HashEye

// SMART CONTRACT SECURITY AUDIT

Primitive

Security assessment by HashEye · prepared for Primitive

HASHEYE AUDITED

PROJECT	Primitive
CLIENT	Primitive
CATEGORY	Blockchain
PUBLISHED	January 1, 2022
REPORT ID	research-primitive-2022-01-01-nquv70

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-primitive-2022-01-01-nquv70.

PrimitiveHyper SecurityAssessment March31,2023 Preparedfor: AlexAngel Primitive Preparedby:NatChin,RobertSchneider,TarunBansal,andKurtWillis

Abouthasheye Foundedin2012andheadquarteredinNewYork,hasheyeprovidestechnicalsecurity assessmentandadvisoryservicestosomeoftheworld’smosttargetedorganizations.We combinehigh-endsecurityresearchwithareal -worldattackermentalitytoreduceriskand fortifycode.With100+employeesaroundtheglobe,we’vehelpedsecurecriticalsoftware elementsthatsupportbillionsofendusers,includingKubernetesandtheLinuxkernel. Wemaintainanexhaustivelistofpublicationsathttps://github.com/hasheye-io/publications, withlinkstopapers,presentations,publicauditreports,andpodcastappearances. Inrecentyears,hasheyeconsultantshaveshowcasedcutting-edgeresearchthrough presentationsatCanSecWest,HCSS,Devcon,EmpireHacking,GrrCon,LangSec,NorthSec, theO’ReillySecurityConference,PyCon,REcon,SecurityBSides,andSummerCon. Wespecializeinsoftwaretestingandcodereviewprojects,supportingclientorganizations inthetechnology,defense,andfinanceindustries,aswellasgovernmententities.Notable clientsincludeHashiCorp,Google,Microsoft,WesternDigital,andZoom. hasheyealsooperatesacenterofexcellencewithregardtoblockchainsecurity.Notable projectsincludeauditsofAlgorand,BitcoinSV,Chainlink,Compound,Ethereum2.0, MakerDAO,Matic,Uniswap,Web3,andZcash. Tokeepuptodatewithourlatestnewsandannouncements,pleasefollowhasheyeon Twitterandexploreourpublicrepositoriesathttps://github.com/hasheye-io.Toengageus directly,visitour“Contact”pageathttps://www.hasheye.io/contact,oremailusat info@hasheye.io. hasheye,Inc. 228ParkAveS#80688 NewYork,NY10003 https://www.hasheye.io info@hasheye.io hasheye 1PrimitiveSecurityAssessment PUBLIC

NoticesandRemarks CopyrightandDistribution Thisreportisconsideredbyhasheyetobepublicinformation;itislicensedtoPrimitive underthetermsoftheprojectstatementofworkandhasbeenmadepublicatPrimitive’s request.Materialwithinthisreportmaynotbereproducedordistributedinpartorin wholewithouttheexpresswrittenpermissionofhasheye. 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 2PrimitiveSecurityAssessment PUBLIC

ExecutiveSummary EngagementOverview
PrimitiveengagedhasheyetoreviewthesecurityofitsHypersmartcontracts.From January3toJanuary31,2023,ateamoffourconsultantsconductedasecurityreviewofthe client-providedsourcecode,witheightperson-weeksofeffort.Detailsoftheproject’s timeline,testtargets,andcoverageareprovidedinsubsequentsectionsofthisreport. ProjectScope
Ourtestingeffortswerefocusedontheidentificationofflawsthatcouldresultina compromiseofconfidentiality,integrity,oravailabilityofthetargetsystem.Weconducted thisauditwithfullknowledgeofthesystem,includingaccesstothesourcecodeand documentation.Weperformedstaticanddynamictestingofthetargetsystemandits codebase,usingbothautomatedandmanualprocesses. SummaryofFindings
Theaudituncoveredsignificantflawsthatcouldimpactsystemconfidentiality,integrity,or availability.Asummaryofthefindingsanddetailsonnotablefindingsareprovidedbelow. EXPOSUREANALYSIS
SeverityCount High9 Medium2 Low4 Informational6 Undetermined2 CATEGORYBREAKDOWN CategoryCount Configuration1 DataValidation11 UndefinedBehavior11 hasheye 5PrimitiveSecurityAssessment PUBLIC

NotableFindings
Significantflawsthatimpactsystemconfidentiality,integrity,oravailabilityarelistedbelow. • Missingdatavalidationthroughoutthecodebase
Throughoutthecodebase,manyfunctionsfailtocheckthatincominguser argumentsareboundbetweentwovalues,resultingintheambiguityofexpected inputparameters.Forexample,missingchecksontypeconversionscouldallow attackerstostealfunds(TOB-HYPR-4),andpoolscouldbedefinedwithazerostrike price(TOB-HYPR-13). • Incorrecthandlingofarithmetic
Thesystemisalsoaffectedbythemishandlingofarithmeticoperations,whichcould allowattackerstostealfunds.Theseissuesstemfromunderlyingassumptionsthat causeintegeroverflowissues(TOB-HYPR-3)androundingissues(TOB-HYPR-13, TOB-HYPR-14,TOB-HYPR-16,TOB-HYPR-17,TOB-HYPR-20,TOB-HYPR-22,and TOB-HYPR-23).WerecommendthatthePrimitiveteamcontinuetoextendthe existingEchidnasuitetotestforthesecornercasesinunexpectedbehavior. hasheye 6PrimitiveSecurityAssessment PUBLIC

