HashEye

// SMART CONTRACT SECURITY AUDIT

Arbitrum

Security assessment by HashEye · prepared for Blockchain

HASHEYE AUDITED

PROJECT	Arbitrum
CLIENT	Blockchain
CATEGORY	Offchain Labs
PUBLISHED	September 1, 2021
REPORT ID	research-arbitrum-2021-09-01-xp7016

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-arbitrum-2021-09-01-xp7016.

ArbitrumTokenBridgeCreator SecurityAssessment(SummaryReport) July29,2024 Preparedfor: HarryKalodner,StevenGoldfeder,andEdFelten OffchainLabs Preparedby:JaimeIglesias,TroySargent,GustavoGrieco,andKurtWillis

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 Abouthasheye1 NoticesandRemarks2 TableofContents3 ProjectSummary4 ProjectTargets5 ExecutiveSummary6 SummaryofFindings7 DetailedFindings8 1.L2runtimecodedoesnotcontainconstructorcode8 2.L2tokenbridgecontractdeploymentcanbegriefed10 3.IncorrectL2Multicalladdresspredicted13 4.RollupownerisassumedtobeanEOA15 5.DepositingbeforethetokenbridgeisfullydeployedcanresultinlossoffUnds17 6.Dangerousaliasingassumption19 7.Uncleardecimalunitsofprovidedamounts21 8.TokenvaluesinDeployHelperarenotadjustedtotokendecimals23 A.VulnerabilityCategories24 B.FixReviewResults27 DetailedFixReviewResults28 hasheye 3OffchainLabsSecurityAssessment PUBLIC

ProjectSummary ContactInformation Thefollowingprojectmanagerwasassociatedwiththisproject: MaryO'Brien,ProjectManager mary.obrien@hasheye.io Thefollowingengineeringdirectorwasassociatedwiththisproject: JosselinFeist,EngineeringDirector,Blockchain josselin.feist@hasheye.io

Thefollowingconsultantswereassociatedwiththisproject: TroySarget,ConsultantKurtWilis,Consultant troy.sargent@hasheye.iokurt.wilis@hasheye.io GustavoGrieco,ConsultantJaimeIglesias,Consultant gustavo.grieco@hasheye.iojaime.iglesias@hasheye.io ProjectTimeline Thesignificanteventsandmilestonesoftheprojectarelistedbelow. DateEvent December4,2023Pre-projectkickoffcall December18,2023Deliveryofreportdraft December18,2023Reportreadoutmeeting July29,2024Deliveryofsummaryreport hasheye 4OffchainLabsSecurityAssessment PUBLIC

ProjectTargets Theengagementinvolvedareviewandtestingofthetargetslistedbelow token-bridge-contracts-privatePR#16 Repositoryhttps://github.com/OffchainLabs/token-bridge-contracts-private/pull/16 Version PR#16(ca71072...ed24e1e) TypeSolidity PlatformEVM token-bridge-contracts-privatePR#21 Repositoryhttps://github.com/OffchainLabs/token-bridge-contracts-private/pull/21 Version PR#21(b2e62e1...3c90260) TypeSolidity PlatformEVM nitro-contractsPR#100 Repositoryhttps://github.com/OffchainLabs/nitro-contracts/pull/100 Version PR#100(b455c31...645e51d) TypeSolidity PlatformEVM hasheye 5OffchainLabsSecurityAssessment PUBLIC

ExecutiveSummary EngagementOverview OffchainLabsengagedhasheyetoreviewthesecurityofitsTokenBridgeCreator.From December4toDecember12,2023,ateamofthreeconsultantsbeganasecurityreviewof theTokenBridgeCreator’ssourcecode.FromDecember13toDecember15,2023,ateam ofthreeconsultantsbeganasecurityreviewofaPRtotheNitrocontract’scodebase. ObservationsandImpact TokenBridgeCreatorisasetofsmartcontractsusedfortheatomicandpermissionless deploymentoftokenbridgecontractsontopofanexistingrollup.Itsmainuse-caseisto makedeploymentand/orconfigurationofOrbitL3chainsmoreconvenient.Thecontract canbeusedforbothL1<>L2andL2<>L3setups. Additionally,wereviewedanumberofsmallchangestotheNitroContractscodebase. Intotal,weuncoveredeightissues,mostofwhicharerelatedtoassumptionsmadebythe designoftheprotocol.Onehigh-severityissue,TOB-ARB-TBC-002,couldallowamalicious actortogriefthedeploymentofabridge.Thisissueandotherswereresolvedduringthefix review. hasheye 6OffchainLabsSecurityAssessment PUBLIC

