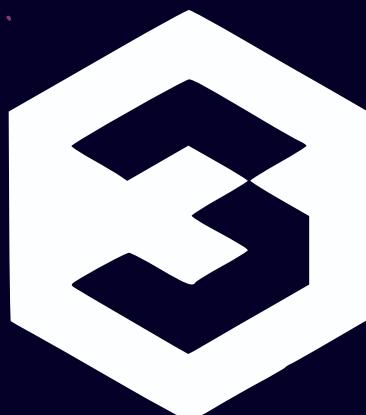


INTRODUCTION TO WEB 3 SECURITY

web



فهرست

2.....	1.1	تمرکزدایی و وب 3 (Decentralization and Web3)
2.....	1.1.1	2.....
2.....	2.1	3.....
3.....	3.1	4.....
4.....	1.3.1	در ادامه جدول زیر راهنماییست برای مسیر شما در موضوع web3
7.....	2	8.....
8.....	2	ویژگی‌های امنیتی (CIA) در فناوری Web3
8.....	1.2	8.....
8.....	2.2	1.....
9.....	3.2	در دسترس بودن (Availability)
10.....	3	امنیت در Web3 و مدل CIA
11.....	3	چگونه آدرس بیت‌کوین ایجاد می‌شود
14.....	4	برنامه‌ی باگ‌بانتی (Bug Bounty) در Web3
14.....	4	چگونه یک برنامه باگ‌بانتی موفق اجرا کنیم؟
16.....	4	Immunefi پلتفرم تخصصی برای پروژه‌های Web3 و قراردادهای هوشمند
18.....	5	جدول نمونه آسیب‌پذیری‌های Web3 با شناسه CVE
19.....	5	تدابع باگ‌بانتی در فاز تولید (Mainnet)
20.....	5	سیستم نظارت و هشدار (Monitoring / Alerting)
6.....	6	1 - تفاوت بین شبکه‌های متمرکز، غیرمتمرکز و توزیع شده
13.....	6	2 - نحوه ساخت بیتکوین



۱. تمرکز دایی و وب ۳ (Decentralization and Web3)

۱.۱ وب ۱.۰

در دورهی وب ۱.۰، وب سایت‌ها بیشتر محتوای ایستا (Static) ارائه می‌دادند، یعنی محتوا به صورت ثابت نوشته شده بود و برخلاف محتواهای پویا، تغییر یا تعامل چندانی نداشت. این محتواها با استفاده از زبان نشانه‌گذاری HTML ساخته می‌شدند.

داده‌ها و محتواها از طریق یک سیستم فایل ایستا (Static File System) ارسال می‌شدند و نه از طریق پایگاه داده Database صفحات وب فقط مقدار بسیار کمی اطلاعات تعاملی داشتند.

فناوری‌های اصلی مورد استفاده در وب ۱.۰ شامل موارد زیر بودند:

- زبان نشانه‌گذاری HTML
- URL رمز‌گذاری شده در پروتکل HTTP پروتکل انتقال

۲.۰ وب ۲.۱

چون بیشتر محتوای وب ۱.۰ ایستا بود، این موضوع باعث شد که وب ۲.۰ شکل بگیرد. اکثر ما فقط با نسخه‌ی فعلی وب، یعنی وب ۲.۰ آشنا هستیم. این نسخه با عنوان‌های دیگری مانند وب تعاملی (read-write web) یا وب اجتماعی نیز شناخته می‌شود. در وب ۲.۰ برای مشارکت در فرایند تولید محتوا نیازی به داشتن تجربه‌ی برنامه‌نویسی نیست. بسیاری از برنامه‌ها طوری طراحی شده‌اند که هر کسی بتواند نسخه‌ی شخصی خودش را از آن‌ها بسازد.



شما می‌توانید ایده‌هایی خلق کرده و آن‌ها را با افراد مختلف در سراسر جهان به اشتراک بگذارید. به عنوان مثال، می‌توانید یک ویدیو را در وب 2.0 منتشر کنید و میلیون‌ها نفر آن را ببینند، درباره‌اش نظر بدهند و با آن تعامل داشته باشند.

اپلیکیشن‌های وب 2.0 شامل شبکه‌های اجتماعی مانند یوتیوب، فیسبوک، فلیکر، اینستاگرام و توییتر هستند. به محبوبیت این سایت‌ها در زمان شروع به کارشان فکر کنید و آن را با محبوبیت امروزشان مقایسه کنید.

