فصل

۵

کیف‌پول

در بیت‌کوین، واژه‌ی «کیف‌پول» برای توصیف چند چیز مختلف به کار می‌رود.

در بالاترین سطح، کیف‌پول یک برنامه‌ی کاربردی است که به عنوان رابط کاربری اصلی بیت‌کوین عمل می‌کند. از این دیدگاه، کیف‌پول مسئول کنترل دسترسی کاربر به پول خود (سکه‌های بیت‌کوین)، مدیریت جفت کلیدهای خصوصی/عمومی و آدرس‌های بیت‌کوین، پیگیری تراز حساب کاربر، و ایجاد تراکُنش‌های امضاشده است.

در سطح پایین‌تر، از دیدگاه یک برنامه‌نویس، «کیف‌پول» به ساختمان داده‌ای گفته می‌شود که برای ذخیره‌سازی و مدیریت کلیدهای کاربر به کار می‌رود. این *ساختمان داده* معمولاً به صورت یک فایل ساخت‌یافته یا پایگاه داده‌ی ساده پیاده‌سازی می‌شود.

در این فصل کیف‌پول را از دیدگاه دوم (دیدگاه برنامه‌نویسی)، یعنی مخزنی برای جفت کلیدهای خصوصی/عمومی، بررسی خواهیم کرد.

مروری بر فناوری کیف‌پول

در این قسمت به معرفی اجمالی فناوری‌هایی می‌پردازیم که برای ساخت یک کیف‌پول بیت‌کوین کاربرپسند، اَمن، و انعطاف‌پذیر به کار می‌روند.

یک سوءتفاهم بزرگ درباره‌ی بیت‌کوین این است که دارایی بیت‌کوین کاربران در داخل کیف‌پول ذخیره می‌شود. در حقیقت، در یک کیف‌پول بیت‌کوین چیزی جز چند جفت کلید خصوصی/عمومی وجود ندارد، و سکه‌های بیت‌کوین در بلاک‌چین (در شبکه‌ی بیت‌کوین) ثبت می‌شوند. کاربران این سکه‌ها را با امضا کردن تراکُنش‌ها با کلیدهایی که در کیف‌پول خود دارند، در شبکه جابجا و کنترل می‌کنند. به بیان ساده، کیف‌پول بیت‌کوین فقط یک دسته *کلید* است.

توجه

کیف‌پول بیت‌کوین حاوی کلیدهای کاربر است، نه سکه‌های بیت‌کوین او. کیف‌پول در واقع یک دسته کلید است که کاربران جفت کلیدهای خصوصی/عمومی خود را در آن ذخیره می‌کنند (فصل ۴ را ببینید). کاربر تراکُنش‌ها را با کلیدهای خصوصی خود امضا می‌کند تا مالکیت خود را بر خروجی‌ها (سکه‌های بیت‌کوین) اثبات کند. سکه‌های بیت‌کوین به صورت خروجی تراکُنش (که اغلب به آنها vout یا txout گفته می‌شود) در بلاک‌چین ذخیره می‌شوند.

دو نـوع کیف‌پول وجود دارد: کیف‌پول‌هایـی که کلیدهای موجود در آنها با یکدیگـر رابطه دارند [به یکدیگر وابسته هستند]، و کیف‌پول‌هایی که چنین نیستند.

در کیف‌پول نوع اول، موسـوم به **کیف‌پول غیرقطعـی** (nondeterministic wallet)، هر کلید [خصوصی] از یک عدد تصادفی مسـتقل استخراج می‌شود، و کلیدها هیچ ارتباطی با یکدیگر ندارند. این کیف‌پول با عنوان JBOK («فقط یک دسته کلید») نیز شناخته می‌شود.

در کیف‌پول نوع دوم، موسوم به **کیف‌پول قطعی** (deterministic wallet)، تمام کلیدها از یک کلید اصلی واحد که به آن بذر (seed) گفته می‌شود، استخراج می‌شوند. کلیدهای این نوع کیف‌پول به یکدیگر وابسته هستند و با داشتن بذر اولیه می‌توان تمامی آنها را از نو تولید کرد. در کیف‌پول‌های قطعی از روش‌های مختلفی برای **اسـتخراج کلید** (key derivation) استفاده می‌شود؛ رایج‌ترین روش استخراج کلید از یک ساختار درخت- مانند استفاده می‌کند و به آن کیف‌پول **قطعی-سلسله‌مراتبی** (hierarchical deterministic) یا کیف‌پول **HD** گفته می‌شود. کیف‌پول‌های قطعی از یک بذر ریشه می‌گیرند. برای سهولت در خاطرسپاری این بذرها، معمولاً آنها را به صورت کلمات معمولی، موسوم به *کُد یادافزا*، می‌نویسند.

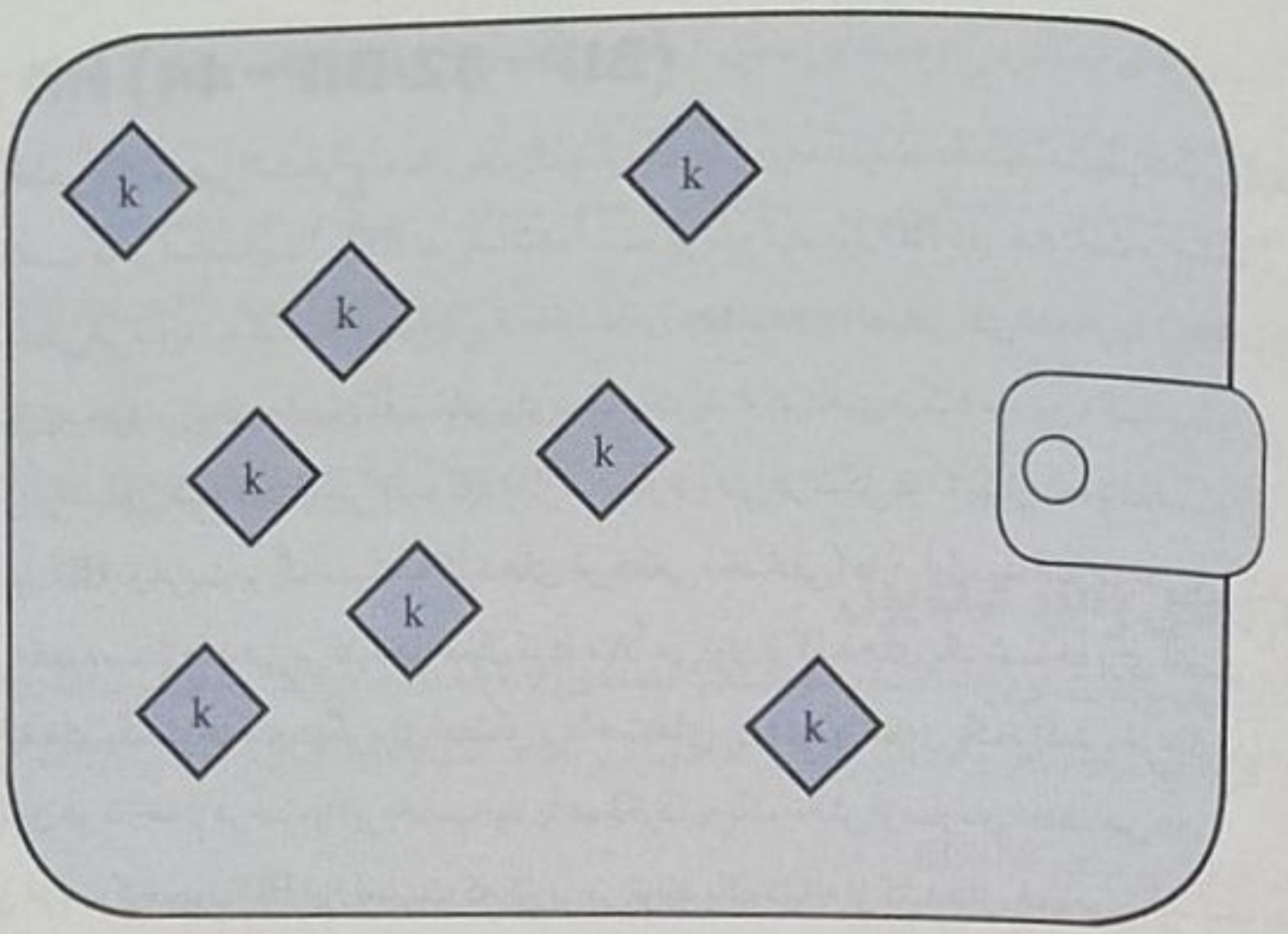
در ادامه هر یک از این فناوری‌ها را با جزئیات بیشتر تشریح می‌کنیم.

کیف‌پول غیرقطعی (تصادفی)

در اولین کیف‌پول بیت‌کوین (که امروزه به آن هسـته‌ی بیت‌کوین گفته می‌شـود) کیف‌پـول مجموعه‌ای از کلیدهای خصوصی تصادفی بود. برای مثال، به محض این که مشتری هسته‌ی بیت‌کوین اولیه اجرا می‌شد، ۱۰۰ کلید خصوصی تصادفی تولید می‌کرد، و اگر در ادامه کلیدهای خصوصی بیشـتری نیاز می‌شـد، آنها را هم به همان صورت تصادفی تولید می‌کرد. این نوع کیف‌پول به خاطر دشـواری مدیریت، انتقال و پشتیبان‌گیری از کلیدها جای خود را به کیف‌پول قطعی داد. عیب اصلی کلیدهای تصادفی این اسـت که باید همیشـه یک کپی از همه‌ی آنها را نگه دارید؛ به عبارت دیگر، باید به طور منظم از کیف‌پول خود نسـخه‌ی پشتیبانی تهیه کنید. همه‌ی کلیدها همیشه باید در دسترس باشند، چون اگر یکی از آنها را گم کنید، بیت‌کوین متناظر با آن را برای همیشـه از دسـت خواهید داد. این ویژگی کلیدهای تصادفی در تضاد مسـتقیم با اصل عدم اسـتفاده‌ی مجدد از یک آدرس [استفاده از هر آدرس مستقل بیت‌کوین فقط برای یک تراکُنش] قرار دارد. همان طور که قبلاً گفتیم، اسـتفاده‌ی مجدد از آدرس‌های بیت‌کوین باعث کاهش امنیت تراکُنش‌ها می‌شـود. به همین دلیل، کیف‌پول غیرقطعی نوع- ۰ گزینه‌ای مناسـب محسوب نمی‌شود، به خصوص اگر بر عدم اسـتفاده‌ی مجدد از آدرس‌های بیت‌کوین اصرار داشته باشـیم، اصلی که رعایت آن مستلزم استفاده از تعداد زیادی کلید خصوصی و پشتیبان‌گیری منظم از تمامی این کلیدها است. هر چند مشتری هسته‌ی بیت‌کوین حاوی یک کیف‌پول [غیرقطعی] نوع- ۰ است، ولی حتی توسعه‌دهندگان هسته‌ی بیت‌کوین هم این کیف‌پول را توصیه نمی‌کنند. در شکل ۵-۱ یک کیف‌پول غیرقطعی با تعدادی کلید خصوصی تصادفی نشان داده شده است.

توجه

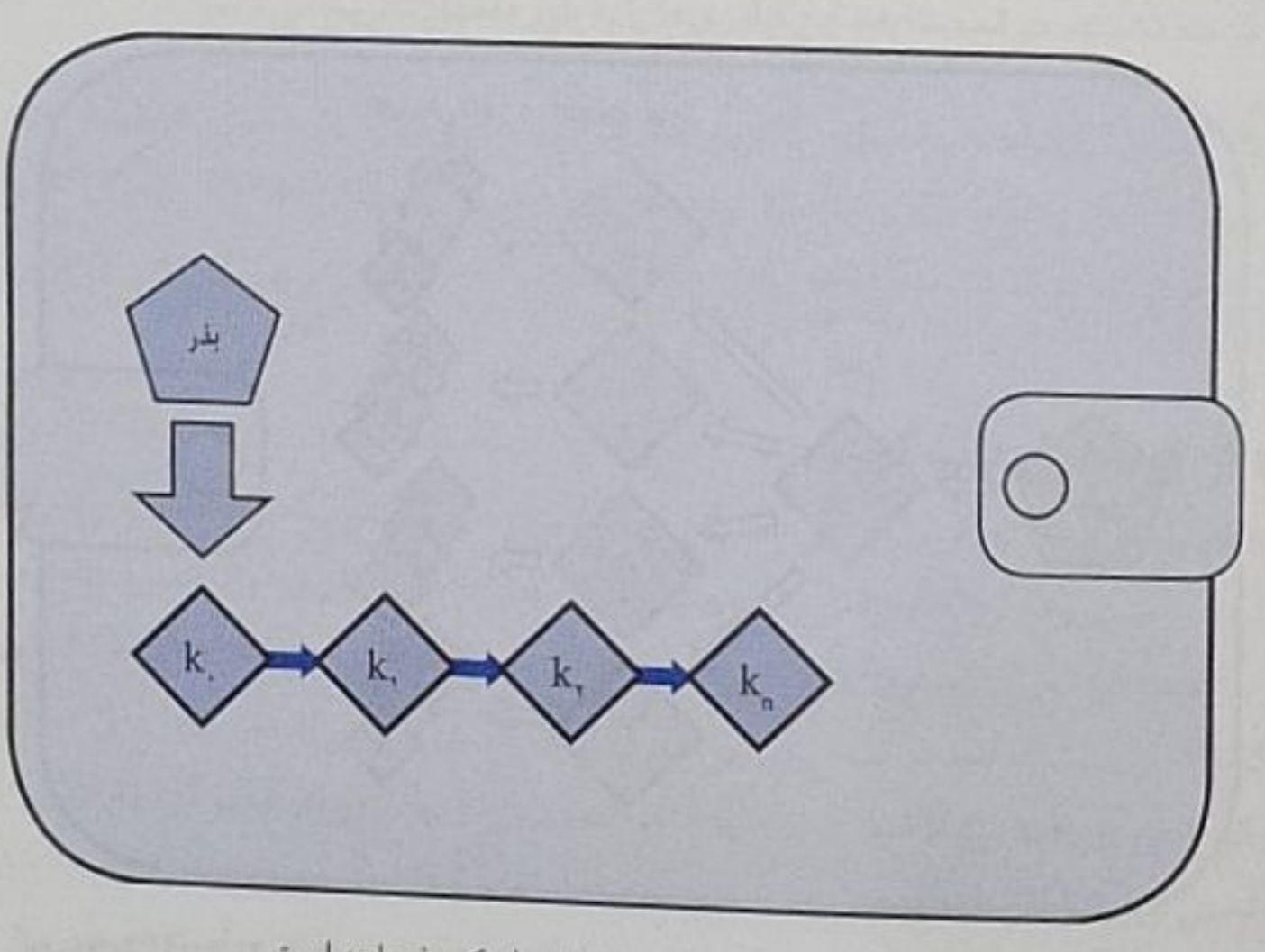
کیف‌پول‌های غیرقطعی جز برای آزمایش‌های ساده مناسب نیستند. چون کار کردن با این کیف‌پول‌ها و تهیه‌ی نسخه‌ی پشتیبانی از آنها سخت و پُرزحمت است. به جای این نوع کیف‌پول از یک **کیف‌پول HD** استاندارد با یک بذر یادافزا برای مقاصد پشتیبان‌گیری استفاده کنید.