SummaryofFindings Thetablebelowsummarizesthefindingsofthereview,includingtypeandseveritydetails. IDTitleTypeSeverity 1L2runtimecodedoesnotcontainconstructor code AccessControlsInformational 2L2tokenbridgecontractdeploymentcanbe griefed DenialofServiceHigh 3IncorrectL2MulticalladdresspredictedConfigurationLow 4RollupownerisassumedtobeanEOADataValidationInformational 5Depositingbeforethetokenbridgeisfully deployedcanresultinlossoffunds DataValidationMedium 6DangerousaliasingassumptionDataValidationLow 7UncleardecimalunitsofprovidedamountsDataValidationLow 8TokenvaluesinDeployHelperarenotadjustedto tokendecimals DataValidationMedium hasheye 7OffchainLabsSecurityAssessment PUBLIC

DetailedFindings 1.L2runtimecodedoesnotcontainconstructorcode Severity:InformationalDifficulty:High Type:AccessControlsFindingID:TOB-ARB-TBC-001 Target: contracts/tokenbridge/arbitrum/L2AtomicTokenBridgeFactory.sol Description ContractsthatarebeingdeployedtoL2viaretryableticketsonL1donotinclude informationontheircreationcode.Thiscanbedangerousandleadtoinconsistenciesin state. ThecontractsdeployedtoL2areencodedinthe L1TokenBridgeRetryableSender ’s sendRetryable function. bytesmemorydata=abi.encodeCall(L2AtomicTokenBridgeFactory.deployL2Contracts, (L2RuntimeCode(l2.routerTemplate.code, l2.standardGatewayTemplate.code, l2.customGatewayTemplate.code, l2.wethGatewayTemplate.code, l2.wethTemplate.code, l2.upgradeExecutorTemplate.code, l2.multicallTemplate.code), l1.router, l1.standardGateway, l1.customGateway, l1.wethGateway, l1.weth, l2StandardGatewayAddress, rollupOwner, aliasedL1UpgradeExecutor)); Figure1.1:TheL2contracts’templatecodeisincludedinaretryableTX. (L1TokenBridgeRetryableSender.sol) hasheye 8OffchainLabsSecurityAssessment PUBLIC

InordertodeploythecodeontheL2side,agenericconstructorcodeisusedtodeploythe givenruntimecode. //createL2routerlogicandupgrade addressrouterLogic=Create2.deploy(0,OrbitSalts.UNSALTED,CreationCodeHelper.getCreationCodeFor(runtimeCode)); Figure1.2:TheL1providedruntimecodeiswrappedwithagenericconstructorcode. (L2AtomicTokenBridgeFactory.sol) Theeffectisthattheoriginalconstructorisstrippedaway,whichcanbedangerousand leadtoerrors.Forexample,thiscouldresultintheremovalofdisablinginitializersfor proxyimplementations,oritcouldleadtoreferencinginvalidaddressesonL2dueto immutableaddressesincludedintheruntimecodethatwerevalidonL1(e.g.,corrupting the DelegateCallAware ’s onlyDelegated modifier). ExploitScenario Inanotherupgrade,animplementationcontractthatisdeployedonL2isleftuninitialized. Amalicioususerinitializestheimplementationandisabletoselfdestructit. Recommendations Shortterm,considerpassinginthecontract’screationcodeinsteadoftheruntimecode.

Longterm,ensurethatallproxyandlogiccontractsareinitializedcorrectlybyadding furtherteststoanykindofcontractdeployedviaaticket. hasheye 9OffchainLabsSecurityAssessment PUBLIC

