**Описание монеты eGOLD v1.1**

**eGOLD** (*electronic gold* - электронное золото) - это электронная криптовалюта, получившая своё название по своему прямому назначению также, как E-mail (*electronic mail* - электронное письмо). E-mail письма прочно вошли в нашу жизнь и вытеснили обычные письма из-за своей простоты, удобства использования и моментальной скорости доставки без каких-либо дополнительных трат. E-mail письма, что немаловажно, дружелюбны к окружающей среде и не требуют использования природных ресурсов. Также и **eGOLD** позиционируется, как экологическая альтернатива физическому золоту и бумажным деньгам с значительными преимуществами над ними, схожими с преимуществами E-mail писем над бумажными.

Стоимость монеты **всегда** будет привязана к тройской унции золота в пропорции 1/100 000. К примеру, по текущему курсу на 25 мая 2020г. **100** монет **eGOLD = 1.73$** или **124.60 руб.**

Любые деньги - это вера сообщества в их меновую ценность по сравнению с товарами или услугами. А истинная цена любых товаров и услуг определяется только вложенным в них трудом, поэтому воздух, которым мы дышим, сейчас не имеет ценны, а вот воздух которым сможем подышать где-то далеко в горах или на море уже будет иметь цену, так как чтобы добраться до данного места потребуется потрудиться и даже заплатить деньги за труды перевозчиков. Вот и получилась цена за воздух. Тоже самое и с остальными товарами и услугами: сколько сил и средств на это тратим, столько и должно это стоить. Убирая издержки в виде ненужных затрат, мы увеличиваем ценность денег, увеличивая количество покупаемых товаров и услуг при том же количестве средств. Тоже самое делают криптовалюты, они убирают издержки. Как пластиковые банковские карты, которые убирают издержки по обмену валюты и трудозатраты на этот обмен. Например, убирается стоимость автобуса, чтобы добраться до банка и время потраченное на это всё. Издержки понижаются ещё при снижение процентов затрачиваемых на обслуживание безналичной оплаты, которую снимают банки за использование пластиковых карт, и эта цена заложена в товаре. В конечном итоге, самая лучшие деньги - это удобные деньги без дополнительных издержек с моментальной безопасной оплатой при сохранении их неизменяемой ценности.

С увеличением популярности монеты это неизбежно приведёт к необходимости увеличения стоимости монеты или увеличения её количества в обращении. При увеличении цены монеты, она потеряет платежную способность, потому что её курс станет плавающим, что приведёт к неудобству в расчётах. Искусственное увеличение объёмов монеты на рынке создателями, приведёт к дисбалансу и концентрации ресурсов в одних руках, что неизбежно подорвёт авторитет монеты и поспособствует инфляции. Поэтому, как и в некоторых других PoS криптовалютах в eGOLD встроен **сеньораж** в виде 4% прибыли в месяц для всех пользователей держащих монеты на своём балансе или по-другому в криптовалютах это называется заработок на **стейкинге**, а таких пользователей в среде блокчейна называют **холдерами**. При этом в eGOLD проценты зачисляются ежесекундно и не требуется долго холдить монеты для получения прироста монет на балансе. Более того, зачисляется ещё и бонусный **1%** держателям нод и есть реферальная система для справедливого распределения ресурсов и поощрения трудозатрат по обслуживанию, популяризации и использованию монеты. Первичный выпуск eGOLD ограничен небольшим количеством относительно её цены, чтобы дать старт всем желающим людям равномерно распределить монеты для их последующей честной генерации и полной децентрализации в отличии от многих криптовалют, где на рынок выпускается только около 10-30% всех монет, а остальное держат у себя создатели.

Если нет возможности купить монеты и желания держать ноду, но есть желание помочь в её развитии и при этом неплохо зарабатывать в неограниченном масштабе, тогда стоит заняться продвижением монеты и создавать кошельки другим людям. Прибыль при этом равна 25% (1% ежемесячно от баланса рефералов 1 уровня) от сеньоража тех людей, кого пригласили , и 12.5% (0.5% ежемесячно от баланса рефералов 2 уровня) от тех, кого они пригласят, и ещё 6.25% (0.25% ежемесячно от баланса рефералов 3 уровня) от тех, кого пригласят те, с кого получают 12.5%. Тратя свои силы и средства на криптовалюту eGOLD, пусть даже те, что пришли от процентов, с лихвой окупятся со временем. Популярность, распространение и использование монеты, а вместе с этим, и доход от неё, полностью в руках всего сообщества и каждого владельца монеты eGOLD!