وب 2.0 این امکان را فراهم کرد که برنامه‌ها در مقیاس بالا اجرا شوند، اما در عین حال باعث شکل‌گیری پلتفرم‌های متتمرکزی شد که کنترل کامل داده‌های کاربران را در اختیار گرفتند. این کنترل به آن‌ها فرصت داد تا از داده‌ها استخراج اطلاعات (**Data Mining**) انجام دهند؛ چیزی که می‌تواند کاربردهای مثبت یا منفی داشته باشد.

با افزایش کنترل این پلتفرم‌های متتمرکز بر کاربران، کم کم کاربران کنترل خود را از دست دادند و این مسئله باعث شد زمینه برای شکل‌گیری وب 3.0 فراهم شود.

3.0 وب 3.1

طراحی وب 2.0 به صورت ذاتی به سیستم‌های متتمرکز وابسته بود. حتی اگر این سیستم‌ها از لحاظ فنی توزیع شده بودند، ولی غیرمتتمرکز (Decentralized) نبودند؛ به این معنا که کنترل این سیستم‌ها در اختیار یک فرد یا گروه محدودی باقی می‌ماند.

این موضوع باعث به وجود آمدن مشکلات بزرگی شد؛ مخصوصاً وقتی مسائل مربوط به حریم خصوصی و استفاده از داده‌ها مطرح شد.

اینجا بود که وب 3.0 شکل گرفت؛ با تمرکز ویژه بر غیرمتتمرکز بودن.



در وب 3.0 هم اجرای برنامه‌ها و هم ذخیره‌سازی داده‌ها باید به صورت غیرمت مرکز انجام شود.

بنابراین، قبل از درک چشم‌انداز وب 3، باید با مفهوم غیرمت مرکز بودن آشنا شد. (Decentralization)

1.3.1 معرفی غیرمت مرکز بودن

"احترام بی‌چون‌وچرا به قدرت، بزرگ‌ترین دشمن حقیقت است"

این جمله معروف از آلبرت اینشتین ما را وادار به تفکر می‌کند: آیا باید سیستم‌هایی وجود داشته باشند که تحت کنترل یک قدرت مرکزی باشند و ما مجبور باشیم به آن قدرت اعتماد کامل داشته باشیم؟ یا اینکه بهتر است سیستم‌ها غیرمت مرکز باشند و تصمیم‌گیری‌ها بر اساس اجماع جمعی انجام شود؟

سیستم‌های مت مرکز، همان‌طور که از نامشان پیداست، می‌توانند تحت سلطه یا تصمیمات یک مرجع خاص قرار گیرند. اما در مقابل، یک سیستم غیرمت مرکز تصمیم‌گیری‌های شامل‌تر و مشارکتی را ترجیح می‌دهد؛ که این موضوع احتمال فساد و سوءاستفاده را در آن سیستم به‌طور چشمگیری کاهش می‌دهد.

در یک ساختار غیرمت مرکز، مالک یا رئیس مشخصی وجود ندارد. تنها گره‌هایی (Node) هستند که در یک شبکه با هم مشارکت دارند. این گره‌ها با هم پیام‌رد و بدل می‌کنند و هر زمان که لازم باشد، از طریق اجماع (Consensus) به یک توافق مشترک می‌رسند.

پروتکل‌هایی که این شبکه‌ها را کنترل می‌کنند، تضمین می‌کنند که کل سیستم از لحاظ داده‌ای در وضعیت یکپارچه و هماهنگ قرار دارد.

هر گره در شبکه دارای یک نسخه به روز از تمام داده‌هایی است که تاکنون ثبت شده‌اند.



شبکه‌های غیر مرکزی حتی می‌توانند داده‌ها را طوری توزیع کنند که بتوان اطلاعات خصوصی خاصی را اعتبارسنجی کرد؛ بدون آنکه نیاز باشد این اطلاعات به شخص ثالثی داده شود.

این نوع اعتبارسنجی بدون افشاء اطلاعات، به لطف استفاده از تکنیک‌هایی مبتنی بر اجماع انجام می‌شود؛ جایی که گره‌های شبکه در یک لحظه مشخص بر سر وضعیت کلی سیستم به توافق می‌رسند.