2.L2tokenbridgecontractdeploymentcanbegriefed Severity:HighDifficulty:Medium Type:DenialofServiceFindingID:TOB-ARB-TBC-002 Target: contracts/tokenbridge/arbitrum/L2AtomicTokenBridgeFactory.sol Description TheretryableticketdeployingtheL2tokenbridgecontractscanfailifthecallisfront-run resultinginablockedstate. Whencreatingthetokenbridge, L1AtomicTokenBridgeCreator sendsouttworetryable tickets,oneforcreatingthe L2AtomicTokenBridgeFactory contractandoneforcalling L2AtomicTokenBridgeFactory.deployL2Contracts .Thesetickets areexpectedtobe atomicinthesensethattheyareallexecutedtogetherandguaranteedtosucceed. However,oncebothretryabletickets arecreatedandpending,amalicioususerhasthe chancetomanuallyredeemthefirstticket,creatingthe L1AtomicTokenBridgeCreator , andtheninsertatransactionbeforethesecondticketisredeemed.Iftheuserincludesa callto L2AtomicTokenBridgeFactory.deployL2Contracts fromanyaccountother thantheexpected L1TokenBridgeRetryableSender ,thentheL2contractswillbe deployedatnon-canonicaladdressesandwillnotmatchtheaddressesstoredinthe contract. TheL2contractaddressesaredependentonthesender.Thiscanbeseeninthe _getProxyAddress function,whichcomputestheaddressofa TransparentUpgradeableProxy thatisdeployedbythe L2AtomicTokenBridgeFactory using create2 giventhesaltcalculatedfromtheprefix, thechainID,andthesender.Thissenderisexpectedtobethe L1TokenBridgeRetryableSender . function_getProxyAddress(bytesmemoryprefix,uint256chainId) internal view returns(address) { returnCreate2.computeAddress(_getL2Salt(prefix,chainId), keccak256(abi.encodePacked(type(TransparentUpgradeableProxy).creationCode, abi.encode(hasheye 10OffchainLabsSecurityAssessment PUBLIC

canonicalL2FactoryAddress,_predictL2ProxyAdminAddress(chainId), bytes("")))), canonicalL2FactoryAddress); } // ... function_getL2Salt(bytesmemoryprefix,uint256chainId)internalviewreturns (bytes32){ returnkeccak256(abi.encodePacked(prefix,chainId, AddressAliasHelper.applyL1ToL2Alias(address(retryableSender)))); } Figure2.1:TheL2proxyaddressiscomputed.(L1AtomicTokenBridgeCreator.sol) The deployL2Contracts functioncanthereforebekeptpermissionless—sinceonlythe deploymentscomingfromthe L1TokenBridgeRetryableSender areconsideredthe canonicalonesfortherollup.However,someofthecontractsaredeployedindependently ofthecaller,atfixedaddressesusinganunsalted create2 call,suchasthe UpgradeExecutor logiccontract. //CreateUpgradeExecutorlogicandupgradetoit. addressupExecutorLogic=Create2.deploy(0,OrbitSalts.UNSALTED,CreationCodeHelper.getCreationCodeFor(runtimeCode)); Figure2.2:TheupgradeexecutorlogicisdeployedatafixedaddressonL2. (L2AtomicTokenBridgeFactory.sol) Astheaddressisfixed,thecontractcannotberedeployedtothesameaddressonceithas alreadybeencreated.Thisessentiallyblocksanyfurthercallsto deployL2Contracts .In particular,thismeansthatitispossibletoblockthesecondretryableticket(comingfrom the L1TokenBridgeRetryableSender)thatdeploysthecontractsattheprecomputed addresses.Theresultisamismatchinthestoreddeploymentaddresses,requiringa manualrecovery. ExploitScenario Bob,amalicioususer,waitsfortherollupownertocall createTokenBridge ,startingthe tokenbridgedeploymentprocess.Ontherollupchain,Bobthenmanuallyredeemsthe firstticket,whichcreatestheL2tokenbridgefactory,andthencalls deployL2Contracts hasheye 11OffchainLabsSecurityAssessment PUBLIC

fromhisownaddress.Therollupownerisnotawarethattheircallfails.Aftermanyfailed bridgingattempts,theissueisdiscovered.Thebridgedfundsarestuck,andtherollup ownerisunabletorecreatetheL2tokenbridgecontractsvia createTokenBridge . Recommendations Shortterm,ensurethattheL2bridgecontractcreationdoesnotendupbeingblocked. Eitherdeploythecurrentlyunsaltedcontractsusing create1 ormakethe create2 salt dependentonthesenderbyusing _getL2Salt(OrbitSalts.UNSALTED) . Longterm,criticallyexaminewhetherassumptions—suchassendingmultipleretryable ticketsbeingatomic—arealwaysvalidandwhetherthesecanbeexploited. hasheye 12OffchainLabsSecurityAssessment PUBLIC

