HashEye

// SMART CONTRACT SECURITY AUDIT

Acronym Foundation

Security assessment by HashEye · prepared for Acronym Foundation

HASHEYE AUDITED

PROJECT	Acronym Foundation
CLIENT	Acronym Foundation
CATEGORY	Blockchain
PUBLISHED	December 1, 2023
REPORT ID	research-acronym-foundation-2023-12-01-9gi7xe

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-acronym-foundation-2023-12-01-9gi7xe.

AcronymFoundation SecurityAssessment December13,2023 Preparedfor: AcronymFoundationTeam AcronymFoundation Preparedby:PriyankaBoseandAnishNaik

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. 228ParkAveS#80688 NewYork,NY10003 https://www.hasheye.io info@hasheye.io hasheye 1AcronymFoundationSecurityAssessment PUBLIC

NoticesandRemarks CopyrightandDistribution Thisreportisconsideredbyhasheyetobepublicinformation;itislicensedtoAcronym Foundationunderthetermsoftheprojectstatementofworkandhasbeenmadepublicat AcronymFoundation'srequest.Materialwithinthisreportmaynotbereproducedor distributedinpartorinwholewithouttheexpresswrittenpermissionofhasheye. 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 2AcronymFoundationSecurityAssessment PUBLIC

TableofContents hasheye 3AcronymFoundationSecurityAssessment PUBLIC

ProjectSummary ContactInformation Thefollowingmanagerswereassociatedwiththisproject: DanGuido,AccountManagerSamGreenup,ProjectManager dan@hasheye.iosam.greenup@hasheye.io Thefollowingengineerswereassociatedwiththisproject: PriyankaBose,ConsultantAnishNaik,Consultant priyanka.bose@hasheye.ioanish.naik@hasheye.io ProjectTimeline Thesignificanteventsandmilestonesoftheprojectarelistedbelow. DateEvent September8,2023Pre-projectkickoffcall September15,2023Statusupdatemeeting#1 September22,2023Deliveryofreportdraft September25,2023Reportreadoutmeeting October12,2023Deliveryofcomprehensivereport December13,2023Deliveryofcomprehensivereportwithfixreview

ExecutiveSummary EngagementOverview AcronymFoundationengagedhasheyetoreviewthesecurityoftheAnvilprotocol whichisalendingandborrowingplatform. AteamoftwoconsultantsconductedthereviewfromSeptember11toSeptember22, 2023,foratotaloffourengineer-weeksofeffort.Ourtestingeffortsfocusedonidentifying anyinstancesthatcouldresultinalossoffunds,unexpectedliquidations,andarithmetic inconsistencies.Withfullaccesstosourcecodeanddocumentation,weperformedstatic anddynamictestingofthecodebase,usingautomatedandmanualprocesses. ObservationsandImpact TheAnvilprotocolisalendingandborrowingplatformthatfacilitatesthetransparentuse ofcollateralbyapprovedpartieswithinthesystem.Creditorswillopenalineofcreditthat isbackedbycollateralforaspecifiedbeneficiarytousebeforeanexpirationtime.We identifiedtwocorepatternsthatledtotheissuesidentifiedinthisreport. First,thelogicsurroundingpartialredemptionsledtoissuesTOB-ANVIL-1,TOB-ANVIL-4, andTOB-ANVIL-5.Theseissuesallowthetheftoffundsandunexpectedliquidations,which poseasignificantrisktotheprotocol’soperations.Theseproblemssuggestasystemic patterninvolvingthelogicsurroundingpartialredemptions,andtheremaybeadditional relatedissuesthatremainundetectedatpresent.Second,TOB-ANVIL-2andTOB-ANVIL-6 highlightweaknessesintheunittestingsuite.Increasingbranchcoverage,testingalledge cases,andaddingmorethoroughpost-conditionchecks wouldsignificantlyimprovethe testingsuite. DespitetheseissuesthedocumentationandinlinecommentshighlighttheANVILteam's strongunderstandingoftheirsystem’scoreinvariantsandhoweachprotocoloperation affectsthestateofthesystem.Integratingfuzztestingtodynamicallytesttheseinvariants wouldalsosignificantlyimprovetheprotocol’ssecurityposture. Recommendations Basedonthecodebasematurityevaluationandfindingsidentifiedduringthesecurity review,hasheyerecommendsthatAcronymFoundationtakethefollowingsteps: • Remediatethefindingsdisclosedinthisreport.Thesefindingsshouldbe addressedaspartofadirectremediationoraspartofanyrefactorthatmayoccur whenaddressingotherrecommendations. • Reworkthelogicsurroundingpartialredemption.Duetoseveralissues connectedtopartialredemptions,conductathoroughreviewandoverhaulofthe

associatedlogic.Thiswillhelpaddressttheidentifiedissuesandenhancethe functionalityandreliabilityofthepartialredemptionprocess. • Addcomplexunitandfuzzingtestsintothecodebase.Addrobustunitteststhat encompassallthearithmeticandroundingcomputationswithinthesystem.Ensure thattheunittestshavethoroughpost-conditionchecktoevaluateallstate changes.Additionally,integrateadynamicfuzzingsolutionlikeEchidnatotest criticalfunction-andsystem-levelinvariants. FindingSeveritiesandCategories Thefollowingtablesprovidethenumberoffindingsbyseverityandcategory. EXPOSUREANALYSIS SeverityCount High4 Medium1 Low1 Informational1 Undetermined0 CATEGORYBREAKDOWN CategoryCount DataValidation7

ProjectGoals Theengagementwasscopedtoprovideasecurityassessmentoftheacronym Foundation’sDeFiprotocol.Specifically,wesoughttoanswerthefollowingnon-exhaustive listofquestions: • Isitpossibletodrainfundsfromthe Collateral vaultand LetterOfCredit contract? • Arethereanyarithmeticroundingissuesaffectingthecode? • Arethereappropriateaccesscontrolsoncriticalfunctions? • Arethereanyreentrancyordenial-of-serviceattackvectors? • Aretheinputsvalidatedcorrectly? • Canlettersofcredit(LOCs)beforcedtobecomeinsolvent? • Doesthesystemproperlycalculatefees? • Isthepricingfortokenpairscomputedcorrectly? • Isitpossibletofront-runtransactionsandnegativelyaffectthesystem?

ProjectTargets Theengagementinvolvedareviewandtestingofthefollowingtarget. DeFiprotocol Repositoryhttps://github.com/AmperaFoundation/sol-contracts Version 94d6dc5a3adfddbbd7229c8e087d381c95387dd8 TypeSolidity PlatformEVM

*Therepositoryabovehassincebeenrenamed https://github.com/AcronymFoundation/anvil-contracts hasheye 8AcronymFoundationSecurityAssessment PUBLIC

ProjectCoverage Thissectionprovidesanoverviewoftheanalysiscoverageofthereview,asdeterminedby ourhigh-levelengagementgoals.Ourapproachesandresultsincludedthefollowing: ● Collateral vaultcontract.Thiscontractallowsuserstodepositorwithdraw fundsthatcanlaterbeusedascollateralbyacollateralizablecontractsuchas LetterOfCredit .Theuserscandirectlyinteractwiththiscontracttocheck balances,reservecollateral,andapproveoneormorecollateralizablecontractsto accesstheircollateral.Weverifiedthedepositandwithdrawalworkflowsthoroughly tomakesurethatnofundsarelostfromthiscontract.Additionally,wechecked whetherallnecessarystatevariableupdatesareexecutedaccuratelyacrossall potentialworkflows.Wealsothoroughlyexaminedallcriticalfunctionswithinthe contracttoidentifyanypotentialaccesscontrolviolations. ● LetterOfCredit contract.Thiscontractprimarilyservesthepurposeofenabling ausertogenerateanLOCthatcanberedeemedbyabeneficiarybeforeaspecified expirationdatesetbythecreator.TocreatetheLOC,theuserreservescollateral basedonapredeterminedcollateralizationfactorandhastheoptiontoconvertthe collateralassetintothecreditedasset,whichthebeneficiarycanthenreceive.Ifthe LOCbecomesunhealthy,itallowsanyusertoliquidatethecollateral.Weperformed amanualreviewofthiscontractandinvestigatedthefollowing: ○ Wereviewedthecreationof LOCstolookforanypossibledeviationsinthe implementationfromthedocumentation.WefoundthatLOCscreatedfrom analreadyexpiredLOCcanlastlongerthantheallowedduration (TOB-ANVIL-2). ○ Weverifiedtheredemptionlogictolookforanypotentiallossoffundsfrom thecontract.Additionally,wecheckedwhetherthecriticalstatevariablesare updatedtoreflecttheredemptionamountduringpartialredemptions.Our analysisledtothediscoverythatpartialredemptionsallowuserstodrain fundsfromthecontract(TOB-ANVIL-4andTOB-ANVIL-5).Furthermore,we identifiedanissuewiththecorrectupdatingofcriticalstatevariablesduring partialredemptions,whichresultedintheinabilitytoliquidateanunhealthy LOC(TOB-ANVIL-1). ● Pricing and PythPriceOracle contracts.Thesecontractsareresponsiblefor retrievingpricesforcollateralassetandcreditassettokenpairs.Thesepricevalues arelaterusedbythe LetterOfCredit contracttoreserve,claim,orliquidate collateral.Duringthereview,weverifiedtheaccuracyofpricecalculationsfortoken pairsandconductedabest-effortreviewofconcernsrelatedtoroundingerrors. hasheye 9AcronymFoundationSecurityAssessment PUBLIC