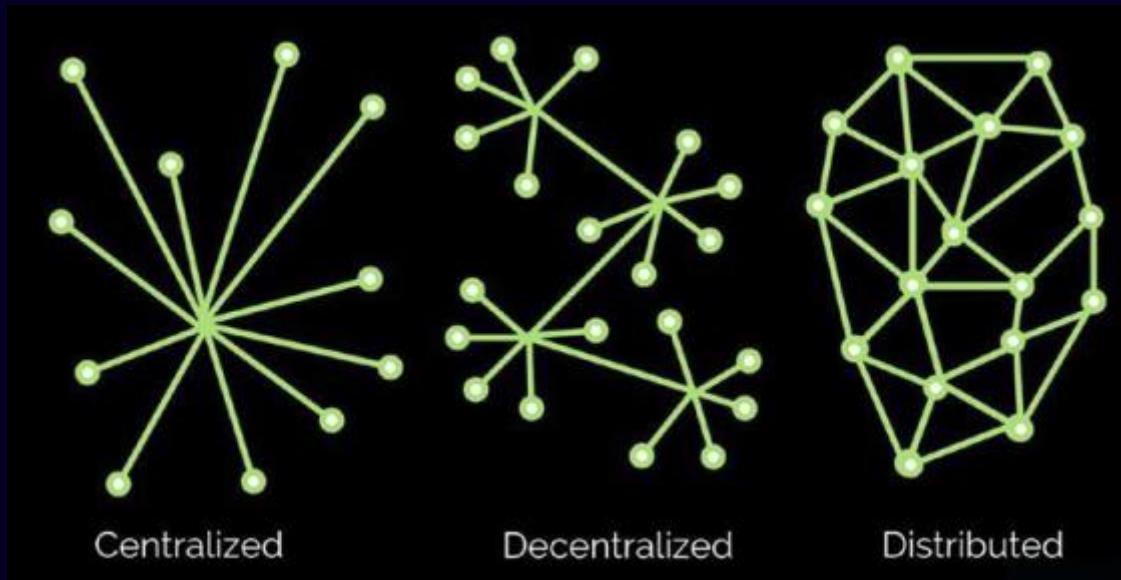
هر گره‌ای که در یک شبکه غیر مرکزی مشارکت دارد، به صورت مستقل از سایر گره‌ها عمل می‌کند.

گره‌ها از استانداردهای مشترک برای ارتباط با یکدیگر استفاده می‌کنند، اما استقلال خود را حفظ کرده و خودشان مسئول مدیریت حریم خصوصی خود هستند، نه اینکه از دستورات یک قدرت مرکزی پیروی کنند.

این ویژگی نه تنها به امنیت شبکه کمک می‌کند، بلکه تضمین می‌کند که اداره‌ی شبکه به صورت دموکراتیک انجام می‌شود.



شکل ۱-۱ نشان می‌دهد که گره‌ها در شبکه‌های متتمرکز، غیرمتتمرکز و توزیع شده چگونه به هم متصل شده و با هم ارتباط برقرار می‌کنند.^۱.



۱ - تفاوت بین شبکه‌های متتمرکز، غیرمتتمرکز و توزیع شده

^۱ پیشنهاد می‌شود مطالب زیر را حتما مطالعه بفرمایید این کتاب صرفا خلاصه وار به مطالب پرداخته است :

- Different Topologies for Networks
- Centralized and Non-distributed
- Centralized but Distributed
- Decentralized Systems