3.IncorrectL2Multicalladdresspredicted Severity:LowDifficulty:Low Type:ConfigurationFindingID:TOB-ARB-TBC-003 Target: contracts/tokenbridge/ethereum/L1AtomicTokenBridgeCreator.sol Description ThecalculationofthedeploymentaddressfortheL2Multicallisincorrect. Thecodefor createTokenBridge predictsthattheaddressof Multicall willbe deployedontheL2usingthe. codehash valueofthe Multicall template. function_predictL2Multicall(uint256chainId)internalviewreturns(address){ returnCreate2.computeAddress(_getL2Salt(OrbitSalts.L2_MULTICALL,chainId), l2MulticallTemplate.codehash, canonicalL2FactoryAddress); } Figure3.1:TheL2Multicalladdressispredicted(L1AtomicTokenBridgeCreator.sol) Acontract’scodehashisthe keccak256 hashofthecontract’sruntimecode.The create2

opcode,however,computestheaddressusingthecontract'screationcode. The Multicall contractiscreatedusingaretryableticketcontainingtheruntimecode andwrapsitwithagenericcreationcode.
//deploymulticall Create2.deploy(0, _getL2Salt(OrbitSalts.L2_MULTICALL), CreationCodeHelper.getCreationCodeFor(l2Code.multicall));
Figure3.2:TheMulticallcontractisdeployedonL2using create2 (L2AtomicTokenBridgeFactory.sol)
ThisdisparitycausesthecreateTokenBridgecontracttopredictanincorrectaddressfor theL2 Multicall contract. Itisworthnotingthatthisissuewasnotdiscoveredduringtestingbecausethetestsdonot checktheinitializationofthe L2Multicall contractproperly;etherj's getCode returns hasheye 13OffchainLabsSecurityAssessment PUBLIC

" 0x" whentheaccounthasnocodeinsteadofreturningemptybytes("").Therefore,the expect statementinfigure3.3passesforanon-deployedcontract(i.e.,because " 0x".length>0).
asyncfunctioncheckL2MulticallInitialization(l2Multicall:ArbMulticall2){
//checkl2Multicallisdeployed constl2MulticallCode=awaitl2Provider.getCode(l2Multicall.address)
expect(l2MulticallCode.length).to.be.gt(0) } Figure3.3:The Multicall contractinitializationunittest (tokenBridgeDeploymentTest.ts) ExploitScenario AusermakesRPC Multicalls ontherollupusingtheaddressstoredintheL1contract. However,theRPCcallsfailbecausethe Multicall addressisinvalid. Recommendations
Shortterm,fixtheprecomputedaddresscalculationorconsiderusingthecontract creationcodedirectly.
function_predictL2Multicall(uint256chainId)internalviewreturns(address){
returnCreate2.computeAddress(_getL2Salt(OrbitSalts.L2_MULTICALL,chainId),
keccak256(CreationCodeHelper.getCreationCodeFor(l2MulticallTemplate.code)),
canonicalL2FactoryAddress); } Figure3.3:TheL2Multicalladdresspredictionisfixed
Longterm,includeend-to-endtestscheckingthatthecomputedaddressactuallymatches thedeployedaddress.Additionally,considercheckingthattheactualdeployedcode matchestheexpectedtemplateinsteadofcheckingwhetherornottheaddresshascode.
Finally,wheneveranexternalAPI(e.g.,etherjs)isused,itisofutmostimportancetocheck thedocumentationtoensurethatthevaluesreturnedbythefunctionsaretheexpected ones. hasheye 14OffchainLabsSecurityAssessment PUBLIC

4.RollupownerisassumedtobeanEOA Severity:InformationalDifficulty:High Type:DataValidationFindingID:TOB-ARB-TBC-004 Target: contracts/tokenbridge/ethereum/{L1AtomicTokenBridgeCreator,L2AtomicT okenBridgeFactory}.sol
Description TherollupowneriscurrentlyassumedtobeanEOA(externallyownedaccount);however, thisisneitherexplicitlycheckednorverified.
Someproxylogiccontractsmustfirstbeinitializedtoprotectagainstanunexpected initializationthatcancausetheexecutionofcriticaloperations(e.g., selfdestruct).
//sweepthebalancetosendtheretryableandrefundthedifference
//itisknownthatanyethpreviouslyinthiscontractcanbeextracted
//thoitisnotexpectedthatthiscontractwillhaveanyeth
retryableSender.sendRetryable{value:isUsingFeeToken?0:address(this).balance}(RetryableParams(inbox, canonicalL2FactoryAddress, msg.sender, msg.sender, maxGasForContracts, gasPriceBid), L2TemplateAddresses(l2RouterTemplate, l2StandardGatewayTemplate, l2CustomGatewayTemplate, isUsingFeeToken?address(0):l2WethGatewayTemplate, isUsingFeeToken?address(0):l2WethTemplate, address(l1Templates.upgradeExecutor), l2MulticallTemplate), l1Deployment, l2Deployment.standardGateway, rollupOwner, msg.sender, AddressAliasHelper.applyL1ToL2Alias(upgradeExecutor), isUsingFeeToken); hasheye 15OffchainLabsSecurityAssessment PUBLIC

