关于 Wormhole 协议燃烧地址的说明

Long Wen

August 4, 2018

Wormhole 协议采用 Proof-of-Burn（PoB）机制进行 Bitcoin Cash 与 Wormhole Cash 之间的兑换。为保证参与 Wormhole Cash 兑换的 Bitcoin Cash 不再流通，Wormhole 开发团队选取了一个无人拥有私钥的 Bitcoin Cash 地址作为燃烧地址：

qqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqu08dsyxz98whc

这是 BCH 的新地址格式，对应的传统地址为：1111111111111111115KMYP7R278，相应的十六进制表示：

00-0000000000000000000000000000000071E76C-501B5A27

按照传统地址规范，第一个字节的 00 表示版本号，最后的 4 个字节 501B5A27 为校验码，中间的 20 个字节 0000000000000000000000000000000071E76C 为 160 比特的哈希值。

需要声明的是，Wormhole 开发团队同样不知道任何的 $pubkey$ 值使其满足等式

$$\mathsf{RIPEMD160}(\mathsf{SHA256}(pubkey)) = 0000000000000000000000000000000071E76C$$

也即 Wormhole 团队不知道该地址的公钥，也不知道相应的私钥。如果选用的 160 比特的全零哈希值，该声明是不证自明的。没有选取全零哈希值的原因是，这一地址在 Wormhole 协议出现之前已经被使用过，参见 [1]。为 Wormhole 协议选取燃烧地址时，除了确保没有人拥有相应地址的公私钥对的诉求之外，主要有以下两点考虑：为 Wormhole 协议的燃烧地址选取一个未被用过的地址、让 Wormhole 协议的燃烧地址具有鲜明的 Wormhole 协议特色。基于以上考虑，开发团队从全零哈希值开始进行累加，直到根据 BCH 新的地址编码规则得到以字符“whc”结尾的燃烧地址。该地址的生成示例参见 Github 地址 [2]，读者可亲自验证上述逻辑。值得一提的是，Counterparty 项目的基础加密货币 XCP 也采用了 PoB 机制，参见 [3, 4]。

一个很自然的问题是，上述地址是否是合法的地址，也即是否真的存在合法的公钥 $pubkey$，满足上述等式？答案是肯定的。地址生成所采用的哈希函数 $\mathsf{RIPEMD160}(\mathsf{SHA256}(pubkey))$，将 $pubkey$ 映射成为 160 比特的哈希值。而 Bitcoin Cash 所采用的椭圆曲线 secp256k1 上的加法点群的阶（order）为

$$n = 0xfffffffffffffffffffffffffffffffebaaedce6af48a03bbfd25e8cd0364141,$$

即私钥 $privkey$ 可能的取值大约为 $n \approx 2^{256}$（此处不讨论几个不适合作私钥的特例情形，例如私钥取 0，1 等分别对应无穷远点以及基点本身的情况），也即公钥 $pubkey$ 的可能取值大约为 $n \approx 2^{256}$[1]。所以可以将哈希函

[1] 读者可能会困惑，即使在压缩公钥（compressed public key）的情形下，也需要 33 个字节来表示公钥的值，所以似乎公钥取值应该至少是 2^{256} 个。之所以会有压缩公钥的表示形式，是因为有了 256 比特的横坐标 x 的值之后，可以根据曲线方程推算出纵坐标 y 的值，然而由于 secp256k1 曲线是关于横坐标轴对称的，所以对于同一个 x 可能存在两个 y 值满足曲线方程，所以需要额外的 1 比特消除这一二义性。这么看下来似乎应该是有 2^{257} 个公钥的可能取值。注意虽然 x 是 256 比特的，但是只有一半的 x 值（专业术语：仅有一半元素有二次剩余）能够根据 secp256k1 的曲线方程解除对应的 y 值。所以公钥取值不会达到 2^{257} 这么多。更进一步，256 比特的 x 并不能取遍所有的 2^{256} 个值，而是只能在有限域 F_p，$p = 0xfffffffffffffffffffffffffffffffffffffffffffffffffffffffefffffc2f$ 中取值注意到有限域的特征（character）p 略小于 2^{256}，所以公钥的可能取值不会超过 2^{256}。