**eGOLD** является экологической **PoS** криптовалютой на постблокчейновой технологии неограниченной пропускной способности параллельных транзакций с использованием быстрого **квантовоусточивого** алгоритма шифрования **FALCON** и что особенно важно - возможностью смены ключей шифрования (открытого и закрытого ключа) с сохранением номера кошелька, при этом новый публичный ключ закрыт под **SHA3** хэшированием и своим специально разработанным хэшированием до момента следующей транзакции, что делает невозможным вычисление закрытого ключа по открытому в связи с его полным сокрытием. В работе с кошельком используется только сменяемый закрытый ключ и постоянный номер кошелька. В криптовалюте eGOLD нет блокчейна, вместо этого используется более совершенный многоуровневый алгоритм следующего поколения для синхронизации транзакций похожий на **DAG (directed acyclic graph – направленный ациклический граф)** с существенным отличием, закрывающим проблемы параллельной синхронизации - это рассылка об успешной транзакции всем нодам веером по цепочке, если эта транзакция не приходила ранее и она прошла проверку на валидность, включая проверку на верность и принадлежность подписи кошельку отправителю. Для работы ноды нужен обычный интернет хостинг с PHP и MySQL. Время принятия одной транзакции около **4 секунды** (обработка транзакции около 1-2 минут). Защита от двойной траты сделана с помощью временной задержки и похожа на защиту криптовалюты **NANOCOIN**. Также в нодах существует проверка на пропущенные транзакции и постоянная проверка валидности кошельков по 51% баланса доступных нод. В eGOLD встроена система самоочищения базы данных, путем введения **горизонта истории транзакций**, это позволяет сохранить неизменным её объем при одинаковом количестве транзакций. К примеру, у **Биткоина** на текущий месяц (июль 2019 года) в среднем около 300 тысяч транзакций в день, а объем базы данных блокчейна постоянно увеличивается и достиг 270 Гигабайт. В eGOLD же при стольких же транзакций в сутки база данных будет примерно **9 Гигабайт** и этот объем, благодаря горизонту и другим методам самоочистки, будет неизменен. Майнинг отсутствует, но есть вознаграждение держателям нод в виде 1 монеты от каждой транзакции ноды + 1% в месяц (месяц равен 30 дням здесь и далее по тексту) держателям нод от баланса кошелька прикрепленного к ноде. Все кошельки получают 4% в месяц монет на баланс кошелька по сложному проценту при начислении раз в секунду - это 4.08% в месяц и **61.59% в год**, а владельцы нод получают дополнительно +1% при совершении транзакций со своих нод от текущего баланса в месяц - это 5.13% в месяц и **82.21% в год** по сложному проценту, помимо вознаграждения от каждой транзакции отправленной с его ноды любыми кошельками. Все владельцы монет получают проценты в качестве **сеньоража** (доход, получаемый от выпуска денег), то есть создателями монет являются все участники. Зачисление в баланс происходит автоматически без создания дополнительных транзакций. Общий первичный выпуск **1 000 000 000 монет**. Монеты могут уничтожаться путём отправления на адрес **G-1**. G-1 - это служебный адрес, не имеющий кошелька. Других таких служебных адресов нет и можно отправлять монеты только на адреса кошельков. После реализации предмайнинга, монеты генерируются на кошельках пользователя в соответствии с правилами маркетинга, никакого ограничения генерации монет нет. Пользователи всегда могут просмотреть общий объём монет в ноде с помощью запроса http://[IP ноды]/egold.php?type=balanceall , где в ответе balanceall - это общий объём монет на всех кошельках без учёта начисления процентов с момента последней транзакции.

В eGOLD в качестве единиц используются только целые числа 1,2,3... без сотых. Комиссия на все транзакции 2 монеты и не зависит от суммы транзакции. При этом 1 монета всегда уходит владельцу ноды к которой подключается кошелёк для совершения транзакций и 1 монета списывается системой для защиты её от атак с использованием многочисленных транзакций. Когда пользователь совершит транзакцию и добавит пароль для доступа к ноде, он сможет воспользоваться сервисами ноды, через которую была совершена эта транзакция: книгой контактов (все контакты зашифрованы от владельца ноды) и E-mail уведомлениями о транзакциях на почту. В монету встроена 3-х уровневая автоматическая реферальная система. Каждый раз при любых транзакциях своих рефералов, владелец рефералов получает вознаграждение от начисленных процентов своих рефералов, зависящий от баланса своих рефералов, прошедшего времени между транзакциями у рефералов и уровня рефералов: 25% (1-ый уровень), 12.5% (2-ой уровень) и 6.25% (3-ий уровень). К примеру: реферал 1-ого уровня получил с 10 000 своих монет на кошельке за месяц 408 монет в виде процентов (4% в месяц по сложному проценту, начисляемому ежесекундно). Тот, кому принадлежит этот реферал, даже при нулевом своем балансе, автоматически получит 102 монеты (отдельно реферал получает 408 монет и отдельно владелец реферала 102) после любой транзакции реферала (перевода средств со своего баланса). Если бы реферал 1-ого уровня был рефералом 2-ого уровня, владелец бы получил 51 монету. Создание кошелька стоит 5 монет, в которые уже входят комиссионные 2 монеты системы, а 3 монеты остаются на вновь созданном кошельке для возможности смены закрытого ключа. При этом создать кошелёк сможет только владелец уже созданного ранее кошелька. Создание кошелька происходит вместе с заданием секретной фразы для нового кошелька, которую потом нужно сменить (не обязательно). Это сделано для популяризации монеты и для защиты её от атак, использующих многочисленные транзакции.

У монеты есть только целая часть и нет сотых, как у доллара центы, а у рубля копейки. В расчёте используется 100 монет вместо 1.00. Цена на золото выбрана по многим причинам. Одна из которых - это относительная стабильность и рост курса по сравнению с фиатными деньгами (за несколько последних месяцев цена на золото в долларах США возросла почти на 20%). Стабильность обусловлена ограниченностью золота в природе и его многовековой ценностью. Не даром, деньги привязанные к запасу золота (золотовалютным резервам), являются самыми стабильными. Хоть eGOLD не обеспечена золотом, но все участники обязуются взять на себя ответственность придерживаться заданного курса также, как мировое сообщество обязалось ценить золото.

