Positive
com

GOLDMINT

GOLDMINT SMART CONTRACTS ASSESSMENT

London, 2018

Positive com

GOLDMINT

Contents

Positive
com

GOLDMINT

Goldmint Smart Contracts Audit

Positive.com experts have assessed the source code of Goldmint smart contracts.

The final version of the audited contracts can be found in the Goldmint/ gm-eth-base-smart-contracts Github repository, branch master. The version used for the final retesting is the commit 9fa48fe330f35b28546240042048dcab2ebf02f9.

The first version of the audited contracts are `GoldmintDAO.sol` and `FiatTables.sol` (later renamed to `Storage.sol`) which are in the Goldmint/GoldminEthereum Github repository, branch `new-features`. The version used for the initial report is the commit c82dadb8e6106b5bf71d71754738aad171902560.

The goal of the audit is to find vulnerabilities, logical flaws and other code errors.

Summary

The contracts `GoldmintDAO.sol` and `Storage.sol` use a common pattern to separate business logic and storage which makes these contracts upgradeable.

Three medium severity and one low level severity issues were found. Best practices aiming to achieve a higher quality of code were recommended. No High severity issues were identified.

During the first retest of the updated contracts, out of the four (4) reported issues, two (2) of them have been FIXED while two (2) of them were still OPEN. Two (2) MEDIUM severity issues remained OPEN. One new MEDIUM severity issue has been introduced.

During the second retest **all** identified issues were found to be FIXED. No further issues were identified.

Positive com

GOLDMINT

Second Retesting

This section includes the second retesting of the audited smart contracts.

The updated version of the audited contracts can be found in the Goldmint/ gm-eth-base-smart-contracts Github repository, branch master. The version used for this retesting is the commit 9fa48fe330f35b28546240042048dcab2ebf02f9.

All reported issues have been fixed.

#	Issue Name	Severity	Status	Comment
1	Missing input validation in `setMntpMigrated()` and `setGoldMigrated()`	Medium	FIXED	A require() statement has been introduced to avoid the case where a request for setting a non-existent address overwrites the first entry of the array.
2	Illegal state transition	Medium	FIXED	A security check is now in place
3	Signed integer overflow on `addFiatTransaction()`	Medium	FIXED	A safeAdd and safeSub function has been introduced to avoid the overflow.
4	Signed integer overflow on `addGoldTransaction()`	Medium	FIXED	A safeAdd and safeSub function has been introduced to avoid the overflow.
5	Unsafe usage of tx.origin	Low	FIXED	The use of tx.origin has been replaced with msg.sender

Positive com

GOLDMINT

Retesting

This section includes the retesting of the audited smart contracts.

The `GoldmintDAO.sol` contract has been renamed to `Goldmint.sol`. The `FiatTables.sol` contract has been renamed to `Storage.sol`.

The updated version of the audited contracts can be found in the Goldmint/GoldminEthereum Github repository, branch master. The version used for this retesting is the commit 895c8fc1cf92fc83207117d2ddac2b2b575858c2.

Out of the four (4) reported issues, two (2) of them have been FIXED while two (2) of them are still OPEN. Two (2) MEDIUM severity issues remain OPEN.

One new MEDIUM severity issue has been introduced.

Medium Severity

Signed integer overflow

In the contract Storage.sol there is no operands validation in `addGoldTransaction()` when adding `_amount` to `goldBalances[_userId]`. This might result in an integer overflow.

Table 1 Retesting table

#	Issue Name	Severity	Status	Comment
1	Missing input validation in `setMntpMigrated()` and `setGoldMigrated()`	Medium	OPEN	The affected code has not been modified
2	Illegal state transition	Medium	FIXED	A security check is now in place
3	Signed integer overflow	Medium	OPEN	The affected code has not been modified
4	Unsafe usage of tx.origin	Low	FIXED	The use of tx.origin has been replaced with msg.sender

Positive com

GOLDMINT

Issue Breakdown

High Severity

No high severity issues were found.

Medium Severity

Missing input validation in `setMntpMigrated()` and `setGoldMigrated()`

There is no validation that `index` is not null in `setMntpMigrated()` and `setGoldMigrated()` after looking up user supplied `_who` argument in `mntpMigrationIndexes` mapping. If there is no such address in `mntpMigrationIndexes` mapping, the first element in `mntpMigrations` and `goldMigrations` will be overwritten.

Illegal state transition

It is possible to start migration again when it has been finished, consider adding a check in `startMigration()` to avoid such case.

Signed integer overflow

There is no operands validation in `addFiatTransaction()` when adding `_amountCents` to `fiatBalancesCents[_userId]`, which might result in an integer overflow.

Low Severity

Unsafe usage of `tx.origin`

Consider passing `msg.sender` as an argument to `addBuyTokensRequest()` and `addSellTokensRequest()` instead of using `tx.origin`, as with `tx.origin` it is possible to impersonate the user if he interacts with a malicious contract. The use of `tx.origin` is generally considered as a bad security practice.

