GhostRider 채굴 알고리즘

By
Tri Nguyen-Pham

I. **목적:** asic 에 저항성이 있고, fpga 의 영향을 최소화하며, fpga 채굴 비용을 높여 진입하기 힘들게 하는 채굴 알고리즘의 개발

기술:

GhostRider 는 널리 알려진 채굴 알고리즘인 x16r (레이븐코인) 과 CryptoNight (모네로)를 통합한 것입니다. X16r 은 기존의 채굴을 위한 해시를 만드는 방법에 무작위성을 부여하여 메모리요구량을 줄이는 것으로써, 이는 gpu 보다 asic 이 더 이득을 볼 수 있는 구조입니다. 반대로 CryptoNight 는 cpu/gpu 메모리를 요구하기 때문에 asic 이 cpu/gpu 보다 더 이득을 보기 힘든 구조입니다. 하지만 x16r 이 가지고 있는 무작위성은 없습니다.

최근 몇 년간, 모네로 팀은 asic 을 대응하기 위해 CryptoNight 에 메모리 요구량과 해시를 만드는 방법의 변수들을 늘리는 포킹을 진행하였습니다. 그러나, 각 포크들도 해시를 만드는 방법은 항상 같았습니다.

GhostRider 방법론:

X16r 의 무작위성이, 높은 메모리 요구량과 함께 asic 효율 곡선에 영향을 미친다는 사실을 통해, GhostRider 의 컨셉은 두 가지 방법을 합치는 것으로 개발되었습니다. 15 개의 서로 다른 코어 기반 알고리즘을 무작위로 선택하여, Cryptonight 해시의 3 가지의 서로 다른 무작위 변수들을 섞는 것이 그것입니다. 이 알고리즘들은 5 가지 무작위 순서의 코어 알고리즘이 3 가지 그룹으로 나뉘어 있으며, 각 그룹마다 1 가지의 무작위 순서의 CN(Cryptonight)변수가 뒤따르고 있습니다. 모든 15 개의 코어 알고리즘은 무작위이지만 같은 체인에서 같은 알고리즘이 반복되는 경우는 없으며, 이는 CN(Cryptonight) 변수들의 순서에도 동일하게 적용됩니다.

무작위 순서 알고리즘: 알고리즘이 각 알고리즘의 순서를 정하기 위해, 이전 블록의 해시를 사용합니다. 이전 블록의 해시를 오른쪽에서 왼쪽으로 읽어나가면서, 각 숫자

혹은 문자에 정해진 알고리즘을 15 개 선택하게 되는 것입니다. 해시는 16 진수로 되어있으며, 하나의 해시당 총 64 개의 자릿수가 있습니다. 만약 숫자 혹은 문자가 F(15 번째 숫자)라면, 0 으로 되돌아 갑니다. 16 진수 숫자에 대응하는 각 알고리즘은 아래의 표에 나와있습니다. 만약 16 진수 숫자가 이전 것과 겹친다면 그냥 다음 자리로 건너뛰게 되고, 이 과정은 15 개의 16 진수 자리수가 모두 나올 때까지 반복됩니다.
비슷하게 CN 변수의 순서도 16 진수와 mod 에 의해 결정됩니다.

16 진수의 각 숫자와 대응하는 알고리즘 표:

0 or F-Blake
1-Bmw
2-Groestl
3-Jh
4-Keccak
5-Skein
6-Luffa
7-Cubehash
8-Shavite
9-Simd
A-Echo
B-Jamsi
C-Fugue
D-Shabal
E-Whirlpool

예시:

만약 이전 블록의 해시가 다음과 같다면;
0000135e13882a45caa301fc03429e416e7ce8d8edebdffe495ab337f9c98582
오른쪽에서 왼쪽으로 한자리씩 읽어나갑니다: 2-Groeslt, 8-Shavite, 5-Skein, 8(skip), 9-Simd, c-Fugue, 9(스킵), f-Blake, 7-Cubehash, 3-Jh, 3(스킵), b-jamsi, a-Echo, 5(스킵), 9(스킵), 4-Keccak, e-whirlpool, f(스킵), f(스킵), d-Shabal, b(스킵), e(스킵), d(스킵),e(스킵), 8(스킵), d(스킵), 8(스킵), e(스킵), c(스킵), 7(스킵), e(스킵), 6-luffa, 1-Bmw. 15 개의 알고리즘이 모두 정해졌으므로 멈추고,
결과값은 다음과 같습니다:
Groeslt->Shavite->Shein->Simd->Fugue->Blake->Cubehash->Jh->jamsi->Echo->Keccak->whirlpool->Shabal->Luffa->Bmw.

CN 변수들도 비슷하게 오른쪽에서 왼쪽으로 한 자리씩 확인하면서 가는데, 이번에는 각 자리의 mod3 +2(3 으로 나눈 나머지 +2)로 계산합니다.

2-CNv4, 8(스킵), 5(스킵), 8(스킵), 9-CNv2,c-(스킵), 9(스킵), f(스킵), 7-CNv3.

순서가 모두 정해졌으니 멈춥니다. 결과는 다음과 같습니다.

CNv4->CNv2->CNv3.

이제 위에서 정한 알고리즘을 5 개씩 3 그룹으로 나누고 CN 변수를 각 그룹 뒤에 붙여 순서를 정합니다. 따라서 최종 순서는 다음과 같습니다.

Groeslt->Shavite->Shein->Simd->Fugue->CNv4->Blake->Cubehash->Jh->jamsi->Echo->CNv2->Keccak->whirlpool->Shabal->Luffa->Bmw->CNv3