Код ноды написан на PHP (требуется версия PHP не ниже 7.1, библиотека PHP-BCMATH, GMP и CURL) + MySQL (рекомендуется использовать актуальные версии). Официальный кошелёк написан на HTML с JavaScript, jQuery, Ajax и сделан в виде одного HTML, что делает возможным его запуск на любых устройствах. FALCON шифрование с использованием закрытого ключа происходит непосредственно в самом HTML файле, а в ноду уже отправляется подписанная транзакция, что полностью защищает секретную фразу от кражи, путем перехвата в момент отправления до ноды. Пара закрытого и открытого ключа тоже генерируется на HTML кошельке.

Для удобства пользователей в ноду встроены модули для хранения контактов и отправки E-mail уведомлений о транзакциях, аналогично SMS уведомлениям банков о зачислении или списании средств. Имена контактов в ноде хранятся в зашифрованном виде. Доступ к контактам и управлению E-mail уведомлениями на ноде осуществляется с помощью пароля. Пароль задается и изменяется при любой транзакции. Для удобства, в кошельке для задания пароля сделана отдельная функция. Сервисы контактов и E-mail уведомлений работают на конкретной ноде и доступны участнику только после совершения транзакции на данной ноде, что стимулирует пользователей использовать определенные ноды, а владельцу ноды с каждой транзакции по ней, идёт вознаграждение в виде 1 монеты. Для работы ноды на балансе кошелька, привязанного к ноде должно быть не менее 100 монет.

Нода eGOLD обладает возможностью самоочищаться от неиспользуемых данных, что защищает её от чрезмерного использования памяти и ресурсов системы, сохраняя одинаковый объем базы данных при одинаковом количестве транзакций путем удаления:

* истории транзакций старше 30 дней (параметр настраивается переменной $history\_day в файле настроек egold\_settings.php)
* ноды у кошелька, если по кошельку не было транзакций и его нодой не пользовались больше 30 дней
* учётных записей для использования отправки E-mail уведомлений и контактов кошелька, если кошелёк не пользовался этой нодой более 365 дней
* кошельков с балансом меньше 10 монет, если ими не пользовались больше 90 дней
* кошельков, если по ним не было входящих и исходящих транзакций больше 30 лет

Исходный код монеты открыт и прост, поэтому монета будет пользоваться популярностью у разработчиков. Каждый сможет сделать себе или всему сообществу модули, решающие те или иные задачи. А владельцы сайтов смогут встраивать ноды в любые свои веб проекты и напрямую принимать eGOLD монеты. Генерация открытого и закрытого ключа, подпись транзакций (оплата), также возможна и с ноды, что позволяет автоматизировать сервисы с выводом средств на кошельки участников. Настройки ноды хранятся в файле egold\_settings.php. Запросы к нодам осуществляются с помощью GET и POST обращения к файлу egold.php, а ответы ноды представлены в виде JSON.

**\* 1.** Запрос к ноде без параметров

**Пример обращения к ноде по ссылке:** http://91.106.203.179/egold.php  
**Пример ответа страницы:** {"time":"1586582875", "noda":"91.106.203.179", "owner":"G-1000-00000-0000-00000", "transactionscount":"12", "datelasttransaction":"1586553129", "email\_domain":"egold.pro", "noda\_site":"[https://www.egold.pro](https://www.egold.pro/)"}

**time** - время на ноде

**noda** (параметр нужен для работы ноды и он прописывается в файле настроек egold\_settings.php под названием **$noda\_ip**) - IP текущей ноды

**owner** (параметр нужен для работы ноды и он прописывается в файле настроек egold\_settings.php под названием **$noda\_wallet**) - кошелёк владельца ноды и нужен для того чтобы нода заработала

**transactionscount** — количество транзакций с момента установки ноды

**datelasttransaction** - время последней транзакции для статистики и проверки обновления ноды

**email\_domain** (необязательный параметр, который прописывается в файле настроек egold\_settings.php под названием **$email\_domain**) - почта с которой приходят уведомления по транзакциям пользователям ноды.

**noda\_site** (необязательный параметр, который прописывается в файле настроек egold\_settings.php под названием **$noda\_site**) - домен сайта ноды по которому можно обратиться к ноде

**\* 2.** Запрос [type=**wallet**] - данные по кошельку

**Пример GET запроса**: http://[IP ноды]/egold.php?type=wallet&wallet=G-1261-21479-1061-51551

**Пример JSON ответа**: {"time":"1567159479","noda":"[IP ноды]","email\_domain":"[домен для отправки уведомлений на почту, если его задали в настройках]", "wallet":"G-1261-21479-1061-51551","ref1":"G-4613-11305-1110-01414","ref2":"G-1514-07813-0081-39731","ref3":"G-1000-00000-0000-00000", "nodawallet":"91.106.203.180", "nodawalletuse":"91.106.203.180", "balance":"11578494", "percent\_4":"121", "percent\_5":"151", "height":"311", "date":"1567243045", "signpubnew":"", "signnew":"", "signpub":"[публичный ключ кошелька]", "sign":"[подпись последней транзакции]"}

В каждом запросе обязательно должен быть указан его тип: **type**. В примере используется тип запроса **wallet** с дополнительным параметром **wallet** равным номеру кошелька. Кошелёк везде может быть записан только в виде цифр (остальное отбрасывается). Пример: 126121479106151551.

