Yenten Whitepaper

An honest cryptocurrency for everybody, minable by the CPU.

Yenten is a cryptocurrency of the cpu, by the cpu, for the cpu. No ASIC.

Yenten wants to help reduce CO2 emissions by just using ordinary hardware to do mining.

Introduction

Bitcoin has long been the premium cryptocurrency for the storage of value. Bitcoin is the 'original' cryptocurrency; an immutable blockchain with a long history. Bitcoin is for this reason why many still see Bitcoin as a stable currency - not because it has not suffered any issues in the past, but because the blockchain technology originally proposed and brought into code reality by Satoshi stands the test of time and leaves a solid record of all operations in a blockchain including any of those that may be disagreeable. We expect transparency in our public trade markets and often transparency is left lacking, however Bitcoin cannot be argued that it shows every right and every wrong, making Bitcoin a trustable currency medium.

Even though the Bitcoin network holds a smaller percentage of the transactions taking place in today's cryptocurrency universe, the number of transactions has steadily risen, especially in times of heavy volatility as many holders of the coin seek to liquidate their holdings. This has lead to some significant troubles and Yenten seeks to provide another store of value which provides the same trusted immutability as Bitcoin. Yenten seeks to solve the issues in Bitcoin's growth by implementing the same trusted blockchain technology directly from Bitcore code whilst decreasing the block times significantly to provide a much faster and responsive network in order to achieve a much higher transaction volume and speeds that are now expected of mass-adopted modern cryptocurrency networks.

In Bitcoin, transactions are encoded in the blockchain with a cryptographic hashing algorithm. These hashes are created by miners using a "proof of work" (PoW) algorithm that combines one or more hashing functions. Bitcoin itself uses the SHA256d hashing function, which has been popularly used for a long time. ASICs (application specific integrated circuits) have been built to create SHA256d hashes at alarming rates which in turn has lead to the Bitcoin network hashrate inflating beyond the level that it was ever intended to be. In order to create a single hash for the Bitcoin network and use it to encode a new block in the blockchain, it now requires a significant amount of power and effort. In fact, using a GPU or CPU to create hashes

for the Bitcoin blockchain is effectively futile against the hashing power provided by SHA256d capable ASICs.
The prevalence of ASICs has lead Bitcoin to a situation where only those significantly invested in hardware are able to gain any kind of reasonable reward from mining it; thereby leaving the transaction processing in the hands of a few actors. Again, this is not an issue of the blockchain itself, but simply a side effect of high block times with an easily solvable hash algorithm.

Yenten: CPU mining only with YescriptR16
Yenten uses YescriptR16 as algorithm, which is a variant of yescript and which performs best on a CPU. Yescript has the following properties:

- Builds upon scrypt, computes classic scrypt and native yescrypt hashes
- High flexibility and large arsenal of defenses
    - One password hashing scheme for many possible use cases that deals with many kinds of attacks
- Scalable from kilobytes (RAM) to terabytes (RAM + ROM) and beyond
    - while providing adequate (or better) security vs. alternatives for same defender's {time, memory} cost
- Scalable to arbitrary SIMD vector width and instruction-level parallelism
- Optional TMTO resistance
    - When enabled, additionally makes attacks' area-time a few times higher
- Optional bcrypt-like GPU unfriendliness (especially important at low memory usage settings)
- Optional multiplication latency hardening (efficient at least on common x86 and ARM CPUs)
- Running time optimally tunable separately from memory usage and parallelism
- Client-side computation of almost final yescrypt hashes (server relief)
    - in a way allowing for a straightforward extension of SCRAM (RFC 5802)
- Hash upgrades to higher cost settings without knowledge of passwords
- Cryptographic security is based on that of SHA-256, HMAC, and PBKDF2
    - The rest of processing may be considered non-cryptographic
    - Known unfortunate peculiarities of HMAC and PBKDF2 are fully avoided

Yenten and YescryptR16: GPU resistance

At small memory cost settings, yescrypt with (at least) the YESCRYPT_PWXFORM flag set discourages GPU attacks by implementing small random lookups similar to those of bcrypt. With current default settings and running the SIMD implementation on a modern x86 or x86-64 CPU (such as Intel's Sandy Bridge or better, or AMD's Bulldozer or better), yescrypt achieves frequencies of small random lookups and of groups of (potentially) parallel small random lookups that are on par with those of bcrypt. (In case of groups of (potentially) parallel lookups, the frequency is normalized for S-box size, since the relevant GPU attack uses the scarce local memory.)

bcrypt's efficiency on current GPUs is known to be extremely poor (making contemporary GPUs and CPUs roughly same speed at bcrypt per-chip), from three independent implementations. The current limiting factors are: GPUs' low local memory size (compared even to bcrypt's 4 KiB S-boxes per instance), high instruction latencies (com- pared to CPUs), and (for another attack) the maximum frequency of random global memory accesses (as limited by global memory bandwidth divided by cache line size).

yescrypt tries to retain bcrypt's GPU resistance while providing greater than bcrypt's (and even than scrypt's) resistance against ASICs and FPGAs. Improving upon bcrypt's GPU resistance is possible, but unfortunately it currently involves yescrypt settings that are suboptimal for modern CPUs (leaving too little parallelism to fully exploit those CPUs for defense), thereby reducing resistance against some non-GPU attacks (even attacks with CPUs, where the parallelism would be re-added from multiple candidate passwords to test at once).