数 RIPEMD160(SHA256($pubkey$)) 看成映射：$\{0,1\}^{256} \to \{0,1\}^{160}$。由于 $2^{256}/2^{160} = 2^{96}$ 并且在随机谕示模型（Random Oracle Model，ROM）下可以将哈希函数 RIPEMD160(SHA256($pubkey$)) 的输出视为 $\{0,1\}^{160}$ 上的均匀分布，则根据鸽巢原理就有：

$$\forall h \in \{0,1\}^{160}, ||\{pubkey : \mathsf{RIPEMD160}(\mathsf{SHA256}(pubkey)) = h\}|| \approx 2^{96},$$

其中 $||\cdot||$ 表示集合中的元素个数。也即对任意的 160 比特哈希值，有大约 2^{96} 个公钥使该哈希值能够成为合法的 BCH 地址，所以至少存在一个公钥值，使得前述燃烧地址是合法的地址。

安全性方面的问题则涉及两个方面：1）如何确认 Wormhole 开发团队确实不知道前述燃烧地址的公私钥对？2）是否存在攻击者，能够根据前述燃烧地址找到 $pubkey/privkey$，从而将地址内的 Bitcoin Cash 占为己有，尤其会有多达 2^{96} 个公钥能够对应到燃烧地址？

燃烧地址所包含的 160 比特哈希值的十六进制表示为 0000000000000000000000000000000071E76C，观察可知，该 160 比特的哈希值最左侧的 137 个比特位全部为零。如果这是开发团队精心构造出来的地址，则构造的过程需要遍历 $pubkey$ 直到 RIPEMD160(SHA256($pubkey$)) 的输出的最左侧的 137 个比特位全部为零。或许读者已经注意到，这一过程跟 BCH 网络的 PoW 求解非常相似：通过大量穷举尝试，使获得的哈希值的最左侧的一些比特位为零。以当前（20180803）的块高度 534965 为例（参见 [5]），块哈希的十六进制表示为：

$$00000000000000000000a5c2378be75b246dae8932e3679f3b8542173c1015547,$$

最左侧的共有 76 个比特为零，根据 PoW 机制可知，为获得该块矿工大约尝试了 2^{76} 次计算。同样的道理，要求最左侧的 137 全部为零的话，需要 Wormhole 开发团队尝试大约 2^{137} 次哈希计算：RIPEMD160(SHA256($pubkey$))。这一计算量远远超过了当前 BCH 网络的全部算力，甚至远远超出了当前所有采用 PoW 机制的数字货币的计算量总和，这是不现实的，也就证明前述安全问题 1：**Wormhole 开发者并不拥有任何与燃烧地址相对应的公私钥对**。

根据燃烧地址中包含的 160 比特的哈希值推算 $pubkey$ 的过程，用密码学的术语可以表述为：成功完成对哈希函数 RIPEMD160(SHA256($\cdot$)) 的原像攻击。这一过程需要大概 2^{160} 次计算，所需的巨大的计算资源使得这一攻击是不可行的。如果存在任何一方能够完成这一过程，则可以对任意的 BCH 账户发动攻击，在这种假设条件下，则如下的 BCH 余额为 27 万的地址会更有吸引力，参见 [6]：

bitcoincash:qp0k6fs6q2hzmpyps3vtwmpx80j9w0r0acmp8l6e9v.