**balance** - текущий баланс с учётом баланса находящегося на проверке

**percent\_4 -** 4% начисление сеньоража в общем случае

**percent\_5** - 5% начисление сеньоража, если при своей ноде и при проведения следующего платежа с неё. Если платёж будет не со своей ноды, нода перестанет работать и сеньораж станет равен 4%

**balancetransactioncheck** - баланс находящийся на транзакции

\* Если есть необработанные транзакции, то нельзя сразу осуществить оплату ещё и на проценты. При этом отображается параметр **balancetransactioncheck** с суммой необработанных транзакций, а весь баланс уменьшается на эту сумму с учётом 2 монет комиссии.

\*Дополнительно в запросе можно просмотреть настройки аккаунта на ноде для отправки уведомления, для этого отправляется параметр пароля **password** хэшем SHAKE256 длинной 256 символов (1024 бита) от предыдущего пароля, который был уже записан на ноду под хэшем SHAKE256 длинной 128 символов. То есть, сначала от предыдущего пароля берётся хэш SHAKE256 на 128 символов на стороне клиента и отправляется в ноду для задания пароля, а потом для проверки пароля на ноде уже от имеющегося хэша берётся ещё раз хэш, но длиною 256 символов, используя функцию gen\_sha3([пароль на ноде],256), отправляется с клиента и проверяется на ноде. Если соответствует два хэша, тогда считаем всё верно и показываем настройки: есть ли почта и с какими порогами.

\*Если пароль будет изменён (будет отличаться от уже установленного), то все сохранённые контакты будут удалены, так как они зашифрованы под старый пароль.

**Пример GET запроса:** : http://[IP ноды]/egold.php?type=wallet&wallet=G-1261-21479-1061-51551&password=[пароль под SHAKE256 с длиной хэша 256 от предыдущего который под SHAKE256 с длиной хэша 128]

**Пример JSON ответа**: {"time":"1567159668","noda":"[IP ноды]","wallet":"G-1261-21479-1061-51551","ref1":"G-4613-11305-1110-01414","ref2":"G-1514-07813-0081-39731","ref3":"G-1000-00000-0000-00000","nodawallet":"91.106.203.180","nodawalletuse":"91.106.203.180","balance":"11578494","percent\_4":"121","percent\_5":"151","height":"311","date":"1567243045","usersemail":"[email]","usersemailup":"10","usersemaildown":"20","usersemaildateupdate":"1567159580","signpubnew":"","signnew":"", "signpub":"[публичный ключ кошелька]", "sign":"[подпись последней транзакции]"}

**\* 3.** Запрос [type=**nodas**] - список доступных нод

Не использует дополнительные параметры и выводит список нод со статистикой обращения к ним и с количеством кошельков подключенных к ним.

Использует дополнительный необязательные параметры:

* balancestart - минимальный баланс кошелька ноды
* balancefinish - максимальный баланс кошелька ноды
* nodausewalletstart - минимальное количество кошельков, использующих ноду
* nodausewalletfinish - максимальное количество кошельков, использующих ноду
* order - параметр сортировки вывода истории в прямом порядке от меньшей даты к большей с параметром =asc, по умолчанию в обратном порядке по дате последнего использования. Также можно отобразить ноды по балансу от большего к меньшему при order=balance.
* start - с какой транзакции в выдаче начать отображать. Здесь и далее в других типах с множественным отображением: при =0 начинаем без пропусков, при =1 сдвигаем выдачу на 1 и так далее.
* limit - количество отображенных нод от параметра start. По умолчанию здесь и далее в других типах с множественным отображением =100.

**Пример GET запроса**: http://[IP ноды]/egold.php?type=nodas

**Пример JSON ответа**: [{"noda":"91.106.203.181","wallet":"461311305111001414","balance":"17516750","walletsuse":"1","datelastuse":"1586553129"}, {"noda":"91.106.203.180","wallet":"126121479106151551","balance":"14365293","walletsuse":"3","datelastuse":"1586553129"}, {"noda":"91.106.203.179","wallet":"100000000000000000","balance":"18402445","walletsuse":"10","datelastuse":"1586587986"}, {"noda":"91.106.203.202","wallet":"151407813008139731","balance":"16955544","walletsuse":"2","datelastuse":"1586531746"}, {"noda":"91.106.206.144","wallet":"125811814158141411","balance":"20661642","walletsuse":"3","datelastuse":"1586086496"}]

**\* 4.** Запрос [type=**history**] - история транзакций

Использует дополнительный необязательные параметры:

* all - при его задании =3 - отображаются все типы истории, включая те, что еще не приняты или отклонённые, all=2 - отображаются ещё не принятые транзакции и принятые, all=1 – отображаются только ещё не принятые транзакции, all=0 -отображаются только принятые транзакции
* history- отображает все входящие и исходящие транзакции кошелька и должен быть равен номеру кошелька
* pin - любое число до 18 цифр (для определения транзакции) Смотрится чёткое соответствие знаком равенства
* wallet - номер кошелька по которому смотрятся только исходящие транзакции
* recipient - номер кошелька по которому смотрятся входящие транзакции
* date - дата от которой смотрятся транзакции
* height - высота от которой смотрятся транзакции
* noda — нода которую использовал кошелёк в последний раз
* order - параметр сортировки вывода истории в прямом порядке от меньшей даты к большей с единственным возможным параметром =asc, по умолчанию в обратном порядке
* history\_exception - параметр для исключения вывода истории с массивом кошельков и высот для исключенных транзакций. Используется для синхронизации транзакций между нодами
* start - с какой транзакции в выдаче начать отображать. Здесь и далее в других типах с множественным отображением: при =0 начинаем без пропусков, при =1 сдвигаем выдачу на 1 и так далее.
* limit - количество отображенных транзакций от параметра start. По умолчанию здесь и далее в других типах с множественным отображением =100, но при history по умолчанию =25.