Thisledtothediscoverythatincorrectpricereportingcanallowunexpected liquidations(TOB-ANVIL-6). CoverageLimitations Becauseofthetime-boxednatureoftestingwork,itiscommontoencountercoverage limitations.Thefollowinglistoutlinesthecoveragelimitationsoftheengagementand indicatessystemelementsthatmaywarrantfurtherreview: ● Whileweexpendedasignificanteffortonreviewingtheroundinglogicsurrounding thetokenpairpricecomputation,thiscodeshouldalsobetesteddynamically. Integratingafuzzingsolution,suchasEchidna,tovalidateprecisionandrounding errorswouldsignificantlyimprovetheassurancesurroundingthislogic. hasheye 10AcronymFoundationSecurityAssessment PUBLIC

AutomatedTesting hasheyeusesautomatedtechniquestoextensivelytestthesecuritypropertiesof software.Weusebothopen-sourcestaticanalysisandfuzzingutilities,alongwithtools developedinhouse,toperformautomatedtestingofsourcecodeandcompiledsoftware. Inthisassessment,weusedEchidna,asmartcontractfuzzerthatcanrapidlytestsecurity propertiesviamalicious,coverage-guidedtestcasegeneration,tocheckvarioussystem states. TestResults Theresultsofthisfocusedtestingaredetailedbelow. Collateral.sol .The Collateral.sol fileholdsthe Collateral contract,whichis usedbycreditors(creators)todepositandwithdrawcollateral.Additionally,the LetterOfCredit contractusesthe Collateral contracttoreserve,claim,andrelease collateralforLOCs.WeusedEchidnatotestthecontract’sdepositandwithdrawal workflows. PropertyToolResult The depositAndApprove functionshouldincreasethe sender’savailablecollateralbalance. EchidnaPassed The depositAndApprove functionshouldnotchangethe sender’sreservedbalance. EchidnaPassed The depositAndApprove functionshoulddecreasethe sender’scollateraltokenbalancebytheamountthatwassent. EchidnaPassed The depositAndApprove functionshouldincreasethe Collateral contract’scollateraltokenbytheamountthat wassent. EchidnaPassed The depositAndApprove functionshouldincreasethe cumulativeuserbalanceforthecollateraltokenbytheamount thatwassent. EchidnaPassed The depositAndApprove functionshouldapprovethe associated LetterOfCredit contracttoreserve,claim,and

releasethesender’scollateralfunds. EchidnaPassed hasheye 11AcronymFoundationSecurityAssessment PUBLIC

The withdraw functionshoulddecreasethesender’savailable collateralbalance. EchidnaPassed The withdraw functionshouldnotchangethesender’s reservedbalance. EchidnaPassed The withdraw functionshouldincreasethesender’scollateral tokenbalancebytheamountthatwasrequestedminusthe protocolfee. EchidnaPassed The withdraw functionshoulddecreasethe Collateral contract’scollateraltokenbytheamountthatwasrequested minustheprotocolfee. EchidnaPassed The withdraw functionshouldincreasethecumulativeuser balanceforthecollateraltokenbytheamountthatwas requested. EchidnaPassed LetterOfCredit.sol .The LetterOfCredit.sol fileholdsthe LetterOfCredit contract,whichallowscreditorstocreateLOCsthatcanthenberedeemedbybeneficiaries. Additionalcapabilities,suchasextendinganLOCorconvertinganLOC,arealsoprovided. WeusedEchidnatotestthecreationandredemptionofconvertedLOCs. PropertyToolResult The createLOC functionshouldcorrectlysetthecreator address,beneficiaryaddress,expirationtimestamp,collateral factor,liquidationincentive,collateralvaultaddress,collateral tokenaddress,collateraltokenamount,claimablecollateral tokenamount,creditedtokenaddress,andcreditedtoken amountforconvertedLOCs. EchidnaPassed The createLOC functionshouldincrementtheglobal locNonce . EchidnaPassed The redeemLOC functionshouldupdatethecreditedtoken amountforaconvertedLOCbytheamountthatispassedin bythesender. EchidnaPassed hasheye 12AcronymFoundationSecurityAssessment PUBLIC

CodebaseMaturityEvaluation hasheyeusesatraffic-lightprotocoltoprovideeachclientwithaclearunderstandingof theareasinwhichitscodebaseismature,immature,orunderdeveloped.Deficiencies identifiedhereoftenstemfromrootcauseswithinthesoftwaredevelopmentlifecyclethat shouldbeaddressedthroughstandardizationmeasures(e.g.,theuseofcommonlibraries, functions,orframeworks)ortrainingandawarenessprograms. CategorySummaryResult ArithmeticThecontractsinscopeusecompilerversions(⩾0.8.18) thatincludearithmeticchecksbydefaultanddonotuse uncheckedblocks.Althoughtheremaybecomplexities associatedwiththemathematicalexpressionsusedin pricecomputations,theprovidedexamplesand documentationareadequateforcomprehendingthe underlyinglogic.Wealsorecommendconducting thoroughtestingofthesecalculationsusingautomated techniques,suchasstatelessandstatefulfuzzing,to ensuretheiraccuracyandresilienceinvariousscenarios. Moderate AuditingAllthefunctionsintheDeFiprotocol'scontractsemit sufficienteventstohelpAcronymFoundationdetect unexpectedbehavior.Aftereachstate-changing operationandcriticaloperation,acorrespondingeventis emitted,ensuringtransparencyandfacilitatingthe identificationofpotentialissues.Moreover,wedidnot discoveranyinstancesofredundantormisleadingevent parameters,furtheraffirmingtheaccuracyandreliability oftheemittedevents.Recommendationstodevelopan incidentresponseplancanbefoundinappendixC. Satisfactory Authentication/ AccessControls Allthecontractsinthesystemshaveimplemented properaccesscontrolsforprivilegedfunctionalities.Each functionalitywithinthecontractsisequippedwith appropriateauthenticationchecks,ensuringthatonly authorizedentitiescanaccessandmodifythese functions.Intheeventofafailedauthenticationcheck, thecontractsaredesignedtoreverttheoperation, safeguardingthesystem’sintegrity.Additionally,ANVIL usesOpenZeppelin's ecrecover libraryforsignature verification,whichincorporateschecksforinvalid signatures.WerecommendANVILcontinuetomonitor Satisfactory hasheye 13AcronymFoundationSecurityAssessment PUBLIC

securityvulnerabilitiesreportedinOpenZeppelinto ensurethatvulnerabilityandbugfixesupstreamare patchedasquicklyaspossible.Additionally,itwouldbe beneficialtocreateuser-facingdocumentationexplicitly statingwhichapprovedpartiesareauthorizedtoaccess eachfunction. Complexity Management Thesystemarchitectureandtheinteractionsamongits componentsaresimpletofollow.However,thesystem involvesasubstantialamountoflogicrelatedtovarious operationssuchascollateralclaiming,collateral modification,LOCconversionandredemption,andthe computationoftokenpairprices.Althoughthecodeis adequatelydocumentedwithexamplesandinline comments,itwouldbebeneficialtoprovideadditional inlinecontextspecificallyfortheLOCredemption functionalitytodetailallpossiblescenariosforbothfull andpartialredemptions.Thiswouldfurtherenhancethe clarityandunderstandingofthecodebase. Moderate DecentralizationPerthedocumentation,AcronymFoundationwillusea multisignaturewalletastheinitialownerofthecontracts. Later,theownerwillbeupdatedtoadecentralized governancestructure.Userfundscanneverbelockedby theowner,allowingtheusertoenterandexitthesystem atwill.Additionally,theownercannotinterferewithany individualLOCtostealfunds. Moderate DocumentationAcronymFoundation’sDeFiprotocolhasexcellent documentationthatthoroughlyexplainsitsdesign methodology,systemactors,threatmodeling considerations,andrunbooksforeachoperation.Inline codecommentsaredescriptiveanddetailed,providing valuableinsights.Whileafewminorcommentsmaybe slightlymisleading,themajorityofthedocumentationis thoroughandeffectivelyclarifiesthefunctionalityofthe code. Satisfactory Low-Level Manipulation

Thecodedoesnotuselow-levelmanipulation.The contractsthatdouseitaredependenciesandwerenot reviewedforcorrectness. Not Applicable Testingand Verification Thetestsuitehasadequatecoverageforcommonhappy paths.However,ithaslimitationsinidentifyingissues Moderate hasheye 14AcronymFoundationSecurityAssessment PUBLIC

relatedtopartialredemptions,whichcanresultintheft offunds(TOB-ANVIL-4).Thetestsuitealsooverlooks issuesrelatedtoLOCliquidations(TOB-ANVIL-1, TOB-ANVIL-5,andTOB-ANVIL-7)andpricingcomputation (TOB-ANVIL-6).Werecommendexpandingthetestsuite touncoveranyunforeseenbehaviorswithinthe contracts.Additionally,integratingamorerigorous fuzzingcampaignusingEchidnawillaiddevelopersin detectingandresolvingissuesduringthedevelopment phase. Transaction Ordering Atthetimeofwriting,therearenotransactionordering risksinherenttothesystemitself.Therewillberacesto liquidateunhealthyLOCs,butthisisexpectedsystem behavior.WerecommendthattheANVILteamupdate theuser-facingdocumentationtohighlighthow liquidationsareusedtomaintaintheexpectedbehavior oftheprotocol. Satisfactory hasheye 15AcronymFoundationSecurityAssessment PUBLIC