شکل ۵-۱ کیف‌پول غیرقطعی (تصادفی) نوع-۰ مجموعه‌ای از کلیدهای خصوصی است.

کیف‌پول قطعی (بذردار)

کیف‌پول قطعی، یا بذردار، حاوی کلیدهایی است که همگی با استفاده از یک تابع دَرهم‌سازی یک‌طرفه از یک نطفه‌ی مشترک (بذر) مشتق شده‌اند. این بذر عددی تصادفی است که با داده‌های دیگر، مانند یک اندیس یا «کُد زنجیره»، ترکیب شده و کلیدهای خصوصی را تولید می‌کند (قسمت بعد را ببینید). در یک کیف‌پول قطعی، فقط با داشتن این بذر می‌توان تمام کلیدهای خصوصی را بازیابی کرد، بنابراین تنها چیزی که لازم است ذخیره شود، همین بذر است. این بذر برای وارد/صادر کردن کل کیف‌پول هم کفایت می‌کند، که در نتیجه باعث تسهیل فرآیند انتقال و جابجایی محتویات یک کیف‌پول بین پیاده‌سازی‌های مختلف خواهد شد. شکل ۵-۲ نمودار منطقی یک کیف‌پول قطعی را نشان می‌دهد.



شکل ۵-۲ کیف‌پول قطعی (بذردار) مجموعه‌ای از کلیدهای مشتق‌شده از یک بذر واحد است.

کیف‌پول HD (BIP-32/BIP-44)

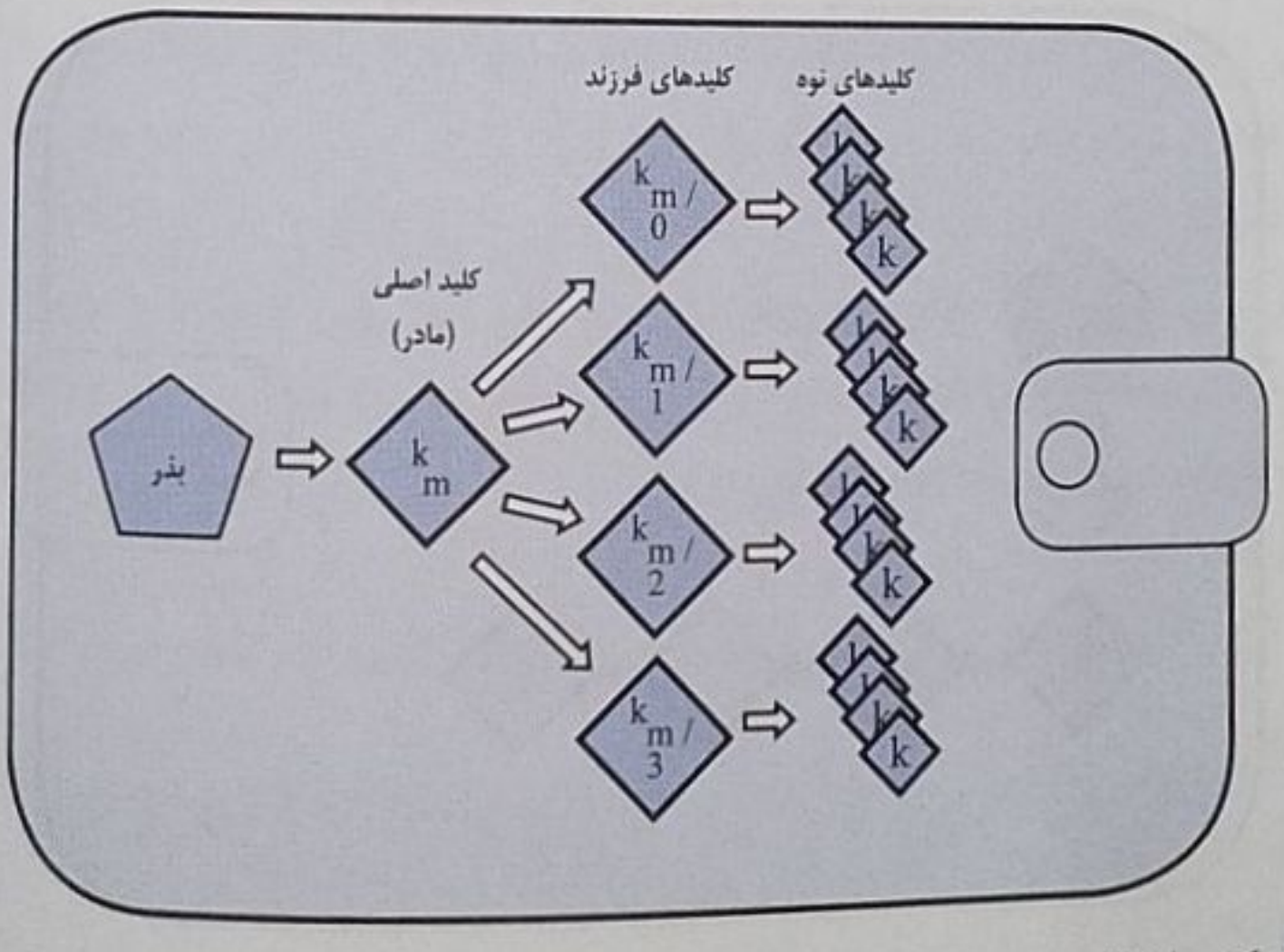
کیف‌پول‌های قطعی برای تسهیل استخراج تعداد زیادی کلید از یک «بذر» واحد توسعه داده شدند. پیشرفته‌ترین کیف‌پول قطعی کیف‌پول HD است که در استاندارد BIP-32 تعریف شده است. در یک کیف‌پول HD کلیدهای استخراج‌شده در یک ساختار درختی سازماندهی می‌شوند، به گونه‌ای که بتوان از یک کلید مادر (parent key) تعداد زیادی کلید فرزند (child key) استخراج کرد؛ هر کلید فرزند خود می‌تواند به عنوان کلید مادر برای تولید یک رشته کلیدهای دیگر (موسوم به کلید نوه) به کار گرفته شود، و این فرآیند می‌تواند بدون هیچ محدودیتی ادامه یابد. این ساختار درختی در شکل ۵-۳ نشان داده شده است.

کیف‌پول HD دو مزیت بزرگ نسبت به کلیدهای غیرقطعی (تصادفی) دارد. اول، ساختار درختی این کیف‌پول اجازه می‌دهد نوعی مفهوم سازماندهی بر کلیدها اعمال کرد، مثلاً می‌توان از کلیدهای یک شاخه برای امضای پرداخت‌های ورودی و از کلیدهای یک شاخه‌های دیگر برای امضای پرداخت‌های ورودی (یا تتمه‌ی یک تراکُنش خرج کردن) استفاده کرد. همچنین می‌توان هر شاخه از درخت را (بر حسب نیاز یا عملکرد) به یک بخش از سازمان اختصاص داد.

دومین مزیت کیف‌پول HD این است که کاربر می‌تواند یک دنباله از کلیدهای عمومی تولید کند، بدون این که نیازی به دسترسی به کلیدهای خصوصی متناظر آنها داشته باشد. این ویژگی به شما اجازه می‌دهد تا بتوانید از کیف‌پول‌های HD در محیط‌های ناامن یا فقط برای دریافت بیت‌کوین استفاده کرده و برای هر تراکُنش یک کلید عمومی متفاوت صادر کنید. در این وضعیت نیازی نیست کلیدهای عمومی از قبل ساخته یا ذخیره شده باشند، در حالی که سرویس‌دهنده همچنان می‌تواند بدون در اختیار داشتن کلیدهای خصوصی به خرج کردن بیت‌کوین ادامه دهد.

بذر و کُد یادافزا (BIP-39)

کیف‌پول HD ساز و کاری بسیار قدرتمند برای مدیریت کلیدها و آدرس‌های بیت‌کوین است. این ابزار در ترکیب با یک روش استانداردسازی‌شده برای تولید بذر از کلمات و عبارات معمولی (که مدیریت و جابجایی کیف‌پول را از قبل هم ساده‌تر می‌کند) قدرت بیشتری می‌یابد. این روش که به نام *کُد یادافزا* شناخته می‌شود، در استاندارد BIP-32 تعریف شده است.



شکل ۵-۳ کیف‌پول HD نوع-۲ درختی است متشکل از کلیدهای تولیدشده از یک بذر واحد.

امروزه، اکثر کیف‌پول‌های بیت‌کوین (و همچنین سایر ارزهای رمزبنیان) از این استاندارد استفاده می‌کنند و می‌توان آنها را به سادگی بین پیاده‌سازی‌های مختلف جابجا کرد، یا بذر آنها را بازیابی کرده و از آن نسخه‌ی پشتیبان تهیه کرد. با یک نگاه به بذرهای زیر می‌توان تشخیص داد که نگهداری، جابجایی و به خاطر سپردن کدام یک از آنها ساده‌تر است:

```
0C1E24E5917779D297E14D45F14E1A1A

army van defense carry jealous true
garbage claim echo media make crunch
```

استانداردهای رایج کیف‌پول

با تکامل فناوری کیف‌پول بیت‌کوین، استانداردهای صنعتی متعددی وضع شد تا کیف‌پول‌های مختلف بتوانند با یکدیگر تعامل داشته باشند، کار با آنها ساده باشد، و همچنین اَمن و انعطاف‌پذیر باشند. استانداردهای رایج کیف‌پول بیت‌کوین عبارتند از:

- کُد یادافزا، بر مبنای BIP-39
- کیف‌پول HD، بر مبنای BIP-32
- ساختار کیف‌پول HD چندمنظوره، بر مبنای BIP-43
- کیف‌پول چندارزی و چندحسابی، بر مبنای BIP-44

احتمال دارد این استانداردها در آینده تغییر کنند یا منسوخ شوند، ولی در حال حاضر آنها فناوری‌های اصلی کیف‌پول بیت‌کوین را تشکیل می‌دهند. طیف وسیعی از تولیدکنندگان کیف‌پول نرم‌افزاری/سخت‌افزار بیت‌کوین این استانداردها را پذیرفته‌اند تا بتوانند کیف‌پول‌های انعطاف‌پذیری تولید کنند که در محیط‌های مختلف قابل استفاده باشند. کاربر می‌تواند فقط با استفاده از کُد یادافزای تولیدشده در یک کیف‌پول، تمامی کلیدها، آدرس‌ها و تراکُنش‌های خود را در کیف‌پول‌های دیگر بازسازی کند. از میان کیف‌پول‌های نرم‌افزاری که از این استانداردها پشتیبانی می‌کنند، می‌توان به مالتی‌بیت HD، بردوالت و مایسلیوم اشاره کرد؛ کیف‌پول‌های سخت‌افزاری کیپ‌کی، لِجر و تریزور نیز بر اساس این استانداردها ساخته شده‌اند. در قسمت بعد این فناوری‌ها را به طور مفصل تشریح می‌کنیم.

توجه

اگر خودتان می‌خواهید کیف‌پول بیت‌کوین بسازید، آن را به صورت یک کیف‌پول HD طراحی کنید که از یک بذر یادافزا برای پشتیبان‌گیری استفاده کرده و استانداردهای صنعتی BIP-32، BIP-39، BIP-43 و BIP-44 را به صورتی که در قسمت بعد خواهید دید، پیاده‌سازی می‌کند.

استفاده از کیف‌پول بیت‌کوین

در فصل ۱ با گابریل، نوجوان کارآفرین برزیلی ساکن ریودوژانیرو، آشنا شدیم که یک فروشگاه اینترنتی برای فروش لباس، پوستر، برچسب و دیگر اقلام منقش به آرم بیت‌کوین را اداره می‌کند. گابریل برای مدیریت اَمن بیت‌کوین‌های خود از *تریزور* (Trezor) که یک کیف‌پول بیت‌کوین سخت‌افزاری است، استفاده می‌کند (شکل ۵-۴ را ببینید). تریزور یک دستگاه USB است که فقط دو دکمه دارد، و تمامی استانداردهای صنعتی گفته‌شده در این فصل را پیاده‌سازی می‌کند. یکی از این دو دکمه برای ذخیره کردن کلیدهای خصوصی، و دیگری برای امضا کردن تراکُنش‌ها است. [به تازگی مدل لمسی تریزور که دکمه ندارد، نیز به بازار عرضه شده است.] از آنجا که تریزور بر مبنای استانداردهای صنعتی کیف‌پول بیت‌کوین ساخته شده، به هیچ فناوری اختصاصی یا یک تولیدکننده‌ی خاص وابسته نیست.

