RWaltz
Software

Security Audit Report for BTCBAM

May 22nd, 2021

www.rwaltz.com

Overview

Project Summary

Project Name	BTCBAM
Description	BTCBAM is a Vienna-based cryptocurrency mining company managed by entrepreneurs and professionals. The company's vision is to create a platform that allows anyone, anywhere in the world, to earn through mining.
Platform	C++
Codebase	GitHub Repository

Audit Summary

Delivery Date	**May 22nd, 2021**
Method of Audit	Static Analysis, Manual Review
Consultants Engaged	1
Timeline	May 10th, 2021 - May 22nd, 2021

Vulnerability Summary

Total Issues	**44**
Error/ Medium	05
Performance/ Minor	30
Warning	09

Executive Summary

Between 10th May to 22nd May 2021, BTCBAM engaged RWaltz to perform a security audit. The objective of the audit requested by BTCBAM was to evaluate the security of BTCBAM's innovations over the Bitcoin Core source code.

As a security-focused team, made up of world-class talent, we prioritize the security of BTCBAM users. True security comes from empowering users directly, and to that end, we will always disclose vulnerabilities we find (as soon as we're certain such disclosure won't harm live users), and describe how we have fixed them or how we intend to fix them.

The BTCBAM Mining System developed a global coin called BTCBAM. They are preparing to put their mark on the sector with a total of 7 projects. We find some medium and minor severity flaws that were dealt by the BTCBAM team. RWaltz recommends performing further testing to validate extended safety and correctness in context to the whole set of code.

Scope

The BTCBAM auditing strategy tasked experts with different specializations to focus on different aspects of the system. The objective of the first phase of the audit requested RWaltz to review changes to the Bitcoin Core code, focusing on the "core consensus" pieces. The review included but was not limited to the following checks:

- Transaction validation
- New data structures
- Input validation
- JoinSplit operations
- Block header changes
- Transaction signing
- Denial of service prevention
- Integer overflows
- Cryptographic weaknesses

Folder in Scope

ID	Location
SRCA	src/*

Findings

ID	Title	Severity
SRCA-01	postfixOperator	Performance
SRCA-02	passedByValue	Performance
SRCA-03	uninitMemberVar	Warning
SRCA-04	assertWithSideEffect	Warning
SRCA-05	containerOutOfBounds	Error/ Medium
SRCA-06	containerOutOfBounds	Warning
SRCA-07	ignoredReturnValue	Warning
SRCA-08	invalidContainer	Error/ Medium
SRCA-09	iterators1	Error/ Medium
SRCA-10	nullPointerRedundantCheck	Warning
SRCA-11	preprocessorErrorDirective	Error/ Medium
SRCA-12	stlFindInsert	Performance
SRCA-13	unknownMacro	Error/ Medium
SRCA-14	useInitializationList	Performance

SRCA-01: postfixOperator

Location	Description:
addrman.h L446, arith_uint256.h L89, base58.cpp L104, core_memusage.h L22, bloom.cpp L309, miner.cpp L201, net_processing.cpp L415, policy\fees.cpp L1010, qt\guiutil.cpp L166, test\transaction_tests.c pp L89, wallet\rpcdump.cpp L752, zmq\zmqnotificationint erface.cpp L137	Prefer prefix ++/-- operators for non-primitive types.

SRCA-02: passedByValue

Location	Description:
util.h L178	Function parameter 'lockfile_name' should be passed by const reference.
httpserver.cpp L128	Function parameter '_prefix' should be passed by const reference.
net_processing.cpp L243	Function parameter 'addrNameIn' should be passed by const reference.
protocol.cpp L137	Function parameter 'ipIn' should be passed by const reference.
rest.cpp L64	Function parameter 'message' should be passed by const reference.

txmempool.cpp L24	Function parameter 'lp' should be passed by const reference.
util.cpp L389	Function parameter 'lockfile_name' should be passed by const reference.
qt\paymentrequestplus.cpp L25,	Function parameter 'err' should be passed by const reference.
rpc\server.cpp L137	Function parameter 'strName' should be passed by const reference.