Notes & Additional Information

This section includes best practice information and various recommendations which if followed will increase the quality of code. Issues listed in this section do not pose a security threat.

GoldmintDAO.sol

- Argument `_isMigrationStarted` is not used in `calculateFee()` function, consider removing it
- Consider assigning `creator` to `msg.sender` in `CreatorEnabled`'s constructor to avoid setting creator in each inherited contract
- The fallback function with a revert call is not needed, as Solidity contracts will by default not accept payments
- ERC20 `decimals` field should be `uint8` instead of `uint` as per ERC20 standard
- Consider adding `increaseApproval` and `decreaseApproval` functions that implement atomic modification of `allowance` with a single call instead of two
- Mapping `balances` does not have its scope defined, consider making it `public` explicitly
- Mapping `allowed` does not have its scope defined, consider making it `internal` explicitly
- Consider checking that `_to` argument is not equal to `0x0` in `transfer()`, `transferFrom()`, `transferRewardWithoutFee()`, `RescueAllRewards()` to avoid accidental transfers
- Consider declaring `_day` argument in `calculateMyRewardDecreased()` as `uint8` instead of `uint` since it cannot be greater than 365

FiatTables.sol

- Consider defining `getGoldmintFeeAccount()` as `constant` since it does not modify contract storage
- Consider explicitly declaring scopes for `changeController()`, `changeFiatFeeController()`, `addSellTokensRequest()` as `public`
- Consider replacing the use of `assert()` with `require()` when checking user input. The `FiatTables()` function uses the `assert()` function to validate the provided user input. The `require()` function is much more forgiving than `assert()` and will not consume the remaining gas of the transaction. `Assert()` should only be used as a fail-safe and should never be reachable by the normal flow of the application. As a rule, if an assertion is triggered it should reflect a bug in the code.
- Consider implementing state machine for request processing in `FiatTables` contract using the pattern described in the official Solidity documentation. E.g. as for now it is not clear whether it is legit to finish a cancelled request
- `cancelRequest()` function always returns a null value, consider removing “return” statement
- Consider adding safe multiplication and division functions in the `calculateSellGoldFee()`, `processBuyRequest()`, `processSellRequest()` functions.
- Consider avoiding casting unsigned integer values to integers as this may result to wrap around issues. Similar to an overflow it may happen when casting to a similar but different variable type, in this case from unsigned integer to integer. An example of this behavior can be found in FiatTables.sol. Even though the function may not proceed with a negative value of `amount`, it is definitely possible for an excessively large `_amountCents` value to wrap around to a negative one. If it is a business requirement to perform type casting from unsigned integer to integer it is advised to first verify that the unsigned integer value does not exceed the maximum `int256` value.

Positive com

GOLDMINT

Appendix A – Contract Profiling

GoldmintDAO.sol

Contract	Function	Visibility	Constant	Returns	Modifiers
SafeMath	safeMul(uint,uint)	internal	false	uint	
SafeMath	safeSub(uint,uint)	internal	false	uint	
SafeMath	safeAdd(uint,uint)	internal	false	uint	
CreatorEnabled	changeCreator(address)	public	false		onlyCreator
StdToken	transfer(address,uint256)	public	false	bool	onlyPayloadSize
StdToken	transferFrom(address,address,uint256)	public	false	bool	
StdToken	balanceOf(address)	public	true	uint256	
StdToken	approve(address,uint256)	public	false	bool	
StdToken	allowance(address,address)	public	true	uint256	
GoldFee	GoldFee()	public	false		
GoldFee	getMin(uint)	public	false	uint	
GoldFee	getMax(uint)	public	false	uint	
GoldFee	calculateFee(bool,bool,uint,uint)	public	true	uint	

Contract	Function	Visibility	Constant	Returns	Modifiers
Gold	Gold(address,address,address)	public	false		
Gold	setControllerContractAddress(address)	public	false		onlyCreator
Gold	setMigrationContractAddress(address)	public	false		onlyCreator
Gold	setGoldmintTeamAddress(address)	public	false		onlyCreator
Gold	setGoldFeeAddress(address)	public	false		onlyCreator
Gold	issueTokens(address,uint)	public	false		onlyCreatorOrController
Gold	burnTokens(address,uint)	public	false		onlyMigrationOrController
Gold	startMigration()	public	false		onlyMigration
Gold	finishMigration()	public	false		onlyMigration
Gold	lockTransfer(bool)	public	false		onlyMigration
Gold	transfer(address,uint256)	public	false	bool	onlyIfUnlocked,onlyPayloadSize
Gold	transferFrom(address,address,uint256)	public	false	bool	onlyIfUnlocked
Gold	transferRewardWithoutFee(address,uint)	public	false		onlyMigration,onlyPayloadSize
Gold	rescueAllRewards(address)	public	false		onlyCreator