شکل ۵-۴ دستگاه تریزور: یک کیف‌پول HD سخت‌افزاری.

وقتی برای اولین بار از کیف‌پول تریزور استفاده می‌کنید، این دستگاه با استفاده از مولد عدد تصادفی سخت‌افزاری داخلی خود یک کُد یادافزا و بذر برای شما تولید کرده و کلمات آن را یکی یکی به شما نشان می‌دهد (شکل ۵-۵ را ببینید).

گابریل تمامی واژه‌های کُد یادافزایی که دستگاه تریزور برای وی تولید کرده، را روی یک تکه کاغذ ثبت کرده است (جدول ۵-۱ را ببینید)، تا بتواند در صورت لزوم (خرابی یا گم شدن دستگاه تریزور) کلیدها، آدرس‌ها و تراکُنش‌های خود را بازیابی کند. با این کُدهای یادافزا، گابریل می‌تواند کلیدها، آدرس‌ها و تراکُنش‌های خود را به هر کیف‌پول (سخت‌افزاری یا نرم‌افزاری) دیگری که بر مبنای این استانداردها ساخته شده باشد، منتقل کند. توجه کنید که ترتیب واژه‌های کُد یادافزا بسیار مهم است، به همین دلیل گابریل در کنار هر واژه مکان آن را نیز ثبت کرده است.

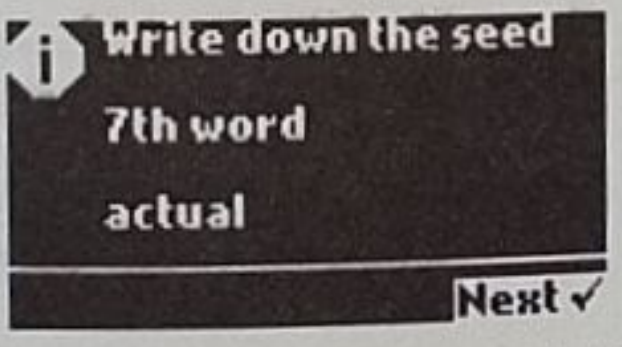
جدول ۵-۱ واژه‌های تشکیل‌دهنده‌ی کُد یادافزا که گابریل آنها را روی کاغذ ثبت کرده است

۱.	army	۷.	garbage
۲.	van	۸.	claim
۳.	defense	۹.	echo
۴.	carry	۱۰.	media
۵.	jealous	۱۱.	make
۶.	true	۱۲.	crunch

نکته

کُد یادافزای ۱۲-کلمه‌ای که در جدول ۵-۱ می‌بینید، یک دنباله‌ی ساده شده است: در واقع، اکثر کیف‌پول‌های سخت‌افزاری دنباله‌های ۲۴-کلمه‌ای تولید می‌کنند. با این حال، طرز کار دنباله‌های یادافزا (صرفنظر از طول آنها) دقیقاً به همان صورتی است که در اینجا شرح داده‌ایم.

در اولین پیاده‌سازی فروشگاه اینترنتی، گابریل از یک آدرس بیت‌کوین واحد (که توسط دستگاه تریزور تولید شده) استفاده می‌کند و همه‌ی مشتریان فروشگاه او پرداخت‌های خود را به همین آدرس انجام می‌دهند. همان طور که خواهید دید، این رویکرد معایبی دارد که می‌توان با به‌کارگیری یک کیف‌پول HD آنها را برطرف کرد.



شکل ۵-۵ نمایش یکی از واژه‌های کُد یادافزا در تریزور.

تشریح فناوری کیف‌پول

در این قسمت به تفصیل درباره‌ی هر یک از استانداردهای صنعتی مهمی که در پیاده‌سازی انواع مختلف کیف‌پول‌های بیت‌کوین به کار می‌روند، صحبت می‌کنیم.

دنباله‌ی کُد یادافزا (BIP-39)

یک *دنباله‌ی کُد یادافزا* رشته‌ای از کلمات متناظر با تعدادی عدد تصادفی است که به عنوان بذر برای استخراج کلیدهای یک کیف‌پول قطعی به کار می‌روند. این دنباله برای بازیابی بذر کیف‌پول، و از آنجا بازیابی تمامی کلیدهای خصوصی موجود در آن، کافی است. یک برنامه‌ی کیف‌پول که از کُد یادافزا برای پیاده‌سازی کیف‌پول قطعی استفاده می‌کند، در اولین قدم یک دنباله‌ی ۱۲ تا ۲۴ کلمه‌ای تولید کرده و به کاربر نشان خواهد داد. این دنباله همان کلماتی هستند که کاربر برای بازیابی یا جابجایی کلیدهای خود به آنها نیاز خواهد داشت. واژه‌های یادافزا فرآیند تهیه‌ی نسخه‌ی پشتیبان از یک کیف‌پول را ساده‌تر می‌کنند، چون خواندن و ثبت کردن آنها در مقایسه با دنباله‌ای از اعداد تصادفی بسیار ساده‌تر است.

توجه

دنباله‌ی کلمات یادافزا را نباید با **کیف‌پول حفظی** (brainwallet) اشتباه گرفت. تفاوت اصلی آنها در این است که در کیف‌پول حفظی واژه‌های یادافزا توسط کاربر انتخاب می‌شوند، در حالی که در یک برنامه‌ی کیف‌پول کلمات یادافزا را خود برنامه به صورت تصادفی تولید می‌کند. این ویژگی امنیت کلمات یادافزا را بسیار بالا می‌برد، چون مغز انسان در تولید دنباله‌های تصادفی بسیار ضعیف عمل می‌کند.

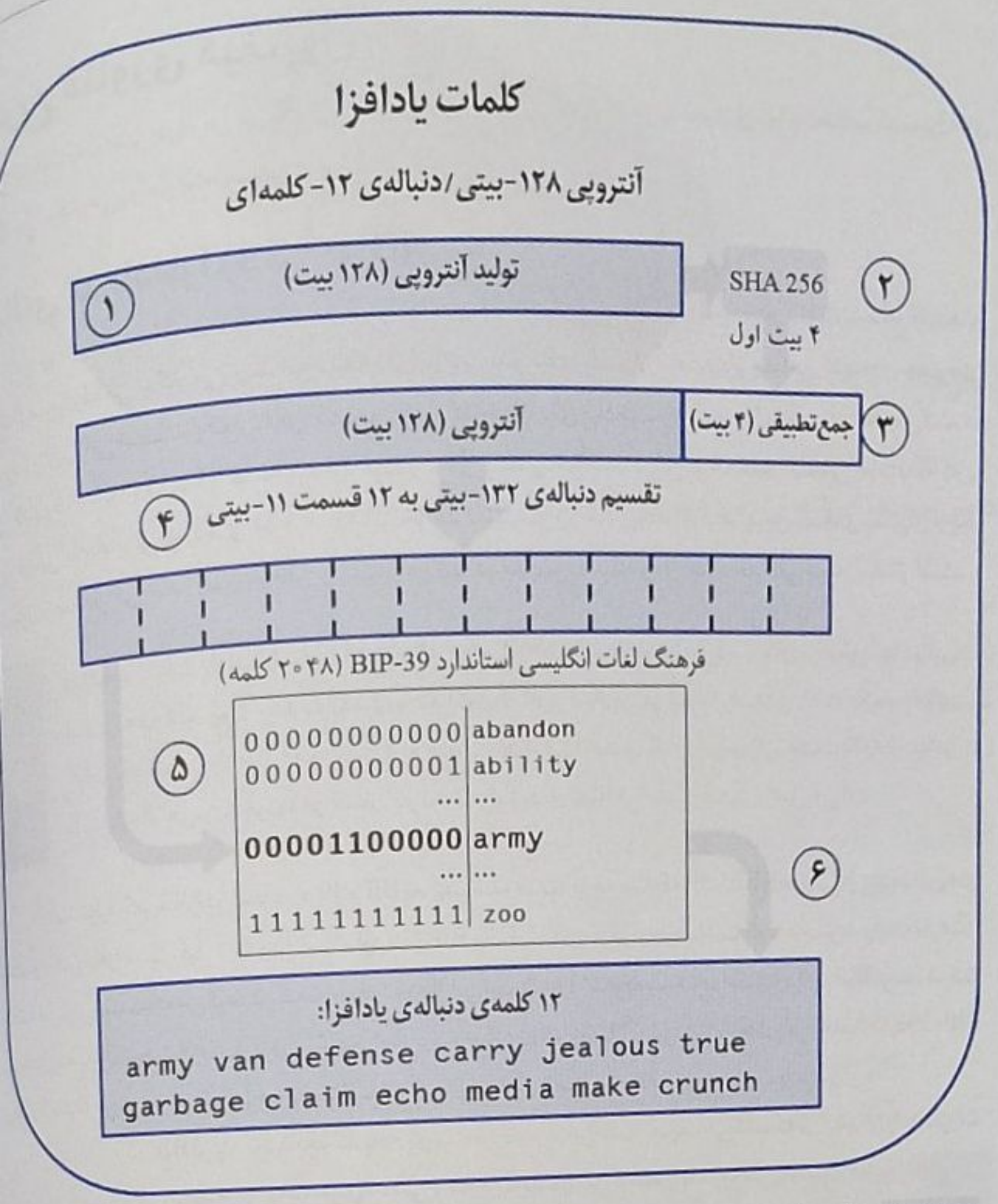
فرآیند تولید کُد یادافزا در استاندارد BIP-39 تعریف شده است. توجه کنید که BIP-39 فقط یکی از پیاده‌سازی‌های استاندارد کُد یادافزا است. قبل از تدوین استاندارد BIP-39 یک استاندارد دیگر وجود داشت که از مجموعه واژه‌ها (فرهنگ لغات) متفاوتی استفاده می‌کرد؛ در کیف‌پول الکتروم از این استاندارد استفاده شده بود. استاندارد BIP-39 توسط شرکت سازنده‌ی کیف‌پول سخت‌افزاری تریزور پیشنهاد شد و با پیاده‌سازی الکتروم سازگار نبود. با این حال، امروزه استاندارد BIP-39 به طور گسترده مورد پذیرش شرکت‌های مختلف قرار گرفته و باید آن را استاندارد غالب این صنعت به شمار آورد.

استاندارد BIP-39 فرآیند تولید یک کُد یادافزا و استخراج بذر از این کُد را تعریف می‌کند، که در اینجا آن را به صورت یک فرآیند ۹-مرحله‌ای توضیح می‌دهیم. برای سادگی و شفافیت بیشتر، این فرآیند را به دو بخش تقسیم کرده‌ایم: بخش اول (مراحل ۱ تا ۶) «تولید واژه‌های دنباله‌ی یادافزا»، و بخش دوم (مراحل ۷ تا ۹) «استخراج بذر از دنباله‌ی یادافزا».

تولید واژه‌های دنباله‌ی یادافزا

برنامه‌ی کیف‌پول واژه‌های دنباله‌ی یادافزا را به صورت خودکار و بر اساس فرآیند تعریف‌شده در استاندارد BIP-39 تولید می‌کند. برنامه‌ی کیف‌پول کار خود را از یک *منبع آنتروپی* شروع می‌کند، سپس یک *جمع تطبیقی* به این عدد تصادفی اضافه کرده، و در آخر از آن به عنوان اندیس برای انتخاب کلمات متناظر در *فرهنگ لغات* خود استفاده می‌کند:

۱. تولید یک دنباله‌ی تصادفی (آنتروپی) به طول ۱۲۸ تا ۲۵۶ بیت.
۲. تولید یک جمع تطبیقی برای این دنباله‌ی تصادفی با انتخاب {طول دنباله تقسیم بر ۳۲} بیت ابتدایی دَرهم SHA256 آن.
۳. اضافه کردن این جمع تطبیقی به انتهای دنباله‌ی تصادفی.
۴. تقسیم این دنباله به قطعات ۱۱-بیتی.
۵. نگاشت هر یک از این مقادیر ۱۱-بیتی به یکی از ۲۰۴۸ واژه‌ی فرهنگ لغات از پیش تعریف‌شده.
۶. کنار هم قرار دادن این واژه‌ها در یک دنباله‌ی یادافزا.



شکل ۵-۶ فرآیند تولید یک عدد تصادفی و تبدیل آن به دنباله‌ی یادافزا.

شکل ۵-۶ چگونگی استفاده از آنتروپی برای تولید کلمات یادافزا را نشان می‌دهد.

در جدول ۵-۲ رابطه‌ی بین طول منبع آنتروپی (عدد تصادفی) و طول دنباله‌ی یادافزا (تعداد کلمات آن) را مشاهده می‌کنید.

جدول ۵-۲ رابطه‌ی طول منبع آنتروپی و تعداد کلمات دنباله‌ی یادافزا

آنتروپی (بیت)	جمع‌تطبیقی (بیت)	آنتروپی + جمع‌تطبیقی (بیت)	طول دنباله‌ی یادافزا (کلمه)
۱۲۸	۴	۱۳۲	۱۲
۱۶۰	۵	۱۶۵	۱۵
۱۹۲	۶	۱۹۸	۱۸
۲۲۴	۷	۲۳۱	۲۱
۲۵۶	۸	۲۶۴	۲۴

استخراج بذر از دنباله‌ی یادافزا

کلمات دنباله‌ی یادافزا از یک آنتروپی با طول ۱۲۸ تا ۲۵۶ بیت تولید می‌شوند. برای استخراج یک بذر که طول آن بیشتر (معمولاً ۵۱۲ بیت) است، باید از تابع کِش‌دهنده‌ی کلید (key-stretching function) PBKDF2 استفاده کنیم. این بذر سپس برای ایجاد یک کیف‌پول قطعی و استخراج کلیدهای آن به کار گرفته می‌شود.