По умолчанию сортировка идёт в обратном порядке от большей даты изменения к меньшей.

**Пример GET запроса**: http://[IP ноды]/egold.php?type=history&wallet=101245147116351512&height=2&start=0&limit=2&order=asc

**Пример JSON ответа** (здесь и далее пропущены ответы "signpub" и "sign"): [{"wallet":"101245147116351512", "recipient":"1", "money":"1", "height":"2", "nodawallet":"100000000000000000", "nodause":"91.106.203.179", "nodaown":"0", "date":"1562182374", "signpubreg":"", "signreg":"", "signpubnew":"", "signnew":"", "signpub":"", "sign":""},{"wallet":"101245147116351512", "recipient":"1", "money":"1", "height":"3", "nodawallet":"101245147116351512", "nodause":"91.106.206.144", "nodaown":"1", "date":"1562182398", "signpubreg":"", "signreg":"", "signpubnew":"", "signnew":"", "signpub":"", "sign":"","checkhistory":"0"}]

*\* Если запрос без параметра* ***all=3*** *с параметром* ***history равным кошельку****, то в конце массива выводится число, означающее общее количество записей в истории по данному запросу без разбивки по страницам. Сделано для того, чтобы знать сколько всего страниц с запросами.*

**Пример GET запроса**: http://[IP ноды]/egold.php?type=history&history=101245147116351512

**Пример JSON ответа** (здесь и далее пропущены ответы "signpub" и "sign"): [{"wallet":"101245147116351512", "recipient":"1", "money":"1", "height":"2", "nodawallet":"100000000000000000", "nodause":"91.106.203.179", "nodaown":"0", "date":"1562182374", "signpubreg":"", "signreg":"", "signpubnew":"", "signnew":"", "signpub":"", "sign":""},{"wallet":"101245147116351512", "recipient":"1", "money":"1", "height":"3", "nodawallet":"101245147116351512", "nodause":"91.106.206.144", "nodaown":"1", "date":"1562182398", "signpubreg":"", "signreg":"", "signpubnew":"", "signnew":"", "signpub":"", "sign":"","checkhistory":"1"},10]

**\* 5.** Запрос [type=**referrals**] - история транзакций по рефералам

Использует дополнительный необязательные параметры:

* wallet - номер кошелька который совершил транзакцию и кому он зачислил реферальные бонусы
* ref - номер кошелька по которому смотрятся все его рефералы всех уровней
* ref1 - номер кошелька по которому смотрятся все его рефералы 1-ого уровня, если нет параметра ref
* ref2 - номер кошелька по которому смотрятся все его рефералы 2-ого уровня, если нет параметра ref
* ref3 - номер кошелька по которому смотрятся все его рефералы 3-его уровня, если нет параметра ref
* height - высота от которой смотрятся транзакции
* date - дата от которой смотрятся транзакции
* order - параметр сортировки вывода истории в прямом порядке от меньшей даты к большей по дате последней транзакции с единственным возможным параметром =asc, по умолчанию в обратном порядке
* start - с какой транзакции в выдаче начать отображать.
* limit - количество отображенных транзакций от параметра start.

По умолчанию сортировка идёт в обратном порядке от большей даты изменения к меньшей.   
  
*\* В конце массива выводится число, означающее общее количество записей в истории реферальных начислений по данному запросу без разбивки по страницам. Сделано для того, чтобы знать сколько всего страниц с запросами.*

**Пример GET запроса**: http://[IP ноды]/egold.php?type=referrals&wallet=126121479106151551&height=2&start=0&limit=2&order=asc

**Пример JSON ответа**: [{"wallet":"126121479106151551","ref1":"461311305111001414","ref2":"151407813008139731","ref3":"100000000000000000","money1":"201214","money2":"100607","money3":"50303","height":"22","date":"1562940727"},10]

**\* 6.** Запрос [type=**referralwallets**] - вывод списка кошельков рефералов у данного кошелька

Использует дополнительный необязательные параметры:

* wallet - отображение кошелька и его рефералов
* ref - номер кошелька по которому смотрятся все его рефералы всех уровней
* ref1 - номер кошелька по которому смотрятся все его рефералы 1-ого уровня, если нет параметра ref
* ref2 - номер кошелька по которому смотрятся все его рефералы 2-ого уровня, если нет параметра ref
* ref3 - номер кошелька по которому смотрятся все его рефералы 3-его уровня, если нет параметра ref
* height - высота от которой смотрятся кошельки по последней транзакции
* date - дата от которой смотрятся кошельки по последней транзакции
* order - параметр сортировки вывода истории в прямом порядке от меньшей даты к большей - возможные параметры =asc - по возрастанию даты, =balanceasc по возрастанию баланс и убывания даты, =balancedesc по убыванию баланса и убывания даты, без параметров сортировка по дате в обратном направлении. Сортировка баланса без учёта начисления процентов
* start - с какой транзакции в выдаче начать отображать.
* limit - количество отображенных транзакций от параметра start.

