HashEye

// SMART CONTRACT SECURITY AUDIT

Offchain BoLD & DAC Rewards

Security assessment by HashEye · prepared for Offchain Labs

HASHEYE AUDITED

PROJECT	Offchain BoLD & DAC Rewards
CLIENT	Offchain Labs
CATEGORY	Offchain Labs
PUBLISHED	June 1, 2024
REPORT ID	research-offchain-bold-and-dac-rewards-2024-06-01-1qtmlp

This report was produced under HashEye's layered review process — **automated detection**, **pattern correlation**, and **senior manual verification** — with every finding signed off by a human reviewer. Full findings detail and on-chain attestation are available on the report page at hasheye.io/audits/research-offchain-bold-and-dac-rewards-2024-06-01-1qtmlp.

BoLDandDACRewardsUpdates SecurityAssessment August5,2024 Preparedfor: HarryKalodner,StevenGoldfeder,andEdFelten OffchainLabs Preparedby:GustavoGriecoandSimoneMonica

Abouthasheye Foundedin2012andheadquarteredinNewYork,hasheyeprovidestechnicalsecurity assessmentandadvisoryservicestosomeoftheworld'smosttargetedorganizations.We combinehigh-endsecurityresearchwithareal -worldattackermentalitytoreduceriskand fortifycode.With100+employeesaroundtheglobe,we'vehelpedsecurecriticalsoftware elementsthatsupportbillionsofendusers,includingKubernetesandtheLinuxkernel. Wemaintainanexhaustivelistofpublicationsathttps://github.com/hasheye-io/publications, withlinkstopapers,presentations,publicauditreports,andpodcastappearances. Inrecentyears,hasheyeconsultantshaveshowcasedcutting-edgeresearchthrough presentationsatCanSecWest,HCSS,Devcon,EmpireHacking,GrrCon,LangSec,NorthSec, theO'ReillySecurityConference,PyCon,REcon,SecurityBSides,andSummerCon. Wespecializeinsoftwaretestingandcodereviewprojects,supportingclientorganizations inthetechnology,defense,andfinanceindustries,aswellasgovernmententities.Notable clientsincludeHashiCorp,Google,Microsoft,WesternDigital,andZoom. hasheyealsooperatesacenterofexcellencewithregardtoblockchainsecurity.Notable projectsincludeauditsofAlgorand,BitcoinSV,Chainlink,Compound,Ethereum2.0, MakerDAO,Matic,Uniswap,Web3,andZcash. Tokeepuptodatewithourlatestnewsandannouncements,pleasefollowhasheyeon Twitterandexploreourpublicrepositoriesathttps://github.com/hasheye-io.Toengageus directly,visitour"Contact"pageathttps://www.hasheye.io/contact,oremailusat info@hasheye.io. hasheye,Inc. 497CarrollSt.,Space71,SeventhFloor Brooklyn,NY11215 https://www.hasheye.io info@hasheye.io hasheye 1OffchainLabsSecurityAssessment PUBLIC

NoticesandRemarks CopyrightandDistribution ©2024byhasheye,Inc. Allrightsreserved.hasheyeherebyassertsitsrighttobeidentifiedasthecreatorofthis reportintheUnitedKingdom. Thisreportisconsideredbyhasheyetobepublicinformation;itislicensedtoOffchain Labsunderthetermsoftheprojectstatementofworkandhasbeenmadepublicat OffchainLabs'srequest.Materialwithinthisreportmaynotbereproducedordistributedin partorinwholewithouttheexpresswrittenpermissionofhasheye. ThesolecanonicalsourceforhasheyepublicationsisthehasheyePublicationspage. Reportsaccessedthroughanysourceotherthanthatpagemayhavebeenmodifiedand shouldnotbeconsideredauthentic. TestCoverageDisclaimer Allactivitiesundertakenbyhasheyeinassociationwiththisprojectwereperformedin accordancewithastatementofworkandagreeduponprojectplan. Securityassessmentprojectsaretime-boxedandoftenreliantoninformationthatmaybe providedbyaclient,itsaffiliates,oritspartners.Asaresult,thefindingsdocumentedin thisreportshouldnotbeconsideredacomprehensivelistofsecurityissues,flaws,or defectsinthetargetsystemorcodebase. hasheyeusesautomatedtestingtechniquestorapidlytestthecontrolsandsecurity propertiesofsoftware.Thesetechniquesaugmentourmanualsecurityreviewwork,but eachhasitslimitations:forexample,atoolmaynotgeneratearandomedgecasethat violatesapropertyormaynotfullycompleteitsanalysisduringtheallottedtime.Theiruse isalsolimitedbythetimeandresourceconstraintsofaproject. hasheye 2OffchainLabsSecurityAssessment PUBLIC