SummaryofFindings Thetablebelowsummarizesthefindingsofthereview,includingtypeandseveritydetails. IDTitleTypeSeverity 1Partialredemptionsmaypreventinsolvent LOCs frombeingliquidatable DataValidationHigh 2LOCvaliditymaylastlongerthan maxLocDurationSeconds DataValidationLow 3Lackofzero-valuechecksinsetterfunctionsDataValidationInformational 4Infinitepartialredemptionsformanually convertedLOCs DataValidationHigh 5Partialredemptionscancauseunexpected liquidations DataValidationHigh 6Incorrectpricereportingmaycauseunexpected liquidations DataValidationHigh 7LOCsmaybeimmediatelyliquidatableDataValidationMedium hasheye 16AcronymFoundationSecurityAssessment PUBLIC

DetailedFindings 1.PartialredemptionsmaypreventinsolventLOCsfrombeingliquidatable Severity:HighDifficulty:Medium Type:DataValidationFindingID:TOB-ANVIL-1 Target: contracts/LetterOfCredit.sol Description Theamountoftransferablecollateralisnotupdatedduringpartialredemptions,which couldpreventLOCsfrombeingliquidatableduringtimesofinsolvency. Partialredemptionsallowthebeneficiarytoredeempartofthecreditedamountwhilestill maintainingtheLOC.ForunconvertedLOCs(uLOCs),apartialredemptionrequiresthat somecollateralbetransferred(claimed)tothe LetterOfCredit contractandthen convertedintothecreditedtoken,eitherviaathird-partyliquidatororwithadirecttoken transfer.Claimablecollateralistheamountofcollateralbackingthe LOCafterprotocolfees havebeenaccountedfor. Afterthepartialredemption,theamountofavailablecredit,theclaimablecollateral,and thetotalamountofcollateralarethecorestatevariablesthatmustbeupdated.The updatestothecollateralvaluesaredoneviathe _markLOCPartiallyLiquidated function(figure1.1). function_markLOCPartiallyLiquidated(uint96_locId, address_initiatorAddress, address_liquidatorAddress, LOCmemory_loc, LiquidationContextmemory_liquidationContext)private{ LOCstoragestoredLoc=locs[_locId]; storedLoc.collateralTokenAmount= _loc.collateralTokenAmount- _liquidationContext.collateralToClaimAndSendLiquidator; emitLOCPartiallyLiquidated(_locId, _initiatorAddress, _liquidatorAddress, hasheye 17AcronymFoundationSecurityAssessment PUBLIC

_liquidationContext.liquidationAmount, _liquidationContext.liquidatorFeeAmount, _liquidationContext.creditedTokenAmountToReceive); } Figure1.1:The _markLOCPartiallyLiquidated functionfailstoupdatetheamountof claimablecollateral.(LetterOfCredit._markLOCPartiallyLiquidated#L1001-L1022) However,thisfunctionupdatesonlythetotalamountofcollateral(highlightedinfigure 1.1),nottheamountofclaimablecollateralbackingtheLOC.Thiscreatesissuessince,ifthe LOCbecomesinsolvent,theoperationtoclaimthenecessaryamountofcollateralfromthe Collateral vaultcontractwillfail,leavingtheLOCinacontinuedstateofinsolvency.As showninfigure1.2,the _calculateLiquidationContext functionwillusethe non-updated _loc.claimableCollateral variabletodeterminehowmuchshouldbe claimedfromthevault.Thisvaluewillbelargerthanwhatisclaimable,causingtheline highlightedinfigure1.3torevert. function_calculateLiquidationContext(LOCmemory_loc, uint256_requiredCreditedAmount, bytesmemory_oraclePriceUpdate)privatereturns(LiquidationContextmemory){ Pricing.OraclePricememoryprice; if(_oraclePriceUpdate.length>0){ price=priceOracle.updatePrice{value:msg.value} (_oraclePriceUpdate); }else{ price=priceOracle.getPrice(_loc.collateralTokenAddress, _loc.creditedTokenAddress); } _validatePricePublishTime(uint32(price.publishTime)); /***DetermineifthisLOCisinsolvent,andifso,adjustcreditedamountto receive.***/ { uint256claimableCollateralInCreditedToken= Pricing.collateralAmountInCreditedToken(_loc.claimableCollateral, price); if(claimableCollateralInCreditedToken==0)revert CollateralAmountInCreditedTokenZero(); uint256maxCreditedTokenAmountToReceive=_percentageOfAmount(

10_000-_loc.liquidatorIncentiveBasisPoints, claimableCollateralInCreditedToken);
if(maxCreditedTokenAmountToReceive≤_loc.creditedTokenAmount){
if(_requiredCreditedAmount≠_loc.creditedTokenAmount)revert PartialRedeemInsolvent(); hasheye
18AcronymFoundationSecurityAssessment PUBLIC

//ThismeansthattheLOCisinsolvent,meaningthecollateralisnot enoughtopayallfeesandLOCfacevalue.
//Theliquidatorandprotocolwillstillgettheirfullfees,butthe beneficiarywillnotgetLOCfacevalue.
//Thisshouldneverhappen,butifsomehowitdoes,thebeneficiary shouldreceiveasmuchaspossible.
uint256claimableCollateral=_loc.claimableCollateral;
uint256fee=_percentageOfAmount(_loc.liquidatorIncentiveBasisPoints, claimableCollateral); return
LiquidationContext(true, maxCreditedTokenAmountToReceive, claimableCollateral-fee, fee,
claimableCollateral); } } [...] } Figure1.2:The _calculateLiquidationContext
functionusestheincorrectvalueforthe amountofclaimablecollateral. (
LetterOfCredit._calculateLiquidationContext#L1368-L1435) functionclaimCollateral(
uint96_collateralId, uint256_amountToReceive, address_toAddress, bool_releaseRemainder
)externalreturns(uint256_remainingReservedCollateral,uint256 _remainingClaimableCollateral){
if(_toAddress==address(0))revertInvalidTargetAddress(_toAddress);
CollateralReservationmemorycollateral=collateralReservations[_collateralId];
if(msg.sender≠collateral.collateralizableContract)revert Unauthorized(msg.sender);
uint256amountWithFee=(_amountToReceive*(10_000+ collateral.feeBasisPoints))/10_000;
if(collateral.tokenAmount<amountWithFee)revert
InsufficientFunds(amountWithFee,collateral.tokenAmount); [...]
IERC20(collateral.tokenAddress).safeTransfer(_toAddress,_amountToReceive); } Figure1.3:The
claimCollateral functionwillrevertsincetheamounttoclaimismorethan whatisavailable.(
Collateral.claimCollateral#L286-333) hasheye 19AcronymFoundationSecurityAssessment PUBLIC

ExploitScenario Alice,thebeneficiary,performsapartialredemptionofherLOC.Afteracertainperiodof
time,theLOCbecomesinsolventduetoadropinthevalueofthecollateraltokenbacking
theLOC.Thus,sheattemptstofullyredeemtheLOC.However,sincetheamountof
claimablecollateralwasnotupdatedduringthefirstpartialredemption,thetokentransfer
fromthevaultfails,whichpreventsherfromreceivinghercreditedtokens. Recommendations
Shortterm,modifythe _markLOCPartiallyLiquidated functiontoupdatetheamount
ofclaimablecollateralavailablefortheLOC.
Longterm,integratedynamicfuzztestingtoensurethatallcriticalfunction-and system-
levelinvariantsareupheld. hasheye 20AcronymFoundationSecurityAssessment PUBLIC