ProjectSummary ContactInformation Thefollowingmanagerswereassociatedwiththisproject:
DanGuido,AccountManagerMaryO’Brien,ProjectManager dan@hasheye.iomary.obrien@hasheye.io
Thefollowingengineerswereassociatedwiththisproject: NatChin,ConsultantRobertSchneider,Consultant natalie.chin@hasheye.iorobert.schneider@hasheye.io TarunBansal,ConsultantKurtWillis,Consultant tarun.bansal@hasheye.iokurt.willis@hasheye.io ProjectTimeline
Thesignificanteventsandmilestonesoftheprojectarelistedbelow. DateEvent January3,2023Pre-projectkickoffcall January9,2023Statusupdatemeeting#1 January17,2023Statusupdatemeeting#2 January24,2023Statusupdatemeeting#3 February2,2023Deliveryofreportdraft February2,2023Reportreadoutmeeting March31,2023Deliveryoffinalreport hasheye 7PrimitiveSecurityAssessment PUBLIC

SummaryofRecommendations
hasheyerecommendsthatPrimitiveaddressthefindingsdetailedinthisreportandtake thefollowingadditionalstepspriortodeployment: • PerformadditionaleconomicanalysisontheHypercurvetoensurethatinputand outputboundsarealwaysknownandchecked.AddtheseboundsintotheHyper smartcontractstoensurethatallfunctionsbehaveasexpected. • IdentifyadditionalglobalsysteminvariantsandcontinuetoextendtheEchidna end-to-endtestsuitetoensurethatitcapturesallfunctions.Runandmaintainthis extendedfuzzingcampaignonaserverwithacorpus. • Documentallimplicitandexplicitusesofroundinginthesystem.Integratethis documentationintoaflowcharttocreateavisualrepresentationoutliningallofthe formulas’roundingdirectionstoensurethattheyalwaysfavorpoolsoverusers. • Followalloftherecommendationsrelatedtoroundingoutlinedinappendix C. •

Performanothersecurityreviewonareleasecandidatepriortoproduction deploymentafterallofthe issuesinthisreporthavebeenaddressedandfixed. hasheye 8PrimitiveSecurityAssessment PUBLIC

ProjectGoals Theengagementwasscopedtoprovideasecurityassessmentof Primitive’sHypersmart contracts.Specifically,wesoughttoanswerthefollowingnon-exhaustivelistofquestions: • Cananattackerstealfunds? • Arethereappropriateaccesscontrolmeasuresinplaceforusersandadmins? • Doesthesystem’sbehaviormatchthespecification? • Cananattackerfreezefundsordenyservicetothesystem? • Isitpossibletoperformsystemoperationswithoutpayingtherequiredfees? • Arethearithmeticlibrariesusedcorrectlyanddotheycorrectlyapplyrounding directions? hasheye 9PrimitiveSecurityAssessment PUBLIC

ProjectTargets Theengagementinvolvedareviewandtestingofthetargetslistedbelow. hyper Repositoryhttps://github.com/primitivefinance/hyper Versions 0bcadb708272276dd77a99f58e57f9f8dfed3c79 (Findings1-17) 577b5c861a9541a91ec39b3618278f93869d0d38 (Findings19-22) 5a75fc5aad144a1066d9ed19320869c03b305c26 TypeSolidity PlatformEthereum solstat Repositoryhttps://github.com/primitivefinance/solstat Version 6e9654163765aac867af1a56fc84462ffdad7d56 TypeSolidity PlatformEthereum hasheye 10PrimitiveSecurityAssessment PUBLIC

ProjectCoverage Thissectionprovidesanoverviewoftheanalysiscoverageofthereview,asdeterminedby ourhigh-levelengagementgoals.Ourapproachesforcoveringallofthetargetcomponents involvedacombinationofstaticanalysis,manualreview,andfuzztestingthroughEchidna: The Hyper contract:The Hyper contractisthemainentrypointforcreatingpoolsand usingthemtodepositandswaptokens.Thecontracthasmanyfeatures,includingthe following: • Tokenandpaircreation:Userscandefinereusable“tokenpairs”structsanduse themtocreatepools;thestructsstorethenumberoftokendecimalsandthe addressofbothtokens.Wecheckedtoensurethatpairscanbecreatedandused whenpoolsarecreated. • Poolcreation:Poolsarecreatedwithacurvetorepresentapoolwithspecific parameters.Thesepoolscanbesetupasmutableorimmutable,whichdetermines whetheraspecified“poolcontroller”canlaterchangethepoolparameters.We reviewedthepoolcreationcodeandimplementedaseriesofglobalsystem invariantstocheckthesystem’sassumptionsoncreatedpools. • Allocatingandunallocating:Thisfunctionalityallowsuserstoaddandremove fundsfromapool.Wereviewedthearithmeticandroundingdirectionsusedin theseoperations. • Swapping:Thisfunctionalityallowsuserstousepoolstoswaptokensagainsttheir pairs.Wereviewedthearithmeticinthecontractstoensurethatthecalculations roundcorrectlyandusethecorrectscalefactors.Wereviewedtwodifferent versionsofthe swap function;theversionsofHyperthatweauditedarelistedinthe “ProjectTargets”section. The HyperLib contract:Thiscontractmanagesstructsthatareusedbythe Hyper contract,includingthecurve,pair,pools,andordersstructs.Theyprovidehelperfunctions tocomputeamounts,validateparameters,andstorestateinformation.Wefocusedonthe arithmeticlogicinthesefunctions,ensuringthatalloperationsroundinthecorrect direction. CoverageLimitations Becauseofthetime-boxednatureoftestingwork,itiscommontoencountercoverage limitations.Thefollowinglistoutlinesthecoveragelimitationsoftheengagementand indicatessystemelementsthatmaywarrantfurtherreview: hasheye 11PrimitiveSecurityAssessment PUBLIC