Figure4.1:Therollupowneraddressispassedasaparametertotheretryableticket (L1AtomicTokenBridgeCreator.sol)
Theaddressisthenincludedasanexecutorroleintheupgradeexecutorcontract. //initupgradeexecutor
address[]memoryexecutors=newaddress[](2); executors[0]=rollupOwner;
executors[1]=aliasedL1UpgradeExecutor;
IUpgradeExecutor(canonicalUpgradeExecutor).initialize(canonicalUpgradeExecutor, executors);
Figure4.2:Therollupownerisgiventheexecutorroleintheupgradeexecutor (L2AtomicTokenBridgeFactory.sol) ThisimplicitlyassumesthattherollupownerwillalwaysbeanEOA,asotherwise,ifitwasa contract,theaddresswouldbealiasedtoanL2address.Ifthiswerethecase,itwouldnot beabletomakethecallstotheupgradeexecutor,becauseithasstoredtheunaliasedL1 address.
Thisassumptionisnotclearlystatedandnotverifiedon-chain.Inordertoprevent centralizationissues,therollup's ownershouldbeexpectedtobeatimelock-controlled multisigcontract.
ExploitScenario Amultisigrollupownercreatesatokenbridge.Therollupownerisaddedasanexecutorto theupgradeexecutor.However,theownerisnowunabletomakeanyupgradecalls becausetheunaliasedaddresshasbeenstored. Recommendations
Shortterm,documenttheassumptionthattherollupownerisexpectedtobeanEOAand considerexplicitlycheckingwhetherornottherollupownerisacontract.Additionally,both

thealiasedandunaliasedaddresscouldbegivenexecutorcontrolintheupgradeexecutor. Longterm,revisitassumptionsthatmaynotbeexplicitlystated;includeexplicitchecksfor these,orensurethatnounexpectedscenarioiscreated. hasheye 16OffchainLabsSecurityAssessment PUBLIC

5.Depositingbeforethetokenbridgeisfullydeployedcanresultinlossof funds Severity:MediumDifficulty:High Type:DataValidationFindingID:TOB-ARB-TBC-005 Target: contracts/tokenbridge/ethereum/L1AtomicTokenBridgeCreator.sol Description IfausertriggersadepositintheL1sideofthetokenbridgebeforeitisfullydeployed,their depositwillnotbeexecutedasexpected,andtheirfundswillnotbemintedontheL2. Tokenbridgecreationreliesontheusageoftworetryableticketsthatwillcorrectlysetup theL2sideofthebridge. /** *@noticeDeployandinitializetokenbridge,bothL1andL2sides,aspartof asingleTX. *@devThisisasingleentrypointofL1tokenbridgecreator.Functiondeploys L1sideoftokenbridgeandthenuses *2retryableticketstodeployL2side.1stretryabledeploysL2factory. Andthen'retryablesender'contract *iscalledtoissue2ndretryablewhichdeploysandinitstherestofthe contracts.L2chainisdetermined *by`inbox`parameter. * *Tokenbridgecanbedeployedonlyonceforcertaininbox.Anyfurther callsto`createTokenBridge`willrevert *becauseL1saltsarealreadyusedatthatpointandL1contractsare alreadydeployedatcanonicaladdresses *forthatinbox. */ Figure5.1:Documentationonthedeploymentofthetokenbridge (L1AtomicTokenBridgeCreator.sol) However,ifthebothticketsarenotimmediatelyredeemed,adepositfromtheL1will produceanincompletedepositinL2. ExploitScenario Alicestartstheprocessofthetokenbridgedeployment.AspikeintheL2activitycauses theretryableticketsnottoexecuteimmediately.BobiseagertousetheL2,sohequickly depositsintothetokenbridge,eventhoughthebridgeisnotfullydeployedyet.IfBob’s hasheye 17OffchainLabsSecurityAssessment PUBLIC

