the supported exit path would therefore have approximately 94.5% exit efficiency under those observed conditions.

The reference must identify its source, quantity, units, observation context, and whether it is a spot valuation or a stated cost basis. Zero, missing, or invalid denominators make the percentage unavailable rather than zero. An output above a valid alternative reference should be reported as such, not silently clamped or described as a guaranteed gain. In the released analytical baseline, exit impact is the proportional shortfall of modeled net proceeds against the selected venue's own spot-value reference, including modeled fees.[27] It is not necessarily the pool's before/after marginal-price movement. Slippage between quote and execution is another distinct quantity; no modeled ratio guarantees a later fill.

C.3 Pool Authentication

A liquidity candidate should not contribute to execution analysis until Mesh has established that the pool structure is consistent with the supported venue.

Venue-specific validation may include program ownership, token mints, vault accounts, reserve relationships, authorities, token-program compatibility, or other structural invariants. The exact requirements differ across liquidity protocols and pool types.

This creates three distinct states:

- an authenticated and quoteable pool can contribute directly to execution modeling;
- an authenticated but unquoteable pool is recognized but cannot safely produce the requested execution estimate;

- an unauthenticated candidate should not contribute reserves or quotes to the security result.

C.4 Per-Size Pool Selection

Mesh should not assume that the same pool provides the strongest execution for every trade size.

Equivalent raw inputs can be compared independently by size among eligible modeled candidates. The released analytical baseline compares one selected candidate per supported venue, prioritizing exact net SOL proceeds, then exact impact, then fixed venue precedence.[27] Broader within-venue candidate evaluation is an expansion of that method, not a claim of exhaustive pool search or optimal split routing. A small and a large position may select different winning venues without changing the input quantity being compared.

The result must preserve provenance. The selected venue, pool address, relevant reserve state, raw input, modeled output, and supported fees should remain attributable to the same execution path.

C.5 Transfer-Level Effects

Execution modeling may also need to incorporate supported token-level behavior. Token-2022 transfer fees, for example, can reduce the amount transferred according to the mint's current configuration.[15]

As Mesh expands, the execution model should incorporate only those effects it can validate reliably. Unsupported mechanics should reduce the completeness of the result rather than being silently ignored while the output is presented as comprehensive.

Appendix D - Token-2022 Validation Model

Token-2022 uses extensions to add functionality and state beyond the base token layout. The extension system uses type-length-value structures that allow token mints and accounts to include additional extension-specific data.[7]

For Mesh, decoding is part of the security boundary. A parser returning a value is not sufficient evidence that the value should be trusted.

D.1 Whole-Set Validation

Mesh is intended to validate the relevant extension set before extension-specific values are permitted to influence security conclusions.

Required checks for supported layouts include:

- TLV framing;
- extension type identifiers;

- declared lengths;
- expected payload sizes;
- authority encodings;
- numeric-field validity;
- duplicate extension handling;
- compatibility between extension types;
- boundaries between adjacent extension records.

The purpose of validating the entire extension region is to prevent one malformed record from affecting the interpretation of another.

If an invalid length causes the decoder to consume bytes belonging to a neighboring extension, validating only the extension currently being queried would be insufficient. The structural integrity of the surrounding set must also be established.

D.2 Duplicate Extensions

Duplicate extension records deserve explicit handling.

The approved whole-set validation contract rejects every duplicate extension type, including duplicates whose payloads are identical. The decoder must not accept the first or last record merely because it can be read. Duplicate type detection is a failure of the inspection's structural admission contract, not another risk score.

Invalid whole-TLV evidence must stop a successful InspectionResult from being issued. Diagnostic observations can be retained separately, but not published as a partial successful score. This approved requirement is distinct from controller-local UNKNOWN handling and must be demonstrated by the integrated release tests; the inspected main-branch baseline is not represented as already satisfying every part of this contract.

D.3 Unsupported Extensions

An extension may pass bounded outer-framing checks while its payload semantics remain unsupported by the current version of Mesh.

Unsupported evidence should remain distinct from malformed evidence. Mesh may be able to verify that a TLV record is correctly framed while lacking the semantic logic necessary to determine the security significance of that extension.