• Swapping:Whilewespentasignificantamountofeffortreviewingswaps,thiscode isincrediblycomplex.Therefore,werecommendcontinuingtodevelopadditional fuzztestsagainsttheswappingcodetoallowfuzzerstoexploreadditionalstates. Ourreviewoftheswappingfunctionalityisinconclusive,andmoreissuesmaystill bepresent. • Settlement:TheHypercontractsimplementasettlementfunctionalitythatiterates overacachedlistoftokenstocarryoutappropriatedebitsandcreditsofreserves. Thislogicismeanttointegratewiththejumpprocessingfunctionalitytoprovide userswithamoregas-efficientwaytohandletokens.Duetotimelimitations,wedid notreviewthisfunctionality. • The Enigma contractandjumpprocessing:Duringouraudit,webrieflyreviewed thefunctionalityofthe Enigma contract.Inthelongterm,thiscontractrequires morein-depthanalysistoensurethatuniqueidentifiersareencodedanddecoded correctly,anditrequiresthoroughfuzztestingtocatchunexpectedbehavior. • Stakingandunstaking:ThePrimitivecodebasecontainsfunctionalitytoallow userstostakeandunstakeassets,providingdifferentinterestratesandrewardsfor puttingfundsintothesystem.Thisfeaturewasdeprioritizedandremovedfromthe scopeofthisreview.ThePrimitiveteamlaterpushedacommittoremovethis functionalityfromtheHypercontracts. • Customfunctiondispatching:TheHypercontractsimplementfunctioncustom dispatching,whichallowsfunctionstobeinvokedthroughthefallbackfunction.Due

totimelimitations,thisfunctionalitywasdeprioritizedandrequiresadditional analysis. hasheye 12PrimitiveSecurityAssessment PUBLIC

AutomatedTesting hasheyehasdevelopeduniquetoolsfortestingsmartcontracts.Inthisassessment,we usedEchidna,asmartcontractfuzzerthatcanrapidlytestsecuritypropertiesviamalicious, coverage-guidedtestcasegeneration,tocheckvarioussystemstates.
Automatedtestingtechniquesaugmentourmanualsecurityreviewbutdonotreplaceit.
Eachtechniquehaslimitations;forexample,Echidnamaynotrandomlygenerateanedge casethatviolatesaproperty.Wefollowaconsistentprocesstomaximizetheefficacyof testingsecurityproperties.WhenusingEchidna,wegenerate100,000,000testcasesper property.
TestHarnessConfiguration Weusedthefollowingtoolsintheautomatedtestingphaseofthisproject:
ToolDescription SlitherAstaticanalysisframeworkthatcanstaticallyverifyalgebraicrelationships betweenSolidityvariables
EchidnaAsmartcontractfuzzerthatcanrapidlytestsecuritypropertiesviamalicious, coverage-guidedtestcasegeneration TestResults
Ourautomatedtestingandverificationfocusedonthefollowingsystemproperties:
Globalsysteminvariants:UsingEchidna,wetestedthefollowingexpectedsystem properties.
IDPropertyToolResult 1The locked variableisalwayssetto 1 outsideofexecution.EchidnaPassed 2TheHyperaccount’s settled variableissetto true immediatelyafterdeployment.Thisvariableissavedinthis stateoutsideofexecution. EchidnaPassed hasheye 13PrimitiveSecurityAssessment PUBLIC

3TheHyperaccount’s prepared variableissetto false outsideofexecution. EchidnaPassed 4Thenumberofwarmtokensoutsideofexecutionisalways 0 .EchidnaPassed 5Thepriorityfeeofacontrolledpoolcanneverbesetto 0 .EchidnaPassed 6The Hyper contract’stokenbalanceisalwaysgreaterthanor equaltothereservesofthetokensavedinthecontract. EchidnaPassed 7Amutablepoolcanneverhaveanonzeropriorityfee.EchidnaPassed 8Apool’smaturityisneverlessthanthelasttimestamp.EchidnaPassed 9Apoolwithanonzerolastpriceneverhaszeroliquidity.EchidnaPassed 10Whencalledonapoolwithanonzero deltaLiquidity value,the getLiquidityDeltas functionneverreturns zero deltaAsset or deltaQuote amounts. EchidnaFAILED (TOB-HYP R-17)
11Apool’slastpriceisnevergreaterthanthestrikeprice.EchidnaFAILED (TOB-HYP R-14)
12Thestrikepriceofapoolcanneverbesetto 0 .EchidnaFAILED (TOB-HYP R-13)
Paircreation:UsingEchidna,wetestedthefollowingpropertiesoftheHypercontract’s statewhentokenpairsarecreated. IDPropertyToolResult
13Thecreationofatokenpairwiththecorrectpreconditions alwayssucceeds. EchidnaPassed 14The pairNonce variableofthe Hyper contractincreasesEchidnaPassed hasheye 14PrimitiveSecurityAssessment PUBLIC

