HashEye

// SMART CONTRACT SECURITY AUDIT

Balancer

Security assessment by HashEye · prepared for Balancer

HASHEYE AUDITED

PROJECT	Balancer
CLIENT	Balancer
CATEGORY	Blockchain
PUBLISHED	January 1, 2020
REPORT ID	research-balancer-2020-01-01-h74863

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-balancer-2020-01-01-h74863.

BalancerV2 SecurityAssessment April5,2021 PreparedFor: MikeMcDonald|BalancerProtocol mike@balancer.finance FernandoMartinelli|BalancerProtocol fernando@balancer.finance PreparedBy: JosselinFeist|hasheye josselin@hasheye.io AlexanderRemie|hasheye alexander.remie@hasheye.io

ExecutiveSummary FromMarch22toApril2,2021,Balancerengagedhasheyetoreviewthesecurityof BalancerV2.hasheyeconductedthisassessmentoverfourperson-weeks,withtwo engineersworkingfromcommits 2c84113 and bce652f fromthebalancer-core-v2 repository. hasheyeperformedapreviousauditofthisprojectoversixperson-weeksinJanuary 2021.Duringthataudit,17issueswerefound,including7relatedtorounding.AppendixH containsourroundingrecommendationsresultingfromthisinitialaudit. Inthefirstweekofthisaudit,wefocusedonunderstandingthecodebase,includingthe components’interactionswitheachother,andlookedforthemostcommonSolidity securityflaws.Weconcentratedonthevault.Inthesecondweek,wefocusedonthepool contractsandfeessystem.WealsousedEchidnatocheckforroundingerrorsinthemath contractsusedbythepools,withafocusonthestablepool. Wefound14issues.Oneofthehigh-severityissues(TOB-BALANCER-001)couldallowan assetmanagertodrainapoolofallofthetokensnotunderhisorhermanagement.We alsofoundmanyarithmeticissuesinthestablepool,includingsomethatcouldleadto tokendraining. Inadditiontothesecurityfindings,weidentifiedcodequalityissuesnotrelatedtoany particularvulnerability,whicharediscussedinAppendixC.AppendixDcontains recommendationsonevaluatingarbitraryERC20tokens,andAppendixEprovides recommendationsonarbitrarypools.AppendixFcontainsascriptbuiltontopofSlitherto reviewthecontracts’ nonReentrant modifiers.AnEchidna-baseddifferentialfuzzerofthe CurveandBalancerstablepoolarithmeticisprovidedinAppendixG. Overall,theBalancerV2projectfollowsbestpractices.Thecode’sstructurehasbeen improvedsinceouroriginalaudit,andBalancerhasavoidedthemainSoliditypitfalls. However,thecode’sunderlyingcomplexityandseveralmovingpartsincreasethe likelihoodofmistakes.Balancerimplementedourroundingrecommendationsforthe WeightedPool ,buttheroundingissuesfoundinthe StablePool indicatethatthecode’s arithmetichasroomforimprovementandthatmoreissuesmightbepresent.Finally,at thetimeoftheaudit,thecodewasunstableandundergoingchanges,includingsomethat werenotreviewedinthisaudit. hasheyerecommendsthatBalancertakethefollowingsteps: • Addressallreportedissues. • ImplementadditionalEchidnateststovalidatetheroundingofallarithmetic.

• Documentthedangersofusingnon-standardERC20tokens(e.g.,deflationary tokens),and,usingAppendixD,developanERC20tokenchecklistforusers. •

Documentthepools’assumptionsanddevelopausercheckl istforarbitrarypools basedonAppendixE. • Performanauditoncethecodebasehasstabilized. 2021hasheyeBalancerV2Assessment|4

ProjectDashboard ApplicationSummary NameBalancerV2 Version 2c84113 , bce652f TypeSolidity PlatformEthereum EngagementSummary DatesMarch22–April2,2021 MethodWhitebox ConsultantsEngaged2 LevelofEffort4person-weeks VulnerabilitySummary TotalHigh-SeverityIssues2 TotalMedium-SeverityIssues3 TotalLow-SeverityIssues5 TotalInformational-SeverityIssues3 TotalUndetermined-SeverityIssues1 Total14 CategoryBreakdown AccessControls1 AuditingandLogging1 DataValidation8 UndefinedBehavior4 Total14 2021hasheyeBalancerV2Assessment|5

CodeMaturityEvaluation CategoryNameDescription AccessControlsSatisfactory.Theprojectusesarobustauthenticationand authorizationsystem. ArithmeticModerate.Manyissuesstemfromalackofproperrounding validation,andmoreofthoseissuesmightbepresent.The codebasewouldbenefitfromcase-by-casehandlingoftherounding direction. AssemblyUseModerate.Whiletheuseofassemblyislimitedanddoesnotleadto anysecurityissues,itcouldbereducedbysharingstructuresacross contracts. CentralizationWeak.Therearelimitsonprivilegedusers’actions.However,an assetmanagercouldeasilyabusehisorherprivilegestodraina poolofallotherassets(TOB-BALANCER-001).Additionally,anadmin couldtoggletheemergencyperiodbetweenactiveandinactiveto selectivelyblocktransactions(TOB-BALANCER-014). CodeStabilityWeak.Thecodewasundergoingsignificantchangesduringthe auditandwilllikelycontinuetoevolvebeforereachingitsfinal version. UpgradeabilityNotApplicable.Thesystemcannotbeupgraded. Function Composition Satisfactory.Theoverallcodeiswellstructured,andmostofthe functionsaresmallandhaveclearpurposes.Thecriticalfunctions canbeeasilyextractedfortesting. Front-RunningModerate.Mostfunctionscontainargumentswithlimitstoprevent suddenchangesthatcouldnegativelyimpactusers’expectations andabilitytoexecutetransactionsasintended.However,a privilegedusercouldfront-runwithdrawalstoquicklyupdatethe withdrawalfee(TOB-BALANCER-011). MonitoringSatisfactory.Mostfunctionsemitevents.However,onefunctionfor changingtheswapfeedoesnot(TOB-BALANCER-004).Additionally, theemissionofatransfereventupontokenburningcouldconfuse usersmonitoringtheeventsofthesystem(TOB-BALANCER-002). SpecificationModerate.Theconstantproductarithmeticiswelldocumentedon https://balancer.finance/.However,theoverallsystemwouldbenefit fromcleanerdocumentationonthisarchitectureandassumption. 2021hasheyeBalancerV2Assessment|6

Revisingthedocumentationonfeesandthecomponents’ interactionswouldalsoimprovecodereadability. Testing& Verification Moderate.Thesystemhassuitableunittestsbutwouldbenefit fromtheadditionoffuzzingorformalmethodstoensurethatthe arithmeticmeetsexpectations.Manyissuesinthisreportwere foundusingEchidna. 2021hasheyeBalancerV2Assessment|7

EngagementGoals Theengagementwasscopedtoprovideasecurityassessmentoftheb alancerV2smart contracts. Specifically,wesoughttoanswerthefollowingquestions: • IsitpossibletocauseaDoSinapool? • Isitpossibleforassetmanagersoradminstoabusetheirabilities? • Isitpossibleforpoolstoaccessfundsbelongingtootherpools? • IstheuseofnativeandwrappedETHimplementedcorrectly? • Dothearithmeticlibrariescorrectlyapplyrounding? • Arefeesappliedcorrectly? • Canswapsbeusedtostealfunds? • Istheflashloanfeatureimplementedcorrectly? • Cantheauthentication/authorizationschemebecircumvented? • Istheinternalbalanceaccountingsystemworkingasexpected? 2021hasheyeBalancerV2Assessment|8

Coverage Authorizer+Authentication.Twosmallcontractsthatprovideauthorizationand authenticationfeaturesthroughoutthecontracts.Wemanuallyreviewedthe implementationtolookforflawsthatcouldallowanattackertosubvertthe authentication/authorizationfeatures. EmergencyPeriod.Implementsanemergencyperiodusedinallofthepoolsandthevault. Wemanuallyreviewedtheimplementationtolookforflawsthatcouldallowtheadminto turntheemergencyperiodonandoffatwill. AssetTransferHandler.Containsfunctionsforsendingandreceiving ERC20tokensand native/wrappedETH.Wemanuallyreviewedtheimplementationforflawsrelatedto wrapping/unwrappingETHandsending/receivingERC20tokens. FlashLoanProvider.Containsasingle flashLoan function.Wereviewedthe implementationtoidentifyanyflawsthatcouldenableanattackertoborrowtokens withoutrepayingtheflashloan. InternalBalance.Storestheinternalbalanceofeachaccountaswellasvarioushelper functionstoaddtoorwithdrawfromtheinternalbalance.Wereviewedthebalanceupdate

functionstodiscernwhetheranattackercouldusethemtogainaccesstootherusers' funds.Wealsocheckedwhetherfeeswerecorrectlyappliedtothebalance. PoolRegistry.Containsthefunctionstoregister,join,orexitapoolandto register/deregistertokensofapool,aswellashelperfunctionstopack/unpackpool variablesinto/fromasingle bytes32 variable.Wereviewedtheimplementationofthe pack/unpackfunctions.Wealsoreviewedallpoolconfiguration-relatedfunctionstofind flawsthatwouldenableapooltoaccessoroverwriteanotherpool'sdataortootherwise disturbtheinternalaccountingofpoolregistrations.Additionally,wereviewedthe applicationofthefeesforjoiningandexitingapool. ProtocolFeesCollector,Fees.Containfunctionstosetglobalfeepercentagesandreceives thecollectedfees,whichcanbewithdrawnbythesystemadmin.Wemanuallychecked thatallfunctionscorrectlyvalidatedtheinputargumentsandthatthefeewithdrawal processdidnotcontainflaws. Swaps.Containthemainfunctionstoinitiateswaps,includingmultipleswapsinonecall. Wecheckedthatthe(chained)swapsdidnotcontainflawsthatcouldallowanattackerto stealfunds.Wecheckedthatthecorrectpoolspecializationfunctionswerecalledinall places. 2021hasheyeBalancerV2Assessment|9