In those situations, only the bounded outer framing has been established. The payload remains semantically opaque, not proven valid or harmless. Material unsupported behavior must remain an explicit limitation and cannot be used to justify SAFE, while structural malformation remains a separate inspection-failure condition.

D.4 Authority-Bearing Extensions

Several Token-2022 extensions contain or depend on privileged authorities.

Examples include transfer-fee configuration and withheld-fee withdrawal authorities, permanent delegates, pause authorities, metadata authorities, and transfer-hook relationships.[7][14][15][16]

Mesh should not interpret those fields in isolation. Authority analysis should determine who controls the capability, whether the same controller holds other sensitive privileges, and whether associated executable program logic remains mutable.

This is one reason Token-2022 analysis is integrated into the broader Mesh architecture rather than treated as an isolated extension scanner.

Appendix E - Historical and Creator-Evidence Coverage

Historical analysis differs from current-state account analysis because it often requires reconstruction of past transactions rather than one direct observation of current account state.

Mesh may use bounded acquisition strategies to keep history analysis reliable, reproducible, and operationally practical. A bounded strategy can establish real evidence without claiming that every historical transaction associated with an address has been recovered.

E.1 Observed History

Historical findings should be phrased in terms of what Mesh actually observed.

If the system recovers one verified deployment associated with a creator address, the supported conclusion is that one deployment was observed. It is not necessarily that only one deployment has ever existed.

This distinction becomes particularly important when the historical evidence state is PARTIAL.

E.2 Coverage

Coverage communicates how confidently Mesh can treat the acquired history as representative of the relevant search scope.

A historical result may be considered more complete where the acquisition strategy successfully covered the defined range required by the methodology. It may remain partial where provider limits, bounded queries, malformed transactions, unsupported historical structures, or other conditions prevent the system from establishing the desired scope.

Historical coverage should influence the certainty of creator conclusions without erasing valid observed events.

E.3 Conflicting Historical Records

Duplicate transaction signatures, conflicting transaction payloads, malformed historical records, or inconsistent provider responses should not be allowed to erase previously validated evidence silently.

Where multiple records refer to the same underlying transaction, Mesh should use explicit consistency and preference rules to preserve valid observations while identifying material conflicts. If disagreement cannot be reconciled safely, historical coverage should degrade rather than allowing the cleaner interpretation to be selected automatically.

Creator history is intended to provide context rather than prove intent. Even complete historical evidence cannot establish that a developer will behave maliciously or responsibly in the future.

Appendix F - Ownership Evidence and Concentration

Ownership analysis begins with token-account balances but should attempt to distinguish between token accounts and the observable controllers behind them.

Raw token-account rankings can overstate or understate meaningful concentration. Several accounts may be controlled by the same address, while some large accounts may belong to recognized liquidity infrastructure, programs, treasuries, custodial systems, or other nonstandard holders.

Mesh therefore attempts to preserve separate concepts for:

- token-account concentration;
- controller-level aggregation;
- recognized automated-market-maker balances;
- standard-wallet exposure;
- unresolved ownership classifications.

F.1 Bounded Samples

Early versions of Mesh may use a bounded sample of the largest observable token accounts rather than attempting exhaustive analysis of every account holding the asset.

Conclusions derived from such a sample should be labeled accordingly. A top-ten concentration metric describes the observed top-ten sample; it does not represent complete beneficial ownership of the entire supply.

The sample can provide material concentration evidence, but it does not prove complete holder resolution. In the released analytical baseline, the ownership classifier and PumpSwap-associated ownership label are separate from the stricter pool-authentication path used for exit quotes; an ownership label must not be described as an authenticated liquidity pool.[45] Broader controller-completeness guarantees remain development requirements rather than consequences of a numeric concentration result.

F.2 Real-World Identity

Onchain controller analysis is not equivalent to real-world identity attribution.

One person or organization may control several addresses that Mesh cannot reliably link. Conversely, an address may represent a multisig, custodian, program, exchange, or infrastructure account serving many independent users.

BAREMAX should therefore avoid claiming definitive beneficial ownership unless the evidence supporting the relationship is independently established. The primary purpose of this domain is to identify observable onchain control and concentration.

Appendix G - Program Upgradeability

Program upgradeability is relevant whenever executable logic influences a security-sensitive interaction.