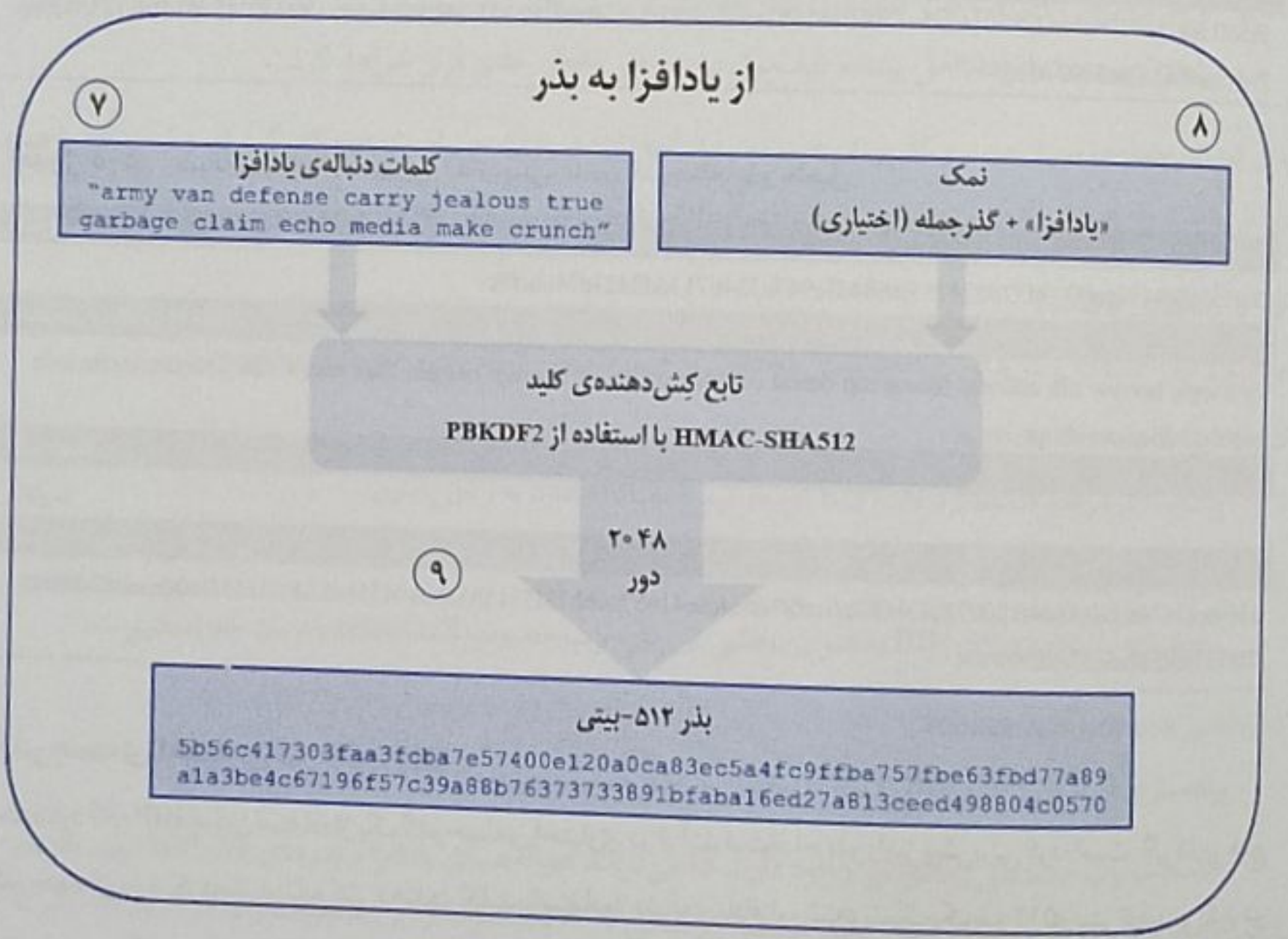
تابع کِش‌دهنده‌ی کلید دو پارامتر می‌گیرد: . دنباله‌ی یادافزا و رشته‌ای موسوم به نمک. هدف از اضافه کردن نمک به تابع کِش‌دهنده‌ی کلید این است که استخراج یک جدول از کلمات محتمل و حملات کوکورانه‌ی مبتنی بر این جدول‌ها را دشوار کند. در استاندارد BIP-39، پارامتر نمک هدف دیگری را هم برآورده می‌کند: امکان وارد کردن یک گذرجمله (passphrase) توسط کاربر برای تقویت حفاظت از بذر (قسمت بعد را ببینید). مراحل استخراج بذر از دنباله‌ی یادافزا را در زیر مشاهده می‌کنید:

۷. پارامتر اول تابع کِش‌دهنده‌ی کلید PBKDF2 همان دنباله‌ی یادافزای تولیدشده در مرحله‌ی ۶ است.
۸. به پارامتر دوم تابع کِش‌دهنده‌ی کلید PBKDF2 «نمک» گفته می‌شود. این نمک از رشته‌ی «یادافزا» به اضافه‌ی یک گذرجمله‌ی اختیاری که از کاربر گرفته می‌شود، تشکیل خواهد شد.
۹. تابع PBKDF2، با ۲۰۴۸ دور اجرای الگوریتم دَرهم‌سازی HMAC-SHA512، دنباله‌ی یادافزا (پارامتر اول) و نمک (پارامتر دوم) را به یک مقدار ۵۱۲-بیتی تبدیل می‌کند. این رشته‌ی ۵۱۲-بیتی همان بذر مورد نظر است.

شکل ۵-۷ چگونگی تولید بذر با استفاده از دنباله‌ی یادافزا را نشان می‌دهد.

توجه

تابع کِش‌دهنده‌ی کلید، به همراه ۲۰۴۸ دور اجرای تابع دَرهم‌سازی، یک ساز و کار بسیار مؤثر در مقابل حملات کوکورانه است. از آنجا که تعداد بذرهایی که با ۵۱۲ بیت می‌توان ساخت، عدد بسیار بزرگ 2^{512} (بیش از 10^{150}) خواهد بود، حتی تصور موفقیت‌آمیز بودن چنین حمله‌ای نیز دشوار است.



شکل ۵-۷ فرآیند استخراج بذر از دنباله‌ی یادافزا.

در جدول‌های ۵-۳ تا ۵-۵ سه نمونه دنباله‌ی یادافزا با طول‌های مختلف و بذر حاصل از آنها را (با و بدون گذرجمله) مشاهده می‌کنید.

جدول ۵-۳ دنباله‌ی یادافزا با آنتروپی ۱۲۸-بیتی، بدون گذرجمله، بذر حاصل

آنتروپی ورودی (۱۲۸ بیت)
0c1e24e5917779d297e14d45f14e1a1a
دنباله‌ی یادافزا (۱۲ کلمه)
army van defense carry jealous true garbage claim echo media make crunch
گذرجمله
(ندارد)
بذر (۵۱۲ بیت)
5b56c417303faa3fcba7e57400e120a0ca83ec5a4fc9ffba757fbe63fbd77a89a1a3be4c67196f57c39a88b76373733891bfaba1 6ed27a813ceed498804c0570

جدول ۵-۴ دنباله‌ی یادافزا با آنتروپی ۱۲۸-بیتی، با گذرجمله، بذر حاصل

آنتروپی ورودی (۱۲۸ بیت)
0c1e24e5917779d297e14d45f14e1a1a
دنباله‌ی یادافزا (۱۲ کلمه)
army van defense carry jealous true garbage claim echo media make crunch
گذرجمله
SuperDuperSecret
بذر (۵۱۲ بیت)
3b5df16df2157104cfdd22830162a5e170c0161653e3afe6c88defeefb0818c793dbb28ab3ab091897d0715861dc8a18358f80b 79d49acf64142ae57037d1d54

جدول ۵-۵ دنباله‌ی یادافزا با آنتروپی ۲۵۶-بیتی، بدون گذرجمله، بذر حاصل

آنتروپی ورودی (۲۵۶ بیت)
2041546864449caff939d32d574753fe684d3c947c3346713dd8423e74abcf8c
دنباله‌ی یادافزا (۲۴ کلمه)
cake apple borrow silk endorse fitness top denial coil riot stay wolf luggage oxygen faint major edit measure invite love trap field dilemma oblige
گذرجمله
(ندارد)
بذر (۵۱۲ بیت)
3269bce2674acbd188d4f120072b13b088a0ecf87c6e4cae41657a0bb78f5315b33b3a04356e53d062e55f1e0deaa082df8d48 7381379df848a6ad7e98798404

گذرجمله‌ی اختیاری در BIP-39

استاندارد BIP-39 امکان استفاده از یک گذرجمله‌ی اختیاری در فرآیند استخراج بذر را نیز پیش‌بینی کرده است. اگر کاربر هیچ گذرجمله‌ای وارد نکرده باشد، تابع کِش دهنده‌ی کلید با استفاده از دنباله‌ی یادافزا و رشته‌ی نمک یک بذر ۵۱۲-بیتی تولید می‌کند. در صورت وجود گذرجمله، تابع کِش دهنده‌ی کلید با همان دنباله‌ی یادافزا بذری متفاوت (با همان طول ۵۱۲ بیت) تولید خواهد کرد.

در حقیقت، اگر یک دنباله‌ی یادافزای واحد داشته باشید، با آن می‌توانید تمامی بذرهای ممکن را با دادن گذرجمله‌های متفاوت تولید کنید. به عبارت دیگر، هیچ گذرجمله‌ای «غلط» نیست؛ تمام گذرجمله‌ها معتبر هستند و یک بذر معتبر (ولی متفاوت) تولید می‌کنند که به یک کیف‌پول منتهی می‌شود. با این حال، تعداد این کیف‌پول‌ها چنان زیاد است (2^{512})، که احتمال دسترسی تصادفی به یک کیف‌پول خاص یا از طریق حدس زدن گذرجمله‌ی آن [کاری که در حملات کورکورانه انجام می‌شود] تقریباً صفر است.

توجه

در استاندارد BIP-39 چیزی به نام گذرجمله‌ی «غلط» وجود ندارد. هر گذرجمله‌ای به یک کیف‌پول منتهی می‌شود، که البته در اکثر قریب به اتفاق موارد خالی است.

این گذرجمله‌ی اختیاری دو ویژگی مهم ایجاد می‌کند:

- یک عامل ثانویه که فقط در ذهن کاربر وجود دارد، و حتی در صورت لو رفتن دنباله‌ی یادافزا، کیف‌پول را محافظت می‌کند.
- نوعی عامل فریبنده که با آن می‌توان نفوذگران را به یک کیف‌پول کوچک، موسوم به **کیف‌پول طعمه** (duress wallet)، هدایت کرد تا کیف‌پول اصلی از خطر حمله در امان بماند.

با این حال، باید یادآوری کنیم که استفاده از گذرجمله خطر از دست رفتن کیف‌پول را هم افزایش می‌دهد:

- اگر صاحب کیف‌پول فوت کند و هیچ کس گذرجمله‌ی انتخابی وی را نداند، حتی با داشتن بذر نیز نمی‌توان محتویات کیف‌پول را بازیابی کرد و محتویات آن کیف‌پول برای همیشه از بین خواهد رفت.
- از طرف دیگر، اگر صاحب کیف‌پول گذرجمله‌ی آن را در جایی بنویسد و این گذرجمله (همراه با دنباله‌ی یادافزا) به دست افراد نااهل بیفتد، کیف‌پول در معرض خطر جدی قرار خواهد گرفت.

با آن که گذرجمله‌ها بسیار مفید و کاربردی هستند، باید آنها را به صورت حساب‌شده به کار گرفت، و این نکته را هم در نظر داشت که بایستی امکان دسترسی به آن توسط افراد خانواده (یا وارثان) صاحب کیف‌پول فراهم شود.

کار با کُدهای یادافزا

استاندارد BIP-39 در زبان‌های برنامه‌نویسی مختلفی به صورت یک کتابخانه پیاده‌سازی شده است:

کتابخانه‌ی python-mnemonic

پیاده‌سازی مرجع استاندارد BIP-39 توسط تیم SatoshiLabs به زبان پایتون.