По умолчанию сортировка идёт в обратном порядке от большей даты изменения к меньшей. Баланс отображается без учёта начисленных процентов

*\* Баланс рефералов выводится без учёта процента от последней транзакции до текущего времени*

*\* В конце массива выводится число, означающее общее количество записей в истории реферальных начислений по данному запросу без разбивки по страницам. Сделано для того, чтобы знать сколько всего страниц с запросами.*

**Пример GET запроса**: http://[IP ноды]/egold.php?type=referralwallets&ref=100000000000000000&height=2&start=0&limit=2&order=asc

**Пример JSON ответа**: [{"wallet":"100000000000000000","ref1":"0","ref2":"0","ref3":"0","noda":"91.106.203.179","nodause":"91.106.203.179","balance":"11324951","date":"1567159965","height":"318"},{"wallet":"271112141960136131","ref1":"100000000000000000","ref2":"0","ref3":"0","noda":"","nodause":"91.106.203.181","balance":"49760991","date":"1567159965","height":"1339"},10]

**\* 7.** Запрос [type=**referralwallets**] - вывод общего количества и объёма монет по уровню рефералов

Обязательно должен быть использован один из параметров:

* ref - номер кошелька по которому смотрятся все его рефералы всех уровней
* ref1 - номер кошелька по которому смотрятся все его рефералы 1-ого уровня, если нет параметра ref
* ref2 - номер кошелька по которому смотрятся все его рефералы 2-ого уровня, если нет параметра ref
* ref3 - номер кошелька по которому смотрятся все его рефералы 3-его уровня, если нет параметра ref

*\* Баланс рефералов выводится без учёта процента от последней транзакции до текущего времени*

**Пример GET запроса**: http://[IP ноды]/egold.php?type=referralresults&ref=461311305111001414

**Пример JSON ответа**: {"count1":"1","balance1":"13969582","count2":"6","balance2":"413713","count3":"101","balance3":"300124234"}

**\* 8.** Запрос [type=**height]** - получение высоты кошелька и возможности отправки транзакции. Служит для вспомогательного запроса перед транзакцией

Используется дополнительный параметр: wallet - кошелёк

**Пример GET запроса**: http://[IP ноды]/egold.php?type=height&wallet=158011521111537971

**Пример JSON ответа**: {"time":"1567159024","noda":"91.106.203.179","balance":99901,"height":"1","date":"1567158999"}

**\* 9.** Запрос [type=**send**] - осуществление транзакции при которой может быть смена публичного и закрытого ключа, смена пароля для управления контактами и E-mail уведомлениями, зарегистрирован новый кошелёк

Использует дополнительный обязательные параметры:

* wallet - кошелёк с которого переводим средства
* recipient - кошелёк получатель на который переводим средства
* money - сумма перевода без учёта комиссии в 2 монеты
* pin - любое число до 18 цифр (для определения транзакции). По умолчанию должен быть равен 0
* height - высота данной транзакции по кошельку
* signpub - открытый ключ переводящего средства кошелька
* sign - подпись транзакции закрытым ключом переводящего средства кошелька формируется из строки: wallet + recipient + money + pin + height + nodause + signpubreg + signreg + signpubnew + signnew. Где **«+»** это сложение строк. Signreg = '30' + sha\_dec(signpubreg) (для регистрации отправляется 3 монеты и высота нового кошелька 0), signpubnew= sha\_dec(signpubnew), signnew= wallet + height. sha\_dec - функция в javascript. Signpubreg, signreg, signpubnew и signnew могут быть пустыми значениями.

Использует дополнительный необязательные параметры:

* signpubnew - хэш для нового закрытого ключа
* signnew - подпись для проверки нового закрытого ключа
* signpubreg - открытый ключ для создания нового кошелька
* signreg - подпись для создания нового кошелька
* password - пароль для управления контактами и E-mail уведомлениями. С клиента получается 128-и символьный пароль. Который сохраняется в ноде как есть и в дальнейшем проверяется его 256-и символьный хэш. Подробнее в пункте 2 по запросу [type=**wallet**]

**Пример GET запроса**: http://[IP ноды]/egold.php?type=send&wallet=100000000000000000&recipient=126121479106151551&money=5&pin=0&height=2&signpub=xxx&sign=xxx

**Пример JSON ответа**: {"time":"1567160163","noda":"91.106.203.179","date":"1567159991","send":"true","recipient":"G-1261-21479-1061-51551"}

**\* 10.** Запрос [type=**synch**] - синхронизации ноды

Не использует никаких дополнительных параметров и устанавливается в **CRON** с **ежеминутным** обращением. Рекомендуется в CRON устанавливать обращение через PHP запрос в виде «/usr/local/bin/php ~/[путь до файла egold.php]/egold.php synch» Где **synch** - это передаваемый параметр **type** для синхронизации, при этом PHP запрос должен быть от версии не ниже 7.1. Также можно использовать обращение вида «http://[IP ноды]/egold.php?type=synch»

**\* 11.** Запрос [type=**synchwallets]** - обращения к синхронизируемой ноде

Использует дополнительный необязательный параметр только в виде POST запроса: wallets - массив кошельков по которым нужно отобразить данные ноды для синхронизации.

Сортировка идёт в прямом порядке от меньшей даты к большей и отображается до 100 кошельков с приоритетом на кошельки перечисленные в массиве.