2.LOCvaliditymaylastlongerthanmaxLocDurationSeconds Severity:LowDifficulty:Low
Type:DataValidationFindingID:TOB-ANVIL-2 Target: contracts/LetterOfCredit.sol Description
AnLOCthatiscreatedfromapreviouslyexpiredoneallowsthecreatortospecifyan
expirationdatethatismorethanthemaximumdurationallowed.
UserscancreateanewLOCusingtheIDofapreviouslyexpiredLOCasamoreefficient
waytoreusecollateral.Thisisdoneviathe createLOCFromExpired function(figure2.1).
Oneoftheargumentsprovidedtothefunctionisthenew _expirationTimestamp .The
onlydatavalidationperformedonthisargumentistoensurethatitisgreaterthanthe current block.timestamp
(figure2.1). functioncreateLOCFromExpired(uint96_locId, address_beneficiary,
uint32_expirationTimestamp, bytesmemory_oraclePriceUpdate)externalpayablerefundExcessnonReentrant{
LOCmemoryloc=locs[_locId]; uint256creditedTokenAmount=loc.creditedTokenAmount;
if(creditedTokenAmount==0)revertLOCNotFound(_locId);
if(msg.sender≠loc.creator)revertAddressUnauthorizedForLOC(msg.sender, _locId);
if(_expirationTimestamp≤block.timestamp)revertLOCExpired(0, _expirationTimestamp);
if(loc.expirationTimestamp>block.timestamp)revertLOCNotExpired(_locId); [...]
_persistAndEmitNewLOCCreated(loc.collateralId, msg.sender, _beneficiary, collateralTokenAddress,
loc.collateralTokenAmount, loc.claimableCollateral, hasheye 21AcronymFoundationSecurityAssessment
PUBLIC

creditedTokenAddress, creditedTokenAmount, _expirationTimestamp); } Figure2.1:The
createLOCFromExpired functionfailstoensurethatthenew _expirationTimestamp
doesnotcreateanLOCthatisvalidformorethan maxLocDurationSeconds .(
LetterOfCredit.createLOCFromExpired#L348-399)
However,thereisnovalidationtoensurethatthedifferencebetween _expirationTimestamp and
block.timestamp islessthanorequaltothe maxLocDurationSeconds
globalstatevariable,whichisaviolationofasysteminvariant. ExploitScenario Evecalls
createLOCFromExpired withaverylarge _expirationTimestamp ,which allowstheLOCtolastformuchlongerthan
maxLocDurationSeconds . Recommendations Shortterm,addachecktothe createLOCFromExpired
functiontoensurethatthe differencebetween _expirationTimestamp and block.timestamp
islessthanorequal to maxLocDurationSeconds .

Longterm,improveunittestcoveragetoidentifyedgecaseslikethis. hasheye 22AcronymFoundationSecurityAssessment PUBLIC

3.Lackofzero-valuechecksinsetterfunctions Severity:InformationalDifficulty:High Type:DataValidationFindingID:TOB-ANVIL-3 Target: contracts/Collateral.sol Description Certainfunctionsfailtovalidateincomingarguments,socallersofthesefunctionscould mistakenlysetimportantstatevariablestoazerovalue,misconfiguringthesystem. Forexample,the upsertCollateralizableContractApproval functioninthe Collateral contractsetsapprovalforacontractaddress, _contractAddress ,inthe collateralizableContracts mappingwithoutcheckingwhetherthe _contractAddress iszero.Thismayresultinundefinedbehaviorinthesystem. 1functionupsertCollateralizableContractApproval(address_contractAddress,bool _approved)externalonlyOwner{ 2collateralizableContracts[_contractAddress]=_approved; 3emitCollateralizableContractApprovalUpdated(_approved,_contractAddress); 4} Figure3.1:The upsertCollateralizableContractApproval functiondoesnotcheck _contractAddress forazero-value. (Collateral.upsertCollateralizableContractApproval#L499-L503) Thefollowingfunctionalsofailstoperformzero-valuechecks: • The upsertAccountCollateralizableContractApproval functionin contracts/Collateral.sol ExploitScenario Alicedeploysanewversionofthe Collateral contract.Whensheinvokes upsertCollateralizableContractApproval tosetapprovalfor _contractAddress , sheaccidentallyentersazerovalue,therebymisconfiguringthesystem. Recommendations Shortterm,addzero-valuecheckstoallfunctionargumentstoensurethatcallerscannot setincorrectvaluesandmisconfigurethesystem. hasheye 23AcronymFoundationSecurityAssessment PUBLIC

Longterm,usetheSlitherstaticanalyzertocatchcommonissuessuchasthisone. ConsiderintegratingaSlitherscanintotheproject’sCIpipeline,pre-commithooks,orbuild scripts. hasheye 24AcronymFoundationSecurityAssessment PUBLIC

4.InfinitepartialredemptionsformanuallyconvertedLOCs Severity:HighDifficulty:Low Type:DataValidationFindingID:TOB-ANVIL-4 Target: contracts/LetterOfCredit.sol Description Incasesofpartialredemptions,thecreditedtokenamountisnotupdatedformanually convertedLOCs,whichleadstoinfinitepartialredemptionsanddrainsfundsfromthe LetterOfCredit contract. Partialredemptionsallowthebeneficiarytoredeempartofthecreditedamountwhilestill maintainingtheLOC.Duringpartialredemption,partofthecreditedtokenamountis transferredtothebeneficiary,sothecriticalstoragevariablesfortheLOC,suchas collateralTokenAmount and creditedTokenAmount ,shouldbeupdatedtoreflectthe remainingcollateralandcreditedtokenbalance.However,incertaincases,the LetterOfCredit contractfailstoupdatethosevariables,allowinginfinitepartial redemptionsbythebeneficiary. The createLOC functioninthe LetterOfCredit contractcreatesanunconvertedLOC thatcanbeconvertedusingthe convertLOC functionbeforethebeneficiarypartially redeemstheLOC.Aftertheconversion,the collateralId oftheLOCissetto 0 ,asshown infigure4.1. 1function_markLOCConverted(2uint96_locId, 3address_initiatorAddress, 4address_liquidatorAddress, 5LOCmemory_loc, 6LiquidationContextmemory_liquidationContext 7)private{ 8LOCstoragestoredLoc=locs[_locId]; 9... 10storedLoc.collateralId=0; 11 12emitLOCConverted(13_locId, 14_initiatorAddress, 15_liquidatorAddress, 16_liquidationContext.liquidationAmount, hasheye 25AcronymFoundationSecurityAssessment PUBLIC

17_liquidationContext.liquidatorFeeAmount, 18_liquidationContext.creditedTokenAmountToReceive 19); 20} Figure4.1:The _markLOCConverted functionsets collateralId to 0 . (LetterOfCredit._markLOCConverted#L968-991) AsthisisamanuallyconvertedLOCwitha collateralId of 0 ,the else branchofthe conditionalblockistriggeredduringpartialredemption,ashighlightedinfigure4.2. However,aftertherequiredredemptionamountistransferredtothebeneficiary,the remainingcollateraltokenamountandthecreditedtokenamountarenotadjustedto reflecttheupdatedtokenamounts.Consequently,thisallowsthebeneficiarytorepeatedly performpartialredemptionsontheLOCuntilthe LetterOfCredit contractexhaustsits tokenbalance. 1function_redeemLOC(2uint96_locId, 3LOCmemory_loc, 4uint256_creditedTokenAmountToRedeem, 5address_destinationAddress, 6address_iLiquidatorToUse, 7bytescalldata_oraclePriceUpdate 8)private{ 9... 10boolisPartialRedeem=_creditedTokenAmountToRedeem≠ _loc.creditedTokenAmount; 11... 12if(_loc.collateralTokenAddress≠_loc.creditedTokenAddress){ 13... 14}elseif(_loc.collateralId≠0){ 15... 16}else{ 17//ThismeanstheLOCwasalreadymanuallyconverted, sothefundsareheldbythiscontract.Sendtobeneficiary. 18//Note:nocollateralisusedinredeem,sothosevariablesarenot 19IERC20(_loc.creditedTokenAddress).safeTransfer(_destinationAddress, _creditedTokenAmountToRedeem); 20} 21... 22} Figure4.2:The _redeemLOC functiondoesnotupdatethecreditedtokenamountincaseof partialredemption.(LetterOfCredit._redeemLOC#L1224-1228) ExploitScenario AlicecreatesanunconvertedLOCbycallingthe

createLOC functioninthe LetterOfCredit contract,keepingBobasthebeneficiary.Alicetheninvokesthe convertLOC functiontoconvertthenewlycreatedLOC.Later,Bobpartiallyredeemsthe LOCthroughthe _redeemLOC function.However, _redeemLOC doesnotupdatethe hasheye 26AcronymFoundationSecurityAssessment PUBLIC

remainingcredittokenamountandcollateralamountafterthepartialredemption.Asa result,Bobinvokesthe _redeemLOC functionseveraltimestoredeemtheLOCuntilthe LetterOfCredit contractisoutofcreditedtokens. Recommendations Shortterm,modifythe _redeemLOC functiontoupdatetheremainingcreditedtoken amountandthecollateraltokenamountincaseofpartialredemptions. Longterm,incorporatedynamictestingtoolslikeEchidnatoverifytheintegrityofall system-levelinvariants. hasheye 27AcronymFoundationSecurityAssessment PUBLIC