VaultAuthorization.Providesauthorizationhelperfunctionsandfunctionstosetarelayer onbehalfofanaccount.Wecheckedthattheauthorizationfunctionsdidnotcontainflaws andthatrelayerswereusedcorrectly. Vault.Actsastheentry-pointcontractforBalancerandiswheretokensarestored,user balancesareupdated,poolsareregistered,andflashloansareprovided.Welookedfor flawsthatcouldallowamaliciousassetmanagertostealassets.Wecheckedthe registrationofpools,theirtokens,andtheadditionandremovalofliquidityforallthree poolspecializationtypes.WereviewedtheuseofnativeETHandtheconversionto wrappedETHforflawsthatcouldenableETHtheft.Wereviewedtheuseofinternal balances,includingwhethertheycouldbeusedtostealtokensfromthesystem. Additionally,weexaminedthestorageofthepackedbalancesinabytes32andthevarious extractionfunctions. Pools(stable,weighted).Wereviewedthepools'components,withafocusontheir arithmeticandaccesscontrols.Weconfirmedthatonlythevaultcouldcallthejoin/exit operations.UsingEchidna,wetestedstableandweightedmathandlookedforflawsthat couldallowanattackertogaintokensfromswapping,joining,orexiting.Duetothetime constraints,andbecauseouroriginalauditfocusedontheweightedpool,weconcentrated oureffortsonthestablepoolarithmetic.Duetothenumberofstablepoolarithmetic findings,webelievethatmoreissuesmightbepresent. Duetothetimeconstraints,hasheyecouldnotexplorethefollowingareas: • BalancerHelpers.sol • EnumerableMap.sol • LogExpMath.sol anditsimpactoncallers 2021hasheyeBalancerV2Assessment|10

AutomatedTestingandVerification hasheyeusedautomatedtestingtechniquestoenhancecoverageofcertainareasofthe contracts,including: • Slither,aSolidityStaticanalysisframework. • Echidna,asmartcontractfuzzer.Echidnacanrapidlytestsecuritypropertiesvia malicious,coverage-guidedtestcasegeneration. Automatedtestingtechniquesaugmentourmanualsecurityreviewbutdonotreplaceit. Eachmethodhaslimitations:Slithermayidentifysecuritypropertiesthatfailtoholdwhen SolidityiscompiledtoEVMbytecode;Echidnamaynotrandomlygenerateanedgecase thatviolatesaproperty. AutomatedTestingwithEchidna WeimplementedthefollowingEchidnaproperties: StableMathArithmeticProperties IDPropertyResult 1 StableMath._calcOutGivenIn cannotleadtofreetokens.FAILED (TOB-BALAN CER-006) 2 StableMath._calcOutGivenIn cannotleadtofreetokenswith largebalances. PASSED 3 StableMath._calcInGivenOut cannotleadtofreetokens.FAILED (TOB-BALAN CER-007) 4 StableMath._calcInGivenOut cannotleadtofreetokenswith largebalances. PASSED 5 StableMath._calcTokenInGivenExactBptOut cannotleadtofree BPTtokens. FAILED (TOB-BALAN CER-008) 6 StableMath._calculateInvariant ismonotonicallyincreasing.FAILED (TOB-BALAN CER-014) 2021hasheyeBalancerV2Assessment|11

7 StableMath._calculateInvariant doesnotdiffersignificantly fromCurve'sinvariant. FAILED (TOB-BALAN CER-009) WeightedMathArithmeticProperties IDPropertyResult 8 WeightedMath._calcOutGivenIn cannotleadtofreetokens.PASSED 9 WeightedMath._calcOutGivenIn cannotleadtofreetokenswith largebalances. PASSED 10 WeightedMath._calcInGivenOut cannotleadtofreetokens.PASSED 11 WeightedMath._calcTokenInGivenExactBptOut cannotleadto freeBPTtokens. PASSED 12 WeightedMath._calculateInvariant ismonotonicallyincreasing.PASSED 13Thesumofallnormalizedweightsinapoolis1.FAILED (TOB-BALAN CER-012) Allthepropertieswererunwithpoolscontainingfourtokens,andwithaTestLimitoffive million. AutomatedTestingwithSlither WeimplementedthefollowingSlitherproperty: PropertyResult All Vault

functionsexpectedtohavethe nonReentrant modifierdohaveit. PASSED (APPENDIXF)

RecommendationsSummary Thissectionaggregatesalltherecommendationsmadeduringtheengagement.Short-term recommendationsaddresstheimmediatecausesofissues.Long-termrecommendations pertaintothedevelopmentprocessandlong-termdesigngoals. ShortTerm □ Documentthefactthatthemanagerhasaccesstoallofthefundsinthepool,or allowonemanagerperpool,insteadofonemanagerpertoken.Thiswillpreventuser confusionandreducetherisksassociatedwithmaliciousmanagers.TOB-BALANCER-001 □ Createa Burn eventorpreventtransfertothezeroaddress.Thiswillensurethatan attackercannotconfuseeventmonitorswithincorrectburnevents.TOB-BALANCER-002 □ Measurethegassavingsfromoptimizations,andcarefullyweighthemagainstthe possibilityofanoptimization-relatedbug.Thecompileroptimizerwasfrequentlythe sourceofbugsandshouldbecarefullyreviewed.TOB-BALANCER-003 □ Addeventsfor BasePool.setSwapFee , Authorizer.changeAuthorizer ,and Authorizer.changeRelayerAllowance .Eventswillfacilitatecontractmonitoringandthe detectionofsuspiciousbehavior.TOB-BALANCER-004 □ Addzero-valuechecksonallfunctionarguments.Thiswillensurethatuserscan't accidentallysetincorrectvalues,misconfiguringthesystem.TOB-BALANCER-005 □ Remove FixedPoint.mul and FixedPoint.div ,andensurethateveryfunctionuses theappropriateroundingdirection.Thesefunctionsareerror-proneandcanleadto roundingissues.TOB-BALANCER-006,TOB-BALANCER-007,TOB-BALANCER-008 □ Createtwoversionsof _calculateInvariant and _getTokenBalanceGivenInvariantAndAllOtherBalances thatroundupordown, respectively,andusethemappropriately.Severalissueswerecausedbyincorrect roundingstrategiesinthesetwofunctions.TOB-BALANCER-006,TOB-BALANCER-007, TOB-BALANCER-008 □ InvestigatethedifferencesbetweentheBalancerandCurveinvariants,and evaluatetheirimpactonarbitrageopportunitiesandpotentialassociatedrisks.The impactsoftheirdifferencesareunclearandshouldbereviewed.TOB-BALANCER-009 □ Disallowswap/join/exitoperationsifthebalanceofthe tokenin hasbeen emptied,oriftheoperationswouldemptythe tokenout .Theweightedpool's method

ofhandlingapoolwithanemptybalanceisfragile,andtheexistingmitigationscouldcease toworkifthecodeisrefactored.TOB-BALANCER-010 □ Addaminimumwithdrawalrequirementforwithdrawaloperations.Otherwise,a maliciousownercouldfront-runthewithdrawaloperationsandmakeuserspay higher-than-expectedfees.TOB-BALANCER-011 □ Checkthatthesumofthenormalizedweightsisequalto10**18intheweighted pool'sconstructor.Thiswillensurethatthepool'sinvariantispreservedatdeployment. TOB-BALANCER-012 □ Applythecorrectroundingin StableMath._calculateInvariant .Thecurrent invariantisnotmonotonicallyincreasing,whichcouldleadtounexpectedbehaviorinallof thestablepool'sformulas.TOB-BALANCER-013 □ Documenttheemergencyperiod-relatedabilitiesofanadminsothatuserswillbe wellinformed.TOB-BALANCER-014

LongTerm □ Documenttheexpectationsformanagersandtheirassociatedrisks.Consider providingsmartcontractsthatcanactasmanagerswithon-chainrestrictions. TOB-BALANCER-001 □ Documentandtesttheexpectedbehaviorofallofthesystem'sevents.Consider usingablockchain-monitoringsystemtotrackanysuspiciousbehaviorinthecontracts. TOB-BALANCER-002 □ Monitorthedevelopmentandadoptionof Solidity compileroptimizationsto assesstheirmaturity.Thedeploymentrequiresoptimizations,whichareinherentlyrisky. TOB-BALANCER-003 □ UseSlither.Slitherwillcatchmanysecurityissueswhilerequiringaloweffort. TOB-BALANCER-004,TOB-BALANCER-005 □ UseEchidnawhendesigninganewpooltoensurethatswap/join/exitoperations andthepool'sinvariantarerobustagainstroundingissues.Manyarithmeticissues werefoundusingEchidna.TOB-BALANCER-006,TOB-BALANCER-007,TOB-BALANCER-008, TOB-BALANCER-010,TOB-BALANCER-012,TOB-BALANCER-013 □ Usedifferentialfuzzingwhendesigningasystemforwhichanother implementationexists.Echidnacanbeusedasadifferentialfuzzertoensurethatthe implementationworksasexpected.TOB-BALANCER-009 □ Identifyanddocumentallparametersthatcanbechangedbyprivilegedusers. Ensurethatupdatestotheseparameterswillhavealimitedimpactonusers'funds. TOB-BALANCER-011 □ Documenttheabilitiesoftheprivilegedrolesthroughoutthesystem.Thatway, userswillnotbesurprisedwhenothersexercisetheirprivileges.TOB-BALANCER-014

FindingsSummary PerBalancer'srequest,thestablepool-relatedfindingsarelocatedattheendofthissection becauseofpossiblefuturedeprecation. #TitleTypeSeverity 1Maliciousmanagercanreinvesttokensto drainthepool Undefined Behavior High 2Transfertozerocanleadtounexpected burns AccessControlsInformational 3Solidity compileroptimizationscanbe dangerous Undefined Behavior

Informational 4MissingeventsforcriticaloperationsAuditingand Logging Informational 5LackofzerocheckonfunctionsDataValidationLow 10Lackofrobustsafeguardsforpoolswith anemptytokeninWeightedPool DataValidationHigh 11Protocolfeefront-runDataValidationLow 12Sumofnormalizedweightscanbe differentfrom1 DataValidationLow 14Emergencyperiodtogglingcanbeusedto selectivelyblocktransactions Undefined Behavior Low 6 StableMath._calcOutGivenIn mayallow freeswaps DataValidationMedium 7 StableMath._calcInGivenOut mayallow freeswaps DataValidationMedium 8StableMath._calcTokenInGivenExactBptO utmayallowanattackertojoinforfree DataValidationMedium 9BalancerStablePool'sinvariantcandiffer significantlyfromCurve'sinvariant Undefined Behavior Undetermined 13StablePool'sinvariantisnotmonotonically increasing DataValidationLow 2021hasheyeBalancerV2Assessment|16