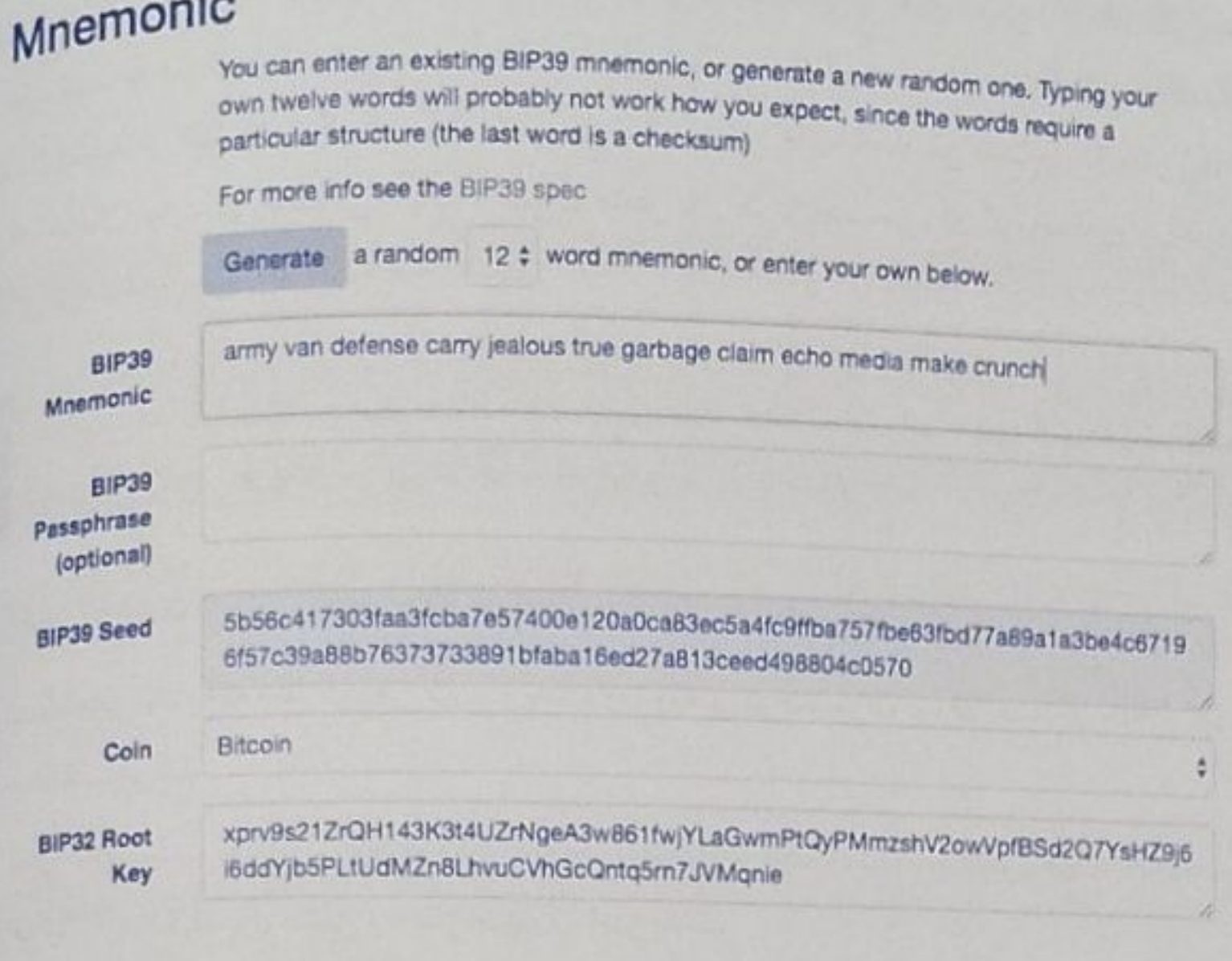
کتابخانه‌ی bitcoinjs/bip39

پیاده‌سازی استاندارد BIP-39 به عنوان بخشی از چارچوب محبوب bitcoinJS به زبان جاواسکریپت.

کتابخانه‌ی libbitcoin/mnemonic

پیاده‌سازی استاندارد BIP-39 به عنوان بخشی از چارچوب محبوب Libbitcoin به زبان C++.

صفحات وب مستقل زیادی نیز وجود دارند که می‌توانند دنباله‌های یادافزا و بذرهای BIP-39 تولید کنند و برای مقاصد تجربی و آزمایشی بسیار مفید هستند. در شکل ۵-۸ یک نمونه از این صفحات وب مولد یادافزا-بذر به آدرس https://iancoleman.github.io/bip39/ را مشاهده می‌کنید.



شکل ۵-۸ یک صفحه‌ی وب مولد BIP-39.

ایجاد کیف‌پول HD از یک بذر

کیف‌پول HD از یک بذر واحد موسوم به بذر ریشه (root seed)، که یک عدد تصادفی ۱۲۸، ۲۵۶ یا ۵۱۲ بیتی است، ساخته می‌شود. این بذر معمولاً (به روشی که در قسمت قبل تشریح کردیم) از یک *دنباله‌ی یادافزا* استخراج می‌شود.

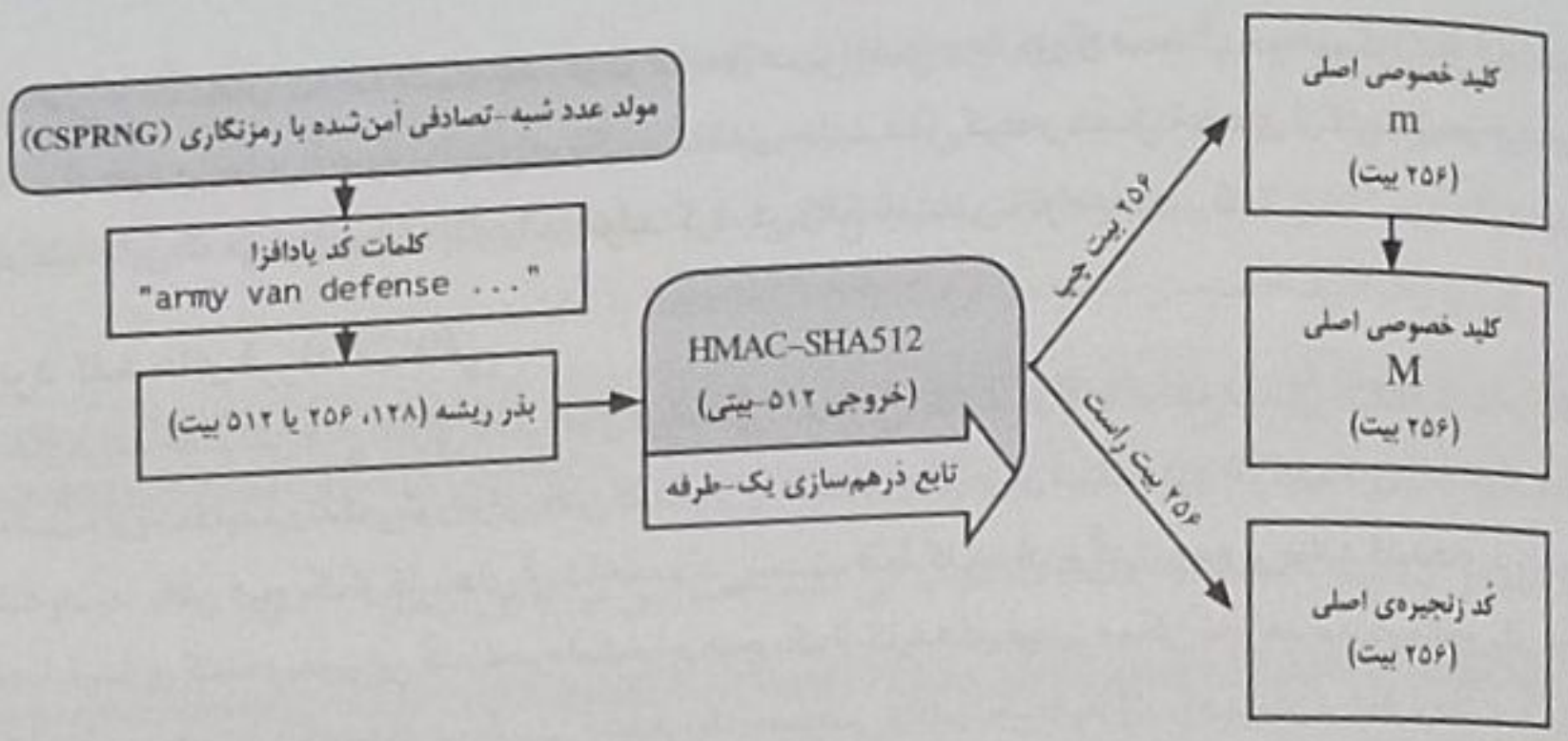
تمامی کلیدهای یک کیف‌پول HD به طور قطعی از این بذر ریشه مشتق می‌شوند، در نتیجه امکان بازیابی کل کیف‌پول (در هر برنامه‌ای که از این استاندارد پیروی کرده باشد) وجود دارد. با این روش می‌توان یک کیف‌پول HD را (حتی اگر حاوی هزاران یا میلیون‌ها کلید باشد) به آسانی و فقط با استفاده از دنباله‌ی یادافزا بازسازی کرده و آن به هر جایی که لازم است، منتقل کرد. فرآیند ایجاد کلیدهای اصلی و کُد زنجیره‌ی اصلی یک کیف‌پول HD در شکل ۵-۹ نشان داده شده است.

در این فرآیند، بذر ریشه به الگوریتم HMAC-SHA512 داده شده و از خروجی دَرهم این الگوریتم یک کلید *خصوصی اصلی* (m) و یک *کُد زنجیره‌ی اصلی* (c) استخراج می‌شود. از کلید خصوصی اصلی (m) و به کمک ضرب منحنی بیضوی [که در فصل ۳ دیدیم] یک کلید خصوصی اصلی دیگر (M) نیز تولید خواهد شد. کاربرد اصلی کُد زنجیره (c) وارد کردن آنتروپی به تابع تولید کلیدهای فرزند از کلید مادر (که در قسمت بعد توضیح خواهیم داد) است.

استخراج کلید فرزند خصوصی

کیف‌پول HD از یک تابع اشتقاق کلید فرزند (child key derivation: CKD) برای استخراج کلیدهای فرزند از کلید مادر استفاده می‌کند. توابع CKD یک تابع دَرهم‌سازی یک-طرفه را برای ترکیب سه پارامتر به کار می‌گیرند:

- یک کلید خصوصی یا عمومی مادر (کلید غیرفشرده‌ی ECDSA)
- یک بذر موسوم به کُد زنجیره (۲۵۶ بیت)
- یک شماره‌ی اندیس (۳۲ بیت)

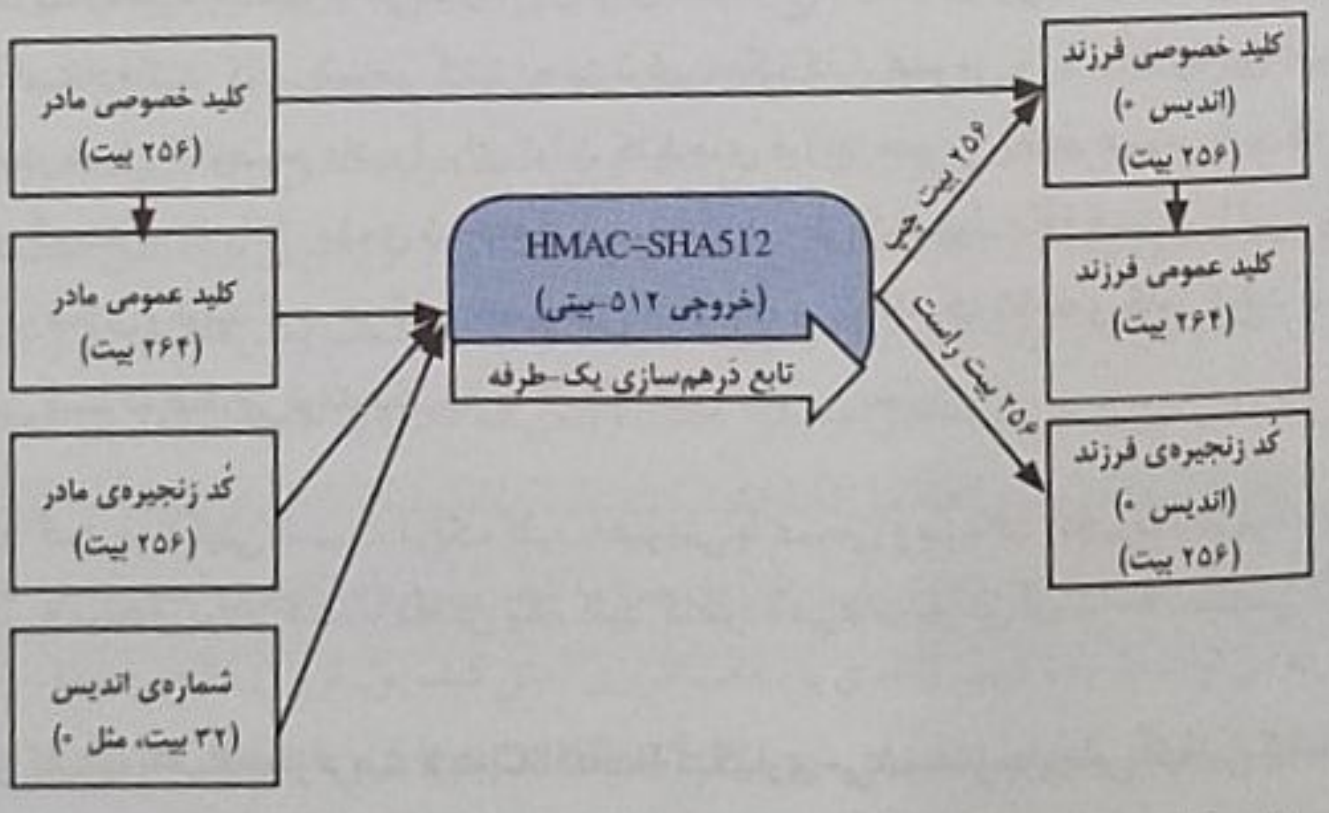


شکل ۵-۹ ایجاد کلیدهای اصلی و کُد زنجیره‌ی اصلی از بذر ریشه.