در ادامه جدول زیر راهنماییست برای مسیر شما در موضوع web3

پله پیش‌نیاز	عنوان	زیر‌فصل‌ها
2	Blockchain	<ul style="list-style-type: none"> 2.1 Types of Blockchains <ul style="list-style-type: none"> 2.1.1 Public Blockchain 2.1.2 Private Blockchain 2.1.3 Permissioned Blockchain 2.2 What Is a Blockchain? 2.3 Blockchain Building Blocks <ul style="list-style-type: none"> 2.3.1 Block 2.3.2 Chain 2.3.3 Network 2.4 Where Is Blockchain Used? 2.5 Evolution 2.6 Consensus <ul style="list-style-type: none"> 2.6.1 Proof of Work 2.6.2 Proof of Stake 2.7 Blockchain Architecture 2.8 Cryptographic Keys 2.9 Blockchain Compared to a Singly Linked List 2.10 Ethereum 2.11 Summary
3	Solidity	<ul style="list-style-type: none"> 3.1 What Is Solidity? 3.2 Ethereum <ul style="list-style-type: none"> 3.2.1 Ethereum Virtual Machine 3.3 Smart Contracts 3.4 Making Sense of Solidity Syntax <ul style="list-style-type: none"> 3.4.1 Pragma 3.4.2 Variables 3.4.3 Value Types 3.4.4 Address 3.4.5 Operators in Solidity 3.4.6 Loops 3.4.7 Decision Flows 3.4.8 Functions in Solidity 3.4.9 Abstract Contracts 3.4.10 Interface 3.4.11 Libraries 3.4.12 Events 3.4.13 Error Handling in Solidity 3.4.14 Solidity and Addresses 3.5 Summary
4	Wallets and Gateways	<ul style="list-style-type: none"> 4.1 Types of Wallets 4.2 So, What Is a Testnet? 4.3 MetaMask <ul style="list-style-type: none"> 4.3.1 Installation 4.3.2 Web3.js <ul style="list-style-type: none"> 4.4.1 web3-eth 4.4.2 web3-shh 4.4.3 web3-bzz 4.4.4 web3-net 4.4.5 web3-utils 4.5 Infura Setup 4.5.1 Interfacing with Ropsten Network via Infura Gateway 4.6 Summary
5	Introduction to Remix IDE	<ul style="list-style-type: none"> 5.1 Remix IDE 5.2 Creating Own Token 5.3 Summary
6	Truffle	<ul style="list-style-type: none"> 6.1 Truffle Installation <ul style="list-style-type: none"> 6.1.1 Installing Node 6.1.2 Install Truffle 6.2 Smart Contract Deployment via Truffle <ul style="list-style-type: none"> 6.2.1 Contract Code 6.2.2 Compile and Deploy the Contract 6.3 Summary
7	IPFS and NFTs	<ul style="list-style-type: none"> 7.1 IPFS <ul style="list-style-type: none"> 7.1.1 IPFS: 30,000-Foot View 7.1.2 Installation 7.2 ERC-721 7.3 Creating an ERC-721 Token and Deploying It to IPFS 7.4 Summary
8	Hardhat	<ul style="list-style-type: none"> 8.1 Installation of Hardhat Framework 8.2 Workflow for Hardhat 8.3 Deployment of the Smart Contract 8.4 Summary



2. ویژگی‌های امنیتی (CIA) در فناوری Web3

این فصل به بررسی ویژگی‌های سه‌گانه‌ی امنیت اطلاعات یعنی محرمانگی (Integrity)، یکپارچگی (Confidentiality) و دردسترس‌بودن (Availability) می‌پردازد که به اختصار به آن‌ها مدل CIA گفته می‌شود.

در حوزه‌ی امنیت سایبری، مدل سه‌گانه‌ی CIA یکی از مفاهیم پایه و بنیادی برای محافظت از اطلاعات حساس سازمان‌هاست. این سه رکن، شالوده‌ی طراحی و اجرای تدابیر مؤثر در امنیت اطلاعات را تشکیل می‌دهند:

1.2 محرمانگی (Confidentiality)

محرمانگی نقش مهمی در جلوگیری از دسترسی غیرمجاز به اطلاعات دارد. این موضوع زمانی اهمیت دوچندان پیدا می‌کند که داده‌هایی مانند اطلاعات شخصی، سوابق مالی یا دارایی‌های فکری در میان باشند.

با اطمینان از اینکه فقط افراد مجاز به این اطلاعات دسترسی دارند، ریسک نشت اطلاعات و آسیب به اعتبار سازمان کاهش می‌یابد.

2.2 یکپارچگی (Integrity)

اطمینان از درستی و قابل اعتماد بودن اطلاعات نیز حیاتی است. هرگونه تغییر یا دستکاری غیرمجاز در داده‌ها می‌تواند منجر به خطا، تقلب یا پیامدهای جدی شود. حفظ یکپارچگی داده‌ها به سازمان‌ها کمک می‌کند تا بتوانند تصمیمات دقیق و بر پایه‌ی اطلاعات واقعی اتخاذ کنند.



3.2 در دسترس بودن (Availability)

سومین ستون این مدل، دسترسی‌پذیری اطلاعات برای کاربران مجاز در زمان مناسب است.

در صورت عدم دسترسی به داده‌های حیاتی، فعالیت‌های روزانه سازمان ممکن است مختلف شود، که می‌تواند باعث کاهش بهره‌وری، نارضایتی مشتریان و در نهایت ضرر مالی شود.