whenapairiscreated. 15Thesameassetandquotetokenalwaysyieldthesamepair ID. EchidnaPassed 16Thecreationofatokenpairwithtwoidenticaltokensalways fails. EchidnaPassed 17Thecreationofatokenpairwithatokenthathaslessthan sixdecimalsalwaysfails. EchidnaPassed 18Thecreationofatokenpairwithatokenthathasmorethan 18decimalsalwaysfails. EchidnaPassed
Curvecreation:UsingEchidna,wetestedthefollowingpropertiesoftheHypercontract’s statewhencurvesarecreated. IDPropertyToolResult 19Acurve’sfeecanneverbesetto 0 .EchidnaPassed 20Acurve’spriorityfeecanneverexceedthefee.EchidnaPassed 21Acurve’sdurationcanneverbesetto 0 .EchidnaPassed 22Acurvecanneverhaveavolatilitygreaterthanthevalueof MIN_VOLATILITY . EchidnaPassed 23Acurve’s createdAt timestampcanneverbesetto 0 .EchidnaPassed
Poolcreation:UsingEchidna,wetestedthefollowingpropertiesoftheHypercontract’s statewhenpoolsarecreated. IDPropertyToolResult 24Thecreationofanon-controlledpoolwiththecorrectEchidnaFAILED hasheye 15PrimitiveSecurityAssessment PUBLIC

preconditionsdoesnotrevert.(TOB-HYP R-13) 25Thecreationofanon-controlledpoolproperlysetsthepool statetoimmutable. EchidnaPassed 26Thecreationofanon-controlledpoolsetsthe jit valueto thedefaultvalueof JUST_IN_TIME_LIQUIDITY_POLICY . EchidnaPassed
27Thecreationofacontrolledpoolwiththecorrect preconditionsneverreverts. EchidnaPassed
28Thecreationofacontrolledpoolsetsthepoolstateto mutable. EchidnaPassed
29Thecreationofapoolsetsthelasttimestampandcurve createdtothecurrenttimestamp. EchidnaPassed 30The getVirtualReserves methodalwaysreturnsvalues lessthanthevaluesreturnedbyHyper’srespective getReserve functionforeachtokenofthepool’spair. EchidnaPassed 31The getAmountsWad methodalwaysreturnsan amountAssetWad valuethatislessthan1e18. EchidnaPassed 32The getAmountsWad methodalwaysreturnsan amountQuoteWad valuethatislessthanthereturnvalueof pool.params.strike() . EchidnaPassed 33Thecreationofacontrolledpoolwithazero-valuepriority feealwaysfails. EchidnaPassed
Updatingpoolparameters:UsingEchidna,wetestedthefollowingpropertiesofthe Hyper contract’sstatewhenpoolparametersareupdated. IDPropertyToolResult
34Updatingpoolparametersdoesnotupdatethelatest timestamp. EchidnaPassed hasheye 16PrimitiveSecurityAssessment PUBLIC

35Updatingpoolparametersmaintainsthesamecontroller address. EchidnaPassed
36Updatingpoolparametersmaintainsthesamecreationtime.EchidnaPassed 37The priorityFee , fee ,
volatility , duration , jit ,and maxTick valuesareupdatedaslongastheyarewithinthe correctbounds.
EchidnaPassed Depositing:UsingEchidna,wetestedthefollowingpropertiesoftheHypercontract’sstate
whenusersdeposittokens. IDPropertyToolResult
38Depositswiththecorrectpreconditionsalwayssucceed.EchidnaPassed 39Whenthe deposit
functioniscalled,thesender’sETH balancedecreasesbythevalueof msg.value . EchidnaPassed
40Depositsdonotalterthe Hyper contract’sETHbalance.EchidnaPassed 41Depositsincreasethe Hyper
contract’sbalanceofthetoken beingdepositedbytheamountbeingdeposited. EchidnaPassed
42Depositsincreasethegivenpool’sreservevalueforthe tokenbeingdepositedbytheamountbeingdeposited.
EchidnaPassed Fundinganddrawing:UsingEchidna,wetestedthefollowingpropertiesoftheHyper
contract’sstatewhenusersfundanddrawtokensfromapool. IDPropertyToolResult
44Fundingoperationswiththecorrectpreconditionsalways succeed. EchidnaPassed 45Callingthe fund
functionwithinsufficientcallerfundsalways fails. EchidnaPassed hasheye
17PrimitiveSecurityAssessment PUBLIC

46Callingthe fund functionwithinsufficientallowancealways fails. EchidnaPassed
47Aftertokensarefunded,thesender’stokenbalance decreasesbythefundedamount. EchidnaPassed
48Aftertokensarefunded,theHyper-registeredbalanceofthe user’stokenincreasesbythefundedamount.
EchidnaFAILED (TOB-HYP R-12) 49Aftertokensarefunded,thereservebalanceofthetoken
increasesbythefundedamount. EchidnaFAILED (TOB-HYP R-12)
50Aftertokensarefunded,thetokenbalanceofthe Hyper contractincreasesbythefundedamount. EchidnaPassed
51Callingthe fund functionwithzerofundsalwayssucceeds.EchidnaPassed 52Callingthe draw
functionwithsufficientbalancealways succeeds. EchidnaPassed 53Callingthe draw
functionwithinsufficientbalancealways fails. EchidnaPassed 54Aftertokensaredrawn,theHyper-
registeredbalanceofthe user’stokendecreasesbythedrawnamount. EchidnaPassed
55Aftertokensaredrawn,thesavedreservebalancedecreases bythedrawnamount. EchidnaPassed
56Aftertokensaredrawn,therecipient’stokenbalancealways increasesbythedrawnamount. EchidnaPassed
56Aftertokensaredrawn,the Hyper contract’stokenbalance decreasesbythedrawnamount. EchidnaPassed
57Callingthe draw functionwiththezeroaddressasthe recipientalwaysfails. EchidnaPassed hasheye
18PrimitiveSecurityAssessment PUBLIC