SRCA-03: uninitMemberVar

Location	Description:
chainparams.h L79	Member variable 'CChainParams::chainTxData' is not initialized in the constructor.
validation.H L376	Member variable 'CScriptCheck::txdata' is not initialized in the constructor.
miner.cppl L93	Member variable 'BlockAssembler::pblock' is not initialized in the constructor.
random.cpp L456	Member variable 'FastRandomContext::bytebuf' is not initialized in the constructor.

txmempool.cpp L22	Member variable 'CTxMemPoolEntry::vTxHashesIdx' is not initialized in the constructor.

SRCA-04: assertWithSideEffect

Location	Description:
txmempool.cpp L667	Assert statement calls a function which may have desired side effects: 'end'.

SRCA-05: containerOutOfBounds

Location	Description:
compat\glibcxx_sanity.cpp L51	Out of bounds access in expression 'test.at(1)' because 'test' is empty and 'at' may be non-zero.

SRCA-06: containerOutOfBounds

Location	Description:
qt\rpcconsole.cpp L172	Either the condition 'stack.empty()' is redundant or expression 'stack.back()' cause access out of bounds.

SRCA-07: ignoredReturnValue

Location	Description:
compat\glibcxx_sanity.c	Return value of function test.at() is not used.
pp L51	

SRCA-08: invalidContainer

Location	Description:
wallet\rpcwallet.cpp L1850	Using iterator to local container 'arrTmp' that may be invalid.

SRCA-09: iterators1

Location	Description:
net_processing.cpp L692	Same iterator is used with different containers 'mapOrphanTransactions' and 'itPrev.second'.

SRCA-10: nullPointerRedundantCheck

Location	Description:
chain.h L471	Either the condition 'pindexWalk' is redundant or there is possible null pointer dereference: pindex.

SRCA-11: preprocessorErrorDirective

Location	Description:
test\arith_uint256_tests.cpp L223	Failed to expand 'CHECKBITWISEOPERATOR'

SRCA-12: stlFindInsert

Location	Description:
util.cpp L676	Searching before insertion is not necessary.

SRCA-13: unknownMacro

Location	Description:
qt\test\test_main.cpp L30	There is an unknown macro here somewhere. Configuration is required. If Q_IMPORT_PLUGIN is a macro then please configure it.

SRCA-14: useInitializationList

Location	Description:
chain.h L382	Variable 'hashPrev' is assigned in constructor body. Consider performing initialization in initialization list
merkleblock.cpp L16	Variable 'header' is assigned in constructor body. Consider performing initialization in initialization list.
miner.cpp L66	Variable 'blockMinFeeRate' is assigned in constructor body. Consider performing initialization in initialization list.
miner.cpp L72	Variable 'blockMinFeeRate' is assigned in constructor body. Consider performing initialization in initialization list.
miner.h L39	Variable 'iter' is assigned in constructor body. Consider performing initialization in initialization list.

netaddress.cpp L632	Variable 'network' is assigned in constructor body.

	Consider performing initialization in initialization list.
sync.cpp L41	Variable 'sourceFile' is assigned in constructor body. Consider performing initialization in initialization list.
wallet\rpcwallet.cpp L1391	Variable 'nAmount' is assigned in constructor body. Consider performing initialization in initialization list.

Conclusion:

RWaltz reviewed BTCBAM's innovations over the Bitcoin Core source code, focused on evaluating its resistance against specific threats to cryptocurrencies. RWaltz identified high-risk and moderate-risk issues during the assessment that affected the performance and availability of the BTCBAM network. The security issues identified did not allow remote code execution nor allowed an attacker to steal funds or compromise the privacy of BTCBAM users.

Moreover as BTCBAM source code is derived from Bitcoin Core, there are no significant changes or modifications made further in the codebase.

BTCBAM is still in a nascent stage, it is experimental and new; users are encouraged to educate themselves about the risks involved in being a direct part of the BTCBAM protocol and network.

We recommend that the BTCBAM team look for opportunities to expand documentation code references to allow for better maintenance, including for future audits and encouraging community contributions.