也就说明了前述安全问题 2：**根据燃烧地址中的 160 比特哈希值反推原像 $pubkey$ 是不可行的**。

最后一个问题：生日攻击（Birthday Attack）是否能够加速整个的攻击过程？因为各种密码学的介绍资料都有提及 160 比特的哈希值最多只能提供 80 比特的安全性，是不是 2^{80} 的计算量就足够完成攻击了？答案是否定的，生日攻击并不适合这个场景。

当讨论哈希函数的安全性时，通常会指涉三个性质：抗原像攻击/单向性（Preimage Resistance/One-Way Function）、抗第二原像攻击/弱抗碰撞性（Second-Preimage Resistance）、抗碰撞攻击/强抗碰撞性（Collision Resistance）：

1. 单向性指的是给定哈希值 h，无法获得输入消息 m 使得 $hash(m) = h$。
2. 弱抗碰撞性指的是给定消息 m_1，无法获得第二个消息 m_2 使得 $hash(m_2) = hash(m_1)$。

3. 强抗碰撞性指的是无法找到两个消息 m_1 和 m_2，使得 $hash(m_1) = hash(m_2)$。

上述三个特性中，攻击者的在攻击过程中的自由度在逐渐提高，自由度的提高意味着完成相应攻击所需的代价也就越低。反向推理则有：如果一个哈希函数满足强抗碰撞特性，则可规约证明该哈希函数满足弱碰撞特性，则可规约证明该函数函数满足单向性。但是对一个哈希函数能够成功完成碰撞攻击并不意味着，该攻击能够破坏哈希函数的抗第二原像攻击的性质，更不意味着能够破坏哈希函数的单向性。例如虽然 CRYPTO 2017 上 Marc Steven 等人给出了对 SHA-1 的碰撞实例 [7]，但是这并没有危及 SHA-1 算法的单向性，对 SHA-1 算法的最好的原像攻击 [8] 只能针对截断的 SHA-1 算法（安全性降低）进行并且仍停留在理论阶段。基于 SHA-1 的 HMAC 算法（Hashed-based Message Authentication Code）的安全性也依然稳固 [9]。

不同的应用场景的安全性会依赖上述的三个哈希函数性质中的某一个，例如 BCH 区块中构建交易的 Merkle Tree 并计算 Merkle Root Hash 时，依赖的是 SHA256(SHA256(·)) 的抗碰撞特性，而在计算 BCH 地址时依赖的是 RIPEMD160(SHA256(·)) 的单向性。生日攻击 [2] 只是给出了哈希函数抗碰撞特性的广义安全边界：160 比特的哈希值，能够提供 80 比特的抗碰撞安全性。而在 BCH 的地址生成中，如前所述，是哈希函数 RIPEMD160(SHA256(·)) 的单向性提供了安全保证，在当前并没有比穷举搜索更好的方式，也即根据燃烧地址中 160 比特的哈希值，反推原像，需要大约 2^{160} 次计算，攻击是切实不可行的。另外，虽然 SHA-1 和 RIPEMD-160 都是输出 160 比特的哈希值，并且已经能够构造出 SHA-1 的实际碰撞 [7]，对 RIPEMD-160 的碰撞攻击仍停留在理论阶段，参见最新成果 [10]，而对 RIPEMD-160 的最新原像攻击可以参考 [11]，也仅是理论攻击。在当前阶段以及可预见的未来，不需要为哈希函数 RIPEMD160(SHA256(·)) 的单向性安全而担心。

综上，已经论述了 Wormhole 燃烧地址的选取方式、燃烧地址的合法性以及燃烧地址的安全性，并通过衡量计算资源的方式说明了 Wormhole 开发团队同样不拥有任何与燃烧地址相对应的公私钥对。

---

[2] 简单介绍下生日攻击的原理，在很多文献中，也用生日悖论来指称同样的事实。搜索哈希函数的碰撞实例与在一个聚会中寻找生日冲突的问题完全相同，所不同的只是处理的问题的量级，在哈希函数的碰撞实例搜寻中，假设为输出为 n 比特的哈希值，则取值空间为 2^n，而在生日冲突的实例搜寻中，取值空间只有 365。随机选取的 t 个哈希值不存在冲突（也即任意两个哈希值均不相同）的概率为：

$$Pr(t\text{个哈希值互不冲突}) = (1-\frac{1}{2^n})(1-\frac{2}{2^n})\cdots(1-\frac{t-1}{2^n}) = \prod_{i=1}^{t-1}(1-\frac{i}{2^n})$$