کارشناسان امنیت سایبری برای تعریف امنیت، از مدل CIA استفاده می‌کنند، چون این مدل چارچوبی جامع برای درک و رسیدگی به جنبه‌های مختلف امنیت اطلاعات ارائه می‌دهد. این سه اصل به‌طور کامل با یکدیگر در ارتباط هستند و باید همزمان و هماهنگ مورد توجه قرار گیرند تا حفاظت مؤثر از اطلاعات و سیستم‌ها محقق شود.

با تمرکز بر این مدل، متخصصان امنیت می‌توانند:

- مطمئن شوند که همه‌ی جوانب امنیت اطلاعات را پوشش داده‌اند.
- تلاش‌ها و منابع را بر اساس اهداف کلیدی امنیتی اولویت‌بندی کنند.
- اطمینان حاصل کنند که تدابیر امنیتی مؤثر و بهینه اجرا شده‌اند.



امنیت در Web3 و مدل CIA

از آنجا که Web3 یک سیستم اطلاعاتی مبتنی بر ارزش و خودمختار است، مدل امنیتی CIA همچنان در آن قابل اعمال و کاربردی (Self-sovereign) است. در هسته‌ی Web3، فناوری بلاکچین وجود دارد که در زمینه‌ی امنیت، مزایا و چالش‌های خاص خود را دارد.

در ادامه، خلاصه‌ای از تحلیل مدل CIA در زمینه Web3 آورده شده است:

• محترمانگی (Confidentiality):

در Web3، به دلیل شبکه‌نشناس بودن (Pseudo-anonymity) در بلاکچین‌های عمومی و عدم استفاده از فناوری‌های حفظ حریم خصوصی، نمی‌توان سطح کافی از محترمانگی را تضمین کرد. بنابراین، صرفاً تکیه بر معماری اصلی بلاکچین برای محترمانگی کافی نیست.

• یکپارچگی (Integrity):

ما در مورد ماهیت تغییرناپذیر (Immutable) دفترکل بلاکچین صحبت می‌کنیم که می‌تواند سطحی از یکپارچگی را فراهم کند. اما این موضوع وابسته به مقاومت زنجیره در برابر حملات Fork و استحکام الگوریتم اجماع (Consensus) در طراحی امنیت و زندگی بودن (Liveness) آن است.

• دردسترس بودن (Availability):

در Web3، میزان در دسترس بودن اطلاعات به تکثیر داده‌ها (Replication)، مقیاس‌پذیری و تأخیر شبکه‌ی بلاکچین، و همچنین تعامل میان داده‌های روی زنجیره (On-chain) و خارج از زنجیره (Off-chain) وابسته است.



3. چگونه آدرس بیتکوین ایجاد می‌شود

آدرس بیتکوین یک شناسه‌ی منحصر به فرد است که برای ارسال و دریافت تراکنش‌های بیتکوین استفاده می‌شود. این آدرس با استفاده از ترکیبی از چندین تکنیک رمزنگاری پیشرفته تولید می‌شود؛ از جمله:

- الگوریتم هش ایمن (SHA)

- الگوریتم امضای دیجیتال منحنی بیضوی (ECDSA)

در ادامه فرآیند ایجاد آدرس بیتکوین به صورت گام‌به‌گام توضیح داده شده است (به شکل 1.1 رجوع شود):

مرحله اول: ایجاد کلید خصوصی

اولین گام برای تولید آدرس بیتکوین، ایجاد یک کلید خصوصی است. کلید خصوصی یک عدد محرمانه و تصادفی است که برای امضای تراکنش‌ها و اثبات مالکیت بیتکوین‌ها کار به می‌رود.

این کلید معمولاً با استفاده از یک تولیدکننده عدد تصادفی شباهمن رمزنگاری شده (CSPRNG) ایجاد می‌شود.

مرحله دوم: ایجاد کلید عمومی

پس از تولید کلید خصوصی، از آن برای تولید یک کلید عمومی استفاده می‌شود. کلید عمومی از طریق منحنی بیضوی (secp256k1) که مربوط به الگوریتم ECDSA است (و بر پایه روابط ریاضی خاصی از کلید خصوصی مشتق می‌شود).



این کلید عمومی یک نقطه روی منحنی است که به طور مستقیم از کلید خصوصی محاسبه می‌شود.

مرحله سوم: هش کردن کلید عمومی

کلید عمومی با استفاده از تابع هش **SHA-256** رمزگاری می‌شود و خروجی آن یک رشته‌ی 64 کاراکتری است که به آن هش کلید عمومی گفته می‌شود.