1.Maliciousmanagercanreinvesttokenstodrainthepool Severity:HighDifficulty:High Type:UndefinedBehaviorFindingID:TOB-BALANCER-001 Target: PoolRegistry.sol Description Eachtokeninapoolhasamanagerwhocanborrowtokens.Amaliciousmanagercould borrowthetokens,swapthemforalltheothertokens,anddrainthepool. withdrawFromPoolBalance allowsamanagerofatokeninapooltowithdrawtokens: functionwithdrawFromPoolBalance(bytes32poolId,AssetManagerTransfer[]memorytransfers) external override nonReentrant noEmergencyPeriod { _ensureRegisteredPool(poolId); PoolSpecializationspecialization=_getPoolSpecialization(poolId); for(uint256i=0;i<transfers.length;++i){ IERC20token=transfers[i].token; _ensurePoolAssetManagerIsSender(poolId,token); uint256amount=transfers[i].amount; if(specialization==PoolSpecialization.MINIMAL_SWAP_INFO){ _minimalSwapInfoPoolCashToManaged(poolId,token,amount); }elseif(specialization==PoolSpecialization.TWO_TOKEN){ _twoTokenPoolCashToManaged(poolId,token,amount); }else{ _generalPoolCashToManaged(poolId,token,amount); } token.safeTransfer(msg.sender,amount); emitPoolBalanceChanged(poolId,msg.sender,token,-(amount.toInt256())); } } Figure1.1: vault/PoolRegistry.sol#L512-L537 . Itisexpectedthatamanagercanimpactorstealonlythetokensunderhismanagement. However,asamanagercanreinvestthetokensdirectlyandswapthemfortheother tokensinthepool,hecandrainthepoolentirely. ExploitScenario Bobdeploysapoolwithfourtokens,oneofwhichEvemanages.Evewithdrawsandswaps allthetokenssheismanagingfortheothers,effectivelydrainingthepool. 2021hasheyeBalancerV2Assessment|17

Recommendations Shortterm,takeoneofthefollowingsteps: • Documentthefactthatthemanagerhasaccesstoallofthefundsinthepool • Allowonemanagerperpool,insteadofonemanagerpertoken Thiswillpreventuserconfusionandreducetherisksassociatedwithmaliciousmanagers. Longterm,documenttheexpectationsformanagersandtheirassociatedrisks.Consider providingsmartcontractsthatcanactasmanagerswithon-chainrestrictions. 2021hasheyeBalancerV2Assessment|18

2.Transfertozerocanleadtounexpectedburns Severity:InformationalDifficulty:Low Type:AccessControlsFindingID:TOB-BALANCER-002 Target: BalancerPoolToken.sol Description BalancerPoolToken usesa Transfer -to-zeroeventtoindicateaburn.Thatsameeventcan betriggeredwithoutburningtokens,whichcouldcauseconfusionamongoff-chain monitors. Whentokensareburned,thetotalsupplydecreases,anda Transfer -to-zeroeventis emitted: function_burnPoolTokens(addresssender,uint256amount)internal{ uint256currentBalance=_balance[sender]; require(currentBalance⩾amount,"INSUFFICIENT_BALANCE"); _balance[sender]=currentBalance-amount; _totalSupply=_totalSupply.sub(amount); emitTransfer(sender,address(0),amount); } Figure2.1: pools/BalancerPoolToken.sol#L139-L146 Userscanemitthesameeventbycalling transfer / transferFrom directly,butthetotal supplywillnotbechanged.Thisdivergencecanbeconfusingtooff-chainmonitorsandcan maketrackingthesystem'seventsmoredifficult. Recommendations Shortterm,eithercreatea Burn eventorpreventtransfertothezeroaddress. Longterm,documentandtesttheexpectedbehaviorofallofthesystem'sevents.Consider usingablockchain-monitoringsystemtotrackanysuspiciousbehaviorinthecontracts. 2021hasheyeBalancerV2Assessment|19

3.Solidityompileroptimizationscanbedangerous Severity:InformationalDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-BALANCER-003 Target: hardhat.config.ts Description InBalancerV2,optionalcompileroptimizationsinSolidityareenabled. Therehavebeenseveraloptimization-relatedbugsinSolidity withsecurityimplications, andoptimizationsareactivelybeingdeveloped.Solidityompileroptimizationsare disabledbydefault,anditisunclearhowmanycontractsinthewildactuallyusethem.Itis thereforeunclearhowwelltheyarebeingtestedandexercised. High-

severitysecurityissuescausedbyoptimizationbugshaveoccurredinthepast.A high-severitybuginthe emscripten -generated solc-js compilerusedbyTruffleand Remixpersisteduntillate2018.ThefixforthisbugwasnotreportedintheSolidity CHANGELOG.Anotherhigh-severityoptimizationbugresultinginincorrectbitshiftresults waspatchedinSolidity 0.5.6.Morerecently,abugstemmingfromincorrectcachingof keccak256 wasreported. AcompilerauditofSolidityfromNovember2018concludedthattheoptionaloptimizations maynotbesafe. Therearelikelylatentbugsrelatedtooptimizationand/ornewbugsthatwillbeintroduced duetofutureoptimizations. ExploitScenario AlatentorfuturebuginSolidity compileroptimizations—orintheEmscriptentranspilation to solc-js —causesasecurityvulnerabilityintheBalancercontracts. Recommendations Shortterm,measurethegassavingsfromoptimizations,andcarefullyweighthemagainst thepossibilityofanoptimization-relatedbug. Longterm,monitorthedevelopmentandadoptionofSolidity compileroptimizationsto assesstheirmaturity. 2021hasheyeBalancerV2Assessment|20

4.Missingeventsforcriticaloperations Severity:InformationalDifficulty:Low Type:AuditingandLoggingFindingID:TOB-BALANCER-004 Target: pools/BasePool.sol,vault/VaultAuthorization.sol Description Severalcriticaloperationsdonottriggerevents,whichwillmakeitdifficulttocheckthe behaviorofthecontractsoncetheyhavebeendeployed. Ideally,thefollowingcriticaloperationsshouldtriggerevents: • BasePool.setSwapFee functionsetSwapFee(uint256swapFee)externalauthenticate{ require(swapFee≤_MAX_SWAP_FEE,"MAX_SWAP_FEE"); _swapFee=swapFee; } Figure4.1: pools/BasePool.sol#L181-L184 • VaultAuthorization.changeAuthorizer functionchangeAuthorizer(IAuthorizernewAuthorizer)external overridenonReentrantauthenticate{ _authorizer=newAuthorizer; } Figure4.2: vault/VaultAuthorization.sol#L42-L44 • VaultAuthorization.changeRelayerAllowance functionchangeRelayerAllowance(addressrelayer,boolallowed) externaloverridenonReentrantnoEmergencyPeriod{ _allowedRelayers[msg.sender][relayer]=allowed; } Figure4.3: vault/VaultAuthorization.sol#L50-L52 Withoutevents,usersandblockchain-monitoringsystemswillnotbeabletoeasilydetect suspiciousbehavior. Recommendations Shortterm,addeventsfor BasePool.setSwapFee , Authorizer.changeAuthorizer ,and Authorizer.changeRelayerAllowance .Eventswillfacilitatecontractmonitoringandthe detectionofsuspiciousbehavior. 2021hasheyeBalancerV2Assessment|21

Longterm,addSlither,whichfoundthisissue,toyourCI. 2021hasheyeBalancerV2Assessment|22

5.Lackofzerocheckonfunctions Severity:LowDifficulty:High Type:DataValidationFindingID:TOB-BALANCER-005 Target: pools/BasePoolFactory.sol Description Certainsetterfunctionsfailtovalidateincomingarguments,socallerscanaccidentallyset importantstatevariablestothezeroaddress. Theseincludethe constructor in BasePoolFactory ,whichsetsthe vault .Thiscontract servesasthebasecontractforthe StablePoolFactory and WeightedPoolFactory contracts.Ifthe vault issettoaddresszero,subsequentcallstocreatea StablePool or WeightedPool willrevert;tosetacorrect vault ,thepoolfactorywillneedtobe redeployed. constructor(IVault_vault){ vault=_vault; } Figure5.3: pools/BasePoolFactory.sol#L26-L28 Thisissueisalsoprevalentinthefollowingfunctions: • AssetTransferHandler.constructor • Authorizer.constructor ExploitScenario Alicedeploysa StablePoolFactory andaccidentallysetsthe vault to address(0) .When sheinvokesthe create functiontocreatea StablePool ,thetransactionrevertsbecause the vault hasbeensetto address(0) . Recommendations Shortterm,addzero-valuechecksonallfunctionargumentstoensurethatuserscan’t accidentallysetincorrectvalues,misconfiguringthesystem. Longterm,useSlither,whichwillcatchfunctionsthatdonothavezerochecks. 2021hasheyeBalancerV2Assessment|23

10.Lackofrobustsafeguardsforpoolswithanemptytokenin WeightedPool Severity:HighDifficulty:High Type:DataValidationFindingID:TOB-BALANCER-010 Target: WeightedMath.sol,WeightedPool.sol Description The WeightedPool hasseveralcornercasesthatcouldeitherenableausertoreceive tokensforfreeortraptheliquidityifoneofthepoolshasanemptybalance.Itmightnot bepossibleforausertoemptyapool’sunderlyingtokenbalance;however,we recommendaddingsafeguardstoimprovethepools’robustness. Casesthatcouldleadtofreetokens _calcTokenInGivenExactBptOut determinesthenumberoftokensthatausermustsend toreceiveagivennumberofthetokensinapool(i.e.,singleassetdeposit): function_calcTokenInGivenExactBptOut(uint256tokenBalance, uint256tokenNormalizedWeight, uint256bptAmountOut, uint256bptTotalSupply, uint256swapFee)internalpurereturns(uint256){ /** //tokenInForExactBPTOut// //a=tokenAmountIn// //b=tokenBalance//totalBPT+bptOut\(1/w)\// //bptOut=bptAmountOuta=b*|├--------------------|^-1|// //bpt=totalBPT\\totalBPT////

```
//w=tokenWeight//
*****************************************************************************/
//Tokenin,soweroundupoverall.
//CalculatethefactorbywhichtheinvariantwillincreaseaftermintingBPTAmountOut
uint256invariantRatio=bptTotalSupply.add(bptAmountOut).divUp(bptTotalSupply);
//Calculatebyhowmuchthetokenbalancehastoincreasetocauseinvariantratio
uint256tokenBalanceRatio=FixedPoint.powUp(invariantRatio,
FixedPoint.ONE.divUp(tokenNormalizedWeight));
uint256tokenBalancePercentageExcess=tokenNormalizedWeight.complement();
uint256amountInAfterFee=tokenBalance.mulUp(tokenBalanceRatio.sub(FixedPoint.ONE));
uint256swapFeeExcess=swapFee.mulUp(tokenBalancePercentageExcess);
returnamountInAfterFee.divUp(swapFeeExcess.complement()); 2021hasheyeBalancerV2Assessment|24
```

} Figure10.1: pools/weighted/WeightedMath.sol#L172-L201 Ifthebalanceofthe tokenIn iszero,the amountIn requiredwillbezero.Anattackermay beabletoswaptokensforfreeifoneofthetokenshasabalanceofzero.Similarissuescan occurinotherfunctions,suchas WeightedMath._calcInGivenOut. Casesthatcouldleadtoatrappedpool Whenauserwantstojoinorexitthepool, _getDueProtocolFeeAmounts iscalled.

```
uint256[]memoryprotocolFeeAmounts=_getDueProtocolFeeAmounts(
```