retryableticketwiththedepositisexecutedbeforetheL2ofthebridgeisready,the retryableticketwillcallanemptyaccountanditwillnotrevert,leavingBobwithouttheL2 counterpartofhistokens. Recommendations Shortterm,consideraddingafront-endchecktoverifythattheL2counterpartdeployment hassucceeded.Provideaclearerrortotheclientthatthetokenbridgeisnotyetfully deployedyetandthatanydepositwillresultinlossoffunds. Longterm,reviewtheassumptionsofthetokenbridgeduringitsusagetoensurethatthe newdeploymentwillnotbreakanyofthem. hasheye 18OffchainLabsSecurityAssessment PUBLIC

6.Dangerousaliasingassumption Severity:LowDifficulty:Medium Type:DataValidationFindingID:TOB-ARB-TBC-006 Target: nitro-contracts/src/bridge/AbsInbox.sol Description ApplyingtheL1-to-L2aliasforuser-providedaddressesdependsonL1contracts,whichcan causealiasingaddressestopointtoinvalidaddresses. Boththe excessFeeRefundAddress andthe callValueRefundAddress —twoaddresses thattheuserprovideswhen createRetryableTicket iscalled— arealiaseddepending onwhethertheL1addresscontainscode.

```
//ifarefundaddressisacontract,weapplythealiastoit //sothatitcanaccessitsfundsontheL2
//sincethebeneficiaryandotherrefundaddressesdon'tgetrewrittenbyarb-os
if(AddressUpgradeable.isContract(excessFeeRefundAddress)){ excessFeeRefundAddress=
AddressAliasHelper.applyL1ToL2Alias(excessFeeRefundAddress); }
if(AddressUpgradeable.isContract(callValueRefundAddress)){
//thisisthebeneficiary.becarefulsincethisistheaddressthatcancancel theretryableintheL2
callValueRefundAddress= AddressAliasHelper.applyL1ToL2Alias(callValueRefundAddress); }
```

Figure6.1:Aliasingofuserprovidedaddresses(AbsInbox.sol) Becauseitisnotpossibletoreliablydeterminewhethertheuserwantstouseanaliasedor unaliasedversionofagivenL2address,thisstepcanleadtomistakesbyerroneously applyinganaliaswhereitwasnotexpected.Thiscouldcausetherefundtobesenttoa nonexistentaddressonL2,whereitcouldberecoveredonlyfromitsL1counterpart,which couldbeanimmutablecontract. ExploitScenario Thefollowingeventsoccur: • Alice(EOA)deploysarandomtokencontractusingnonce0ataddress 0xabc onL1. • Alicealsodeploysamultisigcontractusingnonce0ataddress 0xabc onL2. hasheye 19OffchainLabsSecurityAssessment PUBLIC

• Alicecreatesaretryableticketandspecifiesthe callValueRefundAddress asthe L2multisig(at 0xabc). • Thealiascheckconverts 0xabc toanonexistentaddressonL2duetotheunrelated L1tokencontract. • OnlyL1tokencontractisabletorecoverfunds,butitdoesnotcontainlogictodo so,asitisanimmutabletokencontract. Recommendations Shortterm,clearlydocumentthebehaviorandmakeitcleartoauserthatanaliaswill applyincertaincases.Considernotmakinganyassumptionsonbehalfoftheuserand includeoff-chainchecksandvalidationsinawebapp. Longterm,consideraddingfurtheron-chaincheckstoensurethattheL1contractisable

tosendretryableticketsandrecoverthefundswhenapplyinganalias. hasheye 20OffchainLabsSecurityAssessment PUBLIC

7.Uncleardecimalunitsofprovidedamounts Severity:LowDifficulty:Medium Type:DataValidationFindingID:TOB-ARB-TBC-007 Target: nitro-contracts/src/bridge/AbsInbox.sol Description Whencreatingretryabletickets,thecallermustprovidemultipletokenvaluesinvarious units,whichcouldbepronetoerrors. The createRetryableTicket functionrequirestheusertoprovidevariousparametersin differentunits. function_createRetryableTicket(addressto, uint256l2CallValue, uint256maxSubmissionCost, addressexcessFeeRefundAddress, addresscallValueRefundAddress, uint256gasLimit, uint256maxFeePerGas, uint256amount, bytescalldatadata)internalreturns(uint256){ //ensuretheuser'sdepositalonewillmakesubmissionsucceed uint256amountToBeMintedOnL2=_fromNativeTo18Decimals(amount); if(amountToBeMintedOnL2<(maxSubmissionCost+l2CallValue+gasLimit* maxFeePerGas)){ revertInsufficientValue(maxSubmissionCost+l2CallValue+gasLimit*maxFeePerGas, amountToBeMintedOnL2); } //...) Figure7.1:The _createRetryableTicket function(AbsInbox.sol) Theparameter amount isgiveninthenativetoken’sdecimalunits.Theparameters l2CallValue , maxSubmissionCost ,and maxFeePerGas aredenominatedusing18 decimalunits. hasheye 21OffchainLabsSecurityAssessment PUBLIC