58Fundinganddrawingoperationsalwaysresultinthesame pre-andpost-state. EchidnaPassed
Allocatingandremovingfunds:UsingEchidna,wetestedthefollowingpropertiesofthe Hyper
contract’sstatewhenfundsareallocatedandremoved. IDPropertyToolResult 59Callingthe allocate
functionwiththecorrectpreconditions alwayssucceeds. EchidnaPassed
60Afterfundsareallocated,theliquidityforthespecified poolId increasesbythevalueof deltaLiquidity .
EchidnaPassed 61Afterfundsareallocated,theliquidityforthespecified poolId neverdecreases.
EchidnaPassed 62Afterfundsareallocated,the Hyper contract’sreservetoken
balanceforthetokensincreasesbythevaluesof deltaAsset and deltaQuote ifthecallerdoesnothave
enoughtokensintheirbalance. EchidnaPassed 63Afterfundsareallocated,the Hyper contract’stokenbalance
increasesbythevaluesof deltaAsset and deltaQuote . EchidnaPassed
64Afterfundsareallocated,thecaller’sfreeliquidityposition alwaysincreasesbythevalueof
deltaLiquidity . EchidnaPassed 65Afterfundsareallocated,thephysicalbalancefortheasset
tokenincreases. EchidnaPassed 66Afterfundsareallocated,thephysicalbalanceforthequote
tokenincreases. EchidnaPassed 67Theallocationoffundstoanonexistentpoolalwaysreverts.EchidnaPassed
68Theallocationofzerodeltaliquidityalwaysreverts.EchidnaPassed hasheye
19PrimitiveSecurityAssessment PUBLIC

69Afterfundsareallocated,ifthefeegrowthassetpoolhas changed,thefeegrowthpositionisnonzero.
EchidnaPassed 70Withthecorrectpreconditions,callstothe unallocate functionneverrevert.
EchidnaPassed 71Afterfundsareunallocated,thepoolliquiditydecreasesby theunallocatedamount.
EchidnaPassed 72Afterfundsareunallocated,thecaller’spositionliquidity
decreasesbytheunallocatedamount. EchidnaPassed 73Thetrackedassetreservebalancedoesnotchangeafter
fundsareunallocated. EchidnaPassed 74Thetrackedquotereservebalancedoesnotchangeafter
fundsareunallocated. EchidnaPassed 75Thephysicalassetbalanceofthe Hyper contractdoesnot
changeafterfundsareunallocated. EchidnaPassed 76Whencalledbyacallerwithoutaposition,the unallocate
functionreverts. EchidnaWIP 77Theunallocationoffundstoanonexistentpoolreverts.EchidnaPassed hasheye
20PrimitiveSecurityAssessment PUBLIC

Swapping:UsingEchidna,wetestedthefollowingpropertiesofthe Hyper contract’sstate
whenusersswaptokens.AlthoughEchidnapassedonthe100,000fuzzingruns,we
recommendcontinuingtorunthetestsuitewithafocusontargetingspecificsafeinput
rangestoallowEchidnatoeasilyexploreallareas.Thepropertieslistedbelowthatrequire
additionaladjustmentsaregiventhestatus“WIP”. IDPropertyToolResult
78Whencalledwiththecorrectpreconditions,the swap functiondoesnotrevert. EchidnaWIP

79Swapsafterthegivenpoolreachesmaturityalwaysrevert.EchidnaWIP
80Swapsonanonexistentpoolalwaysrevert.EchidnaWIP 81Swapswithazeroamountalwaysrevert.EchidnaWIP
82Swapswithazerolimitamountalwaysrevert.EchidnaWIP
83Swappinganassettokenintoapoolalwaysdecreasesthe priceofthetoken.* EchidnaWIP
84Swappingaquotetokenintoapoolalwaysincreasesthe priceofthetoken.* EchidnaWIP
85Swappinganassettokenintoapoolincreasesthetoken reserves.* EchidnaWIP
86Pricesofthegiventokensalwayschangeduring swap operations.* EchidnaWIP
87Liquiditydoesnotchangeduring swap operations.*EchidnaWIP 88Feegrowthcheckpointsincreaseduring swap operations.*EchidnaWIP 89Aswapinonedirectionfollowedbyaswapintheopposite directionresultsinthesamestatepre-andpost-swap. EchidnaFAILED (TOB-HYP R-23) hasheye
21PrimitiveSecurityAssessment PUBLIC