TableofContents hasheye 3OffchainLabsSecurityAssessment PUBLIC

ProjectSummary ContactInformation Thefollowingprojectmanagerwasassociatedwiththisproject: MaryO'Brien,ProjectManager mary.obrien@hasheye.io Thefollowingengineeringdirectorwasassociatedwiththisproject: JosselinFeist,EngineeringDirector,Blockchain josselin.feist@hasheye.io Thefollowingconsultantswereassociatedwiththisproject: GustavoGrieco,ConsultantSimoneMonica,Consultant gustavo.grieco@hasheye.iosimone.monica@hasheye.io ProjectTimeline Thesignificanteventsandmilestonesoftheprojectarelistedbelow. DateEvent June10,2024Pre-projectkickoffcall June20,2024Deliveryofreportdraft June20,2024Reportreadoutmeeting August5,2024Deliveryofcomprehensivereport hasheye 4OffchainLabsSecurityAssessment PUBLIC

ExecutiveSummary EngagementOverview
OffchainLabsengagedhasheyetoreviewthesecurityofaseriesofchangestotheBoLD
contractandtheDACRewardscontract.Thesechangesincludefixesforprevioussecurity
assessmentsaswellascodequalityimprovements.TheDACRewardalsowasmodifiedto
supportdistributiononrewardsontheOptimismrollups.
AteamoftwoconsultantsconductedthereviewfromJune10toJune20,2024,foratotal ofthreeengineer-
weeksofeffort.Ourtestingeffortsfocusedonaselectedlistof PRs
providedbyOffchainLabs.Withfullaccesstosourcecodeanddocumentation,we
performedamanualprocesstoreviewthecodechangesaswellassomelimitedmutation
testingtoassessthequalityoftheunittests. ObservationsandImpact
Wespentthemajorityofourtimeonthe BOLDchanges,checkingwhetheranyofthem
wouldallowamaliciouspartytowinachallengebyconfirmationorpreventanhonest
partyfromwinningchallenges(e.g.,bymakingitimpossibleforanhonestpartytocontinue
participatinginachallenge).Wealsocheckedforpossiblemisconfigurationofthenew
parametersandfeatures.Wereviewedchangestothestakingpoolcontractstoassess
whetheramalicioususercouldstealtokensorwithdrawtheirtokensbeforeachallengeis finished.
Wedidnotuncoveranyseriousissuesintheimplementation.However,wefoundone
issueintheDACRewardcodeduetoagasoptimizationthatwouldmakethetransaction
revertforcertainERC20tokens(TOB-DACR-1). Recommendations
Basedonthecodebasematurityevaluationandfindingsidentifiedduringthesecurity
review,hasheyerecommendsthatOffchainLabstakethefollowingsteps: •
Remediatethefindingsuncoveredduringthisreview. •
Improvethequalityofthetestsandaddmutationtestingtothesoftware
developmentlifecycletoimprovethequalityofthetestingsuite(seeappendixB). hasheye
5OffchainLabsSecurityAssessment PUBLIC

FindingSeveritiesandCategories Thefollowingtablesprovidethenumberoffindingsbyseverityandcategory.
EXPOSUREANALYSIS SeverityCount High0 Medium1 Low0 Informational0 Undetermined0 CATEGORYBREAKDOWN
CategoryCount UndefinedBehavior1 hasheye 6OffchainLabsSecurityAssessment PUBLIC

ProjectGoals Theengagementwasscopedtoprovideasecurityassessmentoftheoffchain’sBoLDand
DACRewardscontracts.Specifically,wesoughttoanswerthefollowingnon-exhaustivelist ofquestions: •
Arethefixesforissuesidentifiedinpreviousssecurityauditssufficient? •
Havethecodechangesintroducedanysecurityorcorrectnessissues? hasheye 7OffchainLabsSecurityAssessment
PUBLIC