NeithertheparameternamesnortheNatSpeccommentssuggestthatthevaluesaregiven indifferingunits,whichcancausemistakesinintegration. ExploitScenario Anoptimisticcross-chainbridgeandAMMprotocolisbuiltontopofArbitrum.When integratingwithanOrbitchainwithnon-standarddecimals,theincorrectdecimalunitsare hardcodedintothetokenhandlercontract,causingvaluesthataretoohightobesentto theOrbitchainwhenbridgingthenativetoken. Recommendations Shortterm,considermakingacleardistinctionbetweentheunitsdependingonthechain. Forexample,onL1,allvalueswillalwaysbedenominatedinthenativetoken’sdecimal units,andonL2,allvalueswillalwaysbedenominatedusing18decimalpoints. Longterm,beawareofconfusionsthatcanarisewhenassumptionsintheAPIarenot clearlydocumented.Aimtoremovetheriskofmistakesbyfocusingonclearusabilitywhen interactingwiththebridgecontracts. hasheye 22OffchainLabsSecurityAssessment PUBLIC

8.TokenvaluesinDeployHelperarenotadjustedtotokendecimals Severity:MediumDifficulty:Medium Type:DataValidationFindingID:TOB-ARB-TBC-008 Target: nitro-contracts/src/rollup/DeployHelper.sol Description The DeployHelper contractdeployshelpercontractstotheL2chainthroughretryable ticketscontaininghard-codedvaluesthatcouldbechain-andtokendecimal-dependent. Certainhelpercontractscanbedeployedinthe DeployHelper contractaspartofthe rollupcreationprocess.Thesehelpercontractsaresentviasignedtransactions. //Nick'sCREATE2DeterministicDeploymentProxy //https://github.com/Arachnid/deterministic-deployment-proxy addresspublicconstantNICK_CREATE2_DEPLOYER= 0x3fAB184622Dc19b6109349B94811493BF2a45362; uint256publicconstantNICK_CREATE2_VALUE=0.01ether; bytespublicconstantNICK_CREATE2_PAYLOAD=hex"04f8a58085174876e80083..."; Figure8.1:Aliasingofuser-providedaddresses(AbsInbox.sol) BeforesendingtheseL2transactions,aretryableticketisfirstcreatedinordertofundthe deployeraddress. uint256feeAmount=_value+submissionCost+GASLIMIT*maxFeePerGas; //fundthetargetL2address if(_isUsingFeeToken){ IERC20Inbox(inbox).createRetryableTicket({ to:_l2Address, l2CallValue:_value, maxSubmissionCost:submissionCost, excessFeeRefundAddress:msg.sender, callValueRefundAddress:msg.sender, gasLimit:GASLIMIT, maxFeePerGas:maxFeePerGas, tokenTotalFeeAmount:feeAmount, data:"" }); }else{ Figure8.2:Thetokentotalfeeamountisbeingcalculated(DeployHelper.sol) hasheye 23OffchainLabsSecurityAssessment PUBLIC

ThefeeamountiscomputedinordertocovertheL2value,thesubmissioncost(0 inthe caseofusingafeetoken),andtheretryableTXgascost. Whencreatingaretryableticket,the tokenTotalFeeAmount parameterisexpectedtobe giveninthenativetokendecimalunits,asitisconvertedto18decimals.Thismeansthatif thecustomfeetoken’sdecimalsdiffer,thentheactualtokenvaluethatissentwillbe miscalculated.Whenthebridgereceivesatokenvaluethatistoolittle,itwilltransferthe missingfundsfrom msg.sender (DeployHelper),causingthetransactiontorevert. ExploitScenario Alicecreatesanewrollupwithhercustomfeetokenthatusessixdecimalpoints.When deployingthehelpercontracts,duetothemiscalculation,the Inbox requestsalarge amountoftokens(0.01e18*(1e18-1e6)=10Mtokens)fromthe DeployHelper .Asthe DeployHelper hasnotgivenanytokenspendingapprovaltothe Inbox ,thiswouldresult inatransactionfailure.Ifthetokensaremanuallysenttothe Inbox ,alargepartwouldbe stuckinanunrecoverableaddress. Conversely,ifthefeetokenusesdecimalpointsgreaterthan18,thenitispossiblethattoo littlevaluewouldbesentforthesuccessfulcontractcreation. Recommendations