مرحله چهارم: رمزگذاری و تولید آدرس نهایی

برای تولید آدرس نهایی بیت‌کوین، چندین گام دیگر طی می‌شود:

1. هش کلید عمومی با یک شماره نسخه (**Version Number**) و یک کد بررسی (**Checksum**) ترکیب می‌شود.

2. این ترکیب با استفاده از **Base58** تبدیل به یک رشته متنی می‌شود. یک شیوه‌ی رمزگذاری است که داده‌ها را به صورت حروف و اعداد قابل خواندن برای انسان نمایش می‌دهد.

3. در نهایت، برای کوتاه‌تر و مقاوم‌تر شدن آدرس، تابع هش **RIPEMD-160** نیز بر روی نتیجه اعمال می‌شود.

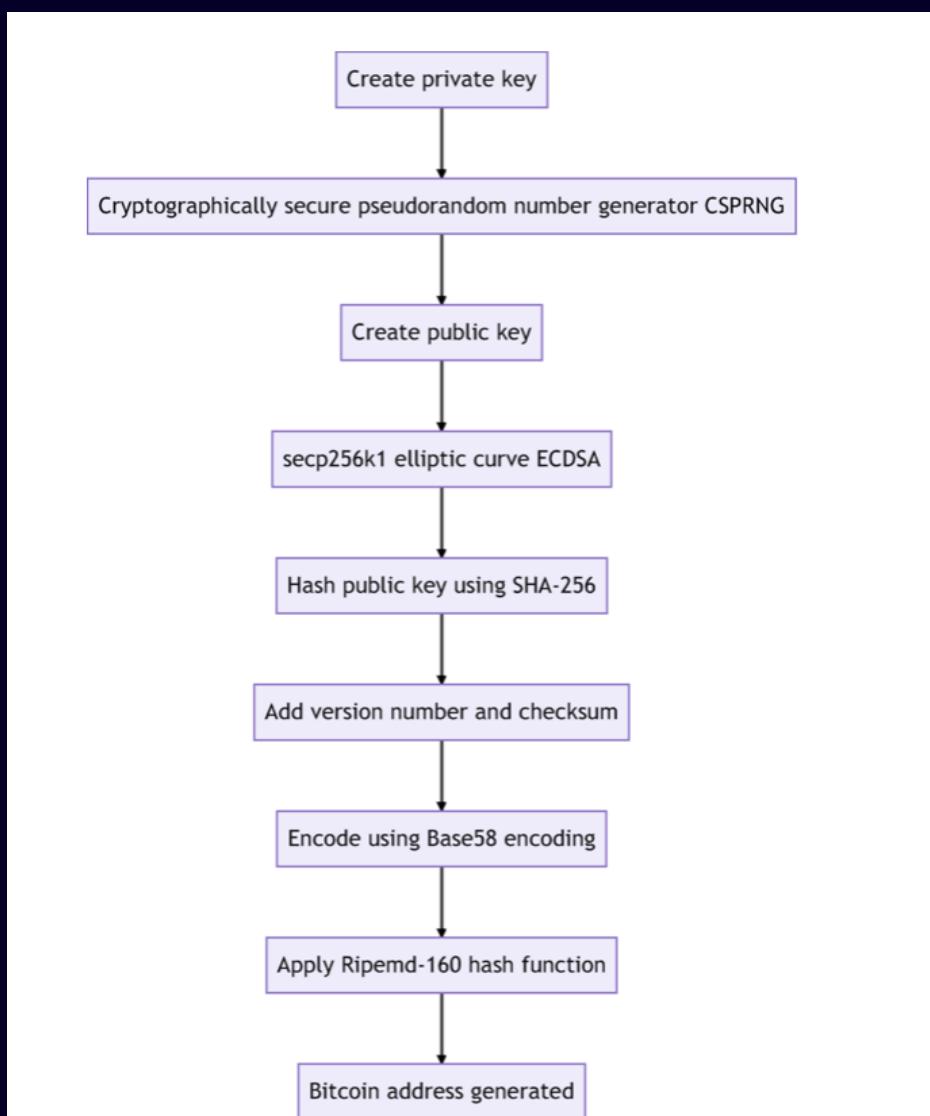
نتیجه‌ی این فرآیند یک آدرس 160 بیتی است که هم مقاوم در برابر خطا و هم کارآمدتر برای استفاده می‌باشد.