ProjectTargets Theengagementinvolvedareviewandtestingofthetargetslistedbelow. BoLD
Repositoryhttps://github.com/OffchainLabs/bold Version 6b42a38fc28f9b6a7864001fe797b9f07dad6357
TypeSolidity PlatformEthereum/Arbitrum DAC-REWARDS Repositoryhttps://github.com/OffchainLabs/fund-
distribution-contracts Version b390734d934e6d8b05b78e5ef38739c4b8e668b6 TypeSolidity
PlatformEthereum/Arbitrum/Optimism hasheye 8OffchainLabsSecurityAssessment PUBLIC

ProjectCoverage Thissectionprovidesanoverviewoftheanalysiscoverageofthereview,asdeterminedby
ourhigh-levelengagementgoals.OurapproachesincludedthefollowingPRs: •
#609addsaneventtotracktimercacheupdates. • #623removes EdgeConfirmedByChildren and
EdgeConfirmedByClaim events, sincetheyareunusedwithbottom-uptimers. •
#627introducesafeaturetopinawithdrawaladdressoncreatinganewstake
(similartohowbeaconchainwithdrawalcredentialworks),thusreducingtheriskof
havingahotprivatekeyinthevalidatorsoftware. •
#631fixesamainlycosmeticerrorwhereattemptstoconfirmanalready-confirmed edgebytimerevertwith
RivalEdgeConfirmed .ThisPRmakestheattemptrevert with EdgeNotPending instead. •
#617mergesthedelaybufferfeaturethatwaspreviouslyaudited. •
#637and#641introduceanoptimizationwhenastakeisslashed:sincetheloser stakeescrowmaynotbeabletocall
withdraw ,theconfiscatedstakeissentdirectly tothewinnerinsteadofaddingtoitswithdrawableamount. •
#639changesthedocumentationofthe firstRival function. •
#619introducesanewstakingpoolsimilartotheexistingassertionstakingpool. •
#628mergeschangesfromStylus. • #643changes ChallengeManager contracttoinherittheallowlistfromthe
Rollup contract,andimposesalimittoensurethateachvalidatorcancreateonlyoneroot edgepersub-challenge.
• #645removestheoldchallengemanager. • #642introducesseveralchanges: •
Theboldupgradeactioncontracthasbeenmodifiedtoupdatethe sequencerinboxwiththedelaybufferfeature. •
Theactioncontractalsoupgradestheinboxtodeprecatemethodsthat assumethat tx.origin==msg.sender
impliestheuseofEOAsender. ThisupgradepreemptsbreakingchangescausedbyfutureEthereumhard forks.
hasheye 9OffchainLabsSecurityAssessment PUBLIC

• ThisPRalsochangesthe BOLDUpgradeAction topersistthe Bridge , Outbox ,and RollupEventInbox
upgrades. • #6intheboldprivaterepositoryaddsthebetaparametersdescribedinthesection
4.8.1oftheboldpaper. • #8intheboldprivaterepositoryappliesthesamefixas#631butisconfirmedby theOSP.

• #652enforcesthatthe delayBlock parameterisconsistentwith delaySeconds to ensurethattheforceinclusionwindowdoesnotinadvertentlydecrease. • #655addressestheissue#8foundintheC4bugcontest.Itisashort-term workaroundthatpreventstheadminfromreducingtheassertionbasestake amount. • #654addressestheissue#5foundintheC4bugcontest. • #651addsmoredescriptiveeventstothesequencerinboxandrollupcontracts. • #650usesOpenZeppelin’senumerablesetforthevalidatorallowlist.Thismakesit easiertoretrievethelistofvalidatorsoff-chain. • #653introducesthefollowingchangestothestakingandwithdrawalflowtomakeit easierandsaferforvalidatorstohandlefunds: • Addsanew returnOldDepositFor function,allowingthewithdrawal addresstotriggerwithdrawals. • Addsanew newStake function,allowingvalidatorstobecomestakedwith anyamount(even0). • #659addressestheissue#2intheC4bugcontest. • #664forbidsapooltobecreatedwithanemptyedgeidorassertionhash. • #665allowsconfigurationof validatorAfkBlocks ,withtheoptiontosetitto0to disablethisfeatureentirely. •#29inthefunddistributionrepositoryallowsrewarddistributioninOptimism chains. WereviewedthiscodelookingforusualflawsinSoliditycodeaswellasanyissuesthat wouldallowamaliciouspartytowinachallengebyconfirmationorpreventanhonest partyfromwinningchallenges(e.g.,bymakingitimpossibleforanhonestpartytocontinue participatinginachallenge).Wealsocheckedforpossiblemisconfigurationofthenew parametersandfeatures.Wereviewedthestakingpoolcontractschangestocheck hasheye 10OffchainLabsSecurityAssessment PUBLIC