Figure10.1: pools/weighted/WeightedPool.sol#L266

```
dueProtocolFeeAmounts=_getDueProtocolFeeAmounts(
```

Figure10.2: pools/weighted/WeightedPool.sol#L375 _getDueProtocolFeeAmounts dividesby currentInvariant .

```
uint256invariantRatio=previousInvariant.divUp(currentInvariant);
```

Figure10.3: pools/weighted/WeightedPool.sol#L500 Ifoneofthebalancesiszero,thepool’sinvariantwillbezero.

```
function_calculateInvariant(uint256[]memorynormalizedWeights,uint256[]memorybalances) internal pure
returns(uint256invariant) {
/**********************************************************************************
//invariant_____// //wi=weightindexi||wi// //bi=balanceindexi||bi^=i// //i=invariant//
*********************************************************************************/
InputHelpers.ensureInputLengthMatch(normalizedWeights.length,balances.length);
invariant=FixedPoint.ONE; for(uint8i=0;i<normalizedWeights.length;i++){
invariant=invariant.mul(FixedPoint.pow(balances[i],normalizedWeights[i])); } }
```

Figure10.4: pools/weighted/WeightedMath.sol#L37-L54 2021hasheyeBalancerV2Assessment|25

Ifoneofthebalancesiszero,thebalancewillbedividedbyzero,causingjoin/exit operationstorevert. Duetothecurrentroundingstrategies,wewerenotabletoemptyapoolthrough swap/exitoperations.However,acoderefactoringoraspecificedgecasemightallowan attackertousetheoperationstoturnaprofit. ExploitScenario Bobdeploysapoolwithfourtokens.Evefindsawaytoemptythepoolofoneofthe tokens,consequentlydrainingitoftheotherthree. Recommendations Shortterm,disallowswap/join/exitoperationsifthe tokenin hasanemptybalanceorif theoperationswouldemptyitout. Longterm,useEchidnawhendesigninganewpooltoensurethattheswap/join/exit operationsarerobustagainstbalance-emptyingattemptsandsimilaredgecases. 2021hasheyeBalancerV2Assessment|26

11.Protocolfeefront-run Severity:LowDifficulty:High Type:DataValidationFindingID:TOB-BALANCER-011 Target: PoolRegistry.sol,InternalBalance.sol,ProtocolFeesCollector.sol Description Privilegeduserscanfront-runwithdrawaloperationstoincreasewithdrawalfees,reducing theamountofassetsotherusers willreceivewhentheyexecutewithdrawals. Withdrawaloperationscarryafee:

```
uint256withdrawFee=toInternalBalance?0:_calculateProtocolWithdrawFeeAmount(amountOut);
_sendAsset(assets[i],amountOut.sub(withdrawFee),recipient,toInternalBalance,false);
```

Figure11.1: vault/PoolRegistry.sol#L388-L389

```
if(taxableAmount>0){
uint256feeAmount=_calculateProtocolWithdrawFeeAmount(taxableAmount); _payFee(token,feeAmount);
amountToSend=amountToSend.sub(feeAmount); } _sendAsset(asset,amountToSend,recipient,false,false);
```

Figure11.2: vault/InternalBalance.sol#L99-L105 Thefeeissetthrough setProtocolWithdrawFee :

```
functionsetWithdrawFee(uint256newWithdrawFee)externalauthenticate{
require(newWithdrawFee≤_MAX_PROTOCOL_WITHDRAW_FEE,"WITHDRAW_FEE_TOO_HIGH");
_withdrawFee=newWithdrawFee; emitWithdrawFeeChanged(newWithdrawFee); }
```

Figure11.3: vault/ProtocolFeesCollector.sol#L80-L84 Withdrawaloperationsdonotspecifytheamountofassetsausershouldreceiveafterthe feehasbeenapplied.Asaresult,ifausersendsawithdrawaltransactionandthefeeis updatedbeforethewithdrawalhasbeenminted,theuserwillreceivea lower-than-expectedamountofassets. ExploitScenario Evesetsthefeeto0.Bobwithdrawsassetsworth$10,000,000.Evefront-runsthe withdrawalandsetsthefeeto0.5%.Asaresult,Bobloses$50,000. Recommendations 2021hasheyeBalancerV2Assessment|27

Shortterm,addaminimumwithdrawalrequirementforwithdrawaloperations.
Longterm,identifyanddocumentallparametersthatcanbechangedbyprivilegedusers.
Ensurethatupdatestotheseparameterswillhavealimitedimpactonusers’funds.
2021hasheyeBalancerV2Assessment|28

12.Sumofnormalizedweightscanbediferentfrom1 Severity:LowDifficulty:High
Type:DataValidationFindingID:TOB-BALANCER-012 Target: WeightedPool.sol Description
Theweightedpooltokennormalizestokenweightssuchthataweightisapercentageof
thetokens’totalweight.Thecomputationexpectsthesumofallweightstobeequalto1,
butroundingmayresultinadifferentsum.
Tocomputenormalizedweights,theweightedpoolconstructorsumstheweightsand
calculateseachweightastheweightdividedbythesumoftheweights:
//Checkvalidweightsandcomputenormalizedweights uint256sumWeights=0;
for(uint8i=0;i<weights.length;i++){ sumWeights=sumWeights.add(weights[i]); }
uint256maxWeightTokenIndex=0; uint256maxNormalizedWeight=0;
uint256[]memorynormalizedWeights=newuint256[](weights.length);
for(uint8i=0;i<normalizedWeights.length;i++){ uint256normalizedWeight=weights[i].div(sumWeights);
require(normalizedWeight⩾_MIN_WEIGHT,"MIN_WEIGHT"); normalizedWeights[i]=normalizedWeight;
if(normalizedWeight>maxNormalizedWeight){ maxWeightTokenIndex=i;
maxNormalizedWeight=normalizedWeight; } } Figure12.1: pools/weighted/WeightedPool.sol#L72-L91
Duetothearithmeticprecisionlossin weights[i].div(sumWeights) ,thesumofthe
normalizedweightscanbeslightlyhigherorlowerthan1.Thismaybreaktheunderlying assumptionsofthepool.
ExploitScenario Bobdeploysapoolwith4tokens.Theirweightsare1,1,1,and3.Thesumofthe
normalizedweightsisequalto10**18+1,whichbreaksoneofthepool’sinvariants. Recommendations
2021hasheyeBalancerV2Assessment|29

Shortterm,checkthatthesumofthenormalizedweightsisequalto10**18inthe
weightedpool’sconstructor.Thiswillensurethatthepool’sinvariantispreservedat deployment.
Longterm,identifythesystem’sinvariants,anduseEchidnatocheckitsrobustness.
2021hasheyeBalancerV2Assessment|30

14.Emergencyperiodtogglingcanbeusedtoselectivelyblocktransactions Severity:LowDifficulty:High
Type:UndefinedBehaviorFindingID:TOB-BALANCER-014 Target: EmergencyPeriod.sol Description The
EmergencyPeriod contractallowstheadmintorepeatedlytogglethe emergency
periodbetweenactiveandinactiveuntilthe _emergencyPeriodEndDate haspassed.As
such,amaliciousadmincouldcausetransactionsoftheadmin’schoosingtofailby front-
runningthemandthenquicklyenablingtheemergencyperiod.
function_setEmergencyPeriod(boolactive)internal{
require(block.timestamp<_emergencyPeriodEndDate,"EMERGENCY_PERIOD_FINISHED");
_emergencyPeriodActive=active; emitEmergencyPeriodChanged(active); } Figure14.1:
lib/helpers/EmergencyPeriod.sol#L53-L57 functionsetEmergencyPeriod(boolactive)externalauthenticate{
_setEmergencyPeriod(active); } Figure14.2: pools/BasePool.sol#L186-L188
functionsetEmergencyPeriod(boolactive)externalauthenticate{ _setEmergencyPeriod(active); }
Figure14.3: vault/Vault.sol#L69-L71 ExploitScenario
Evebecomesanadminbeforetheemergencyperiodhasended.Evemonitorsthe
mempoolfortransactionsexecutedbyBob.Whenshefindsone,shefront-runsthe
transaction,quicklyenablestheemergencyperiod,andthendeactivatesthatperiod. Recommendations
Shortterm,documenttheemergencyperiod-relatedabilitiesofanadminsothatuserswill bewellinformed.
Longterm,clearlydocumenttheabilitiesoftheprivilegedrolesthroughoutthesystem.
Thatway,userswillnotbesurprisedwhenothersexercisetheirprivileges.
2021hasheyeBalancerV2Assessment|31

6.StableMath._calcOutGivenInmayallowfreeswaps Severity:MediumDifficulty:High
Type:DataValidationFindingID:TOB-BALANCER-006 Target: StableMath.sol Description Theout-given-
infunctionhasroundingtocalculateswaps,whichcouldenableanattacker toobtaintokensatnocost.
StableMath._calcOutGivenIn computesthenumberoftokensthatoneuserwillreceive
basedonthenumberoftokenssent: function_calcOutGivenIn(uint256amplificationParameter,
uint256[]memorybalances, uint256tokenIndexIn, uint256tokenIndexOut, uint256tokenAmountIn
)internalpurereturns(uint256){
/**
//outGivenIntokenxfory-polynomialequationtosolve// //ay=amountouttocalculate//
//by=balancetokenout// //y=by-ay(finalBalanceOut)// //D=invariantDD^(n+1)//
//A=amplificationcoefficienty^2+(S----------D)*y-------------=0//
//n=numberoftokens(A*n^n)A*n^2n*P// //S=sumoffinalbalancesbuty// //P=productoffinalbalancesbuty//
**/
//Amountout,sowerounddownoverall.