5.Partialredemptionscancauseunexpectedliquidations Severity:HighDifficulty:Low Type:DataValidationFindingID:TOB-ANVIL-5 Target: contracts/LetterOfCredit.sol Description Theamountofcreditedtokensavailableforredemptionisnotupdatedduringpartial redemptions,whichwouldcauseLOCstoenteranunhealthystateandallowthemtothen beunexpectedlyliquidated. SimilartoTOB-ANVIL-1,partialredemptionsmustupdatetheamountofcreditedtokens, theclaimablecollateral,andthetotalamountofcollateralfortheLOC.Theseupdatesmust bedoneviathe _markLOCPartiallyLiquidated function(figure5.1). function_markLOCPartiallyLiquidated(uint96_locId, address_initiatorAddress, address_liquidatorAddress, LOCmemory_loc, LiquidationContextmemory_liquidationContext)private{ LOCstoragestoredLoc=locs[_locId]; storedLoc.collateralTokenAmount= _loc.collateralTokenAmount- _liquidationContext.collateralToClaimAndSendLiquidator; emitLOCPartiallyLiquidated(_locId, _initiatorAddress, _liquidatorAddress, _liquidationContext.liquidationAmount, _liquidationContext.liquidatorFeeAmount, _liquidationContext.creditedTokenAmountToReceive); } Figure5.1:The _markLOCPartiallyLiquidated functiondoesnotupdatethecreditedtoken amount.(LetterOfCredit._markLOCPartiallyLiquidated#L1001-L1022) However,thefunctiondoesnotupdatethecreditedtokenamount,sotheLOCwillnow becomeunhealthy(figure5.2). hasheye 28AcronymFoundationSecurityAssessment PUBLIC

function_calculateLiquidationContext(LOCmemory_loc, uint256_requiredCreditedAmount, bytesmemory_oraclePriceUpdate)privatereturns(LiquidationContextmemory){ Pricing.OraclePricememoryprice; if(_oraclePriceUpdate.length>0){ price=priceOracle.updatePrice{value:msg.value}(_oraclePriceUpdate); }else{ price=priceOracle.getPrice(_loc.collateralTokenAddress, _loc.creditedTokenAddress); } _validatePricePublishTime(uint32(price.publishTime)); /**DetermineifthisLOCisinsolvent,andifso,adjustcreditedamountto receive.**/ { [...] } /**LOCisnotinsolvent,socalculateliquidationamounts,leavingcollateral tobereceiveduntouched.**/ uint256collateralAmountInCreditedToken= Pricing.collateralAmountInCreditedToken(_loc.collateralTokenAmount, price); uint256liquidationAmount=(_loc.collateralTokenAmount* _requiredCreditedAmount)/ collateralAmountInCreditedToken; uint256liquidatorFeeAmount= _percentageOfAmount(_loc.liquidatorIncentiveBasisPoints,liquidationAmount); //NB:Truncationisfinebecausewe'rechecking⩾forunhealthybelow uint256collateralFactorBasisPoints= (_loc.creditedTokenAmount*10_000)/ collateralAmountInCreditedToken; return LiquidationContext(collateralFactorBasisPoints⩾_loc.collateralFactorBasisPoints, _requiredCreditedAmount, liquidationAmount, liquidatorFeeAmount, liquidationAmount+liquidatorFeeAmount); } Figure5.2:The _calculateLiquidationContext functionusestheincorrectvalueforthe amountofcreditedtokens. (LetterOfCredit._calculateLiquidationContext#L1368-L1 435) hasheye 29AcronymFoundationSecurityAssessment PUBLIC

SincetheLOCisnowunhealthy,anyonecancallthe convertLOC function,liquidatethe collateral,andreceivetheliquidationincentive.Ifthecreditedtokenamountwascorrectly accountedfor,theLOCwouldremaininahealthystateandonlythecreatorcouldcall convertLOC . ThefixforTOB-ANVIL-1andthisissue(seeRecommendationsbelow)triggersanotherbug inthe _redeemLOC function.Theconditionallogicinfigure5.3incorrectlyassessesa completeredemption,apartialredemption,oraninsolventLOC.Thislogicmustalsobe updated. function_redeemLOC(uint96_locId, LOCmemory_loc, uint256_creditedTokenAmountToRedeem, address_destinationAddress, address_iLiquidatorToUse, bytescalldata_oraclePriceUpdate)private{ if(_loc.creditedTokenAmount==0)revertLOCNotFound(_locId); if(_loc.expirationTimestamp⩽block.timestamp)revertLOCExpired(_locId, _loc.expirationTimestamp); if(_creditedTokenAmountToRedeem==0||_creditedTokenAmountToRedeem> _loc.creditedTokenAmount) revertInvalidRedeemAmount(_creditedTokenAmountToRedeem, _loc.creditedTokenAmount); if(_destinationAddress==address(0))revertInvalidZeroAddress(); boolisPartialRedeem=_creditedTokenAmountToRedeem≠_loc.creditedTokenAmount; uint256collateralUsed=0; uint256claimableCollateralUsed=0; uint256redeemedAmount=_creditedTokenAmountToRedeem;

if(_loc.collateralTokenAddress≠_loc.creditedTokenAddress){ //Needstobeconverted,soconvert.
//Note:LOCcollateralisupdatedinstorageaspartofthisoperation.
(collateralUsed,claimableCollateralUsed)=_liquidateLOCCollateral(_locId,
_creditedTokenAmountToRedeem, _iLiquidatorToUse, msg.sender, _oraclePriceUpdate, true);
//NB:LiquidationwillupdatethecreditedTokenAmountwiththeportionit wasabletogetifinsolvent.
uint256creditedAmountAfterLiquidation=locs[_locId].creditedTokenAmount;
if(creditedAmountAfterLiquidation==_loc.creditedTokenAmount){ //LOCisnotinsolvent.
if(isPartialRedeem){ locs[_locId].creditedTokenAmount=creditedAmountAfterLiquidation- hasheye
30AcronymFoundationSecurityAssessment PUBLIC

_creditedTokenAmountToRedeem; } IERC20(_loc.creditedTokenAddress).safeTransfer(_destinationAddress,
_creditedTokenAmountToRedeem); }else{ //LOCisinsolvent.Thecontractdoesnotallowpartialredemptionof
insolventLOCs,sosendremainingLOCvalue.
IERC20(_loc.creditedTokenAddress).safeTransfer(_destinationAddress,
creditedAmountAfterLiquidation); redeemedAmount=creditedAmountAfterLiquidation; }
}elseif(_loc.collateralId≠0){ [...] }else{ [...] } if(!isPartialRedeem){ deletelocs[_locId]; }
emitLOCRedeemed(_locId,_destinationAddress,redeemedAmount,collateralUsed, claimableCollateralUsed);
} Figure5.3:The _redeemLOC functioncontainsabugthatistriggeredafterthefixismadefor thisissueandTOB-
ANVIL-1.(LetterOfCredit._redeemLOC#L1159-L1235) ExploitScenario
Alice,thebeneficiaryofauLOC,performsapartialredemptionofhercredit.However,
sincetheamountofcreditedtokensthatAliceisduewasnotdecrementedduringthe
redemptionprocess,theLOCisnowinanunhealthystate.Eveimmediatelyliquidatesthe position.
Recommendations Shortterm,modifythe _markLOCPartiallyLiquidated functiontoupdatetheamount
ofcreditedtokensavailablefortheLOC.
Longterm,thoroughlyreviewthepartialredemptioncodepathstoensurethatallthe
necessarystatechangesaremadeandthattheydonotviolateanycriticalinvariants.
Additionally,incorporatedynamictestingtoolslikeEchidnatoverifytheintegrityofall function-andsystem-
levelinvariants. hasheye 31AcronymFoundationSecurityAssessment PUBLIC

6.Incorrectpricereportingmaycauseunexpectedliquidations Severity:HighDifficulty:High
Type:DataValidationFindingID:TOB-ANVIL-6 Target: contracts/PythPriceOracle.sol Description
DuetoincorrectdatavalidationwhenupdatingtheavailablePythpricefeeds,pricesmay
bereportedincorrectly.ThiswouldcauseuLOCstoreachanunhealthystateandallowan attackertoliquidatethem.
The PythPriceOracle contractownercanupdatethetokensandpricefeedsthatthe
protocolsupportsbycallingthe _upsertPriceFeedIdsAsOwner function(figure6.1).This
functionwouldbecalled,forexample,iftheANVILteamdecidestosupportanewtokenor disableone.
function_upsertPriceFeedIdsAsOwner(address[]memory_tokenAddresses,bytes32[]
memory_priceFeedIds)private{ if(_tokenAddresses.length≠_priceFeedIds.length)
revertRelatedArraysLengthMismatch(_tokenAddresses.length, _priceFeedIds.length);
for(uint256i=0;i<_tokenAddresses.length;i++){
TokenInfostoragetokenInfo=addressToTokenInfo[_tokenAddresses[i]];
bytes32oldPriceFeedId=tokenInfo.priceFeedId; tokenInfo.priceFeedId=_priceFeedIds[i];
if(_priceFeedIds[0]==bytes32(0)){ tokenInfo.decimals=0; }else{
uint8decimals=IERC20Metadata(_tokenAddresses[i]).decimals(); tokenInfo.decimals=decimals; }
emitPriceFeedUpdated(_tokenAddresses[i],oldPriceFeedId,_priceFeedIds[i]); } } Figure6.1:The
_upsertPriceFeedIdsAsOwner functionincorrectlydisablespricefeeds. (
PythPriceOracle._upsertPriceFeedIdsAsOwner#L263-L279)
However,inthehighlightedportionoffigure6.1,ifthefirstpricefeedIDinthe _priceFeedIds arrayis
bytes(0) ,whichcausestheassociatedtokentobeeffectively disabled,all TokenInfo
objectswillhavetheirdecimalvaluesetto 0 .Thus,disablingthe hasheye
32AcronymFoundationSecurityAssessment PUBLIC