whetheritispossibletostealtokensandwhetheramalicioususercanwithdrawtheir tokensbeforethechallengeisfinished. CoverageLimitations Becauseofthetime-boxednatureoftestingwork,itiscommontoencountercoverage limitations.Thefollowinglistoutlinesthecoveragelimitationsoftheengagementand indicatessystemelementsthatmaywarrantfurtherreview: • WehavenotreviewedindetailhowtheBoLDcodebaseevolved.Instead,weused thediff/PRsprovidedbyOffchainLabstoboundthescopeofthisassessment. hasheye 11OffchainLabsSecurityAssessment PUBLIC

SummaryofFindings Thetablebelowsummarizesthefindingsofthereview,includingtypeandseveritydetails. IDTitleTypeSeverity 1routeTokenfunctionmayfailforcertainERC20 tokens Undefined Behavior Medium hasheye 12OffchainLabsSecurityAssessment PUBLIC

DetailedFindings 1.routeTokenfunctionmayfailforcertainERC20tokens Severity:MediumDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-DACR-1 Target: src/FeeRouter/ArbChildToParentRewardRouter.sol , src/FeeRouter/OpChildToParentRewardRouter.sol , src/FeeRouter/ParentToChildRewardRouter.sol Description The routeToken functioncanrevertforERC20tokensthatrequiretheallowancetobeset to0whencallingthe approve function,suchasUSDT. Thefunctionhasanoptimizationwhereitapproves amount+1 tothegateway,whichwill transfertheamountoftokensinafollowingcall.Theideaistoneverchangetheallowance storageslotfromanon-zerotoazerovalueandsavinggas.However,tokenssuchasUSDT requirethecurrentallowancetobe0whenausercallsthe approve function;inthiscase, thismakesthe routeToken functionrevert. functionrouteToken(addressparentChainTokenAddr,uint256maxSubmissionCost,uint256 gasLimit,uint256maxFeePerGas) public payable { ... //approveamountongateway,adding1sostorageslotdoesn'tgetsetto0, savinggas. IERC20(parentChainTokenAddr).approve(gateway,amount+1); ... } Figure1.1:Snippetofthe routeToken function (src/FeeRouter/ParentToChildRewardRouter.sol#L162-L163) ExploitScenario The ParentToChildRewardRouter contractisdeployedwith parentChainTokenAddr settotheUSDTaddress.Ausercallsthe routeToken function,butitunexpectedlyreverts. hasheye 13OffchainLabsSecurityAssessment PUBLIC

Recommendations Shortterm,removethisoptimizationorusethe forceApprove functioninthe SafeERC20 libraryafterverifyingthatthisfunctionwouldstillsavegas. Longterm,whendevelopingacontractthatinteractswithERC20tokens,considerall possibledifferentimplementationsandwhichimplementationsyourcontractshould support. hasheye 14OffchainLabsSecurityAssessment PUBLIC

A.VulnerabilityCategories Thefollowingtablesdescribethevulnerabilitycategories,severitylevels,anddifficulty levelsusedinthisdocument. VulnerabilityCategories CategoryDescription AccessControlsInsufficientauthorizationorassessmentofrights AuditingandLoggingInsufficientauditingofactionsorloggingofproblems AuthenticationImproperidentificationofusers ConfigurationMisconfiguredservers,devices,orsoftwarecomponents

CryptographyAbreachofsystemconfidentialityorintegrity DataExposureExposureofsensitiveinformation DataValidationImproperrelianceonthestructureorvaluesofdata DenialofServiceAsystemfailurewithanavailabilityimpact ErrorReportingInsecureorinsufficientreportingoferrorconditions PatchingUseofanoutdatedsoftwarepackageorlibrary SessionManagementImproperidentificationofauthenticatedusers TestingInsufficienttestmethodologyortestcoverage TimingRaceconditionsorotherorder-of-operationsflaws UndefinedBehaviorUndefinedbehaviortriggeredwithinthesystem