نقش کُد زنجیره وارد کردن آنتروپی به فرآیند تولید کلید فرزند است، به طوری که دانستن اندیس و کلید فرزند به تنهایی برای استخراج سایر کلیدهای فرزند کافی نباشند. به عبارت دیگر، کسی نباید بتواند با دانستن یک کلید فرزند سایر کلیدهای خواهر آن را استخراج کند، مگر این که کُد زنجیره را هم در اختیار داشته باشد. بذر کُد زنجیره‌ی اولیه (در ریشه‌ی درخت کلیدها) از بذر ریشه ساخته می‌شود، در حالی که کُدهای زنجیره‌ی بعدی از کُد زنجیره‌ی مادر همان شاخه تولید خواهند شد. این سه پارامتر (کلید مادر، کُد زنجیره، اندیس) به طریق ذیل با یکدیگر ترکیب و دَرهم‌سازی شده و کلیدهای فرزند را می‌سازند.

کلید عمومی مادر، کُد زنجیره و شماره‌ی اندیس به وسیله‌ی الگوریتم HMAC-SHA512 ترکیب و دَرهم‌سازی می‌شوند و یک رشته‌ی دَرهم ۵۱۲-بیتی تولید می‌کنند. این دَرهم ۵۱۲-بیتی به دو نیمه‌ی ۲۵۶-بیتی تقسیم می‌شود. نیمه‌ی سمت راست نقش کُد زنجیره را برای کلیدهای فرزند آن شاخه بازی خواهد کرد، و نیمه‌ی سمت چپ نیز بعد از اضافه شدن به کلید خصوصی مادر برای تولید کلیدهای خصوصی فرزند به کار گرفته خواهد شد. شکل ۵-۱۰ چگونگی تولید اولین شاخه از کلیدهای فرزند (اندیس ۰، شاخه‌ی «صفر») را نشان می‌دهد.

با تغییر دادن اندیس می‌توان شاخه‌های متعددی از کلیدهای فرزند تولید کرد (شاخه‌ی ۱، شاخه‌ی ۲، و غیره). هر کلید مادر می‌تواند 2^{31} ($2,147,483,647$) فرزند داشته باشد (2^{31} نصف بازه‌ی 2^{32} است، چون نیمه‌ی دیگر این بازه برای



شکل ۵-۱۰ پردازش کلید خصوصی مادر برای تولید یک شاخه از کلیدهای خصوصی فرزند.

نوع خاصی از اشتقاق کنار گذاشته شده که در ادامه‌ی همین فصل درباره‌ی آن صحبت خواهیم کرد). از آنجا که هر کلید فرزند خود می‌تواند به عنوان کلید مادر یک شاخه‌ی جدید عمل کرده و نسل جدیدی از کلیدها به وجود آورد، تعداد کلیدهایی که می‌توان از یک بذر واحد تولید کرد، در واقع نامتناهی خواهد بود.

کاربرد کلیدهای فرزند اشتقاقی

کلیدهای فرزند خصوصی هیچ تفاوتی با کلیدهای غیرقطعی (تصادفی) ندارند. از آنجا که تابع اشتقاق کلید فرزند تابعی یک‌طرفه است، از یک کلید فرزند نمی‌توان برای یافتن کلید مادر یا کلیدهای خواهر آن استفاده کرد. اگر کلید فرزند n ام یک شاخه را داشته باشید، یافتن هیچ یک از کلیدهای آن شاخه ممکن نیست. فقط کلید مادر و کُد زنجیره می‌توانند کلیدهای فرزند یک شاخه را بازسازی کنند، و بدون این کُد زنجیره استخراج هیچ یک از کلیدهای نوه نیز ممکن نخواهد بود. در واقع، برای ایجاد یک شاخه‌ی جدید و تولید کلیدهای نوه بایستی کلید فرزند خصوصی و کُد زنجیره‌ی فرزند را همزمان در اختیار داشته باشید.

اما این کلیدهای فرزند چه کاربردی دارند؟ از یک کلید فرزند خصوصی می‌توان (درست مثل هر کلید خصوصی دیگر) برای تولید کلید عمومی و آدرس بیت‌کوین، و امضا کردن تراکُنش‌ها با این آدرس استفاده کرد.

یک کلید فرزند خصوصی، و کلید عمومی و آدرس بیت‌کوین استخراج شده از آن، هیچ تفاوتی با کلیدها و آدرس‌های تصادفی ندارند. دنیای خارج از کیف‌پول HD هیچ راهی برای تشخیص این که یک کلید به صورت کاملاً تصادفی تولید شده یا خروجی یک تابع استخراج کلید است، ندارد. همین که یک کلید فرزند تولید شد، عملکرد آن دقیقاً مشابه کلیدهای «معمولی» خواهد بود.

کلید گسترده

همان طور که قبلاً دیدیم، تابع استخراج کلید می‌تواند برای تولید کلیدهای فرزند در هر سطحی از درخت کلید به کار رود. این تابع سه پارامتر می‌گیرد: یک کلید مادر، یک کُد زنجیره، و شماره‌ی اندیس شاخه‌ی فرزند مطلوب. کلید مادر و کُد زنجیره پارامترهای اساسی این تابع هستند، که مجموع آن دو با نام کلید گسترده (extended key) خوانده می‌شود. «کلید گسترده» را می‌توان «کلید گسترش‌پذیر» نیز تصور کرد، چون این نوع کلید را می‌توان برای استخراج کلیدهای فرزند به کار گرفت.

کلید گسترده چیزی نیست جز یک رشته‌ی ۵۱۲-بیتی که از به هم چسباندن کلید ۲۵۶-بیتی و کُد زنجیره‌ی ۲۵۶-بیتی به دست می‌آید. دو نوع کلید گسترده وجود دارد. *کلید خصوصی گسترده* ترکیبی از یک کلید خصوصی و یک کُد زنجیره است که می‌توان از آن برای استخراج کلیدهای فرزند خصوصی (و از آنجا کلیدهای فرزند عمومی) استفاده کرد. *کلید عمومی گسترده* نیز ترکیب یک کلید عمومی و یک کُد زنجیره است که می‌تواند [به روشی که در فصل ۳ توضیح دادیم] برای تولید کلیدهای فرزند عمومی (*فقط عمومی*) به کار گرفته شود.

کلیدهای گسترده در واقع ریشه‌ی شاخه‌های ساختار درختی کیف‌پول HD هستند. با این ریشه می‌توان کل شاخه را از نو بازسازی و تولید کرد. هر کلید خصوصی گسترده می‌تواند یک شاخه‌ی کامل تولید کند، در حالی که یک کلید عمومی گسترده *فقط* می‌تواند یک شاخه از کلیدهای عمومی بسازد.

کلید گسترده ترکیبی است از یک کلید خصوصی یا عمومی و یک کُد زنجیره، و می‌تواند یک شاخه‌ی کامل و مستقل تولید کند. با داشتن یک کلید گسترده می‌توان به کل آن شاخه دسترسی داشت.

کلیدهای گسترده با استفاده از فرمت Base58Check کُدگذاری می‌شوند تا جابجایی آنها بین کیف‌پول‌های سازگار با BIP-32 به سادگی میسر باشد. الگوریتم Base58Check به کار رفته برای کُدگذاری کلیدهای گسترده از یک «شماره‌ی

ویرایش» خاص استفاده می‌کند، به طوری که رشته‌های حاصل به ترتیب دارای پیشوندهای «xprv» و «xpub» برای کلیدهای خصوصی گسترده و کلیدهای عمومی گسترده‌ی باشند و تشخیص آنها به سادگی ممکن باشد. از آنجا که کلیدهای گسترده ۵۱۲ یا ۵۱۳ بیت طول دارند، بسیار طولانی‌تر از رشته‌های Base58Check هستند که قبلاً دیده‌ایم.

در زیر یک کلید خصوصی گسترده با فرمت Base58Check را مشاهده می‌کنید:

```
xprv9tyUQV64JT5qs3RSTJkXCWKMyUgoQp7F3hA1xzG6ZGu6u6Q9VMNjGr67Lctvy5P8oyaYAL9CAWrUE
9i6GoNMKUga5biW6Hx4tws2six3b9c
```

کلید عمومی گسترده‌ی Base58Check متناظر با این کلید خصوصی نیز چنین است:

```
xpub67xpozcx8pe95XVuZLHXZeG6XWXHpGq6Qv5cmNfi7cS5mtjJ2tgypeQbBs2UAR6KECeeMVKZBPLrt
JunSDMstweyLXhRgPxdp14sk9tJPW9
```

استخراج کلید فرزند عمومی

همان طور که قبلاً گفتیم، یکی از خصوصیات بسیار مفید کیف‌پول HD امکان استخراج کلیدهای فرزند عمومی از یک کلید عمومی مادر *بدون* نیاز به کلید خصوصی متناظر آن است. برای استخراج یک کلید فرزند عمومی دو روش وجود دارد: اشتقاق آن از کلید فرزند خصوصی، یا استخراج مستقیم آن از کلید عمومی مادر. به عبارت دیگر، در یک شاخه از ساختار درختی کیف‌پول HD می‌توان از یک کلید عمومی گسترده برای استخراج تمامی کلیدهای عمومی (و فقط کلیدهای عمومی) استفاده کرد.

با این روش میانبُر می‌توان فقط با استفاده از یک کلید عمومی گسترده سیستمی بسیار اَمن برای توزیع کلیدهای عمومی ایجاد کرد، سیستمی که در آن نیازی به هیچ گونه کلید خصوصی نیست. این سیستم می‌تواند به تعداد نامتناهی کلید عمومی و آدرس بیت‌کوین تولید کرده و (برای دریافت بیت‌کوین) در اختیار دیگران قرار دهد، ولی خود قادر به خرج کردن بیت‌کوین‌های دریافتی نیست، چون کلید خصوصی متناظر با آن کلیدهای عمومی را ندارد. چنین سیستمی را می‌توان با خیال راحت روی یک سرویس‌دهنده‌ی وب ناامَن قرار داد. در همان حال، با کلید خصوصی گسترده‌ی متناظر با این کلید عمومی گسترده می‌توان [روی یک سیستم مستقل و اَمن] کلیدهای خصوصی مورد نیاز برای امضای تراکُنش‌ها و خرج کردن بیت‌کوین‌های دریافتی را استخراج کرد.

یکی از کاربردهای رایج این سیستم قرار دادن کلید عمومی گسترده روی سرویس‌دهنده‌ی میزبان فروشگاه اینترنتی (تجارت الکترونیک) است. این سرویس‌دهنده‌ی وب می‌تواند به کمک تابع اشتقاق کلید عمومی برای هر تراکُنش (تسویه حساب سبد خرید کاربر) یک آدرس بیت‌کوین جدید تولید کند. توجه کنید که هیچ کلید خصوصی روی این سرویس‌دهنده‌ی وب قرار ندارد، بنابراین خطر دزدیده شدن و لو رفتن آن نیز منتفی است. بدون کیف‌پول HD، تنها راه برای انجام این کار تولید هزاران و میلیون‌ها کلید عمومی در یک سرویس‌دهنده‌ی اَمن و سپس بارگذاری آنها در سرویس‌دهنده‌ی میزبان فروشگاه‌های اینترنتی است، روشی که علاوه بر خسته‌کننده و پُرزحمت بودن آن، نیاز به مراقبت دائمی دارد تا مبادا سرویس‌دهنده‌ی تجارت الکترونیک با کمبود کلید روبرو شود.

کاربرد دیگر این سیستم در پیاده‌سازی انباره‌ی سرد یا کیف‌پول سخت‌افزاری است. در این سناریو، کلید خصوصی گسترده می‌تواند در یک کیف کاغذی یا سخت‌افزاری (مثل کیف‌پول تریزور) ذخیره شده، و کلید خصوصی گسترده روی اینترنت (مانند سرویس‌دهنده‌ی میزبان فروشگاه اینترنتی) قرار داده شود. کاربران در هنگام پرداخت، آدرس‌های «دریافت بیت‌کوین» (کلیدهای عمومی) را از این سرویس‌دهنده می‌گیرند، در حالی که کلیدهای

خصوصی در یک جای آمن نگهداری می‌شوند. برای خرج کردن بیت‌کوین، کاربر (صاحب فروشگاه اینترنتی) می‌تواند از این کلید خصوصی گسترده در یک برنامه‌ی امضای بیت‌کوین غیرآنلاین استفاده کرده یا تراکُنش‌های خود را به وسیله‌ی یک کیف‌پول سخت‌افزاری (مانند تریزور) امضای کند. فرآیند استخراج کلیدهای عمومی فرزند از یک کلید عمومی مادر در شکل ۵-۱۱ نشان داده شده است.