Shortterm,convertthevaluegivenin18decimalunitstothenativetokendecimalunits (roundedupifneeded).
Longterm,implementfurthertestingthatincludesboundsonexpecteddeploymentcosts.
Additionally,avoidpatternsthatcancauseconfusioninfunctionAPIs. hasheye
24OffchainLabsSecurityAssessment 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 hasheye
25OffchainLabsSecurityAssessment PUBLIC

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. hasheye
26OffchainLabsSecurityAssessment PUBLIC

B.FixReviewResults Whenundertakingafixreview,hasheyereviewsthefixesimplementedforissues
identifiedintheoriginalreport.Thisworkinvolvesareviewofspecificareasofthesource
codeandsystemconfiguration,notcomprehensiveanalysisofthesystem.
FromDecember14toDecember15,2023,hasheyereviewedthefixesandmitigations
implementedbytheOffchainLabsteamfortheissuesidentifiedinthisreport.We
reviewedeachfixtodetermineitseffectivenessinresolvingtheassociatedissue.
Insummary,oftheeightissuesdescribedinthisreport,OffchainLabshasresolvedthree
issuesandleftfiveissuesunresolved.Foradditionalinformation,pleaseseetheDetailed
FixReviewResultsbelow. IDTitleStatus 1L2runtimecodedoesnotcontainconstructorcodeUnresolved
2L2tokenbridgecontractdeploymentcanbegriefedResolved 3IncorrectL2MulticalladdresspredictedResolved
4RollupownerisassumedtobeanEOAResolved
5Depositingbeforethetokenbridgeisfullydeployedcanresultinloss offunds Unresolved
6DangerousaliasingassumptionUnresolved 7UncleardecimalunitsofprovidedamountsUnresolved
8TokenvaluesinDeployHelperarenotadjustedtotokendecimalsUnresolved hasheye
27OffchainLabsSecurityAssessment PUBLIC

DetailedFixReviewResults TOB-ARB-TBC-001:L2runtimecodedoesnotcontainconstructorcode Unresolved.
TOB-ARB-TBC-002:L2tokenbridgecontractdeploymentcanbegriefed
ResolvedinPR26andPR28.ContractsarenolongerbeingdeployedusingtheunsaltedCREATE2
method,whichcouldleadtofront-runningissues;instead,allcontractsincludethealiased
retryablesenderaspartofthedeploymentsalt,thereforelinkingitwiththesender.
Additionally,theabilitytoresendtheL2deploymentretryabletickethasbeenaddedwhich
wouldhelpmitigatethescenarioinwhichtheoriginalticketsareredeemedoutoforder blockingthedeployment.
TOB-ARB-TBC-003IncorrectL2Multicalladdresspredicted
ResolvedinPR27,thecontract’sruntimecodehashisnolongerbeingusedtopredictthe
L2address;insteaditscreationcodeis.Additionally,thetestshavebeenupdatedtocheck
thatthedeployedcontractcodematchesthatoftheexpectedtemplateandtheprevious
codelengthcheckshavebeenupdatedtomatchetherjs’sbehavior. TOB-ARB-TBC-
004:RollupownerisassumedtobeanEOA ResolvedinPR27,therollupownerisnolongerassumedtobeanEOA-
insteadacheckis performedtodeterminewhetherit'sacontractornotand,inthecasethatitis,the

addressisaliasedmitigatingtheissue. TOB-ARB-TBC-
005:DepositingbeforethetokenbridgeisfullydeployedcanresultinlossoffundsUnresolved. TOB-ARB-TBC-
006:DangerousaliasingassumptionUnresolved. TOB-ARB-TBC-007:UncleardecimalunitsofprovidedamountsUnresolved. TOB-ARB-TBC-008:TokenvaluesinDeployHelperarenotadjustedtotokendecimalsUnresolved.