SeverityLevels SeverityDescription InformationalTheissuedoesnotposeanimmediateriskbutisrelevanttosecuritybest practices. UndeterminedTheextentoftheriskwasnotdeterminedduringthisengagement. LowTheriskissmallorisnotonetheclienthasindicatedisimportant. MediumUserinformationisatrisk;exploitationcouldposereputational,legal,or moderatefinancialrisks. HighTheflawcouldaffectnumeroususersandhaveseriousreputational,legal, orfinancialimplications. DifficultyLevels DifficultyDescription UndeterminedThedifficultyofexploitationwasnotdeterminedduringthisengagement. LowTheflawiswellknown;publictoolsforitsexploitationexistorcanbe scripted. MediumAnattackermustwriteanexploitorwillneedin-depthknowledgeofthe system. HighAnattackermusthaveprivilegedaccesstothesystem,mayneedtoknow complextechnicaldetails,ormustdiscoverotherweaknessestoexploitthis issue.

B.MutationTesting Duringourreview,werantwomutationtestingtools,Necessistand slither-mutate ,to identifypotentialshortcomingsinthetestsuiteandimplementation. TouseNecessist,runthefollowingcommand: necessist--frameworkfoundry test/challengeV2/EdgeChallengeManager.t.sol Touse slither-mutate ,runthefollowingcommand: slither-mutatesrc/challengeV2/EdgeChallengeManager.sol --test-cmd='forgetest' Thesamecommandscanbeusedforothercontracts. NotethatduetoaSolidityissueregardingunicodecharacters#14733, slither-mutate willuseanincorrectsourcemappingforthemutation;asaresult,anyunicodecharacters shouldberemoved.The EdgeChallengeManager.sol includestheunicodecharactersβ and□inthecommentsonlines86,94,and103. WetestedthefollowingcontractsL ● Necessist ○ ChallengeEdgeLib.t.sol ○ EdgeChallengeManager.t.sol ○ EdgeChallengeManagerLib.t.sol ○ Rollup.t.sol ● Slither-mutate ○ AssertionStakingPool.sol ○ EdgeChallengeManager.sol ○ EdgeChallengeManagerLib.sol ○ RollupCore.sol ○ RollupUserLogic.sol ○ RollupAdminLogic.sol Afterreviewingtheresults,weidentifiedmissingtestcoverageforthecasesdescribedin thefollowingsections. 1.Timercacheisnevertestedforacachehit The updateTimerCache functionisnevertestedforacachehitinthehighlightedline. functionupdateTimerCache(EdgeStorestoragestore,bytes32edgeId,uint256newValue,

uint256maximumCachedTime) internal returns(bool,uint256) { uint256currentAccuTimer=validateCurrentTimer(store,edgeId, maximumCachedTime); newValue=newValue>type(uint64).max?type(uint64).max:newValue; //onlyupdatewhenincreased if(newValue>currentAccuTimer){ store.edges[edgeId].totalTimeUnrivaledCache=uint64(newValue); return(true,newValue); } return(false,currentAccuTimer); } FigureB.1:The updateTimerCache function (src/challengeV2/libraries/EdgeChallengeManagerLib.sol#L516-L528) Additionally,thebranchwhere newValue>type(uint64).max isnevertested. 2.Failuretocheckreturnedvalueofcurrenttimer Relatedtothepreviousfinding,the updateTimerCache functionisnotproperlytested,as thecurrenttimercanbereducedtozerowithoutproducinganyfailureinthetests: functionvalidateCurrentTimer(EdgeStorestoragestore,bytes32edgeId,uint256 maximumCachedTime) internal view returns(uint256) { uint256currentAccuTimer=store.edges[edgeId].totalTimeUnrivaledCache; if(currentAccuTimer≥maximumCachedTime){ revertCachedTimeSufficient(currentAccuTimer,maximumCachedTime); } returncurrentAccuTimer; } FigureB.2:The validateCurrentTimer function (src/challengeV2/libraries/EdgeChallengeManagerLib.sol#L500-L510) Thisfunctioniscalledinothercontexts,butthereturnedvalueisnevercheckedinthose cases. 3.Totalunrivaledtimeisalwayszerointhetests Inthecontextofthetests,the timeUnrivaled functionalwaysreturnszero: functionupdateTimerCacheByClaim(EdgeStorestoragestore, bytes32edgeId, bytes32claimingEdgeId,

uint8numBigStepLevel, uint256maximumCachedTime)internalreturns(bool,uint256){ //calculatethetimeunrivaledwithoutinheritance uint256totalTimeUnrivaled=timeUnrivaled(store,edgeId);