استفاده از کلید عمومی گسترده در یک فروشگاه اینترنتی

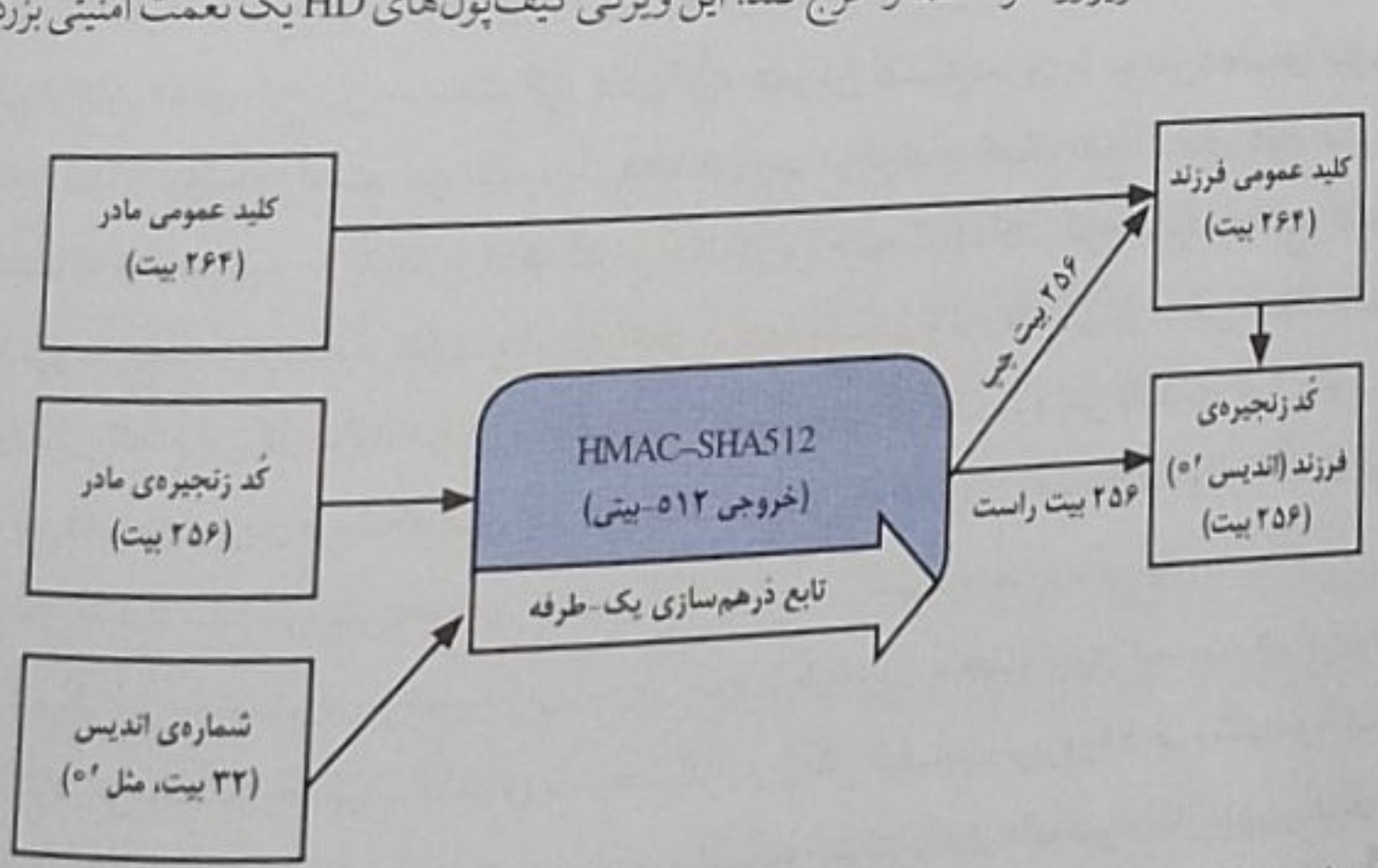
اجازه دهید با ادامه‌ی داستان گابریل (نوجوان کارآفرین برزیلی) ببینیم یک کیف‌پول HD در عمل چگونه کار می‌کند.

این فروشگاه اینترنتی در ابتدا فقط یک سرگرمی کوچک بود، که گابریل آن را بر اساس صفحه‌ی وردپرس خود ساخته بود [وردپرس یکی از محبوب‌ترین ابزارهای ایجاد وبلاگ است]، فروشگاهی بسیار ساده با فقط چند صفحه و یک فرم سفارش با یک آدرس بیت‌کوین واحد. گابریل این آدرس بیت‌کوین را با دستگاه تریزور خود تولید کرده بود. بدین ترتیب، مبلغ تمامی خریدها به این آدرس واحد [که توسط کیف‌پول تریزور کنترل می‌شد] واریز می‌شدند.

وقتی مشتری خریدی در این فروشگاه انجام می‌داد، برای پرداخت بهای آن فرم سفارش را پُر کرده و مبلغ بیت‌کوین را به آدرسی که در همان فرم منتشر شده بود، ارسال می‌کرد. در پاسخ به خرید مشتری، این فرم یک سیگنال حاوی جزئیات خرید مشتری به ایمیل گابریل می‌فرستاد تا فرآیند پردازش (بسته‌بندی کالا و ارسال) آن را شروع کند. در روزهای اول که گابریل فقط چند سفارش در هفته دریافت می‌کرد، این سیستم به خوبی کار می‌کرد.

با این حال، این فروشگاه کوچک بسیار موفق از آب در آمد و مشتریان محلی زیادی را به خود جذب کرد. چندی نگذشت که سر گابریل خیلی شلوغ شد. از آنجا که تمامی پرداخت‌ها به یک آدرس بیت‌کوین انجام می‌شد، برای گابریل بسیار دشوار بود سفارش‌ها را به طور صحیح با تراکُنش‌های مربوطه منطبق کند، به خصوص وقتی در فاصله‌ای کوتاه چند سفارش با مبالغ یکسان دریافت می‌کرد.

راه‌حل این مشکل در همان کیف‌پول HD گابریل نهفته بود: استخراج تعداد زیادی کلید فرزند عمومی بدون نیاز به انتقال کلیدهای خصوصی متناظر آنها به سرویس‌دهنده‌ی وب میزبان فروشگاه اینترنتی. گابریل می‌توانست یک کلید عمومی گسترده (xpub) روی این سرویس‌دهنده قرار دهد، و کار استخراج آدرس‌های غیرتکراری برای هر سفارش را به آن بسپارد. از آنجا که کلیدهای xpub فقط برای دریافت بیت‌کوین کاربرد دارند، اکنون گابریل می‌تواند با خیال راحت بیت‌کوین‌های دریافتی را با استفاده از دستگاه تریزور خود امضا و خرج کند. این ویژگی کیف‌پول‌های HD یک نعمت امنیتی بزرگ است.



شکل ۵-۱۱ استخراج کلیدهای عمومی فرزند از یک کلید عمومی مادر.

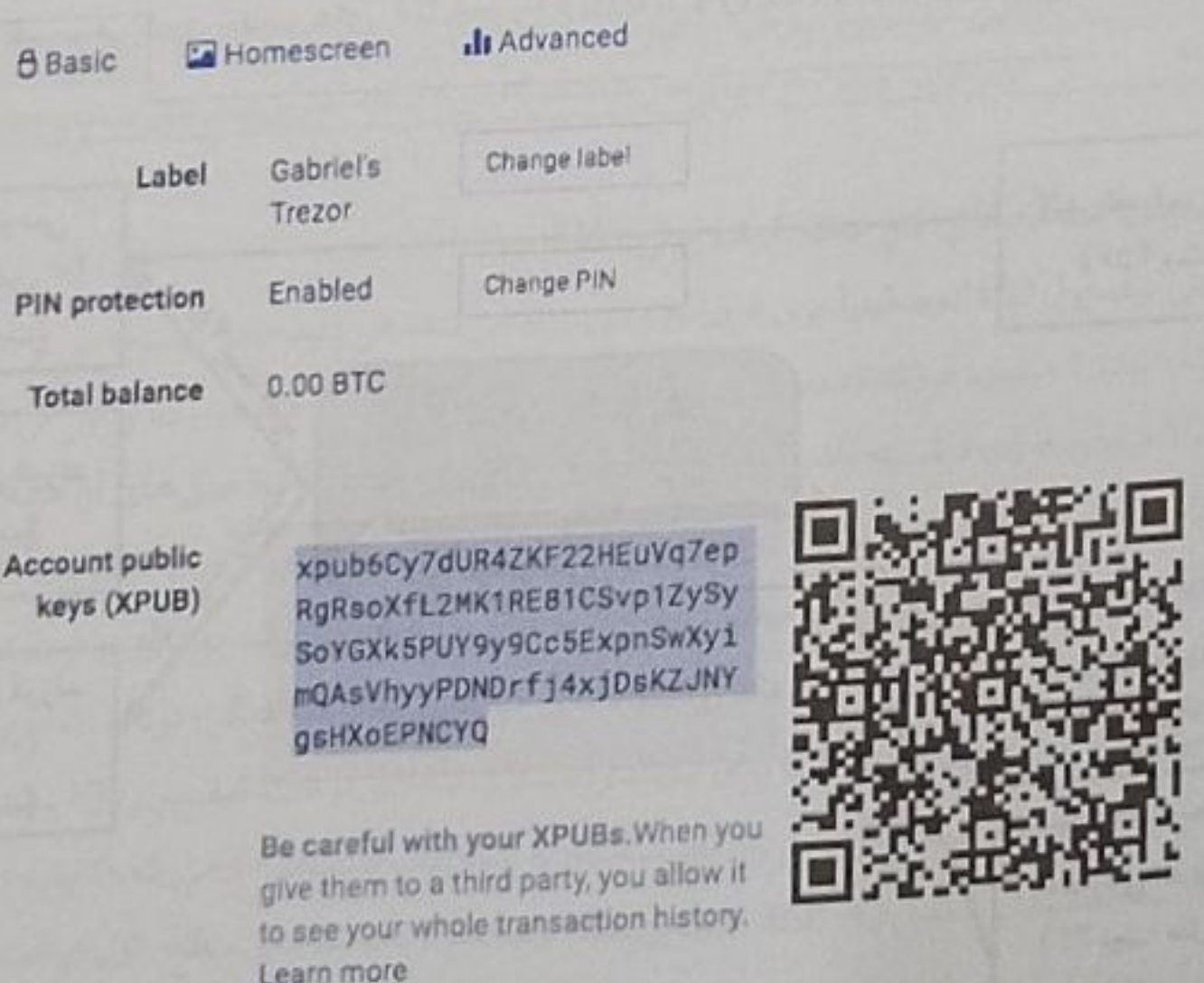
سرویس‌دهنده‌ی میزبان فروشگاه اینترنتی گابریل به امنیت چندان بالایی نیاز ندارد، چون هیچ کلید خصوصی روی آن ذخیره نمی‌شود.

برای صادر کردن این xpub، گابریل از یک نرم‌افزار وب- محور در کنار کیف‌پول سخت‌افزاری تریزور استفاده می‌کند. برای صادر کردن کلیدهای عمومی لازم است دستگاه تریزور به پورت USB کامپیوتر متصل شود. توجه کنید که کیف‌پول‌های سخت‌افزاری هرگز کلیدهای خصوصی را صادر نمی‌کنند و این کلیدها همیشه روی همان دستگاه باقی خواهند ماند. اجرای برنامه‌ی صدور xpub را در شکل ۵- ۱۲ مشاهده می‌کنید.

گابریل سپس این xpub را در نرم‌افزار نقل و انتقال بیت‌کوین فروشگاه اینترنتی خود کپی می‌کند. او برای این منظور از برنامه‌ی وب- محور *مایسلیوم گیر* (Mycelium Gear) که یک وصلینه‌ی منبع- باز است، و برای اکثر سرویس‌دهنده‌ها و مرورگرهای وب معروف ارائه شده، استفاده می‌کند. مایسلیوم گیر با استفاده از این xpub برای هر خرید یک آدرس بیت‌کوین یکتا تولید خواهد کرد.

اشتقاق کلید فرزند تقویت‌شده

توانایی استخراج یک شاخه کلید عمومی کامل از یک xpub بسیار سودمند است، ولی یک خطر بالقوه نیز به همراه دارد. البته دسترسی به یک xpub منجر به دسترسی به کلیدهای خصوصی فرزند نخواهد شد. با این حال، از آنجا که این xpub حاوی کُد زنجیره است، اگر یکی از کلیدهای خصوصی فرزند لو برود، می‌توان از آن کلید خصوصی و این کُد زنجیره برای بازتولید تمامی کلیدهای خصوصی فرزند همان شاخه استفاده کرد. به عبارت دیگر، فقط با داشتن یک کلید خصوصی فرزند به همراه کُد زنجیره‌ی مادر می‌توان تمامی کلیدهای خصوصی فرزند در آن شاخه را به دست آورد. بدتر آن که، این کلید خصوصی فرزند می‌تواند به همراه یک کُد زنجیره‌ی مادر برای استخراج کلید خصوصی مادر نیز به کار گرفته شود.



شکل ۵-۱۲ صادر کردن یک xpub (کلید عمومی گسترده) از کیف‌پول سخت‌افزاری تریزور.