对指数函数的泰勒级数展开有 $e^{-x} = 1-x+x^2/2!-x^3/3!+\cdots, for\ \ x << 1$，取展开式的前两项则有：如 $e^{-x} \approx 1-x$。带入上式可得：

$$Pr(t\text{个哈希值互不冲突}) = \prod_{i=1}^{t-1} e^{-\frac{i}{2^n}} \approx e^{-\frac{1+2+\cdots+t-1}{2^n}} = e^{-\frac{t\cdot(t-1)}{2\cdot 2^n}}$$

用 p 表示 t 个哈希值中存在冲突的时间，$p = Pr(t\text{个哈希值中存在冲突})$，则有：

$$p = 1 - Pr(t\text{个哈希值互不冲突}) \approx 1 - e^{-\frac{t\cdot(t-1)}{2\cdot 2^n}}$$

继续计算可得：$ln(1-p) \approx -t(t-1)/2^{n+1}$，$t(t-1) \approx 2^{n+1}ln(\frac{1}{1-p})$，由于通常有 $t >> 1$，则有近似：$t^2 \approx t(t-1)$，因此：

$$t \approx \sqrt{2^{n+1}ln\frac{1}{1-p}} \approx 2^{(n+1)/2}\sqrt{ln\frac{1}{1-p}}$$

取 $p = 0.5, n = 160$ 时，$t \approx 2^{80.23}$，也即当随机选取大约 2^{80} 个 160 比特的哈希值时，出现哈希冲突的概率为 0.5。这也就是生日悖论所提供的关于哈希函数的碰撞特性的安全边界：n **比特的哈希值能够提供大约** $n/2$ **的比特安全强度**。

References

[1] BTC.com. Bitcoin Cash Address: 0000000000000000000000000000000000000000. https://bch.btc.com/1111111111111111111114oLvT2

[2] Github Project. genburn. Codes for Generating Wormhole's Burn Address. https://github.com/copernet/genburn

[3] Blockchain.com. XCP Proof-of-Burn Address. https://www.blockchain.com/btc/address/1CounterpartyXXXXXXXXXXXXXXXUWLpVr

[4] Counterparty blog: Why Proof-of-Burn. https://counterparty.io/news/why-proof-of-burn/

[5] BTC.com. Bitcoin Cash Block Height 534965. https://bch.btc.com/000000000000000007ab116ea2dd1c7031b01895daa16e27d78986c141b888e

[6] BTC.com. Bitcoin Cash Address: qp0k6fs6q2hzmpyps3vtwmpx80j9w0r0acmp8l6e9v. https://bch.btc.com/19hZx234vNtLazfx5J2bxHsiWEmeYE8a7k.

[7] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, Yarik Markov: The First Collision for Full SHA-1. CRYPTO (1) 2017: 570-596. https://shattered.io/static/shattered.pdf

[8] Thomas Espitau, Pierre-Alain Fouque, Pierre Karpman: Higher-Order Differential Meet-in-The-Middle Preimage Attacks on SHA-1 and BLAKE. IACR Cryptology ePrint Archive 2015: 515 (2015). https://eprint.iacr.org/2015/515.pdf

[9] Mihir Bellare: New Proofs for NMAC and HMAC: Security without Collision Resistance. J. Cryptology 28(4): 844-878 (2015). https://cseweb.ucsd.edu/~mihir/papers/hmac-new.pdf

[10] Fukang Liu, Florian Mendel, Gaoli Wang: Collisions and Semi-Free-Start Collisions for Round-Reduced RIPEMD-160. ASIACRYPT (1) 2017: 158-186. https://eprint.iacr.org/2017/800.pdf

[11] Yanzhao Shen, Gaoli Wang: Improved Preimage Attacks on RIPEMD-160 and HAS-160. KSII TRANSACTIONS ON INTERNET AND INFORMATION SYSTEMS VOL. 12, NO. 2, 2018. http://www.itiis.org/digital-library/manuscript/file/1926/TIIS+Vol+12,+No+2-11.pdf