: نتیجه گیری

فرآیند تولید آدرس بیتکوین از طریق این الگوریتم‌ها و مراحل، باعث می‌شود که امنیت و یکپارچگی شبکه‌ی بیتکوین حفظ شود.

همچنین، این فرآیند به کاربران اجازه می‌دهد تا آدرس‌هایی قابل اشتراک‌گذاری و نسبتاً خوانا داشته باشند، درحالی‌که این آدرس‌ها از نظر فنی غیرقابل پیش‌بینی و غیرقابل جعل هستند.



2 - نحوه ساخت بیتکوین Figure



4. برنامه‌ی باگ‌بانتی (Bug Bounty) در Web3

پس از انجام ممیزی امنیتی (Security Audit)، اجرای یک برنامه‌ی فعال باگ‌بانتی برای حفظ امنیت پروژه Web3 کاملاً حیاتی است.

برخلاف اتکا به یک یا چند متخصص امنیت داخلی، باگ‌بانتی‌ها فرصت استفاده از توان تخصصی جامعه جهانی از هکرهای کلاه‌سفید و پژوهشگران امنیتی را فراهم می‌کنند. این افراد با سطح‌های مختلف مهارت و پیش‌زمینه‌های متنوع، می‌توانند امنیت پروژه را از زوایای گوناگون بررسی کنند.

در واقع، داشتن یک برنامه‌ی باگ‌بانتی مؤثر، به معنای این است که قراردادهای هوشمند شما توسط صدھا یا هزاران فرد خبره به صورت دقیق بررسی می‌شوند.

1.4 چگونه یک برنامه باگ‌بانتی موفق اجرا کنیم؟

1 انتخاب سازوکار مشوق مناسب

نخستین اصل حیاتی در اجرای باگ‌بانتی، اندازه‌ی مناسب پاداش‌ها (Incentives) است. بسیاری از پژوهشگران امنیتی، باگ‌بانتی را به عنوان شغل اصلی خود انتخاب کرده‌اند. اگر پاداش‌ها کم باشند، احتمال اینکه زمان ارزشمند خود را صرف بررسی پروژه‌ای کنند، بسیار پایین است.

با ارائه‌ی پاداش‌های جذاب و مناسب با میزان خطر آسیب‌پذیری‌ها، می‌توان متخصصان را تشویق کرد تا وقت و انرژی کافی صرف بررسی پروژه کنند.



برای مثال:

- اگر آسیب‌پذیری منجر به سرقت احتمالی میلیون‌ها دلار شود، پاداش کشف آن باید به اندازه‌ی ریسک آن تهدید باشد، فارغ از اینکه روش کشف آن چگونه بوده است.
- می‌توان پاداش‌ها را بر اساس سطح تهدید تعریف کرد؛ مانند:
 - Low: \$250
 - Medium: \$1,000
 - High: \$10,000
 - Critical: \$100,000+

2 تعریف محدوده (Scope) دقیق

برای جلوگیری از گزارش‌های بی‌ربط، باید محدوده دقیق برنامه با گروه‌بندی مشخص شود:

مثال:

- فقط قراردادهای هوشمند منتشرشده در شبکه اصلی (Mainnet) شامل برنامه باشند.
- آدرس‌های دقیق قراردادها مشخص شوند.
- کدهای فرانت‌اند، بک‌اند یا API‌ها نیز در صورت تمایل با لینک گیت‌هاب اضافه شوند.
- مواردی که خارج از محدوده (Out of Scope) هستند، نیز به‌وضوح اعلام شوند؛ مثل کد تستی یا حملات DoS



3 استفاده از پلتفرم‌های معتبر باگ‌بانانی

به جای اجرای برنامه باگ‌بانانی در وبسایت خودتان، بهتر است از پلتفرم‌های معتبر و محبوب بین متخصصان استفاده شود:

پلتفرم تخصصی برای پروژه‌های **Web3** و قراردادهای هوشمند **Immunefi**

نمونه دیگر پلتفرم‌ها:

پرکاربرد در پروژه‌های امنیتی گسترده **HackerOne** •

Bugcrowd •

HackenProof •

4 اجازه افشاء عمومی هماهنگ‌شده (Coordinated Disclosure)

یکی از اصول مهم شفافیت در برنامه‌های باگ‌بانانی، این است که کشف‌کنندگان بتوانند پس از برطرف شدن آسیب‌پذیری، یافته‌های خود را به صورت عمومی منتشر کنند. این کار هم به پروژه اعتبار می‌دهد، هم به کشف‌کننده امتیاز می‌دهد، و هم به سایر توسعه‌دهندگان در آینده کمک می‌کند.