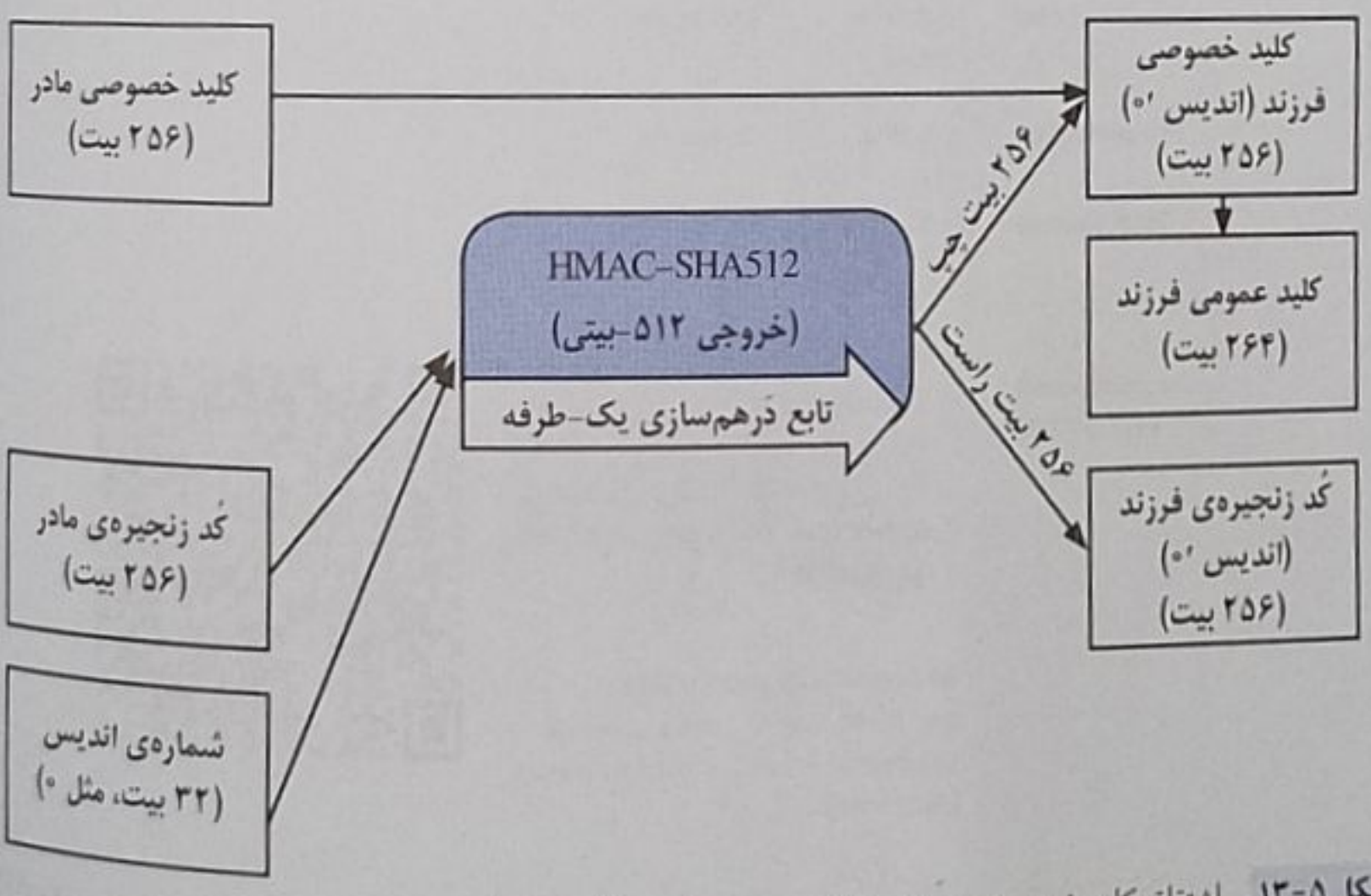
برای مقابله با این خطر، کیف‌پول HD از یک تابع استخراج کلید جایگزین موسوم به **اشتقاق تقویت‌شده** (hardened derivation) استفاده می‌کند، که رابطه‌ی بین کلید عمومی مادر و کُد زنجیره‌ی فرزند را «قطع» می‌کند. تابع اشتقاق تقویت‌شده به جای کلید عمومی مادر از کلید خصوصی مادر برای استخراج کُد زنجیره‌ی فرزند استفاده می‌کند. به این ترتیب یک «دیوار آتش» در دنباله‌ی مادر/فرزند ایجاد می‌شود و دیگر با داشتن یک کُد زنجیره نمی‌توان به کلید خصوصی مادر یا کلیدهای خصوصی خواهر متناظر با آن دسترسی یافت. تابع اشتقاق تقویت‌شده تقریباً مشابه تابع اشتقاق کلید خصوصی فرزند معمولی است، با این استثنا که (به جای کلید عمومی مادر) از کلید خصوصی مادر به عنوان ورودی تابع دَرهم‌سازی استفاده می‌کند؛ شکل ۵-۱۳ را ببینید.

وقتی از تابع اشتقاق تقویت‌شده استفاده می‌کنید، کلید خصوصی فرزند و کُد زنجیره‌ی حاصل به کلی با خروجی تابع اشتقاق معمولی متفاوت خواهند بود. خروجی این تابع یک «شاخه» از کلیدها است که آسیب‌پذیر نیستند چون از کُد زنجیره‌ی مورد استفاده در تولید آنها نمی‌توان برای استخراج کلیدهای خصوصی بهره‌برداری کرد و با آنها فقط می‌توان کلیدهای عمومی گسترده ساخت. در حقیقت، فرآیند اشتقاق تقویت‌شده در سطح بالای کلیدهای عمومی گسترده‌ی به کار رفته یک «شکاف» در درخت کلید کیف‌پول به وجود می‌آورد.

به بیان ساده، اگر می‌خواهید از یک xpub برای اشتقاق شاخه‌ای از کلیدهای عمومی استفاده کنید، بدون این که خود را در معرض خطر لو رفتن یک کُد زنجیره قرار دهید، باید (به جای یک کلید مادر معمولی) آنها را از یک کلید مادر تقویت‌شده استخراج کنید. بهترین روش آن است که کلیدهای فرزند سطح-یک کلید اصلی همیشه با تابع اشتقاق تقویت‌شده استخراج شوند تا خطر لو رفتن کلید اصلی از بین برود.

عدد اندیس برای اشتقاق معمولی و تقویت‌شده

عدد اندیس به کار رفته در تابع اشتقاق یک عدد صحیح ۳۲-بیتی است. برای تمایز بین کلیدهایی که با تابع اشتقاق معمولی استخراج شده‌اند و آنهایی که با استفاده از تابع اشتقاق تقویت‌شده به دست آمده‌اند، این عدد اندیس به دوبازه‌ی مساوی تقسیم می‌شود. اندیس‌های ۰ تا $2^{31}-1$ (0x0 تا 0x7FFFFFFF) *فقط* برای اشتقاق معمولی به کار



شکل ۵-۱۳ اشتقاق کلید فرزند تقویت‌شده: کلید عمومی مادر حذف شده است.

می‌روند، در حالی که اندیس‌های 2^{31} تا $2^{32}-1$ (0x80000000 تا 0xFFFFFFFF) *فقط* برای اشتقاق تقویت‌شده به کار گرفته می‌شوند. به عبارت دیگر، اگر عدد اندیس کوچکتر از 2^{31} باشد، با یک کلید فرزند معمولی روبرو هستیم، و اگر عدد اندیس مساوی یا بزرگتر از 2^{31} باشد، یک کلید فرزند تقویت‌شده داریم.

برای سادگی خواندن و نمایش اندیس‌ها، در کلیدهای فرزند تقویت‌شده هم شماره‌گذاری اندیس از صفر شروع می‌شود، ولی آنها را با علامت پریم (′) متمایز می‌کنیم. برای مثال، اولین کلید فرزند معمولی دارای اندیس ۰ است، در حالی که اندیس اولین کلید فرزند تقویت‌شده (اندیس 0x80000000) با ′۰ نشان داده می‌شود؛ اندیس دومین کلید فرزند تقویت‌شده (اندیس 0x80000001) نیز ′۱ خواهد بود، و الی‌آخر. وقتی در یک کیف‌پول HD اندیسی به صورت ′i می‌بینید، در واقع معنای آن اندیس $i+2^{31}$ است.

شناسه‌ی کلید (مسیر) در کیف‌پول HD

برای شناسایی و ارجاع به کلیدهای یک کیف‌پول HD از نام‌گذاری بر اساس «مسیر» استفاده می‌شود، که در آن هر سطح با کاراکتر *اسلَش* (/) از سطح قبلی جدا شده است؛ جدول ۵-۶ را ببینید. کلیدهای خصوصی مشتق‌شده از کلید خصوصی اصلی با کاراکتر «m»، و کلیدهای عمومی مشتق‌شده از کلید عمومی اصلی با کاراکتر «M» شروع می‌شوند. بنابراین، اولین کلید خصوصی فرزند کلید خصوصی اصلی m/0، و اولین کلید عمومی فرزند کلید عمومی اصلی M/0خواهند بود؛ دومین نوه‌ی کلید خصوصی اول نیز m/0/1 است، و الی‌آخر.

جدول ۵-۶ شناسه‌های کیف‌پول HD

مسیر کیف‌پول HD	کلید ارجاع‌شده
m/0	اولین کلید خصوصی فرزند (۰) کلید خصوصی اصلی (m)
m/0/0	اولین کلید خصوصی نوه‌ی (۰) کلید فرزند اول (m)
m/0′/0	اولین نوه‌ی «معمولی» اولین کلید فرزند «تقویت‌شده» (m/0′)
m/1/0	اولین کلید خصوصی نوه‌ی (۰) کلید فرزند دوم (m/1)
M/23/17/0/0	اولین نوه‌ی-نوه‌ی کلید عمومی نسل هیجدهم (اندیس ۱۷) کلید عمومی نوه‌ی نسل بیست و چهارم (اندیس ۲۳)

حرکت در میان شاخه‌های درخت کیف‌پول HD

ساختار درختی کیف‌پول HD انعطاف‌پذیری فوق‌العاده‌ای ارائه می‌کند. هر کلید مادر گسترده می‌تواند ۴ میلیارد فرزند (شاخه) داشته باشد: ۲ میلیارد فرزند «معمولی» و ۲ میلیارد فرزند «تقویت‌شده». هر یک از این فرزندها نیز به نوبه‌ی خود می‌تواند ۴ میلیارد فرزند داشته باشد، و الی‌آخر. عمق این درخت و تعداد نسل‌های آن عملاً نامحدود است. با این حال، با وجود این انعطاف‌پذیری خارق‌العاده، حرکت در میان شاخه‌های درخت کیف‌پول HD بسیار دشوار است. به دلیل نامحدود بودن ترکیب‌های محتمل ساختار داخلی شاخه‌ها و زیرشاخه‌های درخت کیف‌پول HD، این دشواری به خصوص خود را در انتقال کلیدها از یک برنامه‌ی کیف‌پول به برنامه دیگر خود را نشان می‌دهد.

برای مواجهه با این پیچیدگی، دو BIP برای استانداردسازی ساختار درخت کیف‌پول HD پیشنهاد شده است. در BIP-43 پیشنهاد شده که از اندیس اولین فرزند تقویت‌شده به عنوان یک شناسه‌ی خاص که «کارکرد» ساختار درختی را تعیین می‌کند، استفاده شود. بر اساس BIP-43، کیف‌پول HD باید فقط از یک شاخه‌ی سطح-اول درخت استفاده کرده، و از عدد اندیس برای ایجاد تمایز بین کلیدها، و تعریف کاربرد و شناسایی آنها استفاده کند. برای مثال، یک کیف‌پول HD که فقط از شاخه‌ی m/i′ استفاده می‌کند، برای یک کاربرد خاص در نظر گرفته شده که آن کاربرد با اندیس «i» مشخص می‌شود.

استاندارد BIP-44 پیشنهاد می‌کند که برای ایجاد یک کیف‌پول چندحسابی از اندیس ′۴۴ (تحت استاندارد BIP-43) موسوم به عدد «هدف» (purpose) استفاده شود. به عبارت دیگر، کیف‌پول‌های HD که استاندارد BIP-44 را پیاده‌سازی کرده‌اند، فقط از یک شاخه‌ی درخت، شاخه‌ی m/44′/، استفاده می‌کنند. در این استاندارد، ساختار درخت شامل ۵ سطح از پیش تعریف‌شده است:

```
m / purpose' / coin_type' / account' / change / address_index
```

در این کیف‌پول، سطح-اول [«purpose»] همیشه ′۴۴ است. سطح-دوم [«coin_type»] نوع ارز رمزبنیان را مشخص می‌کند، و اجازه می‌دهد این کیف‌پول با انواع مختلف ارزهای رمزبنیان کار کرده و هر نوع ارز را در یک زیرشاخه از سطح-دوم سازماندهی کند. در حال حاضر سه زیرشاخه در این سطح تعریف شده است: m/44′/0′ برای بیت‌کوین، m/44′/1′ برای بیت‌کوین تست‌نِت، و m/44′/2′ برای لایت‌کوین.

سطح-سوم [«account»] به کاربر اجازه می‌دهد تا کیف‌پول خود را برای مقاصد حسابداری مختلف سازماندهی کند. برای مثال، یک کیف‌پول HD می‌تواند شامل دو «حساب» بیت‌کوین، m/44′/0′ و m/44′/1′ باشد که هر یک از آنها ریشه‌ی زیرشاخه‌ی خاص خود هستند.

در سطح-چهارم [«change»]، این کیف‌پول HD دو زیرشاخه دارد، یکی برای ایجاد آدرس‌های دریافت و دیگری برای ایجاد آدرس‌های تتمه. توجه کنید که تا زیرشاخه‌ی سوم کلیدها از نوع تقویت‌شده هستند، ولی از زیرشاخه‌ی چهارم به بعد از کلیدهای معمولی استفاده می‌شود. این ویژگی به کیف‌پول HD اجازه می‌دهد تا کلیدهای عمومی گسترده‌ی این سطح را برای استفاده در محیط‌های غیرامن صادر کند. در واقع، در این سطح و سطح بعد است که کیف‌پول HD آدرس‌های فرزند مورد نیاز برای استفاده در تراکُنش‌های بیت‌کوین را تولید می‌کند.

آدرس‌های واقعی پرداخت/دریافت در سطح-پنجم [«address_index»] تولید و نگهداری می‌شوند. برای مثال، M/44′/0′/0′/0/2 شناسه‌ی سومین آدرس دریافت (کلید عمومی) یک تراکُنش بیت‌کوین در حساب اول است. به چند مثال دیگر در جدول ۵-۷ توجه کنید.

جدول ۵-۷ چند نمونه شناسه‌ی کلید در یک کیف‌پول HD با استاندارد BIP-44

مسیر کیف‌پول HD	کلید ارجاع‌شده
M/44′/0′/0′/0/2	سومین کلید عمومی دریافت برای اولین حساب بیت‌کوین
M/44′/0′/3′/1/14	پانزدهمین کلید عمومی تتمه‌ی پول برای چهارمین حساب بیت‌کوین
m/44′/2′/0′/0/1	دومین کلید خصوصی در اولین حساب لایت‌کوین، برای امضا کردن تراکُنش‌ها (پرداخت)