At much larger memory cost settings, yescrypt with (at least) the YESCRYPT_RW flag set additionally discourages GPU attacks through discouraging time-memory tradeoffs (TMTO) and thereby limiting the number of concurrent instances that will fit in a GPU card's global memory. The more limited number of concurrent instances (compared e.g. to classic scrypt, which is TMTO-friendly) prevents the global memory access latency from being hidden or even leaves some computing resources idle all the time (like it also happens with YESCRYPT_PWXFORM above due to limited local memory).

Setting both flags at once achieves the best effect, regardless of memory cost setting.

Yenten and YescryptR16: ASIC and FPGA resistance

Yescrypt with (at least) the YESCRYPT_PWXFORM flag set performs rapid random lookups (as described above), typically from a CPU's L1 cache, along with 32x32 to 64-bit integer multiplications. Both of these operations have latency that is unlikely to be made much lower in specialized hardware than it is in CPUs. (This is in contrast with bitwise operations and additions found in Salsa20/8, which is the only type of computation performed by classic scrypt in its SMix and below. Those allow for major latency reduction in hardware.) For each sub-block of data processed in BlockMix, yescrypt computes multiple sequential rounds of pwxform, thereby imposing a lower bound on how quickly BlockMix can proceed, even if a given hardware platform's memory bandwidth would otherwise permit for much quicker processing.

yescrypt with (at least) the YESCRYPT_RW flag set additionally discourages time-memory tradeoffs (TMTO), thereby reducing attackers' flexibility. Perhaps more importantly, yescrypt's YESCRYPT_RW increases the area- time cost of attacks, and this higher cost of attacks is achieved at a lower (defensive) running time. Specifically, scrypt achieves its optimal area-time cost at 2*N combined iterations of the loops in SMix, whereas yescrypt achieves its optimal area-time cost at 4/3*N iterations (thus, at 2/3 of classic scrypt's running time) and, considering the 2x area-time reduction that occurs along with exploitation of TMTO in classic scrypt, that cost is higher by one third (+33%). Normalized for the same running time (which lets yescrypt use 1.5 times higher N), the area-time cost of attacks on yescrypt is 3 times higher than that on scrypt.

Like with GPU attacks, setting both flags at once achieves the best effect also against specialized hardware.

YescryptR16: all details

All yescrypt details can be found here:
https://github.com/bsdphk/PHC/blob/master/Yescrypt/yescrypt-phc.pdf

Yenten: an honest Coin

Yenten has no ICO, no airdrops, no premine.

Yenten: technical specifications

Algorithm: YescryptR16 (GPU is slower than CPU)
Block time: 2.0 minutes
Max Block size: 2M
Block reward of block #1: 50 YTN
Total YTN setting: 84,000,000 YTN
Calculated max YTN: about 80,000,000 YTN
SubsidyHalvingInterval: 800,000 blocks
Difficulty re-target: every block (DarkGravityWave v3-1)
Premine: none
P2P Port: 9981
RPC Port: 9982

Yenten: Roadmap

Yenten has a simple roadmap, like that of the original bitcoin. It's a cryptocurrency that will grow because of the community that trusts and uses this coin.

DISCLAIMER:

This Yenten White Paper is for information purposes only. The autohor does not guarantee the accuracy of or the conclusions reached in this white paper, and this white paper is provided "as is". The author does not make and expressly disclaims all representations and warranties, express, implied, statutory or otherwise, whatsoever, including, but not limited to: (i) warranties of merchantability, fitness for a particular purpose, suitability, usage, title or noninfringement; (ii) that the contents of this white paper are free from error; and (iii) that such contents will not infringe third-party rights. The author and its affiliates shall have no liability for damages of any kind arising out of the use, reference to, or reliance on this white paper or any of the content contained above, even if advised of the possibility of such damages. In no event will the author or its affiliates be liable to any person or entity for any damages, losses, liabilities, costs or expenses of any kind, whether direct or indirect, consequential, compensatory, incidental, actual, exemplary, punitive or special for the use of, reference to, or reliance on this white paper or any of the content contained above, including, without limitation, any loss of business, revenues, profits, data, use, goodwill or other intangible losses.

Large parts of the text of this whitepaper are copied from the 'Denarius Whitescroll' and 'yescrypt - a Password Hashing Competition submission'