uint256invariant=_calculateInvariant(amplificationParameter,balances);
balances[tokenIndexIn]=balances[tokenIndexIn].add(tokenAmountIn);
uint256finalBalanceOut=_getTokenBalanceGivenInvariantAndAllOtherBalances(amplificationParameter,
balances, invariant, tokenIndexOut);
balances[tokenIndexIn]=balances[tokenIndexIn].sub(tokenAmountIn);
2021hasheyeBalancerV2Assessment|32

returnbalances[tokenIndexOut].sub(finalBalanceOut).sub(1); } Figure6.1:
pools/stable/StableMath.sol#L89-L124 _calcOutGivenIn performs: ▯▯▯▯▯▯▯▯▯ =▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯(▯▯▯, ▯
▯▯▯▯▯▯▯▯) ▯▯▯▯▯▯▯▯▯[▯▯▯▯▯▯▯▯] = ▯▯▯▯▯▯▯▯▯[▯▯▯▯▯▯▯▯] + ▯▯▯▯▯▯▯▯▯▯▯▯▯▯ ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯ = ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
[..]▯▯▯▯▯▯▯▯▯(▯▯▯, ▯▯▯▯▯▯▯▯▯, ▯▯▯▯▯▯▯▯▯▯▯▯▯, ▯▯▯▯▯▯▯▯▯▯▯) _calculateInvariant and
_getTokenBalanceGivenInvariantAndAllOtherBalances
containoperationsthatcanroundupordownaccordingtotheirvalues. Forexample,
_getTokenBalanceGivenInvariantAndAllOtherBalances uses FixedPoint.mul
,whichcanroundupordownbasedonthevalue: tokenBalance=tokenBalance.mul(tokenBalance) .add(c) .divUp(
Math.mul(tokenBalance,2) .add(b) .sub(invariant)); Figure6.2: pools/stable/StableMath.sol#L489
Asaresult, _calcOutGivenIn canroundinadirectionthatallowsanattackertoreceive
freetokensbyeitherswappingwith0 tokenin forasmallamountof tokenout or
accumulatingdustthroughcallsto StableMath._calcOutGivenIn.
Weclassifiedthisissueasoneofmediumseverity,aswewerenotabletogeneratefree
tokenswithlargebalances.However,thatmightstillbepossibleincertainedgecases. ExploitScenario • Ampis
4168774334271564283904 . • Thepoolhas4tokens,andthebalancesare 35303029 , 1524785991 , 323536 ,and
323534 . • Eveswaps62,439,391ofthetokensinindex1for0tokensinindex0. Recommendations ShortTerm •
Remove FixedPoint.mul and FixedPoint.div ,andensurethateveryfunctionuses
theappropriateroundingdirection. • Createtwoversionsof _calculateInvariant and
_getTokenBalanceGivenInvariantAndAllOtherBalances thatroundupordown,
respectively,andusethemappropriately. 2021hasheyeBalancerV2Assessment|33

Longterm,useEchidnawhendesigninganewpooltoensurethattheswap/join/exit
operationsarerobustagainstroundingissues. 2021hasheyeBalancerV2Assessment|34

7.StableMath._calcInGivenOutmayallowfreeswaps Severity:MediumDifficulty:High
Type:DataValidationFindingID:TOB-BALANCER-007 Target: StableMath.sol Description Thein-given-
outfunctionhasroundingtocalculateswaps,whichcouldenableanattacker toobtaintokensatnocost.
StableMath._calcOutGivenOut computesthenumberoftokensthatoneuserwillreceive
basedonthenumberoftokenssent: function_calcInGivenOut(uint256amplificationParameter,
uint256[]memorybalances, uint256tokenIndexIn, uint256tokenIndexOut, uint256tokenAmountOut
)internalpurereturns(uint256){
/**
//inGivenOuttokenxfory-polynomialequationtosolve// //ax=amountintocalculate// //bx=balancetokenin//
//x=bx+ax(finalBalanceIn)// //D=invariantDD^(n+1)// //A=amplificationcoefficientx^2+(S----------
D)*x-------------=0// //n=numberoftokens(A*n^n)A*n^2n*P// //S=sumoffinalbalancesbutx//
//P=productoffinalbalancesbutx//
**/
//Amountin,soweroundupoverall.
uint256invariant=_calculateInvariant(amplificationParameter,balances);
balances[tokenIndexOut]=balances[tokenIndexOut].sub(tokenAmountOut);
uint256finalBalanceIn=_getTokenBalanceGivenInvariantAndAllOtherBalances(amplificationParameter,
balances, invariant, tokenIndexIn);
balances[tokenIndexOut]=balances[tokenIndexOut].add(tokenAmountOut);
2021hasheyeBalancerV2Assessment|35

returnfinalBalanceIn.sub(balances[tokenIndexIn]).add(1); } Figure7.1:
pools/stable/StableMath.sol#L129-L164 _calcOutGivenOut performs: ▯▯▯▯▯▯▯▯▯ =▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯(▯▯▯,
▯▯▯▯▯▯▯▯▯) ▯▯▯▯▯▯▯▯▯[▯▯▯▯▯▯▯▯▯] = ▯▯▯▯▯▯▯▯▯[▯▯▯▯▯▯▯▯▯] - ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯ ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯ = ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯
[..]▯▯▯▯▯▯▯▯▯(▯▯▯, ▯▯▯▯▯▯▯▯▯, ▯▯▯▯▯▯▯▯▯▯▯▯▯, ▯▯▯▯▯▯▯▯▯▯▯) _calculateInvariant and
_getTokenBalanceGivenInvariantAndAllOtherBalances
containoperationsthatcanroundupordownaccordingtotheirvalues. Forexample,
_getTokenBalanceGivenInvariantAndAllOtherBalances uses FixedPoint.mul
,whichcanroundupordownbasedonthevalue: tokenBalance=tokenBalance.mul(tokenBalance) .add(c) .divUp(
Math.mul(tokenBalance,2) .add(b) .sub(invariant)); Figure7.2: pools/stable/StableMath.sol#L489
Asaresult, _calcOutGivenOut canroundinadirectionthatallowsanattackertoreceive
freetokensbyaccumulatingdustwithcallsto StableMath._calcOutGivenOut .
Weclassifiedthisissueasoneofmediumseverity,aswewerenotabletogeneratefree
tokenswithlargebalances.However,thatmightstillbepossibleincertainedgecases. ExploitScenario •
Ampis10**18. • Thebalancesare20,369,17,465,037,809,and1. •

Eveswaps1weiofthetokensinindex0andreceives37,808weiofthetokensin index2. • Evethenswaps37,808weiofthetokensinindex2andreceives20,369weiofthe tokensinindex0. • Asaresult,Evereceives20,368weiofthetokensinindex0forfree. Recommendations ShortTerm • Remove FixedPoint.mul and FixedPoint.div ,andensurethateveryfunctionuses theappropriateroundingdirection. 2021hasheyeBalancerV2Assessment|36

• Createtwoversionsof _calculateInvariant and _getTokenBalanceGivenInvariantAndAllOtherBalances thatroundupordown, respectively,andusethemappropriately. Longterm,useEchidnawhendesigninganewpooltoensurethattheswap/join/exit operationsarerobustagainstroundingissues. 2021hasheyeBalancerV2Assessment|37

8.StableMath._calcTokenInGivenExactBptOutmayallowanattackertojoin forfree Severity:MediumDifficulty:High Type:DataValidationFindingID:TOB-BALANCER-008 Target: StableMath.sol Description The"join"operationfortoken-in-for-exact-bpt-outhasrounding,whichcouldallowan attackertojoinapoolwithoutpayinganytokens. StableMath._calcTokenInGivenExactBptOut determinesthenumberoftokensthata usermustsendtoreceiveagivennumberofBpttokens(forasingleasset): function_calcTokenInGivenExactBptOut(uint256amp, uint256[]memorybalances, uint256tokenIndex, uint256bptAmountOut, uint256bptTotalSupply, uint256swapFee)internalpurereturns(uint256){ //Tokenin,soweroundupoverall. //Getcurrentinvariant uint256currentInvariant=_calculateInvariant(amp,balances); //Calculatenewinvariant uint256newInvariant= bptTotalSupply.add(bptAmountOut).divUp(bptTotalSupply).mulUp(currentInvariant); //Firstcalculatethesumofalltokenbalanceswhichwillbeusedtocalculate //thecurrentweightoftoken uint256sumBalances=0; for(uint256i=0;i<balances.length;i++){ sumBalances=sumBalances.add(balances[i]); } //getamountInAfterFee uint256newBalanceTokenIndex=_getTokenBalanceGivenInvariantAndAllOtherBalances(amp, balances, newInvariant, tokenIndex); uint256amountInAfterFee=newBalanceTokenIndex.sub(balances[tokenIndex]); 2021hasheyeBalancerV2Assessment|38

//GettokenBalancePercentageExcess uint256currentWeight=balances[tokenIndex].divDown(sumBalances); uint256tokenBalancePercentageExcess=currentWeight.complement(); uint256swapFeeExcess=swapFee.mulUp(tokenBalancePercentageExcess); returnamountInAfterFee.divUp(swapFeeExcess.complement()); } Figure8.1: pools/stable/StableMath.sol#L129-L164 _calcTokenInGivenExactBptOut performs: ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯ =▯▯▯▯ ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯(▯▯▯, ▯▯▯▯▯▯▯▯▯) ▯▯▯▯▯▯▯▯▯▯▯▯▯ = ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯ + ▯▯▯▯▯▯▯▯▯▯▯▯▯ ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯ ▯* ▯ ▯▯▯▯ ▯▯▯▯▯▯▯▯▯▯▯▯▯ ▯▯▯▯▯▯▯▯▯▯▯▯▯ = ▯ ∑▯▯▯▯▯▯▯▯▯[▯] ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯ = ▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯[..]▯▯▯▯▯▯▯▯▯(▯▯▯, ▯ ▯▯▯▯▯▯▯▯, ▯▯▯▯▯▯▯▯▯▯▯▯▯, ▯▯▯▯▯▯▯▯▯▯▯) _calculateInvariant and _getTokenBalanceGivenInvariantAndAllOtherBalances containoperationsthatcanroundupordownaccordingtotheirvalues. Forexample, _getTokenBalanceGivenInvariantAndAllOtherBalances uses FixedPoint.mul ,whichcanroundupordownbasedonthevalue. tokenBalance=tokenBalance.mul(tokenBalance) .add(c) .divUp(Math.mul(tokenBalance,2) .add(b) .sub(invariant)); Figure8.2: pools/stable/StableMath.sol#L489 Asaresult, _calcTokenInGivenExactBptOut canroundinadirectionthatallowsan attackertoreceiveBpttokenswithoutpayinganytokens. Weclassifiedthisissueasoneofmediumseverity,aswewerenotabletogeneratefree tokenswithlargebalances.However,thatmightstillbepossibleincertainedgecases. ExploitScenario • Ampis 287584007913129639935. • Thereare4tokensinthepool,eachofwhichhasabalanceof 100,000,000,000,000,000. • ThecurrentBptsupplyis5,690,710,977,914,755,780,762,205. 2021hasheyeBalancerV2Assessment|39

• Evejoinsthepooltoobtain1,000,021Bpttokenswithouthavingtomakea payment. Recommendations ShortTerm • Remove FixedPoint.mul and FixedPoint.div ,andensurethateveryfunctionuses theappropriateroundingdirection • Createtwoversionsof _calculateInvariant and _getTokenBalanceGivenInvariantAndAllOtherBalances thatroundupordown, respectively,andusethemappropriately. Longterm,useEchidnawhendesigninganewpooltoensurethattheswap/join/exit operationsarerobustagainstroundingissues. 2021hasheyeBalancerV2Assessment|40