90AswapofaquotetokenwithmorethanⓃdecimalplaces fortheassettokenreverts. EchidnaPassed
*Invarianttestswerewrittenjointlywith Primitivethroughouttheaudit.
Mathfunctioninvariants:UsingEchidna,wetestedthefollowingpropertiesofthe Hyper contract’sstate.
IDPropertyToolResult 91The expWad function’soutputdomainis [0,∞) .EchidnaPassed 92The expWad functionisstrictlymonotonicallyincreasing.EchidnaPassed 93The expWad function’smaximumerroris 0 when x is approximately 0 . Thisismeasuredas . □□□ := □□□ |□| < ε (1 - □□□□□□(□))*□□□□(□) EchidnaPassed 94The sqrt functionisstrictlymonotonicallyincreasing.EchidnaPassed 95The sqrt functionroundsdown.EchidnaPassed 96The sqrt function’smaximumerroris 0 . Thisismeasuredas. □□□ :=□□ □ 0 < □ |□□□□(□*□) - □| EchidnaPassed 97The pdf function’soutputdomainis [0,1/√(2π)) . Note:Theoutputdomainis [0,1/√(2π)] ;however,the upperboundshouldnotbeincludedwhenrounding correctly.Theresultisinconsistentlyroundedupwhen x is 0 . EchidnaFAILED (TOB-HYP R-22) 98The pdf functionismonotonicallydecreasing. Itsmaximumerroronvalueswithinitsinputdomainis approximately 0.4e18 . Thisismeasuredas. □□□ := □□□ □ < □ □□□(□) - □□□(□) EchidnaFAILED (TOB-HYP R-22) hasheye
22PrimitiveSecurityAssessment PUBLIC

99The pdf function’smaximumerroris 0 when x is approximately 0 . Thisismeasuredas. □□□ := □□□ □ < ε □□□(□) - 1 (2π) EchidnaPassed 100The erfc function’soutputdomainis [0,2] .EchidnaFAILED (TOB-HYP R-22) 101The erfc functionismonotonicallydecreasing. Itsmaximumerroronvalueswithinitsinputdomainis approximately 1.36e57 . Thisismeasuredas. □□□ := □□□ □ < □ □□□□(□) - □□□□(□) EchidnaFAILED (TOB-HYP R-22) 102The erfc function’smaximumerrorisapproximately 3e12 when x isapproximately 0 . Thisismeasuredas . □□□ := □□□ □ < ε (□□□□(□) - 1) * □□□□(□) EchidnaFAILED (TOB-HYP R-22) 103The ierfc functionrevertswhengivenvaluesoutsideofits inputdomain. Note:Itreturnsthehard-codedvalue +/-1e20 onvalues outsideofitsdomain,butitshouldreturn +/-∞ or,better yet,revert. EchidnaFAILED (TOB-HYP R-22) 104The ierfc functionismonotonicallydecreasing. Itsmaximumerrorisapproximately 4.9e12 onvalues withinitsinputdomain. Thisismeasuredas. □□□ := □□□ □ < □ □□□□□(□) - □□□□□(□) EchidnaFAILED (TOB-HYP R-22) 105The ierfc function’smaximumerroris 0 when x is approximately 1 . EchidnaPassed hasheye 23PrimitiveSecurityAssessment PUBLIC

CodebaseMaturityEvaluation hasheyeusesatraffic-lightprotocoltoprovideeachclientwithaclearunderstandingof theareasinwhichitscodebaseismature,immature,orunderdeveloped.Deficiencies identifiedhereoftenstemfromrootcauseswithinthesoftwaredevelopmentlifecyclethat shouldbeaddressedthroughstandardizationmeasures(e.g.,theuseofcommonlibraries, functions,orframeworks)ortrainingandawarenessprograms. CategorySummaryResult ArithmeticTheHypercodebaseusesasignificantamountof complexarithmetic.Wefoundissuesrelatedtoprecision loss,theriskthatfundscouldbetrapped,andincorrect roundingdirections,whichattackerscouldexploitfor profit(TOB-HYPR-3,TOB-HYPR-4,TOB-HYPR-5, TOB-HYPR-6,TOB-HYPR-7,TOB-HYPR-14,TOB-HYPR-16, TOB-HYPR-19,TOB-HYPR-20,TOB-HYPR-22,and TOB-HYPR-23).Theseareasofthecodebaserequire additionaldocumentationspecifyingthearithmetic operations’expectedbehaviorandadditional invariant-basedtestingtoensurethattheybehaveas expected. Weak AuditingThefunctionsintheHypercontractsemitsufficient eventstohelpPrimitivedetectunexpectedbehavior.A respectiveeventisproperlyemittedforeach state-changingfunction.However,therearesomecases inwhichusersreceiveobscureunderflow/overflowerror messagesduetotheuseofassemblyoperationsto checkagainstvalidinput(appendixG).Werecommend writingeacheventsothatitexplainsandshowsthe updatedstate. Moderate Authentication/ AccessControls MostoftheexpectedaccesscontrolsacrossthesystemaredocumentedHowever,documentationonrolesand privilegesneedstobeaddedtothecodebase.Moreover, theaccesscontroltestsneedtobemoreconclusiveto testbothhappyandunhappypaths. Moderate hasheye 24PrimitiveSecurityAssessment PUBLIC