مثال‌های واقعی از باگ‌باتی در Web3

پروژه	میزان پاداش	نوع آسیب‌پذیری		توضیح
		Critical	High	
Polygon (MATIC)	\$2,000,000	Critical		کشف آسیب‌پذیری که می‌توانست موجب سرقت تمام موجودی شود.
Aurora Protocol	\$6,000,000	Critical		آسیب‌پذیری در پل انتقال که مهاجم می‌توانست بدون سرمایه دارایی برداشت کند.
Balancer	\$500,000	High		کشف نقص در محاسبات مربوط به استخراج نقدینگی.
Aave	\$250,000	Medium		کشف نقص در منطق وامدهی که می‌توانست باعث ایجاد شرایط بهره‌برداری شود.



جدول نمونه آسیب‌پذیری‌های Web3 با شناسه CVE

شناسه CVE	سال	پروژه / نرم افزار	سطح خطر (CVSS)	نوع آسیب‌پذیری
CVE-2022-32925	2022	Solidity (Compiler)	9.8 (Critical)	Memory Corruption
CVE-2021-39137	2021	Ethereum Geth	7.5 (High)	DoS (Denial of Service)
CVE-2020-26241	2020	OpenEthereum (Parity)	8.6 (High)	Improper Input Validation
CVE-2021-44228	2021	استفاده شده در Log4j (Web3 بسیاری از Backend ها)	10.0 (Critical)	Remote Code Execution
CVE-2020-26259	2020	MetaMask	6.5 (Medium)	Untrusted Input
CVE-2023-28432	2023	Hyperledger Fabric	7.5 (High)	Authorization Bypass
CVE-2019-19751	2019	Parity Ethereum	9.8 (Critical)	JSON-RPC Exposure



نکات جدول:

- آسیب‌پذیری‌هایی مانند **CVE-2021-44228 (Log4Shell)** نشان می‌دهند که حتی مازول‌هایی که مستقیماً به بلاک‌چین ربط ندارند، می‌توانند به امنیت آسیب‌بزند.
- پروژه‌هایی که قراردادهای هوشمند را توسعه می‌دهند، باید نه فقط کدهای **Solidity**، بلکه تمامی وابستگی‌ها (فرانت‌اند، بک‌اند، زیرساخت، ابزارهای CI/CD) را بررسی کند.
- برخی آسیب‌پذیری‌ها تنها از طریق برنامه‌های باگ‌بانتی کشف شده‌اند و همین مسئله اهمیت اجرای مداوم چنین برنامه‌هایی را نشان می‌دهد.

تداویم باگ‌بانتی در فاز تولید (**Mainnet**)

در فاز تولید (زمانی که قراردادهای هوشمند با پول واقعی کار می‌کنند)، اجرای باگ‌بانتی اهمیت دوچندان پیدا می‌کند.

هیچ برنامه‌ای ۱۰۰٪ ایمن نیست، به‌ویژه در Web3. بنابراین توصیه می‌شود:

- برنامه باگ‌بانتی را پس از لانچ نیز فعال نگه دارید.
- دامنه‌ی باگ‌بانتی را به سایر اجزای پروژه API‌ها، بک‌اند، فرانت‌اند (نیز گسترش دهید).



سیستم نظارت و هشدار (Monitoring / Alerting)

استفاده از سیستم‌هایی مانند **RCSM (Real-time Smart Contract Monitoring)** برای بررسی اجرای قراردادها به صورت لحظه‌ای ضروری است. اگر تراکنشی از آدرس غیرمجاز فرآخوانی شود یا رفتاری مشکوک دیده شود، سیستم هشدار می‌دهد.

نمونه ابزارهای موجود:

Tenderly • Smart Contract Watch •

جمع‌بندی

باگ‌بانتی‌ها یکی از مؤثرترین ابزارهای امنیتی در Web3 هستند که با درگیر کردن جامعه جهانی، سطح امنیت پروژه را بالا می‌برند. یک برنامه‌ی موفق، باید:

- پاداش‌های منطقی و جذاب داشته باشد
- دامنه مشخص و شفاف تعریف کند
- روی پلتفرم‌های معتبر اجرا شود
- به کشف‌کنندگان امکان افشای عمومی بعد از وصله را بدهد
- و در تمام طول عمر پروژه، فعال باقی بماند.