**Пример POST запроса с параметром [wallets=100000000000000000]**: http://[IP ноды]/egold.php?type=synchwallets

**Пример JSON ответа**: {"time":"1567160232","noda":"91.106.203.179","synchwallets"

:{"100000000000000000":{"wallet":"100000000000000000","ref1":"310121260159274912","ref2":"0","ref3":"0","noda":"","nodause":"","balance":"3","height":"0","date":"1560612600","signpubnew":"","signnew":"","signpub":"","sign":""},"413712101472131111":{"wallet":"413712101472131111","ref1":"126121479106151551","ref2":"461311305111001414","ref3":"151407813008139731","noda":"","nodause":"","balance":"3","height":"0","date":"1560917136","signpubnew":"","signnew":"","signpub":"","sign":""}}}

**\* 12.** Запрос [type=**contacts]** - отображение контактов кошелька на ноде с которой он совершает транзакции

Для использования этого запроса сначала необходимо совершить транзакцию на ноде вместе с заданием пароля для управления настройками на этой ноде. Контакты будут храниться и отображаться только при запросам к этой ноде. Максимально можно хранить 100 контактов

Использует дополнительный обязательные параметры:

* wallet - кошелёк по которому смотрятся контакты
* password - пароль для доступа к данным кошелька, сохраненным на ноде сравнивается с присланным, используя функцию gen\_sha3([пароль на ноде],256). То есть параметр пароля **password** подхэшем SHAKE256 длинной 256 символов (1024 бита).С клиента получается 128-и символьный пароль. Который сохраняется в ноде как есть и в дальнейшем проверяется его 256-и символьный хэш. Подробнее в пункте 2 по запросу [type=**wallet**] и пункте 8по запросу [type=**send**]

Использует дополнительный необязательные параметры:

* contacts - массив контактов передаваемых ноде только по средствам POST запроса. Если этот параметр не передать, тогда отображается список уже имеющихся контактов. Если передан пустым, то все контакты удаляются. На сервер передаются параметры **recipient** и **name**. Фактически, они могут быть любыми строками с символами букв английского языка и цифрами. Длина должна быть не более 255 символов. Параметры для отправки соответствуют параметрам в **JSON** ответе для отображения контактов. Пример **JSON** ниже.

**Пример JSON ответа по сохранению и изменению контактов**: {"time":"1567160321","noda":"91.106.203.179","contact":"save"}

**Пример JSON ответа по удалению контактов (если отправлен запрос без контактов)**: {"time":"1567160493","noda":"91.106.203.179","contact":"del"}

**Пример JSON ответа по отображению контактов**: [{"recipient":"200000000000000000","name":"name2"},{"recipient":"300000000000000000","name":"name3"}]

**\* 13.** Запрос [type=**email**] - E-mail уведомления по транзакциям кошелька на ноде с которой он совершает транзакции

Для использования этого запроса сначала необходимо совершить транзакцию на ноде вместе с задание пароля для управления настройками на этой ноде. Только ноды со своим прописанным в файле настроек доменом для почты могут отправлять E-mail уведомления. Об этом можно узнать при обращении к ноде с пустым запросом http://[IP ноды]/egold.php или из списка доступных нод http://[IP ноды]/egold.php?type=nodas по параметру "email":"true"

Использует дополнительный обязательные параметры:

* wallet - кошелёк с которого переводим средства
* password - пароль для доступа к данным кошелька, сохраненным на ноде сравнивается с присланным, используя функцию gen\_sha3([пароль на ноде],256). То есть параметр пароля **password** подхэшем SHAKE256 длинной 256 символов (1024 бита).С клиента получается 128-и символьный пароль. Который сохраняется в ноде как есть и в дальнейшем проверяется его 256-и символьный хэш. Подробнее в пункте 2 по запросу [type=**wallet**] и пункте 8по запросу [type=**send**]

Использует дополнительные необязательные параметры:

* email - почта на которую будут отправляться уведомления. Если не будет этого параметра при присутствии почты на ноде, она будет удалена и все остальные параметры очищены.
* up - целое число указывающее на минимальный порог по сумме исходящих транзакций при котором будут отправлены уведомления
* down - целое число указывающее на минимальный порог по сумме входящих транзакций при котором будут отправлены уведомления

**Пример JSON ответа по сохранению почты и параметров**: {"time":"1567160588","noda":"91.106.203.179","emailwallet":"save"}

**Пример JSON ответа по удалению почты и параметров**: {"time":"1567160606","noda":"91.106.203.179","emailwallet":"del"}

**\* 14.** Запрос [type=**balanceall**] - общий баланс всех кошельков без учёта начисленных процентов

*\* Баланс всех кошельков выводится без учёта процента от последней транзакции до текущего времени*

**Пример GET запроса**: http://[IP ноды]/egold.php?type=balanceall

**Пример JSON ответа**:

{"time":"1583326557","noda":"91.106.203.179","balanceall":"18446744074137928231"}

**\* 15.** Запрос [type=**walletscount**] - общее количество кошельков

**Пример GET запроса**: http://[IP ноды]/egold.php?type=walletscount

**Пример JSON ответа**:

{"time":"1583326557","noda":"91.106.203.179","walletscount":"234"}

**\* 16. Пример совершения транзакции на PHP**

<?php

include './egold\_crypto/falcon.php';//подключаем шифрование

function egold\_send($params){

$url = 'http://[IP ноды которую будем использовать]/egold.php';