Complexity Management Somefunctionsinthe Hyper contractarewellisolated andeasytotest.However,certainfunctions,suchas allocate and unallocate ,usenestedhelperfunctions toupdatethestateofpools,reserves,andpositions, whichmakethemslightlyhardertotrack.Certainareas inthecodebaseuseassemblyoverhigh-levelSolidity code,whichwouldbenefitfromadditional documentation.Moreover,the Enigma contractand customfunctiondispatchingfeaturerequirefurther

analysis. Moderate DecentralizationPoolcontrollershavesignificantcontroloverthesystem, astheyhavetheabilitytoupdatetheparametersofa curve.Asaresult,usershavetocheckthattheirpools aresetupwiththevaluesthattheyintended;andfor mutablepools,usershavetocheckthattheupdated parametersstillreflecttheoptiontheywanttopurchase. Thecorecontractsofthesystemarenotupgradeable. Moderate DocumentationTheHypercontractscontainasignificantamountof documentation,whichprovideaone-to-one cross-referencebetweenthetermsanddefinitionsinthe whitepaperandthoseinthecode.Thetermsrelatedto thesystemareexplainedinthewhitepaperthrough equationderivations,interactivecurvesthroughDesmos, anddetailedexplanationsofsystemcomponents.In manycases,thedocumentationprovideduswith sufficientcontexttounderstandthecomplexbehaviorof thesystem.However,thecontractslackper-function documentationforcomplicatedlogicsuchasthe swap function.Additionally,wefoundminorissuesrelatedto incorrectand/oroutdatedNatSpeccomments (TOB-HYPR-2,TOB-HYPR-10);thoseareasofthecodebase shouldbecloselyreviewedtoensuretheybehaveas expected. Primitivedevelopedthoroughdocumentationofthe codebase’sglobalsysteminvariantsthatspecifythe expectedstateofpoolsandpairsandper-function preconditionsandpostconditionsthatallowustoeasily reasonaboutwhethertheinvariantshold.Although manyoftheseinvariantsareimplemented,we recommendthatPrimitivecontinuetodevelopadditional Moderate hasheye 25PrimitiveSecurityAssessment PUBLIC

documentationoninvariants,specificallythoserelatedto theadjustmentofpoolparametersandthetrackingof ticks. Transaction OrderingRisks Wefoundoneissuethatwouldallowanattackertoprofit immediatelyafteracontrollerhaschangedparameters ofamutablepool(TOB-HYPR-15).However,additional investigationisrequiredtodeterminewhether transactionorderingposesanotherrisktotheother sectionsoftheHypercodebase,particularlytothe customfunctiondispatchingfeature. Further Investigation Required Low-Level Manipulation Assemblyversionsoffunctionsareusedinmanyareasof thesystem;theseversionsdonotincludechecksthat high-levelSolidityversionswouldhaveprovided (TOB-HYPR-22).Manyoftheseassemblyblocksare missinginlinecommentsdescribingtheiroperationsand aremissinghigh-levelreferenceimplementationsthat couldbeusedindifferentialfuzzing. Weak Testingand Verification Unittestsareusedforthemajorityofthecodebase; however,thetestsaremissingunhappypathtestingand cornercasesininputbounds.Althoughbasicfuzztests arealsopresent,werecommendthatPrimitivecontinue toextendthecurrentend-to-endfuzzteststocatch undesiredbehaviorthatcouldoccurduetopoolsof multipletokendecimalsandduetocomplexoperations. Weak hasheye 26PrimitiveSecurityAssessment PUBLIC

SummaryofFindings Thetablebelowsummarizesthefindingsofthereview,includingtypeandseveritydetails. IDTitleTypeSeverity 1Lackofzero-valuechecksonfunctionsDataValidationInformational 2Documentationdiscrepancyin computePriceWithChangeInTau Undefined Behavior Informational 3Riskoftokentheftduetopossibleinteger underflowinslt DataValidationHigh 4Riskoftokentheftduetouncheckedtype conversion DataValidationHigh 5UserscanswapwithoutpayinganyfeesDataValidationMedium 6Swapfunctionreturnsincorrectlyscaledoutput tokenamount DataValidationHigh 7Liquidityproviderscanwithdrawtotalfeesearned byapool Undefined Behavior High 8Assettokenpricedeviatesfromthepricecurveof thepool Undefined Behavior Undetermined 9Newpaircreationcanoverwriteexistingpairs Undefined Behavior High 10ErrorinInvariant.getX Undefined Behavior Informational 11Poolswithoverflowingmaturitydatescanbe created DataValidationLow hasheye 27PrimitiveSecurityAssessment PUBLIC

12MintingfundstotheHypercontractarbitrarily increasesthenextcaller’sbalance ConfigurationInformational 13Poolstrikepricecouldbezeroduetolackof lowerboundcheckonmaxTick DataValidationHigh 14Roundingerrorallowsliquiditytobeadded withoutdepositingtokens DataValidationHigh 15AttackerscansandwichchangeParameterscalls tostealfunds Undefined Behavior High 16Limitedprecisioninstrikepricesduetofixedtick spacing DataValidationLow 17Functionsthatroundbyadding1resultin unexpectedbehavior Undefined Behavior Informational 18Solidity compileroptimizationscanbe problematic Undefined Behavior Informational 19getAmountOutreturnsincorrectvaluewhen calledbycontroller DataValidationLow 20Mismatchedbaseunitcomparisoncaninflate limittolerance DataValidationMedium 21IncorrectimplementationofedgecasesingetY function Undefined Behavior Low 22Lackofproperboundhandlingforsolstat functions Undefined Behavior Undetermined 23Attackerscanstealfundsbyswappinginboth directions Undefined Behavior High hasheye 28PrimitiveSecurityAssessment PUBLIC