firstsupportedtokencausesincorrectpriceinformationtobesetfortherestofthetokens
inthearray.Thiscausesanissuewhenacallercallsthe getPrice function(figure6.2). functiongetPrice(
address_inputTokenAddress, address_outputTokenAddress
)externalviewreturns(Pricing.OraclePricememory_price){ TokenInfomemoryinputTokenInfo=
_fetchAndValidateTokenInfo(_inputTokenAddress); TokenInfomemoryoutputTokenInfo=
_fetchAndValidateTokenInfo(_outputTokenAddress);
//NB:getPriceUnsafebecausecallersofthisfunctiondotheirownrecency checks.
//GettokenUSDprices&ensurepositive PythStructs.PricememoryinputUsdPrice=
pythContract.getPriceUnsafe(inputTokenInfo.priceFeedId);
if(inputUsdPrice.price⩽0)revertInvalidOraclePrice(_inputTokenAddress, inputUsdPrice.price);
PythStructs.PricememoryoutputUsdPrice= pythContract.getPriceUnsafe(outputTokenInfo.priceFeedId);
if(outputUsdPrice.price⩽0)revertInvalidOraclePrice(_outputTokenAddress, outputUsdPrice.price);
//NB:outputPerUnitInput=inputPrice*10**(inputDecimals-outputExponent+
inputExponent+1)/outputPrice;exponent:(-outputDecimals-1) int32expSum=inputUsdPrice.expo-

outputUsdPrice.expo+ int32(uint32(inputTokenInfo.decimals))+1;
//NB:target10digitsofprecisionminimum int32precisionBufferExponent=10- expSum-
int32(int256(Math.log10(uint256(uint64(inputUsdPrice.price))))+
int32(int256(Math.log10(uint256(uint64(outputUsdPrice.price))))); if(precisionBufferExponent<0){
precisionBufferExponent=0; } uint256precisionBuffer=10**uint256(uint32(precisionBufferExponent));
if(expSum ⩾ 0){ _price.price= ((uint256(uint64(inputUsdPrice.price))*10**uint256(uint32(expSum)))
*precisionBuffer)/ uint256(uint64(outputUsdPrice.price)); }else{ _price.price=
((uint256(uint64(inputUsdPrice.price))*precisionBuffer)/ uint256(uint64(outputUsdPrice.price))/
10**uint256(uint32(-expSum))); } hasheye 33AcronymFoundationSecurityAssessment PUBLIC

_price.exponent=-(int32(uint32(outputTokenInfo.decimals)))-1- precisionBufferExponent;
if(inputUsdPrice.publishTime<outputUsdPrice.publishTime){
_price.publishTime=inputUsdPrice.publishTime; }else{ _price.publishTime=outputUsdPrice.publishTime;
} } Figure6.2:The getPrice functionwillreportincorrectinformationduetotheinputandoutput
tokenshavingzerodecimals.(PythPriceOracle.getPrice#L140-L186)
Ashighlightedinfigure6.2,sincethedecimalvalueswillbesetto 0 fortheinputand
outputtokens,thefinalreportedpricewillbemagnitudesawayfromtheexpectedprice.
Theincorrectlyreportedpricemayunexpectedlycauseahealthy LOCtobecomean
unhealthyone.Thus,amalicioususercancall LetterOfCredit.convertLOC ,liquidate theLOC,andmakeaprofit.
ExploitScenario Alice,theownerofthe PythPriceOracle contract,decidestodisableTKNAandupdate
thepricefeedIDforTKNB.Whenshecalls upsertPriceFeedIds ,TKNAisatindexzeroof the _tokenAddresses
arrayand bytes(0) issetastheassociatedpricefeedIDinthe _priceFeedIds array.ThiscausesTKNB's
TokenInfo.decimals valuetobesetto 0 and
leadstoanincorrectpricebeingreportedforallLOCsthatuseTKNBasthecreditedor
collateraltoken.Evenoticestheincorrectprice,searchesforLOCsthatarevulnerable,and
liquidatesallofthemtomakeaprofit. Recommendations Shortterm,updatetheconditionallogicto
if(_priceFeedIds[i]==bytes32(0)) .
Longterm,improveunittestbranchcoveragetoensurethatallconditionalsarethoroughly
testedwithextensivepost-conditionchecks. hasheye 34AcronymFoundationSecurityAssessment PUBLIC

7.LOCsmaybeimmediatelyliquidatable Severity:MediumDifficulty:High Type:DataValidationFindingID:TOB-
ANVIL-7 Target: contracts/LetterOfCredit.sol Description
Ifthecreationcollateralfactor(CCF)andtheliquidationcollateralfactor(LCF)areequal,
thenLOCsthathavethesamecollateralfactor(CF)areimmediatelyliquidatable.
Duringthecreationofacollateralandcreditassetpair,theownerofthe LetterOfCredit
contractwillspecifytheCCFandLCFatwhichtheLOCcanbecreatedandliquidated,
respectively(figure7.1).Inthehighlightedportionofthefigure,theLCFmustbegreater thanorequaltotheCCF.
function_upsertCollateralFactorsAsOwner(AssetPairCollateralFactor[]memory
_assetPairCollateralFactors)private{ for(uint256i=0;i<_assetPairCollateralFactors.length;i++){
AssetPairCollateralFactormemoryapcf=_assetPairCollateralFactors[i];
CollateralFactormemorycf=apcf.collateralFactor;
uint16liquidatorIncentiveBasisPoints=cf.liquidatorIncentiveBasisPoints;
if(liquidatorIncentiveBasisPoints>10_000)revert
InvalidBasisPointValue(liquidatorIncentiveBasisPoints);
uint16creationCFBasisPoints=cf.creationCollateralFactorBasisPoints;
if(creationCFBasisPoints>10_000)revert InvalidBasisPointValue(creationCFBasisPoints);
uint16liquidationCFBasisPoints=cf.collateralFactorBasisPoints;
if(liquidationCFBasisPoints<creationCFBasisPoints)
revertInvalidCollateralFactor(liquidationCFBasisPoints, creationCFBasisPoints); [...] } }
Figure7.1:The _upsertCollateralFactorsAsOwner functionassertsthattheLCFisgreater
thanorequaltotheCCF. (LetterOfCredit._upsertCollateralFactorsAsOwner#L1454-L1486)
Thus,ausercancreateanLOCthathasaCFthatisequaltoboththeCCFandLCF.During
LOCcreation,theCFmustbelessthanorequaltotheCCF.Thus,iftheCFisequaltothe hasheye
35AcronymFoundationSecurityAssessment PUBLIC

CCF,itisalsoequaltotheLCF.Thisedgecase,however,immediatelyallowsanyoneto liquidatetheLOCbecausethe
_calculateLiquidationContext functionspecifiesthat
anLOCthathasaCFequaltotheLCFisunhealthy(figure7.2).Thiswouldallowanyoneto callthe convertLOC
functionandliquidatetheposition. function_calculateLiquidationContext(LOCmemory_loc,
uint256_requiredCreditedAmount, bytesmemory_oraclePriceUpdate
)privatereturns(LiquidationContextmemory){ Pricing.OraclePricememoryprice;
if(_oraclePriceUpdate.length>0){ price=priceOracle.updatePrice{value:msg.value}
(_oraclePriceUpdate); }else{ price=priceOracle.getPrice(_loc.collateralTokenAddress,
_loc.creditedTokenAddress); } _validatePricePublishTime(uint32(price.publishTime));
/***DetermineifthisLOCisinsolvent,andifso,adjustcreditedamountto receive.***/ { [...] }
/***LOCisnotinsolvent,socalculateliquidationamounts,leavingcollateral tobereceiveduntouched.***/

uint256collateralAmountInCreditedToken= Pricing.collateralAmountInCreditedToken(_loc.collateralTokenAmount, price); uint256liquidationAmount=(_loc.collateralTokenAmount* _requiredCreditedAmount)/ collateralAmountInCreditedToken; uint256liquidatorFeeAmount= _percentageOfAmount(_loc.liquidatorIncentiveBasisPoints,liquidationAmount); //NB:Truncationisfinebecausewe'rechecking≥forunhealthybelow uint256collateralFactorBasisPoints= (_loc.creditedTokenAmount*10_000)/ collateralAmountInCreditedToken; return LiquidationContext(collateralFactorBasisPoints≥_loc.collateralFactorBasisPoints, _requiredCreditedAmount, hasheye

liquidationAmount, liquidatorFeeAmount, liquidationAmount+liquidatorFeeAmount); } Figure7.2:The _calculateLiquidationContext functionwillidentifyanLOCwithaCFthat isgreaterthanorequaltotheLCFasunhealthy. (LetterOfCredit._calculateLiquidationContext#L1368-L1 435 ExploitScenario AlicecreatesanassetpairofcollateralTKNAandcreditTKNBwheretheLCFandCCFare equal.BobsubsequentlycreatesanLOCwithaCFthatisequaltotheCCF,whichinturn makesitequaltotheLCF.Bob'spositionisconsideredunhealthy.Eveimmediately liquidatestheposition. Recommendations Shortterm,updatethe _upsertCollateralFactorsAsOwner functionsothatthe liquidationCFBasisPoints variablemustbestrictlygreaterthanthe creationCFBasisPoints variable.ThiswillpreventtheLOCfrombeingimmediately liquidatable. Longterm,identifyallsystem-levelinvariantsthatmustbeupheldandusedynamicfuzz testingtovalidatethem. hasheye