9.BalancerStablePool'sinvariantcandifersignificantlyfromCurve's invariant Severity:UndeterminedDifficulty:Medium Type:UndefinedBehaviorFindingID:TOB-BALANCER-009 Target: StableMath.sol Description Balancerimplements StablePool withaformulabasedonCurve'sformula.Thetwo implementationsemploydifferentordersofoperationsandroundingstrategies,leadingto differentresults.Theimpactofthisdifferenceisunclearandrequiresfurtherinvestigation. Curve'sinvariantisasfollows: defget_D(xp:uint256[N_COINS],amp:uint256)→uint256: """ Dinvariantcalculationinnon-overflowingintegeroperations iteratively A*sum(x_i)*n**n+D=A*D*n**n+D** (n+1)/(n**n*prod(x_i)) Convergingsolution: D[j+1]=(A*n**n*sum(x_i)-D[j]** (n+1)/(n**nprod(x_i)))/(A*n**n-1) """ S:uint256=0 for_xinxp: S+=_x ifS==0: return0 Dprev:uint256=0 D:uint256=S Ann:uint256=amp*N_COINS for_iinrange(255): D_P:uint256=D for_xinxp:

D_P=D_P*D/(_x*N_COINS+1)#+1istoprevent/0 Dprev=D D=(Ann*S/A_PRECISION+D_P*N_COINS)*D/((Ann-A_PRECISION)*D/ A_PRECISION+(N_COINS+1)*D_P) #Equalitywiththeprecisionof1 ifD>Dprev: ifD-Dprev⩽1: returnD else: ifDprev-D⩽1: returnD
#convergencetypicallyoccursin4roundsorless,thisshouldbeunreachable!

#ifitdoeshappenthepoolisborkedandLPscanwithdrawvia`remove_liquidity` raise
Figure9.1:Curve’sinvariant Balancer’sinvariantisasfollows:
function_calculateInvariant(uint256amplificationParameter,uint256[]memorybalances) internal pure returns(uint256) {
/** //invariant//
//D=invariantD^(n+1)// //A=amplificationcoefficientAn^nS+D=ADn^n+----------//
//S=sumofbalancesn^nP// //P=productofbalances// //n=numberoftokens//
**/
//Weroundupinvariant. uint256sum=0; uint256numTokens=balances.length;
for(uint256i=0;i<numTokens;i++){ sum=sum.add(balances[i]); } if(sum==0){ return0; }
uint256prevInvariant=0; uint256invariant=sum;
uint256ampTimesTotal=Math.mul(amplificationParameter,numTokens); for(uint256i=0;i<255;i++){
uint256P_D=Math.mul(numTokens,balances[0]); for(uint256j=1;j<numTokens;j++){
P_D=Math.divUp(Math.mul(Math.mul(P_D,balances[j]),numTokens),invariant); } prevInvariant=invariant;
invariant=Math.divUp(Math.mul(Math.mul(numTokens,invariant),
invariant).add(Math.mul(Math.mul(ampTimesTotal,sum),P_D)),
Math.mul(numTokens.add(1),invariant).add(Math.mul(ampTimesTotal.sub(1),P_D)));
if(invariant>prevInvariant){ if(invariant.sub(prevInvariant)⩽1){

break; } }elseif(prevInvariant.sub(invariant)⩽1){ break; } } returninvariant; }
Figure9.2:Balancer’sinvariant, pools/stable/StableMath.sol#L36-L84
WhileCurveandBalancerusethesameformula,theyusedifferentordersofoperations androundingmethods.Additionally,Curverevertsiftheprecisionloopdoesnotconverge after255iterations,whileBalancerreturnsthevalue,potentiallywithasignificantlossof precision. WeimplementedadifferentialfuzzerthroughEchidna(describedinAppendixG),which showedthattheimplementationscanreturnsignificantlydifferentvalues.Forexample, with4tokenswithbalancesof1;1;54,066;and108,075,532andanamplitudeof 83,437,555,Curvereturns964,800,whileBalancerreturns101,906,431,leadingtoa differenceinmagnitudeof10**8.
Thisdifferencemeritsfurtherconsideration,asitisunclearwhetheritcouldleadto unexpectedarbitrageorotherissues. ExploitScenario
EveexploitsunexpectedarbitrageopportunitiesbetweenBalancerandCurve,generatinga profit.BobnoticestheoperationandlosesconfidenceinBalancer’sarithmetic. Recommendations
Shortterm,investigatethedifferencesbetweentheBalancerandCurveinvariants,and evaluatetheirimpactsonarbitrageopportunitiesandpotentialassociatedrisks.
Longterm,usedifferentialfuzzingwhendesigningasystemforwhichanother implementationexists.

13.StablePool’sinvariantisnotmonotonicallyincreasing Severity:LowDifficulty:High
Type:DataValidationFindingID:TOB-BALANCER-013 Target: StableMath.sol Description
StableMath._calculateInvariant returnsthepool’sinvariant,whichindicatesthevalue ofthepool.Theinvariantisexpectedtobemonotonicallyincreasingbutisnot. StableMath._calculateInvariant returnsthepool’sinvariantandisanapproximationof thefollowing: Figure13.1:Thepool’sinvariant
function_calculateInvariant(uint256amplificationParameter,uint256[]memorybalances) internal pure returns(uint256) {
/** //invariant//
//D=invariantD^(n+1)// //A=amplificationcoefficientAn^nS+D=ADn^n+----------//
//S=sumofbalancesn^nP// //P=productofbalances// //n=numberoftokens//
**/
//Weroundupinvariant. uint256sum=0; uint256numTokens=balances.length;
for(uint256i=0;i<numTokens;i++){ sum=sum.add(balances[i]); } if(sum==0){ return0; }

uint256prevInvariant=0; uint256invariant=sum;
uint256ampTimesTotal=Math.mul(amplificationParameter,numTokens); for(uint256i=0;i<255;i++){
uint256P_D=Math.mul(numTokens,balances[0]); for(uint256j=1;j<numTokens;j++){
P_D=Math.divUp(Math.mul(Math.mul(P_D,balances[j]),numTokens),invariant); } prevInvariant=invariant;
invariant=Math.divUp(Math.mul(Math.mul(numTokens,invariant),
invariant).add(Math.mul(Math.mul(ampTimesTotal,sum),P_D)),

Math.mul(numTokens.add(1),invariant).add(Math.mul(ampTimesTotal.sub(1),P_D)));
if(invariant>prevInvariant){ if(invariant.sub(prevInvariant)⩽1){ break; }
}elseif(prevInvariant.sub(invariant)⩽1){ break; } } returninvariant; } Figure13.2:
pools/stable/StableMath.sol#L36-L84
Thisinvariantisexpectedtoincreaseifthebalanceofthepoolincreases.Duetothe incorrectapplicationof
_calculateInvariant ,thisassumptioncanbebroken.Asaresult, thepool'sinvariantcoulddecreaseovertime.
ExploitScenario Bobdeploysastablepoolwith4tokens.Theirbalancesare1;2;
17,464,062,808,818,790,875;and14,953,662,333,476,747,andthe amp parameteris
1,001,313,087,410,729,399.
Thecurrentinvariantis6,600,767,538,511,083,210.Thebalanceofeachtokenincreasesby
1wei.Thepoolinvariantisnow6,600,767,538,511,083,209(i.e.,theoriginalinvariant,less
1).Asaresult,thepoolhaslostvalue,whilethebalancehasincreased. Recommendations
Shortterm,applythecorrectroundingin StableMath._calculateInvariant.
Longterm,identifythesystem'sinvariants,anduseEchidnatocheckitsrobustness.
2021hasheyeBalancerV2Assessment|45

A.VulnerabilityClassifications VulnerabilityClasses ClassDescription
AccessControlsRelatedtoauthorizationofusersandassessmentofrights
AuditingandLoggingRelatedtoauditingofactionsorloggingofproblems
AuthenticationRelatedtotheidentificationofusers
ConfigurationRelatedtosecurityconfigurationsofservers,devices,or software
CryptographyRelatedtoprotectingtheprivacyorintegrityofdata
DataExposureRelatedtounintendedexposureofsensitiveinformation
DataValidationRelatedtoimproperrelianceonthestructureorvaluesofdata
DenialofServiceRelatedtocausingasystemfailure
ErrorReportingRelatedtothereportingoferrorconditionsinasecurefashion
PatchingRelatedtokeepingsoftwareuptodate
SessionManagementRelatedtotheidentificationofauthenticatedusers
TimingRelatedtoraceconditions,locking,ortheorderofoperations
UndefinedBehaviorRelatedtoundefinedbehaviortriggeredbytheprogram SeverityCategories
SeverityDescription InformationalTheissuedoesnotposeanimmediateriskbutisrelevanttosecurity
bestpracticesorDefenseinDepth. UndeterminedTheextentoftheriskwasnotdeterminedduringthisengagement.
LowTheriskisrelativelysmallorisnotariskthecustomerhasindicatedis important.
MediumIndividualusers'informationisatrisk;exploitationcouldpose 2021hasheyeBalancerV2Assessment|46

reputational,legal,ormoderatefinancialriskstotheclient.
HighTheissuecouldaffectnumeroususersandhaveseriousreputational,
legal,orfinancialimplicationsfortheclient. DifficultyLevels DifficultyDescription
UndeterminedThedifficultyofexploitationwasnotdeterminedduringthis engagement.
LowCommonlyexploitedpublictoolsexist,orsuchtoolscanbescripted.
MediumAnattackermustwriteanexploitorwillneedin-depthknowledgeofa complexsystem.
HighAnattackermusthaveprivilegedinsideraccesstothesystem,may
needtoknowextremelycomplextechnicaldetails,ormustdiscover otherweaknessestoexploitthisissue.
2021hasheyeBalancerV2Assessment|47

B.CodeMaturityClassifications CodeMaturityClasses CategoryNameDescription
AccessControlsRelatedtotheauthenticationandauthorizationofcomponents
ArithmeticRelatedtotheproperuseofmathematicaloperationsand semantics
AssemblyUseRelatedtotheuseofinlineassembly
CentralizationRelatedtotheexistenceofasinglepointoffailure
CodeStabilityRelatedtotherecentfrequencyofcodeupdates UpgradeabilityRelatedtocontractupgradeability
Function Composition Relatedtoseparationofthelogicintofunctionswithclearpurposes Front-
RunningRelatedtoresilienceagainstfront-running
KeyManagementRelatedtotheexistenceofproperproceduresforkeygeneration, distribution,andaccess
MonitoringRelatedtotheuseofeventsandmonitoringprocedures
SpecificationRelatedtotheexpectedcodebasedocumentation Testing& Verification
Relatedtotheuseoftestingtechniques(unittests,fuzzing,symbolic execution,etc.) RatingCriteria
RatingDescription StrongThecomponentwasreviewed,andnoconcernswerefound.
SatisfactoryThecomponenthadonlyminorissues. ModerateThecomponenthadsomeissues.
WeakThecomponentledtomultipleissues;moreissuesmightbepresent. 2021hasheyeBalancerV2Assessment|48