checkClaimIdLink(store,edgeId,claimingEdgeId,numBigStepLevel);
totalTimeUnrivaled+=store.edges[claimingEdgeId].totalTimeUnrivaledCache;
returnupdateTimerCache(store,edgeId,totalTimeUnrivaled,maximumCachedTime); } FigureB.3:The updateTimerCacheByClaim function (src/challengeV2/libraries/EdgeChallengeManagerLib.sol#L537-L549) 4.Overflowassertionsarenevertested Thereisagapinthetestingofoverflowassertions,whichaffectsimportantsectionsofthe createNewAssertion function: functioncreateNewAssertion(AssertionInputscalldataassertion, bytes32prevAssertionHash, bytes32expectedAssertionHash)internalreturns(bytes32newAssertionHash,booloverflowAssertion){ ...
if(assertion.afterState.machineStatus≠MachineStatus.ERRORED&& afterStateCmpMaxInbox<0){
//Ifwedidn'treachthetargetnextinboxposition,thisisan overflowassertion. overflowAssertion=true;
//Thisshouldn'tbenecessary,butmightaswellconstrainthe assertiontobenon-empty
require(afterGS.comparePositions(beforeGS)>0,"OVERFLOW_STANDSTILL"); } ... FigureB.4:Snippetofthe createNewAssertion function (src/rollup/RollupCore.sol#L378-L531) 4.Edgestakingfuzztestmissingprecondition The testProperInitialization fuzztestismissingapreconditionthatsometimes causesittofail.Thenewcoderequiresthattheassertionisnotzero.
functiontestProperInitialization(bytes32edgeId)public{ IEdgeStakingPoolstakingPool=
stakingPoolCreator.createPool(address(challengeManager),edgeId);
assertEq(address(stakingPoolCreator.getPool(address(challengeManager),edgeId)),
address(stakingPool)); assertEq(address(stakingPool.challengeManager()),address(challengeManager));
assertEq(stakingPool.edgeId(),edgeId); hasheye 19OffchainLabsSecurityAssessment PUBLIC

assertEq(address(stakingPool.stakeToken()),address(token)); } FigureB.5:The testProperInitialization function (test/stakingPool/EdgeStakingPool.t.sol#L44-L52) Ifthe edgeId iszero,thenthecodewillrevertinthepoolcreation: constructor(address_challengeManager, bytes32_edgeId)AbsBoldStakingPool(address(EdgeChallengeManager(_challengeManager).stakeToken())) {
if(_edgeId==bytes32(0)){ revertEmptyEdgeId(); } challengeManager=_challengeManager; edgeId=_edgeId;
} FigureB.6:The EdgeStakingPool constructor (src/assertionStakingPool/EdgeStakingPool.sol#L35-L44) 5.The makeStakeWithdrawableAndWithdrawBackIntoPool functionisnevertested The makeStakeWithdrawableAndWithdrawBackIntoPool functionofthe AssertionStakingPool contractisnevertested. 6.confirmEdgeByTimecornercaseisnevertested Thecasewherethetotaltimeunrivaledislessthantheconfirmationthresholdblocksis neverreachedandneedsanadditionalunittest. functionconfirmEdgeByTime(EdgeStorestoragestore, bytes32edgeId, uint64claimedAssertionUnrivaledBlocks, uint64confirmationThresholdBlock
)internalreturns(uint256){ if(!store.edges[edgeId].exists()){ revertEdgeNotExists(edgeId); }
uint256totalTimeUnrivaled=timeUnrivaledTotal(store,edgeId);
//sincesiblingassertionshavethesamepredecessor,theycanbeviewedas
//rivaledges.Addingtheassertionunrivaledtimeallowsustostartthe confirmation
//timerfromthemomentthefirstassertionismade,ratherthanhavingtowait untilthe
//secondassertionismade. totalTimeUnrivaled+=claimedAssertionUnrivaledBlocks; hasheye 20OffchainLabsSecurityAssessment PUBLIC

if(totalTimeUnrivaled<confirmationThresholdBlock){
revertInsufficientConfirmationBlocks(totalTimeUnrivaled, confirmationThresholdBlock); } ... } FigureB.7:The confirmEdgeByTime function (src/challengeV2/libraries/EdgeChallengeManagerLib.sol#L744-L773) hasheye 21OffchainLabsSecurityAssessment PUBLIC