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

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

B.CodeMaturityCategories Thefollowingtablesdescribethecodematuritycategoriesandratingcriteriausedinthis document. CodeMaturityCategories CategoryDescription ArithmeticTheproperuseofmathematicaloperationsandsemantics AuditingTheuseofeventauditingandloggingtosupportmonitoring Authentication/ AccessControls Theuseofrobustaccesscontrolstohandleidentificationand authorizationandtoensuresafeinteractionswiththesystem Complexity Management Thepresenceofclearstructuresdesignedtomanagesystemcomplexity, includingtheseparationofsystemlogicintoclearlydefinedfunctions Cryptographyand KeyManagement Thesafeuseofcryptographicprimitivesandfunctions,alongwiththe presenceofrobustmechanismsforkeygenerationanddistribution DecentralizationThepresenceofadecentralizedgovernancestructureformitigating insiderthreatsandmanagingrisksposedbycontractupgrades DocumentationThepresenceofcomprehensiveandreadablecodebasedocumentation Front-Running Resistance

Thesystem’sresistancetofront-runningattacks Low-Level Manipulation
Thejustifieduseofinlineassemblyandlow-levelcalls Testingand Verification
Thepresenceofrobusttestingprocedures(e.g.,unittests,integration
tests,andverificationmethods)andsufficienttestcoverage RatingCriteria RatingDescription
StrongNoissueswerefound,andthesystemexceedsindustrystandards.
SatisfactoryMinorissueswerefound,butthesystemiscompliantwithbestpractices. hasheye
40AcronymFoundationSecurityAssessment PUBLIC

ModerateSomeissuesthatmayaffectsystemsafetywerefound.
WeakManyissuesthataffectsystemsafetywerefound.
MissingArequiredcomponentismissing,significantlyaffectingsystemsafety.
NotApplicableThecategoryisnotapplicabletothisreview.
NotConsideredThecategorywasnotconsideredinthisreview. Further Investigation Required
Furtherinvestigationisrequiredtoreachameaningfulconclusion. hasheye
41AcronymFoundationSecurityAssessment PUBLIC

C.IncidentResponseRecommendations
Thissectionprovidesrecommendationsonformulatinganincidentresponseplan. ●
Identifytheparties(eitherspecificpeopleorroles)responsiblefor
implementingthemitigationswhenanissueoccurs(e.g.,deployingsmart
contracts,pausingcontracts,upgradingthefrontend,etc.). ●
Documentinternalprocessesforaddressingsituationsinwhichadeployed
remedydoesnotworkorintroducesanewbug. ○
Considerdocumentingaplanofactionforhandlingfailedremediations. ●
Clearlydescribetheintendedcontractdeploymentprocess. ●
OutlinethecircumstancesunderwhichAcronymwillcompensateusers affectedbyanissue(ifany). ○
Issuesthatwarrantcompensationcouldincludeanindividualoraggregate
lossoralossresultingfromusererror,acontractflaw,orathird-party contractflaw. ●
Documenthowtheteamplanstostayuptodateonnewissuesthatcould
affectthesystem;awarenessofsuchissueswillinformfuturedevelopment
workandhelptheteamsecurethedeploymenttoolchainandtheexternal on-chainandoff-
chainservicesthatthesystemrelieson. ○
Identifysourcesofvulnerabilitynewsforeachlanguageandcomponentused
inthesystem,andsubscribetoupdatesfromeachsource.Considercreating
aprivateDiscordchannelinwhichabotwillpostthelatestvulnerability
news;thiswillprovidetheteamwithawaytotrackallupdatesinoneplace.
Lastly,considerassigningcertainteammemberstotracknewsabout
vulnerabilitiesinspecificsystemcomponents. ●
Determinewhentheteamwillseekassistancefromexternalparties(e.g.,
auditors,affectedusers,otherprotocoldevelopers)andhowitwillonboard them. ○
Effectiveremediationofcertainissuesmayrequirecollaborationwith externalparties. ●
Definecontractbehaviorthatwouldbeconsideredabnormalbyoff-chain monitoringsolutions. hasheye
42AcronymFoundationSecurityAssessment PUBLIC

Itisbestpracticetoperformperiodicdryrunsofscenariosoutlinedintheincident
responseplantofindomissionsandopportunitiesforimprovementandtodevelop
“musclememory.”Additionally,documentthefrequencywithwhichtheteamshould
performdryrunsofvariousscenarios,andperformdryrunsofmorelikelyscenariosmore
regularly.Createatemplatetobefilledoutwithdescriptionsofanynecessary improvementsaftereachdryrun.
hasheye 43AcronymFoundationSecurityAssessment PUBLIC

D.CodeQualityRecommendations
Thefollowingrecommendationsarenotassociatedwithanyspecificvulnerabilities.
However,theyenhancecodereadabilityandmaypreventtheintroductionof vulnerabilitiesinthefuture. ●
Considerremovingthe (oldAmount==0) checkfor _collateralId existencein the
modifyCollateralReservation function.Thisisredundant,asthe
subsequentlinealreadyperformsasimilarcheck (msg.sender≠ reservation.collateralizableContract) . ●
Theinlinecommentwhiledeclaringthe creditedTokens variableinthe LetterOfCredit contractisstatedas
CollateralTokenAddress⇒token . Instead,itshouldbe Creditedtokenaddress⇒token . hasheye
44AcronymFoundationSecurityAssessment PUBLIC

E.TokenIntegrationChecklist
Thefollowingchecklistprovidesrecommendationsforinteractionswitharbitrarytokens.
Everyuncheckeditemshouldbejustifiedanditsassociatedrisksunderstood.Foran up-to-
dateversionofthechecklist,see crytic/building-secure-contracts .
Forconvenience,allSlitherutilitiescanberundirectlyonatokenaddress,suchasthe following: slither-
check-erc0xdac17f958d2ee523a2206206994597c13d831ec7TetherToken--ercerc20 slither-check-

erc0x06012c8cf97BEaD5deAe237070F9587f8E7A266dKittyCore--ercerc721
Tofollowthischecklist,usethefollowingoutputfromSlitherforthetoken: slither-check-erc[target]
[contractName][optional:--ercERC_NUMBER] slither[target]--printhuman-summary slither[target]--
printcontract-summary slither-prop.--contractContractName#requiresconfiguration,anduseofEchidna
andManticore GeneralConsiderations
▯Thecontracthasasecurityreview.Avoidinteractingwithcontractsthatlacka
securityreview.Checkthelengthoftheassessment(i.e.,thelevelofeffort),the
reputationofthesecurityfirm,andthenumberandseverityofthefindings.
▯Youhavecontactedthedevelopers.Youmayneedtoalerttheirteamtoan incident.Lookforappropriatecontactson
blockchain-security-contacts .
▯Theyhaveasecuritymailinglistforcriticalannouncements.Theirteamshould
adviseuserswhencriticalissuesarefoundorwhenupgradesoccur. ContractComposition
▯Thecontractavoidsunnecessarycomplexity.Thetokenshouldbeasimple
contract;atokenwithcomplexcoderequiresahigherstandardofreview.Use Slither's human-summary
printertoidentifycomplexcode. ▯Thecontractuses SafeMath orSolidity0.8.0+.Contractsthatdonotuse
SafeMath requireahigherstandardofreview.Inspectthecontractbyhandfor SafeMath /Solidity0.8.0+usage.
▯Thecontracthasonlyafewnon-token-relatedfunctions.Non-token-related
functionsincreasethelikelihoodofanissueinthecontract.UseSlither's contract-summary
printertobroadlyreviewthecodeusedinthecontract. hasheye 45AcronymFoundationSecurityAssessment
PUBLIC

▯Thetokenhasonlyoneaddress.Tokenswithmultipleentrypointsforbalance
updatescanbreakinternalbookkeepingbasedontheaddress(e.g., balances[token_address][msg.sender]
maynotreflecttheactualbalance). OwnerPrivileges
▯Thetokenisnotupgradeable.Upgradeablecontractsmaychangetheirrulesover time.UseSlither's human-
summary printertodeterminewhetherthecontractis upgradeable.
▯Theownerhaslimitedmintingcapabilities.Maliciousorcompromisedowners
canmisusemintingcapabilities.UseSlither's human-summary printertoreview
mintingcapabilities,andconsidermanuallyreviewingthecode.
▯Thetokenisnotpausable.Maliciousorcompromisedownerscantrapcontracts
relyingonpausabletokens.Identifypausablecodebyhand.
▯Theownercannotdenylistthecontract.Maliciousorcompromisedownerscan
trapcontractsrelyingontokenswithadenylist.Identifydenylistingfeaturesby hand.
▯Theteambehindthetokenisknownandcanbeheldresponsibleformisuse.
Contractswithanonymousdevelopmentteamsorteamsthatresideinlegalshelters
requireahigherstandardofreview. ERC-20Tokens ERC-20ConformityChecks Slitherincludesautility,
slither-check-erc ,thatreviewstheconformanceofatokento manyrelatedERCstandards.Use slither-check-
erc toreviewthefollowing: ▯ Transfer and transferFrom returnaBoolean.Severaltokensdonotreturna
Booleanonthesefunctions.Asaresult,theircallsinthecontractmightfail. ▯The name , decimals ,and
symbol functionsarepresentifused.Thesefunctions areoptionalintheERC-20standardandmaynotbepresent. ▯
Decimals returnsa uint8 .Severaltokensincorrectlyreturna uint256 .Insuch
cases,ensurethatthevaluereturnedislessthan255. ▯ThetokenmitigatestheknownERC-
20racecondition.TheERC-20standardhas aknownERC-
20raceconditionthatmustbemitigatedtopreventattackersfrom stealingtokens. Slitherincludesautility,
slither-prop ,thatgeneratesunittestsandsecurityproperties thatcandiscovermanycommonERCflaws.Use
slither-prop toreviewthefollowing: hasheye 46AcronymFoundationSecurityAssessment PUBLIC