MissingThecomponentwasmissing. NotApplicableThecomponentisnotapplicable.
NotConsideredThecomponentwasnotreviewed. Further Investigation Required
Thecomponentrequiresfurtherinvestigation. 2021hasheyeBalancerV2Assessment|49

C.CodeQualityRecommendations
Thefollowingrecommendationsarenotassociatedwithspecificvulnerabilities.However,

theyenhancecodereadabilityandmaypreventtheintroductionofvulnerabilitiesinthe future. PoolRegistry • Removethecounterlibrary.Thelibraryisusedonlytwice,oncetoreaditsvalue andoncetoincrementit.Usingalibraryforthesepurposesiscumbersome. BasePool • RemoveBasePoolAuthorizationfromtheBasePool'sconstructor.Thereisno constructorfor BasePoolAuthorization (or Authentication),butusersmay incorrectlyassumethatcallingitwillresultininitialization. • Consideremittinganeventwiththeregisteredpoolparametersinthe constructor.Thiswillmakeiteasyforoff-chainsystemstomonitorthecreationof newpools. BasePoolFactory • RemoveunusedimportIBasePool.Thecontractdoesnotusethisinterface. StablePoolUserDataHelpers • RenameminBPTAmountInasminBPTAmountOutinexactTokensInForBptOut. TheABI-decodedargumentsignifiesBPTout,notin. WeightedPoolUserDataHelpers • RenameminBPTAmountInasminBPTAmountOutinexactTokensInForBptOut. TheABI-decodedargumentsignifiesBPTout,notin. CodeSizeOptimizations Thecontractsareapproachingthecodesizelimit.Werecommendmakingthefollowing changestoreducethecontracts’codesize: • Replacemodifierswithinternalcalls.Thecompilergeneratesboilerplatecodefor modifiersthatisnotoptimizedandleadstocodeduplication.Replacingmodifiers withinternalcallswillreducethecodesizeinsomecases(see#6584). 2021hasheyeBalancerV2Assessment|50

• Remove trackExempt fromAssetTransfersHandler._sendAsset.Theparameteris false inallofthecalls(asidefromthoseinthemockcontracts)andcanbe removed. • Removethebitmaskfrom PoolRegistry._getPoolAddress .Thebitwiseshifting totherightalreadyensuresthattheleadingbitsarezero.Moreover,thevalueis casttoanaddressthatappliesthesamemask. 2021hasheyeBalancerV2Assessment|51

D.TokenIntegrationChecklist Thefollowingchecklistprovidesrecommendationsforinteractionswitharbitrarytokens. Everyuncheckeditemshouldbejustified,anditsassociatedrisks,understood.An up-to-dateversionofthechecklistcanbefoundincrytic/building-secure-contracts. Forconvenience,allSlitherutilitiescanberundirectlyonatokenaddress,suchasthe following: slither-check-erc0xdac17f958d2ee523a2206206994597c13d831ec7TetherToken Tofollowthischecklist,usethebelowoutputfromSlitherforthetoken: -slither-check-erc[target] [contractName][optional:--ercERC_NUMBER] -slither[target]--printhuman-summary -slither[target]--printcontract-summary -slither-prop.--contractContractName#requiresconfiguration,anduseof EchidnaandManticore GeneralSecurityConsiderations ☐ Thecontracthasasecurityreview.Avoidinteractingwithcontractsthatlacka securityreview.Checkthelengthoftheassessment(i.e.,thelevelofeffort),the reputationofthesecurityfirm,andthenumberandseverityofthefindings. ☐ Youhavecontactedthedevelopers.Youmayneedtoalerttheirteamtoan incident.Lookforappropriatecontactsonblockchain-security-contacts. ☐ Theyhaveasecuritymailinglistforcriticalannouncements.Theirteamshould adviseusers(likeyou!)whencriticalissuesarefoundorwhenupgradesoccur. ERCConformity Slitherincludesautility, slither-check-erc ,thatreviewstheconformanceofatokento manyrelatedERCstandards.Useslither-check-erctoreviewthefollowing: ☐ Transfer and transferFrom returnaboolean.Severaltokensdonotreturna booleanonthesefunctions.Asaresult,theircallsinthecontractmightfail. ☐ Thename , decimals ,and symbol functionsarepresentifused.Thesefunctions areoptionalintheERC20standardandmaynotbepresent. ☐ Decimals returnsa uint8 .Severaltokensincorrectlyreturna uint256 .Insuch 2021hasheyeBalancerV2Assessment|52

cases,ensurethatthevaluereturnedisbelow255. ☐ ThetokenmitigatestheknownERC20racecondition.TheERC20standardhasa knownERC20raceconditionthatmustbemitigatedtopreventattackersfrom stealingtokens. ☐ ThetokenisnotanERC777tokenandhasnoexternalfunctioncallin transfer and transferFrom .Externalcallsinthetransferfunctionscanleadtoreentrancies. Slitherincludesautility, slither-prop ,thatgeneratesunittestsandsecurityproperties thatcandiscovermanycommonERCflaws.Useslither-proptoreviewthefollowing: ☐ Thecontractpassesallunittestsandsecuritypropertiesfrom slither-prop . RunthegeneratedunittestsandthencheckthepropertieswithEchidnaand Manticore. Finally,therearecertaincharacteristicsthataredifficulttoidentifyautomatically.Conduct amanualreviewofthefollowingconditions: ☐ Transfer and transferFrom shouldnottakeafee.Deflationarytokenscanleadto unexpectedbehavior. ☐ Potentialinterestearnedfromthetokenistakenintoaccount.Sometokens distributeinteresttotokenholders.Thisinterestmaybetrappedinthecontractif nottakenintoaccount. ContractComposition ☐ Thecontractavoidsunnecessarycomplexity.Thetokenshouldbeasimple contract;atokenwithcomplexcoderequiresahigherstandardofreview.Use Slither’s human-summary printertoidentifycomplexcode. ☐ Thecontractuses SafeMath .Contractsthatdonotuse SafeMath

requireahigher standardofreview.Inspectthecontractbyhandfor SafeMath usage. ▯ Thecontracthasonlyafewnon-token-relatedfunctions.Non-token-related functionsincreasethelikelihoodofanissueinthecontract.UseSlither’s contract-summary printertobroadlyreviewthecodeusedinthecontract. ▯ Thetokenhasonlyoneaddress.Tokenswithmultipleentrypointsforbalance updatescanbreakinternalbookkeepingbasedontheaddress(e.g., balances[token_address] [msg.sender]maynotreflecttheactualbalance). OwnerPrivileges ▯ Thetokenisnotupgradeable.Upgradeablecontractsmaychangetheirrulesover time.UseSlither’s human-summary printertodetermineifthecontractis upgradeable. 2021hasheyeBalancerV2Assessment|53

▯ Theownerhaslimitedmintingcapabilities.Maliciousorcompromisedowners canabusemintingcapabilities.UseSlither’s human-summary printertoreview mintingcapabilities,andconsidermanuallyreviewingthecode. ▯ Thetokenisnotpausable.Maliciousorcompromisedownerscantrapcontracts relyingonpausabletokens.Identifypausablecodebyhand. ▯ Theownercannotblacklistthecontract.Maliciousorcompromisedownerscan trapcontractsrelyingontokenswithablacklist.Identifyblacklistingfeaturesby hand. ▯ Theteambehindthetokenisknownandcanbeheldresponsibleforabuse. Contractswithanonymousdevelopmentteamsorteamsthatresideinlegalshelters requireahigherstandardofreview. TokenScarcity Reviewsoftokenscarcityissuesmustbeexecutedmanually.Checkforthefollowing conditions: ▯ Thesupplyisownedbymorethanafewusers.Ifafewusersownmostofthe tokens,theycaninfluenceoperationsbasedonthetokens’repartition. ▯ Thetotalsupplyissufficient.Tokenswithalowtotalsupplycanbeeasily manipulated. ▯ Thetokensarelocatedinmorethanafewexchanges.Ifallthetokensareinone exchange,acompromiseoftheexchangecouldcompromisethecontractrelyingon thetoken. ▯ Usersunderstandtherisksassociatedwithalargeamountoffundsorflash loans.Contractsrelyingonthetokenbalancemustaccountforattackerswitha largeamountoffundsorattacksexecutedthroughflashloans. ▯ Thetokendoesnotallowflashminting.Flashmintingcanleadtosubstantial swingsinthebalanceandthetotalsupply,whichnecessitatestrictand comprehensiveoverflowchecksintheoperationofthetoken. 2021hasheyeBalancerV2Assessment|54

E.RisksAssociatedwithArbitraryPools Balanceraimstoacceptarbitrarypools,thecodeofwhichhasnotbeenvettedby developers.Arbitrarypoolsintroduceadditionalissuesthatcouldallowanattackertosteal users'funds.Werecommendthatusersreviewthepoolcodetoensureitbehavesas expected.Poolsshouldmeetthefollowingcriteria: • Theyshouldnotbeupgradeable.Upgradeablepoolshaveinheritedrisks. • Theyshouldbeunabletoself-destruct.Destructiblepoolshaveinheritedrisks; forexample,theymaybesubjecttomaliciousupgradesthrough create2 . • Poolsshouldhaveaclearfeeschemaandshouldnotallowuserstostealfunds usinghighfees.Usersmustbeawareoftheamountoffeesthatpoolsearnonand takefromtransactions. • Poolsshouldnothavecallsto deregisterTokens .Ifapoolcancall deregisterTokens ,itmaybeabletotrapusers’funds. • Pooltokens’weightshouldbeimmutableandbounded.Otherwise,anattacker couldabuseupdatestoatoken’sweighttogainaprofit. • Tokenswithdecimals≠18shouldbehandledcorrectly.Tokenswithdifferent decimals canleadtoincorrectpricecomputations. • Poolsshouldhavedocumentedparameters.Eachpool’sparametersmustbe documentedandmadecleartousers. • Poolsshouldfavorimmutableparameters.Immutableparametersreducethe risksassociatedwithprivilegedusers. 2021hasheyeBalancerV2Assessment|55