Under Solana's loader-v3 deployment model, a program may remain upgradeable while an upgrade authority exists. Revoking that authority makes the program immutable under that deployment mechanism.[11]

Mesh should therefore distinguish three states:

Program State	Interpretation
MUTABLE	Mesh established that an active upgrade authority remains.
IMMUTABLE	Mesh established the conditions required to demonstrate that the program can no longer be upgraded under the supported deployment model.
UNKNOWN	Mesh could not acquire or validate enough program evidence to establish either state.

The UNKNOWN state is important because a failed ProgramData lookup or malformed program relationship must not be interpreted as proof of immutability.

Upgradeability should be interpreted within the identified loader and supported validation rules. An immutable executable can still consult mutable account state, and a Token-2022 hook-configuration authority may select another program even when the currently selected program is immutable.[20] These are separate control relationships. The table states the intended evidence meanings; it must not be read as proof that every baseline error path already produces these exact outcomes.

Appendix H - Risk Interpretation and Consumer Labels

The consumer-facing labels used by Mesh are intended to summarize domain-specific findings without replacing the evidence underneath them.

The proposed consumer vocabulary is SAFE, CAUTION, HIGH RISK, and UNKNOWN. The inspected engine instead exposes domain-specific strings such as LOWER, ELEVATED, HIGH, and UNKNOWN through its existing result contract.[9][45] These vocabularies are not an automatic one-to-one mapping; in particular, LOWER does not meet the proposed SAFE standard merely because it is the lowest resolved risk classification.

SAFE represents a positive security conclusion within the validated scope of Mesh. Material supported checks must be sufficiently resolved, and no significant supported risk should remain.

CAUTION is proposed consumer wording for an established condition requiring attention. Existing ELEVATED engine results retain their implemented meaning until a versioned presentation policy is validated. A cautionary message accompanying UNKNOWN does not transform missing evidence into an established intermediate risk level.

HIGH RISK represents a supported condition or combination of conditions judged sufficiently serious to warrant a strong security warning.

UNKNOWN represents insufficient validated evidence to justify a stronger conclusion. UNKNOWN is not intended to function as a midpoint between safe and dangerous.

Overall interpretation should preserve severe domain findings rather than averaging them away. Likewise, material unresolved evidence should remain visible even when other domains appear favorable.

The evidence, rather than the label, remains the authoritative basis for the result.

Appendix I - Methodology Versioning and Reproducibility

The BAREMAX methodology is expected to evolve as new protocols, threat classes, production failures, and analytical techniques are incorporated into Mesh. That evolution should be traceable.

Document revision, application source, release record, result schema and observation time are separate identifiers. This October 28 revision describes deployed application commit `4cf69b52daf12b7c5e91eed0d2052ee8d035ee7b` and its release record at `bfa472a1e2e9e16237408ddc498fc1ef89507882`. The LP evidence candidate is identified separately in Section 10.1. Result-version fields retain their implemented meaning.[9][17] Changes to a document do not change the scoring methodology or reconstruct missing historical observations.

Where practical, a Mesh analysis should preserve enough information to reconstruct the relevant security context, including:

- analysis timestamp;
- network;
- methodology or result version;
- relevant asset or wallet identifiers;
- material evidence states;
- domain conclusions;
- liquidity provenance where applicable;
- program and authority relationships;
- important unresolved conditions.

This information supports reproducibility, incident analysis, debugging, regression testing, and future comparison between methodology versions.

A historical result should not necessarily be expected to match a result produced months later if the blockchain state or BAREMAX methodology has changed. The relevant requirement is that BAREMAX should be able to explain why the conclusions differ.

Appendix J - Scope of the Whitepaper

This whitepaper describes the design philosophy, architecture, threat model, current development direction, and intended evolution of BAREMAX Mesh. It is not intended to serve as exhaustive protocol documentation or a permanent specification of every implementation detail.

Detailed decoder layouts, individual RPC strategies, test fixtures, protocol-specific pool authentication rules, internal thresholds, software interfaces, and similar implementation details should remain in engineering documentation unless their inclusion is necessary to explain a public security guarantee.

This separation is deliberate. The whitepaper should establish the principles users, developers, and potential partners can reasonably expect BAREMAX Labs to follow, while technical documentation can evolve more rapidly as the system expands.