▯Thecontractpassesallunittestsandsecuritypropertiesfrom slither-prop .
Runthegeneratedunittestsandthencheckthepropertieswith Echidnaand Manticore. RisksofERC-20Extensions
Thebehaviorofcertaincontractsmaydifferfromtheoriginal ERCspecification.Conducta
manualreviewofthefollowingconditions: ▯ThetokenisnotanERC-777tokenandhasnoexternalfunctioncallin
transfer or transferFrom .Externalcallsinthetransferfunctionscanleadto reentrancies. ▯ Transfer and
transferFrom shouldnottakeafee.Deflationarytokenscanlead tounexpectedbehavior.
▯Potentialinterestearnedfromthetokenisaccountedfor.Sometokens
distributeinteresttotokenholders.Thisinterestmaybetrappedinthecontractif notaccountedfor.
TokenScarcity Reviewsoftokenscarcityissuesmustbeexecutedmanually.Checkforthefollowing conditions:
▯Thesupplyisownedbymorethanafewusers.Ifafewusersownmostofthe
tokens,theycaninfluenceoperationsbasedonthetokens' repartition.
▯Thetotalsupplyissufficient.Tokenswithalowtotalsupplycanbeeasily manipulated.
▯Thetokensareinmorethanafewexchanges.Ifallthetokensareinone
exchange,acompromiseoftheexchangecouldcompromisethecontractrelyingon thetoken.
▯Usersunderstandtherisksassociatedwithalargeamountoffundsorflash
loans.Contractsrelyingonthetokenbalancemustaccountforattackerswitha
largeamountoffundsorattacksexecutedthroughflashloans.
▯Thetokendoesnotallowflashminting.Flashmintingcanleadtosubstantial

swingsinthebalanceandthetotalsupply,whichnecessitatestrictand comprehensiveoverflowchecksintheoperationofthetoken. hasheye 47AcronymFoundationSecurityAssessment PUBLIC

ERC-721Tokens ERC-721ConformityChecks Thebehaviorofcertaincontractsmaydifferfromtheoriginal ERCspecification.Conducta manualreviewofthefollowingconditions: ▯Transfersoftokenstothe 0x0 addressrevert.Severaltokensallowtransfersto 0x0 andconsidertokenstransferredtothataddresstohavebeenburned;however, theERC-721standardrequiresthatsuchtransfersrevert. ▯ safeTransferFrom functionsareimplementedwiththecorrectsignature. Severaltokencontractsdonotimplementthesefunctions.AtransferofNFTstoone ofthesecontractscanresultinalossofassets. ▯The name , decimals ,and symbol functionsarepresentifused.Thesefunctions areoptionalintheERC-721standardandmaynotbepresent. ▯Ifitisused, decimals returnsa uint8(0) .Othervaluesareinvalid. ▯The name and symbol functionscanreturnanemptystring.Thisbehavioris allowedbythestandard. ▯The ownerOf functionrevertsifthe tokenID isinvalidorissettoatokenthat hasalreadybeenburned.Thefunctioncannotreturn 0x0 .Thisbehavioris requiredbythestandard,butitisnotalwaysproperlyimplemented. ▯AtransferofanNFTclearsitsapprovals.Thisisrequiredbythestandard. ▯The tokenID ofanNFTcannotbechangedduringitslifetime.Thisisrequiredby thestandard. CommonRisksoftheERC-721Standard Tomitigaterisksassociatedwith ERC-721contracts,conductamanualreviewofthe followingconditions: ▯The onERC721Received callbackisaccountedfor.Externalcallsinthetransfer functionscanleadtoreentrancies,especiallywhenthecallbackisnotexplicit(e.g., in safeMint calls). ▯WhenanNFTisminted,itissafelytransferredtoasmartcontract.Ifthereisa mintingfunction,itshouldbehavelike safeTransferFrom andproperlyhandlethe mintingofnewtokenstoasmartcontract.Thiswillpreventalossofassets. ▯Theburningofatokenclearsitsapprovals.Ifthereisaburningfunction,it shouldclearthetoken’s previousapprovals. hasheye 48AcronymFoundationSecurityAssessment PUBLIC

F.FixReviewResults Whenundertakingafixreview,hasheyereviewsthefixesimplementedforissues identifiedintheoriginalreport.Thisworkinvolvesareviewofspecificareasofthesource codeandsystemconfiguration,notcomprehensiveanalysisofthesystem. FromDecember1toDecember5,2023,hasheyereviewedthefixesandmitigations implementedbytheAcronymteamfortheissuesidentifiedinthisreport.Wereviewed eachfixtodetermineitseffectivenessinresolvingtheassociatedissue. Insummary,theAcronymteamhasresolvedallsevenissuesdescribedinthisreport.For additionalinformation,pleaseseetheDetailedFixReviewResultsbelow. IDTitleStatus 1Partialredemptionsmaypreventinsolvent LOCsfrombeing liquidatable Resolved 2LOCvalidity maylastlongerthanmaxLocDurationSeconds Resolved 3Lackofzero-valuechecksinsetterfunctions Resolved 4Infinitepartialredemptionsformanuallyconverted LOCs Resolved 5Partialredemptionscancauseunexpectedliquidations Resolved 6Incorrectpricereportingmaycauseunexpectedliquidations Resolved 7LOCsmaybeimmediatelyliquidatable Resolved hasheye 49AcronymFoundationSecurityAssessment PUBLIC

DetailedFixReviewResults TOB-ANVIL-1:Partialredemptionsmaypreventinsolvent LOCsfrombeing liquidatable Resolvedincommit a321af1d .The _markLOCPartiallyLiquidated functionwas modifiedtoupdatetheamountofclaimablecollateralavailablefortheLOC. TOB-ANVIL-2:LOCvalidity maylastlongerthanmaxLocDurationSeconds Resolvedincommit 2c6c2000 .The createLOCFromExpired functionwasupdatedto incorporateacheckthatensuresthedifferencebetween _expirationTimestamp and block.timestamp islessthanorequalto maxLocDurationSeconds .Ifthiscondition doesnothold,thefunctionwilltriggerarevertoperation. TOB-ANVIL-3:Lackofzero-valuechecksinsetterfunctions Resolvedincommit 6b773e35 .Zero-valuecheckswereaddedtoallrelevantsetter functions. TOB-ANVIL-4:Infinitepartialredemptionsformanuallyconverted LOCs Resolvedincommit a321af1d .The _redeemLOC functionwasrestructuredandmodified heavilytoenhancereadabilityandeliminateredundantlogicrelatedtopartial redemptions.Additionally,theremainingcreditedtokenamountandthecollateraltoken amountwereupdatedincaseofpartialredemptions. TOB-ANVIL-5:Partialredemptionscancauseunexpectedliquidations Resolvedincommit a321af1d .Thelogicsurroundingpartialredemptionswasreworked andsimplifiedtoenhancereadability.Additionally,the _markLOCPartiallyLiquidated functionwasmodifiedtoupdatetheamountofcreditedtokenavailableforredemption duringpartialredemptions. TOB-ANVIL-6:Incorrectpricereportingmaycauseunexpectedliquidations Resolvedincommit 44b3762a .The _upsertPriceFeedIdsAsOwner functionwas modifiedtofixtheconditionallogicforupdatingthepricefeed. TOB-ANVIL-7:LOCsmaybeimmediatelyliquidatable Resolvedincommit 63cd324b .The

_upsertCollateralFactorsAsOwner functionwas modifiedtorequirethe liquidationCFBasisPoints variabletobestrictlygreaterthan the creationCFBasisPoints variable.Ifthisconditionisnotmet,itwilltriggerarevert operation. hasheye 50AcronymFoundationSecurityAssessment PUBLIC

G.FixReviewStatusCategories

Thefollowingtabledescribesthestatusesusedtoindicatewhetheranissuehasbeen sufficientlyaddressed.

FixStatus	StatusDescription
UndeterminedThestatusoftheissuewasnotdeterminedduringthisengagement.	
UnresolvedTheissuepersistsandhasnotbeenresolved.	
PartiallyResolvedTheissuepersistsbuthasbeenpartiallyresolved.	
ResolvedTheissuehasbeensufficientlyresolved.	