F.CheckingnonReentrantModifierswithSlither TheBalancercodebasehasmanyfunctionsthatcaninteractwiththird-partypools.Asa result,mostofthefunctionsareprotectedbya nonReentrant modifier.WeusedSlitherto identifyandreviewthefunctionsthatarenotprotectedbyamodifier. Thefollowingscriptusesawhitelistapproach,inwhicheveryreachablefunctionmustbe eitherprotectedwitha nonReentrant modifierorwhitelisted. Noissuewasfoundusingthescript. fromslitherimportSlither fromslither.core.declarationsimportContract fromtypingimportList contracts=Slither(".",ignore_compile=True) def_check_access_controls(contract:Contract,modifiers_access_controls:List[str],whitelist:List[str]): print(f"###Check{contract}accesscontrols") no_bug_found=True forfunctionincontract.functions_entry_points: iffunction.is_constructor: continue iffunction.view: continue ifnotfunction.modifiersor(notany((str(x)inmodifiers_access_controls)forxinfunction.modifiers)): ifnotfunction.nameinwhitelist: print(f"\t-{function.canonical_name}shouldhaveannonre-eentrant modifier") no_bug_found=False ifno_bug_found: print("\t-Nobugfound") _check_access_controls(

contracts.get_contract_from_name("Vault"), ["nonReentrant"],
["queryBatchSwap","receive","setEmergencyPeriod"],)

G.DiferentialFuzzingonBalancer<>Curve
TheBalancercodebaseimplementsastablepoolwithaformulabasedonCurve’s formulas.UsingEchidna,wedevelopedadifferentialfuzzertofuzztestthetwo implementations. Thedifferentialfuzzerrunsboth theCurveandBalancerinvariantfunctionsandlooksfor casesinwhichthefunctionsreturnvalueswithadifferenceofatleast10**8. WediscoveredissueTOB-BALANCER-09usingthedifferentialfuzzer. import"./StableMath.sol"; interfaceCurve{
functionget_D(uint256[4]memory,uint256)externalviewreturns(uint256); }
contractDifferentialFuzzingisStableMath{ usingFixedPointforuint256;
uint256internalconstantA_PRECISION=100;//Curveamplificationprecision Curvec; constructor()public{
addresscurve_addr; bytesmemorycurve_bytecode=
hex"610330566004361015610 00d57610326565b600035601c52600051341561002157600080fd5b63c763592c81
1415610324576000610140526101806000600481835 2015b602061018051026004013561016052610140805 16101
605181818301101561006657600080fd5b808201905090508152505b815160010180835281141561003f575b5050
6101405115156100985760006000526020600 0f35b600061016052610140516101805260843560048082 02821582
8483041417610 0bf57600080fd5b809050905090506101a0526101c0600060ff818352015b610180516101e05261
0220600060048183520 15b60206102205102600401356102005261 01e05161018051808202821582848304141761
011557600080fd5b8090509050905061020051600480820282158284830414176101365760 0080fd5b8090509050
9050600181818301101561014e57600080fd5b808201905090508080610160576 00080fd5b82049050905 06101e0
525b81516001018083528114156100ea575b505061018051610160526101a051610140518082028215828483 0414
176101a157600080fd5b80905090509050606480820490509050 6101e051600480820282158284830414176101cb
57600080fd5b80905090509050818183011015610 1e157600080fd5b808201905090506101805180820282158284
830414176102005760 0080fd5b809050905090506101a0516064808210156 1021a57600080fd5b808203905 09050
6101805180820282158284830414176102395760 0080fd5b80905090509050606480820490509050600561 01e051
80820282158284830414176102635760 0080fd5b80905090509050818183011015610279576 00080fd5b80820190
5090508080610 28b57600080fd5b82049050905061018052610160516101805111156102da576001610180516101
6051808210156102ba57600080fd5b808203905090501115156102d557610180516000526020600 0f35b61030c56
5b600161016051610180518082101561 02f157600080fd5b808203905090501115156 1030c57610180516000 5260

206000f35b81516001018083528114156100d6575b50506000 6000fd5b505b60006000fd5b61000461033003 6100
0460003961000461033003600 0f3"; uintsize=curve_bytecode.length; assembly{
curve_addr:=create(0,0xa0,size) } c=Curve(curve_addr); } uintlimit=10**8;
functiontest(uint[4]memorybalances,uintamp)public{
uintcurve_value=c.get_D(balances,amp*A_PRECISION); uint[]memorybalances_dynamic_array=newuint[](4);
balances_dynamic_array[0]=balances[1]; balances_dynamic_array[1]=balances[1];
balances_dynamic_array[2]=balances[2]; balances_dynamic_array[3]=balances[3];
uintbalance_value=_calculateInvariant(amp,balances_dynamic_array); if(curve_value>balance_value){
assert(curve_value-balance_value<limit); } if(curve_value<balance_value){ assert(balance_value-curve_value<limit); } } } FigureG:Echidna-baseddifferentialfuzzer.

H.Fixed-PointRoundingRecommendations TheBalancercodebaseusesfixed-pointarithmetic.Ourinitialrecommendationsregarding theroundingstrategiestoapplytothe WeightedPool areincludedbelow.We recommendedapplyingastrategyofroundingdownorupsuchthattheoperationsare alwaysbeneficialtothepool.
Theprocessofdeterminingtheroundingdirectionforeachoperationisdescribedbelow. RoundingPrimitives The mul and div operationsarearithmeticprimitivesofthefixedpoint: □□□□□□□□□□□□□□(□,□) = □ * □ + 10**18 2 10**18 functionmul(uint256a,uint256b)internalpurereturns(uint256){ uint256c0=a*b;
require(a==0||c0/a==b,"ERR_MUL_OVERFLOW"); uint256c1=c0+(ONE/2);
require(c1≥c0,"ERR_MUL_OVERFLOW"); uint256c2=c1/ONE; returnc2; } FigureF.1:
math/FixedPoint.sol#L82-L89 □□□□□□□□□□□□□□(□,□) = □ * 10**18 + □ 2 □
functiondiv(uint256a,uint256b)internalpurereturns(uint256){ require(b≠0,"ERR_DIV_ZERO");
uint256c0=a*ONE; require(a==0||c0/a==ONE,"ERR_DIV_INTERNAL");//muloverflow uint256c1=c0+(b/2);
require(c1≥c0,"ERR_DIV_INTERNAL");//addrequire uint256c2=c1/b; returnc2; } FigureF.2:
math/FixedPoint.sol#L100-L108
Everyarithmeticoperation(forbothswapsandliquidity)isbasedontheseprimitives.

Fixed-PointPrimitives Innon-fixed-pointarithmetic,multiplicationdoesnotloseprecision(ifweignoreoverflow).
Onlydivisionoperationsneedthetwoprimitivestoapproximateresults. Roundingdownisthedefaultbehavior: □□□ □□□□ (□,□) = □ □ Thebelowapproximationcanbeusedtoroundupadivisionoperation.(SeeNumber Conversion,RolandBackhouse.) □□□ □□ (□,□) = □ +□ -1 □

andcanthenbeusedasprimitivestobuildthefollowing: □□□ □□□□ □□□ □□ ● □□□□□□□□□□□□□ □□ ● □□□□□□□□□□□□□ □□□□ ● □□□□□□□□□□□□□ □□□□ ● □□□□□□□□□□□□□ □□ DeterminingtheRoundingDirection Todeterminetheroundingdirection(i.e.,upordown),onecanreasonoutevery operation'soutcome. Forexample, _inGivenOut calculatesthenumberof aI Intokensthatmustbepaidto receive a0 Outtokens: □□ = □□ * ((□ □ □□ - □0) □0 □□) - 1) Tobenefitthepool, aI musttendtowardahighvalue(□),resultinginthefollowing: ● must □ □□ * ((□□ □□ - □0) □0 □□) - 1) ● must □ ((□□ □□ - □0) □0 □□) - 1) ● must □ (□□ □□ - □0) □0 □□

2021hasheyeBalancerV2Assessment|60

○ >=1 □□ □□ - □0 ● must□(Seethe"PowerRounding"sectionbelow.) □0 □□ ● must□ □□ □□ - □0 Sothefollowingformulamustapply: □□ = □□ * □ ((□□ □□ - □0 □) □0 □□ □) - 1) Thesameanalysiscanbeappliedinallofthesystem'sformulas. PowerRounding Severaloperationsrequirethecomputationof.Thefollowingdescribestherulesfor □ □ □ applyingroundingon: □ □ ● Ifc ⩾ 1, ○ □ requires □ and □ □ □ □ □ ○ □ requires □ . □ □ □ □ □ ● Ifc<1, ○ □ requires □ and □ □ □ □ □ ○ □ requires □ . □ □ □ □ □ Notethatthisdoesnotapplytoroundinginthe pow function(LogExpMath.sol#L229). (1-x)vs.(x-1)Rounding Severaloperationsrequirethecomputationofor.Thefollowing (1-□)(□-1) describestherulesforapplyingtherounding: 2021hasheyeBalancerV2Assessment|61

● For, (1-□) ○□ requires □ and (1-□)□ ○□ requires □. (1-□)□ ● For , (□-1) ○□ requires □ and (□-1)□ ○□ requires□. (□-1)□ RoundingResults Thefollowingexplainswhichroundingdirectionshouldbeapplied,withoutconsideringthe fees. joinPool HowmuchamountmustbepaidtoreceivepoolAmountOutofBPTtokens(□□□□□ □□□□). □□□□□ □□□□ □ □□□□□ □□□□ = □ □□□□□□□□□□□□□□□ * □

□□□□□□□□□□□□□ □□□□□□□□□ □ exitPool HowmuchamountwillbereceivedinexchangeforpoolAmountInofBPT □□□□□ □□□□□□□□ sent(). □□□□□ □□□□□□□□ □ □□□□□ □□□□□□□□ = □□□□□□□□□□□□□□□ * □

□□□□□□□□□□□□ □□□□□□□□□ □ _tokenInForExactBPTOut HowmanytokensshouldbepaidforbptAmountOutofBPTtokens(). □□□□□ □□□□ □ □□□□□ □□□□ = □□□□□□□□□□□□□ * □ ((□□□□□□□□□□□□□ + □□□□□□□□□□□□ □□□□□□□□□□□□□ □) 1 □□□□□□□□□□□□□□□□□□□□ □ - 1) 2021hasheyeBalancerV2Assessment|62

_bptInForExactTokensOut Howmuchthatmustbepaidtoreceiveoftokens(). □□□ □□ □□□□□□□□□□ □ □□□ □□ □ □□□ □□ = □□□□□□□□□□□□□ * □ (1 - □ ∏ □ (1 - (□□□□□□□□□□ □ □□□□□□□ □ □) □□□□□□ □) _inGivenOut HowmuchaIIntokensmustbepaidtoreceivea0ofOuttokens(aI). □ □□ = □□ * □ ((□□ □□ - □0 □) □0 □□ □) - 1) _outGivenIn HowmuchaOOuttokensthatwillbereceivedwhenaIIntokensarepaid(a0). □) □0=□0 * □ (1 - (□□ □□ +□□ □) □□ □□ □ 2021hasheyeBalancerV2Assessment|63