# 基于广义生日难题的非对称POW

# 摘要

PoW是现代加密货币和拒绝服务保护的核心概念，但是发展太快了，需要GPU ，ASIC， 有专门设备的用户。这种尝试依赖于内存密集型计算目的是消除架构间的不一致，导致了计划缓慢或者失败。

我们提出一种基于计算难题的非对称PoW，需要大量计算但是很容易验证。我们提出的Equihash(疑似是对称哈希)是一种基于广义生日难题的PoW，且增强了Wagner算法。引进了算法绑定新技术，阻止了成本摊销（amortization），证明了可能的并行实现受到内存带宽限制。方法权衡了时空，如果内存太小会受到惩罚。

我们的方案实际可用可部署的：在1.8GHz CPU 700MB RAM上30s。如果内存减则计算量增加1000倍，proof只有120bytes。

# 简介

解释了一下pow和dos。比特币矿池越来越大且越来越耗电。对于DoS保护，早期的TLS难题方案对最强大的对手不再有效。没有一个方案已经被实际的pow采用。

1. *内存难度计算*

为了减少ASIC和CPU之间的差距，Dwork etal提出了增强型内存计算，以及之后的内存难度计算。对于使用大量内存的用户来说这样可以降低内存的消耗。由于内存是一种非常昂贵的资源，在面积和平摊芯片成本方面，ASICs的效率只比普通的x86机器稍微高一些。此外，僵尸网络更不适应需要GB内存计算，因为他们更容易引起注意。 然而，人们可以辩称，减少的ASIC优势可能会提供额外的激励。

对于僵尸网络的创造者，从而降低了普通互联网用户的安全。

1. 对于相当大的内存来说他们打满，如果需要在合理时间内运行又必须使用小内存。（后边没看懂）
2. 提出的方案是对称的，参考了使用的内存。
3. 这些方案还没有对可能的优化和摊销进行彻底的分析。

一个相当快的非对称内存方案应该是一个通用工具，有效应对DOS，垃圾邮件，以及新的对等加密货币。

1. *最近的非对称候选方案*

两种解决对称和性能难题的方案。第一个是Dziembowski提出的proof of space。以及Tromp提出的Cuckoo-cycle PoW。

1. *我们的贡献*

我们提出了一系列快速，非对称内存，最优/分摊费用，有限的平行证明，基于已研究的计算难题。首先我们展示一个广范围的计算难题，能用可调参数进行非对称pow，ASIC和僵尸网络保护由最优时空权衡算法决定。

我们的主要方案是基于广义生日难题的pow，这是一种对称哈希。还引进了Wagner算法。（下边是实验环境）。为了增强方案的可信度，我们提高了现存针对生日难题的分摊策略。我们还展示了如何从两个不同的困难问题上构建一个PoW，并在算法绑定技术不适用的情况下得到最佳的权衡。在这方面，我们探讨了硬背包问题，在最近的论文中，最好的算法已经被仔细研究过。

1. *文章结构*

# POW和难以计算的问题

*A．特性*

这部分我们列举pow需要哪些性能，解释了一种一般化的放来关于如何把计算难题转化为pow，以及必备条件。

难题定义：

R：决定难度的参数集合。

I：参照R的输入集合

S：可能的解集

我们假设一个算法，AR可以在任意I上解决PR，即找到一个S使得P(R,I,S)=true

基于p的pow方案是一个交互协议，操作如下：

1. 验证者发起一次质疑，输入I∈*I*，R∈*R*
2. 证明者找到S发送给验证者
3. 验证者计算P(R,I,S)

一个非接触版更容易。*I*包含一些公共值（比特币最后一个区块的哈希值）和验证者的ID。证明者发出S这样任意一方都能验证证明。

非正式地，A应该难度适中。我们希望所有的证明者内存都够，处于同等地位，这样不需要私密的最优或者分期偿还。我们总结对于P和A的要求，以及其他的性能为了成为ASIC和僵尸网络对抗在如下列表。

**Progress-free进度**。为了防止中心化，缓和网络时延，挖矿必须是一个随机进程。任意时间的证明都是非零的和前边事件独立的，这样挖矿必须接近泊松分布随着一定数量的证明在给定框架下，服从泊松分布，且AR算法运行时间符合下边的指数分布：

泊松分布通常由难度过滤模拟，时间确定的算法β，需要nonce N作为输入，哈希函数G串起来，输出的尾部有确定数目的0.这样算法β必须免费分期偿还的，也就是给β生产输出q应该是q倍的时间。

**大AT（area-time）开销** 我们希望ASIC或者FPGA对于算法的实现要求较大的空间开销，高时空消耗不应该比台式机用性价比参数实现更好。空间由要求的内存最大化。对于一个有规律的使用者，AR的最佳实现要求充足的内存M.一般的电脑有1GB内存。芯片上的1GB内存很贵。

**小规模证明和即时验证** 解必须足够短，可以用小内存快速验证，为了防止验证者dos攻击。假设一些验证者用轻量级硬件，如智能手机RAM和带宽有限。

**陡峭的时空权衡**

**灵活性**

**并行限制**

**免费最优化**

*B.基于难题的内存难度pow*

验证者需要的内存比证明者的内存要少很多。带有内存捷径的内存困难的PoW长时间以来都是一个开放难题，现有的方案对于验证者和证明者都分配同等的内存。

然而几乎所有的难题都能转成一个pow。针对特定问题读者可以创造自己的pow。一个著名的NP问题是最自然的候选人。最佳算法运行在指数时间，即。另一方面，验证在|I|上是多项式的。因此在T上是多重对数的。对于基于NP完全的PoW的验证同运行时间比应该很快。NP完全的最佳算法通常需要数量可观的内存以及微不足道的时空权衡。SAT，cliqyes以及汉密尔顿路径都是PoW的候选。问题不是NP完全的，但是最佳算法是只输时间，解密是log。Section II-A中的属性很难全部满足。定义三个难题，开发审查实现的最优算法。

* 广义生日问题
* 硬背包问题
* 信息集解码问题

广义生日是最合适的。

# EQUIHASH：广义生日的POW

这部分讲解广义生日问题和Wagner的算法。

1. 问题

广义生日问题是这样的，给定一个列表L，列表里是n比特字符串{Xi},找到一个独一无二的{Xi},满足：

1. Wagner算法
2. 分析
3. 算法绑定
4. 时空权衡
5. 并行

# 我们的主要观点

# 广义生日难题的时空折中方案

# 相似问题

# 致谢

# 结论