Contract	Function	Visibility	Constant	Returns	Modifiers
GoldmintMigration	getMntpMigration(uint)	public	true	address,string,uint,bool,uint64,string	
GoldmintMigration	getGoldMigration(uint)	public	true	address,string,uint,bool,uint64,string	
GoldmintMigration	GoldmintMigration(address,address)	public	false		
GoldmintMigration	lockMntpTransfers(bool)	public	false		onlyCreator
GoldmintMigration	lockGoldTransfers(bool)	public	false		onlyCreator
GoldmintMigration	startMigration()	public	false		onlyCreator
GoldmintMigration	pauseMigration()	public	false		onlyCreator
GoldmintMigration	finishMigration()	public	false		onlyCreator
GoldmintMigration	destroyMe()	public	false		onlyCreator
GoldmintMigration	migrateMntp(string)	public	false		
GoldmintMigration	isMntpMigrated(address)	public	true	bool	
GoldmintMigration	setMntpMigrated(address,bool,string)	public	false		onlyCreator

Contract	Function	Visibility	Constant	Returns	Modifiers
GoldmintMigration	migrateGold(string)	public	false		
GoldmintMigration	isGoldMigrated(address)	public	true	bool	
GoldmintMigration	setGoldMigrated(address,bool,string)	public	false		onlyCreator
GoldmintMigration	calculateMyRewardMax(address)	public	true	uint	
GoldmintMigration	calculateMyRewardDecreased(uint,uint)	public	true	uint	
GoldmintMigration	calculateMyReward(uint)	public	true	uint	
GoldmintMigration	()	external	false		

FiatTables.sol

Contract	Function	Visibility	Constant	Returns	Modifiers
SafeMath	safeAdd(uint,uint)	internal	false	uint	
CreatorEnabled	changeCreator(address)	public	false		onlyCreator
StringMover	stringToBytes32(string)	public	true	bytes32	
StringMover	stringToBytes64(string)	public	true	bytes32,bytes32	

Contract	Function	Visibility	Constant	Returns	Modifiers
StringMover	bytes32ToString(bytes32)	public	true	string	
StringMover	bytes64ToString(bytes32,bytes32)	public	true	string	
Storage	Storage()	public	false		
Storage	setControllerAddress(address)	public	false		onlyController
Storage	addDoc(string)	public	false	uint	onlyController
Storage	getDocCount()	public	true	uint	
Storage	getDocAsBytes64(uint)	public	true	bytes32,bytes32	
Storage	addFiatTransaction(string,int)	public	false	uint	onlyController
Storage	getFiatTransactionsCount(string)	public	true	uint	
Storage	getAllFiatTransactionsCount()	public	true	uint	
Storage	getFiatTransaction(string,uint)	public	true	int	
Storage	getUserFiatBalance(string)	public	true	int	
Storage	addBuyTokensRequest(string,string)	public	false	uint	onlyController

Contract	Function	Visibility	Constant	Returns	Modifiers
Storage	addSellTokensRequest(string,string)	public	false	uint	onlyController
Storage	getRequestsCount()	public	true	uint	
Storage	getRequest(uint)	public	true	a,userId,hashA,hashB,buy,state	
Storage	cancelRequest(uint)	public	false		onlyController
Storage	setRequestProcessed(uint)	public	false		onlyController
GoldFiatFee	GoldFiatFee(string)	public	false		
GoldFiatFee	getGoldmintFeeAccount()	public	false	bytes32	
GoldFiatFee	setGoldmintFeeAccount(string)	public	false		onlyCreator
GoldFiatFee	calculateBuyGoldFee(uint,int)	public	true	int	
GoldFiatFee	calculateSellGoldFee(uint,int)	public	true	int	
FiatTables	FiatTables(address,address,address,address)	public	false		
FiatTables	changeController(address)	public	false		onlyCreator
FiatTables	changeFiatFeeContract(address)	public	false		onlyCreator

Contract	Function	Visibility	Constant	Returns	Modifiers
FiatTables	addDoc(string)	public	false	uint	onlyCreator
FiatTables	getDocCount()	public	true	uint	
FiatTables	getDoc(uint)	public	true	string	
FiatTables	addFiatTransaction(string,int)	public	false	uint	onlyCreator
FiatTables	getFiatTransactionsCount(string)	public	true	uint	
FiatTables	getAllFiatTransactionsCount()	public	true	uint	
FiatTables	getFiatTransaction(string,uint)	public	true	int	
FiatTables	getUserFiatBalance(string)	public	true	int	
FiatTables	addBuyTokensRequest(string,string)	public	false	uint	
FiatTables	addSellTokensRequest(string,string)	public	false	uint	
FiatTables	getRequestsCount()	public	true	uint	
FiatTables	getRequest(uint)	public	true	address,string,string,bool,uint8	
FiatTables	cancelRequest(uint)	public	false		onlyCreator

Contract	Function	Visibility	Constant	Returns	Modifiers
FiatTables	processRequest(uint,uint,uint)	public	false		onlyCreator
FiatTables	processBuyRequest(string,address,int,uint)	internal	false		
FiatTables	processSellRequest(string,address,int,uint)	internal	false		
FiatTables	issueGoldTokens(address,uint)	internal	false		
FiatTables	burnGoldTokens(address,uint)	internal	false		