$ch = curl\_init();

curl\_setopt($ch, CURLOPT\_URL, $url);

curl\_setopt ($ch, CURLOPT\_RETURNTRANSFER, 1);

curl\_setopt($ch, CURLOPT\_POST, 1);

curl\_setopt($ch, CURLOPT\_POSTFIELDS, $params);

curl\_setopt($ch, CURLOPT\_TIMEOUT\_MS, 15\*1000);

$json = curl\_exec ($ch);

curl\_close ($ch);

return json\_decode($json, true);

}

$falcon\_k= '[Закрытый ключ]';//закрытый ключ уже зарегистрированного кошелька

$falcon\_p= Falcon\createPublicKey($falcon\_k);//создание публичного ключа

$str\_s= [кошелёк с которого отправляем].[кошелёк на который отправляем].[сумма - 2 монеты комиссии].[пин - любые цифры в количестве от 1 до 18 символом].[текущая высота с которой отправляем +1].[ IP ноды которую будем использовать];//строка которую подписываем

$falcon\_s= Falcon\sign($falcon\_k, $str\_s);//подпись строки закрытым ключом

$params = array(//POST параметры для отправки

'type' => 'send',

'wallet' => [кошелёк с которого отправляем],

'recipient' => [кошелёк на который отправляем],

'money' => [сумма - 2 монеты комиссии],

'pin' => [пин число от 0],

'height' => [текущая высота с которой отправляем +1],

'signpub' => $falcon\_p,

'sign' => $falcon\_s

);

$json\_send= egold\_send($params);

if(isset($json\_send['send']) && $json\_send['send']=='true'){/\*транзакция прошла\*/}

?>

**\* 17. Пример создания кошелька на PHP**

<?php

include './egold\_crypto/falcon.php';//подключаем шифрование

function egold\_send($params){

$url = 'http://[IP ноды которую будем использовать]/egold.php';

$ch = curl\_init();

curl\_setopt($ch, CURLOPT\_URL, $url);

curl\_setopt ($ch, CURLOPT\_RETURNTRANSFER, 1);

curl\_setopt($ch, CURLOPT\_POST, 1);

curl\_setopt($ch, CURLOPT\_POSTFIELDS, $params);

curl\_setopt($ch, CURLOPT\_TIMEOUT\_MS, 15\*1000);

$json = curl\_exec ($ch);

curl\_close ($ch);

return json\_decode($json, true);

}

$falcon\_k= '[Закрытый ключ]';//закрытый ключ уже зарегистрированного кошелька

$falcon\_p= Falcon\createPublicKey($falcon\_k);//создание публичного ключа

$str\_s= [кошелёк с которого отправляем].[кошелёк на который отправляем].[сумма - 2 монеты комиссии].[пин - любые цифры в количестве от 1 до 18 символов].[текущая высота с которой отправляем +1].[ IP ноды которую будем использовать];//строка которую подписываем

$falcon\_s= Falcon\sign($falcon\_k, $str\_s);//подпись строки закрытым ключом

$params = array(//POST параметры для отправки

'type' => 'send',

'wallet' => [кошелёк с которого отправляем],

'recipient' => [кошелёк на который отправляем],

'money' => [сумма - 2 монеты комиссии],

'pin' => [пин число от 0],

'height' => [текущая высота с которой отправляем +1],

'signpub' => $falcon\_p,

'sign' => $falcon\_s

);

$json\_send= egold\_send($params);

if(isset($json\_send['send']) && $json\_send['send']=='true'){/\*транзакция прошла\*/}

function bchexdec($hex){//длинные числа

$dec = 0; $len = strlen($hex);

for ($i = 1; $i <= $len; $i++)$dec = bcadd($dec, bcmul(strval(hexdec($hex[$i - 1])), bcpow('16', strval($len - $i))));

return $dec;

}

function sha\_dec($str){return substr(bchexdec(gen\_sha3($str,19)),0,19);}//генерация хеша из 19 чисел

list($falcon\_k\_reg,$falcon\_p\_reg)= Falcon\createKeyPair(128, [случайная строка на 50 000 символов]);//создание новой пары открытого и закрытого ключа

$str\_s\_reg='30'.sha\_dec($falcon\_p\_reg);//строка для подписи закрытым ключом нового кошелька

$falcon\_s\_reg= Falcon\sign($falcon\_k\_reg, $str\_s\_reg);//подпись ключом нового кошелька

$str\_s= [кошелёк с которого отправляем].'00'.'3'.'0'.[текущая высота с которой отправляем +1].[ IP ноды которую будем использовать].$falcon\_p\_reg.$falcon\_s\_reg;//строка которую подписываем ключом уже имеющегося кошелька

$falcon\_s= Falcon\sign($falcon\_k, $str\_s);//подпись строки закрытым ключом уже имеющегося кошелька

$params = array(//POST параметры для отправки

'type' => 'send',

'wallet' => [кошелёк с которого отправляем],

'recipient' => '00',

'money' => '3',

'pin' => '0',

'height' => [текущая высота с которой отправляем +1],

'signpubreg' => $falcon\_p\_reg,

'signreg' => $falcon\_s\_reg,

'signpub' => $falcon\_p,

'sign' => $falcon\_s

);

$json\_send= egold\_send($params);

if(isset($json\_send['walletnew']) && strlen(preg\_replace("/[^0-9]/i",'',$json\_send['walletnew']))==18){/\*кошелёк создан\*/}

?>