The core commitments described throughout the whitepaper are more durable than any individual implementation: evidence should be validated before it influences security conclusions, uncertainty should remain explicit, material risks should remain explainable, user protection should remain non-custodial, and expansion should occur only where the underlying security engine can support it reliably.

References

External protocol and market references retain their original September 22-23 access dates unless noted otherwise. Project implementation references were reconciled with the October 28 release record. Repository links are project-maintained engineering evidence and may require access; they are not independent certifications.

Publication dates, original access dates and the current project-release checkpoint are recorded separately. References support the specific statements they accompany.

[1] Solana Foundation, “Fees.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/core/fees

[2] Visa Inc., “Visa Launches Stablecoin Settlement in the United States, Marking a Breakthrough for Stablecoin Integration.” December 16, 2025. Accessed September 22, 2026.

https://corporate.visa.com/en/sites/visa-perspectives/newsroom/visa-launches-stablecoin-settlement-in-the-united-states.html

[3] Securitize, “BlackRock and Securitize Debut New BUIDL Share Class on Solana Network.” March 25, 2025. Accessed September 22, 2026.

https://investors.securitize.io/news/news-details/2025/BlackRock-and-Securitize-Debut-New-BUIDL-Share-Class-on-Solana-Network-03-25-2025/default.aspx

[4] Franklin Templeton, “Franklin Templeton Expands Its Digital Asset ETP Suite With Launch of Franklin Solana ETF (SOEZ).” December 3, 2025. Accessed September 22, 2026.

https://www.franklintempleton.com/press-releases/news-room/2025/franklin-templeton-expands-its-digital-asset-etp-suite-with-launch-of-franklin-solana-etf-soez

[5] Solana Foundation, “Overview of Institutional Real World Assets on Solana.” July 30, 2026. Accessed September 22, 2026.

https://solana.com/news/overview-of-institutional-real-world-assets-on-solana

[6] M. Souppaya, K. Scarfone, and D. Dodson, “Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities.” NIST Special Publication 800-218. February 2022. Accessed September 22, 2026.

https://csrc.nist.gov/pubs/sp/800/218/final

[7] Solana Foundation, “Token Extensions.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions

[8] Solana Foundation, “Core Concepts.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/core

[9] BAREMAX project, “InspectionSummary Interface.” Repository snapshot at commit bfa472a1e2e9e16237408ddc498fc1ef89507882. Reviewed against the October 28, 2026 release record.

https://github.com/brosanwich/BAREMAX/blob/bfa472a1e2e9e16237408ddc498fc1ef89507882/app/inspection-summary.ts

[10] Solana Foundation, “Accounts.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/core/accounts

[11] Solana Foundation, “Programs.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/core/programs

[12] Solana Foundation, “SPL Token Basics.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/basics

[13] Solana Foundation, “Terminology.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/references/terminology

[14] Solana Foundation, “Token Extensions: Transfer Hook.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions/transfer-hook

[15] Solana Foundation, “Transfer Fees.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions/transfer-fees

[16] Solana Foundation, “Permanent Delegate.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions/permanent-delegate

[17] BAREMAX project, “Structured Analysis Boundary.” Repository snapshot at commit bfa472a1e2e9e16237408ddc498fc1ef89507882. Reviewed against the October 28, 2026 release record.

https://github.com/brosanwich/BAREMAX/blob/bfa472a1e2e9e16237408ddc498fc1ef89507882/app/ANALYSIS-BOUNDARY.md

[18] OWASP Foundation, “Fail Securely.” Accessed September 22, 2026.

https://community.owasp.org/Fail_securely

[19] Solana Program contributors, “Token-2022 processor: PermanentDelegate authority handling.” Token-2022 source code, program@v10.0.0. Accessed September 22, 2026.

https://raw.githubusercontent.com/solana-program/token-2022/program@v10.0.0/program/src/processor.rs

[20] Solana Program contributors, “Token-2022 transfer-hook processor: configuration updates.” Token-2022 source code, program@v10.0.0. Accessed September 22, 2026.

https://raw.githubusercontent.com/solana-program/token-2022/program@v10.0.0/program/src/extension/transfer_hook/processor.rs

[21] Solana Foundation, “Non-Transferable Tokens.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions/non-transferrable-tokens

[22] Solana Foundation, “Pausable Mint.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions/pausable

[23] Solana Foundation, “Scaled UI Amount.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions/scaled-ui-amount

[24] Solana Foundation, “Metadata Pointer & Token Metadata.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/extensions/metadata

[25] Pump.fun, “The Pump.fun Bonding Curve.” Pump.fun Documentation. Accessed September 22, 2026.

https://pump.fun/docs/bonding-curve

[26] Raydium, “Concentrated-liquidity Math.” Raydium Documentation. Accessed September 22, 2026.

https://docs.raydium.io/algorithms/clmm-math

[27] BAREMAX project, “Authenticated Single-venue Exit Aggregation.” Repository snapshot at commit bfa472a1e2e9e16237408ddc498fc1ef89507882. Reviewed against the October 28, 2026 release record.

https://github.com/brosanwich/BAREMAX/blob/bfa472a1e2e9e16237408ddc498fc1ef89507882/app/MULTI-VENUE-EXITS.md

[28] Raydium, “Constant-product AMM.” Raydium Documentation. Accessed September 22, 2026.

https://docs.raydium.io/algorithms/constant-product

[29] Raydium, “Raydium Documentation.” Accessed September 22, 2026.

https://docs.raydium.io/

[30] Solana Foundation, “Solana Wallets.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/wallets

[31] Solana Developers, “Solana Actions and Blockchain Links (Blinks).” @solana/actions SDK Documentation. Accessed September 23, 2026.

https://solana-developers.github.io/solana-actions/

[32] BAREMAX project, “BAREMAX - README.” Repository snapshot at commit bfa472a1e2e9e16237408ddc498fc1ef89507882. Reviewed against the October 28, 2026 release record.

https://github.com/brosanwich/BAREMAX/blob/bfa472a1e2e9e16237408ddc498fc1ef89507882/README.md

[33] BAREMAX project, “BAREMAX Web Interface.” Repository snapshot at commit bfa472a1e2e9e16237408ddc498fc1ef89507882. Reviewed against the October 28, 2026 release record.

https://github.com/brosanwich/BAREMAX/blob/bfa472a1e2e9e16237408ddc498fc1ef89507882/web/README.md

[34] BAREMAX project, “October 28 beta release notes.” Project-maintained release record; reviewed October 28, 2026. Release record.

https://github.com/brosanwich/BAREMAX/pull/58

[35] Solana Foundation, “How to Connect a Wallet with React.” Solana Cookbook. Accessed September 22, 2026.

https://solana.com/developers/cookbook/wallets/connect-wallet-react

[36] Solana Foundation, “Partial Signing.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/core/transactions/partial-signing

[37] Phantom, “Establish a Connection.” Phantom Developer Documentation. Accessed September 22, 2026.

https://docs.phantom.com/solana/establishing-a-connection

[38] Solana Foundation, “simulateTransaction.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/rpc/http/simulatetransaction

[39] Solana Foundation, “Writing to the Network.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/intro/quick-start/writing-to-network

[40] Solana Foundation, “Fee Structure.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/core/fees/fee-structure

[41] Solana Foundation, “Minimal Mobile Kit Expo Example.” Solana Developer Templates. Accessed September 22, 2026.

https://solana.com/developers/templates/kit-expo-minimal

[42] Solana Mobile, “Mobile Wallet Adapter.” Solana Mobile Documentation. Accessed September 22, 2026.

https://docs.solanamobile.com/developers/mobile-wallet-adapter

[43] Solana Foundation, “Clusters and Public RPC Endpoints.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/references/clusters

[44] Solana Foundation, “Create a Token Account.” Solana Documentation. Accessed September 22, 2026.

https://solana.com/docs/tokens/basics/create-token-account

[45] BAREMAX project, “Analysis Engine.” Repository snapshot at commit bfa472a1e2e9e16237408ddc498fc1ef89507882. Reviewed against the October 28, 2026 release record.

https://github.com/brosanwich/BAREMAX/blob/bfa472a1e2e9e16237408ddc498fc1ef89507882/app/analysis-engine.ts