DetailedFindings 1.Lackofzero-valuechecksonfunctions Severity:InformationalDifficulty:High Type:DataValidationFindingID:TOB-HYPR-1 Target: contracts/Hyper.sol Description Certainsetterfunctionsfailtovalidateincomingarguments;therefore,callersofthese functionscanaccidentallysetimportantstatevariablestothezeroaddress. Forexample,the constructor functioninthe Hyper contract,whichsetstheWETH contract,lackszero-valuechecks. constructor(addressweth){ WETH=weth; __account__.settled=true; } Figure1.1:The constructor functionin Hyper.sol IftheWETHaddressissettothezeroaddress,theadminmustredeploytheHyper

contractstoresettheaddress’svalue. ExploitScenario Alicedeploysanewversionofthe Hyper contractbutmistakenlyentersthezeroaddress fortheWETHaddress.Shemustredeploythesystemtoresetthevalueof WETH. Recommendations Shortterm,addzero-valuecheckstoallfunctionargumentstoensurethatuserscannot accidentallysetincorrectvalues,misconfiguringthesystem. Longterm,usetheSlitherstaticanalyzertocatchcommonissuessuchasthisone. ConsiderintegratingaSlitherscanintotheproject’scontinuousintegrationpipeline, pre-commithooks,orbuildscripts. hasheye 29PrimitiveSecurityAssessment PUBLIC

2.DocumentationdiscrepancyincomputePriceWithChangeInTau Severity:InformationalDifficulty:Undetermined Type:UndefinedBehaviorFindingID:TOB-HYPR-2 Target: contracts/libraries/Price.sol Description Theformulaprovidedinthe @custom fieldoftheNatSpeccommentforthe computePriceWithChangeInTau functiondoesnotmatchtheformulaimplementedin thefunctionbody.Thediscrepancybetweenthedocumentationandimplementationcan causeenduserstomisunderstandwhatthefunctionactuallydoes. /** *@devComputeschangeinpricegivenachangeintimeinseconds. *@paramstkWAD *@paramvolpercentage *@paramprcWAD *@paramtauseconds *@paramepsilonseconds *@custom:mathP(τ-ε)=(P(τ)^(√(1-ε/τ))/K^2)e^((1/2)(t^2)(√(τ)√(τ-ε)- (τ-ε))) */ Figure2.1:TheNatSpeccommentforthe computePriceWithChangeInTau functionin Price.sol Thefunctionbodyimplementsthefollowingformula: Δ(τ - ε) = ((Δ(τ)/Δ)^(√(1 - ε/τ))) * Δ * Δ^((1/2)(σ^2)(√(τ)√(τ- ε) - (τ - ε))) ThedocumentedandimplementedformulaswillresultindifferentvaluesforP(τ-ε). ExploitScenario Alicecalculatestheresultofherinvestmentusingformulasfromthecodebase’sNatSpec comments.Theresultofhercalculationsconvinceshertosubmittransactions.WhenAlice completesherseriesoftransactions,theendresultdiffersfromherexpectations. Recommendations Shortterm,correcttheformulaerrorintheNatSpeccommentfor computePriceWithChangeInTau sothatitmatchestheimplementation. hasheye 30PrimitiveSecurityAssessment PUBLIC

Longterm,thoroughlyproofreadtheNatSpeccommentsthroughoutthecodebase, especiallywheretheydescribeimportantformulasandoperations. hasheye 31PrimitiveSecurityAssessment PUBLIC

3.Riskoftokentheftduetopossibleintegerunderflowinslt Severity:HighDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-3 Target: contracts/Assembly.sol Description Anattackercanstealfundsfromapoolusingafunctionthatfailstorevertonunexpected input. The addSignedDelta functionisusedbythe allocate , unallocate ,and unstake functionstoaltertheliquidityofapool.Thesefunctionslackcheckstoensurethatthe inputvaluesarewithinpermissiblelimits;instead,theyrelyonthe addSignedDelta functiontorevertonunexpectedinputs. The addSignedDelta functionchecksforthesignofthe delta value;itsubtractsitstwo’s complementfromthe input valueif delta isanegativeinteger.Afterthesubtraction operation,thecodechecksforunderflowusingthe slt (signed-less-than)functionfrom theEVMdialectoftheYULlanguage.The slt functionassumesthatbothargumentsarein atwo’scomplementrepresentationoftheintegervaluesandcheckstheirsignsusingthe mostsignificantbit.Ifanunderflowoccursinthepreviousubtractionoperation, slt ’s output valuewillbeinterpretedasanegativeintegerwithitsmostsignificantbitsetto 1 . Becausethe slt functionwillfindthatthe output valueislessthanthe input valuebased ontheirsigns,itwillreturn 1 .However,the addSignedDelta functionexpectstheresultof slt tobe 0 whenanunderflowoccurs;therefore,itfailstocapturetheunderflow conditioncorrectly. functionaddSignedDelta(uint128input,int128delta)purereturns(uint128output) { bytesmemoryrevertData=abi.encodeWithSelector(InvalidLiquidity.selector); assembly{ switchslt(delta,0)//delta<0?1:0 //negativedelta case1{ output:=sub(input,add(not(delta),1)) switchslt(output,input)//output<input?1:0 case0{ //notlessthan revert(add(32,revertData),mload(revertData))//0x1fff9681 } } hasheye 32PrimitiveSecurityAssessment PUBLIC

//positiondelta case0{ output:=add(input,delta) switchslt(output,input)//(output<input?1:0)==0?1:0 case1{ //lessthan revert(add(32,revertData),mload(revertData))//0x1fff9681 } } } } Figure3.1:Thevulnerable addSignedDelta functionin Assembly.sol ExploitScenario Aliceallocates100USDCintoapool.Eve,anattacker,callsthe unallocate functionwith the100USDCasthe deltaLiquidity argument.The addSignedDelta functionfailsto revertontheintegerunderflowthatresults,andEveisabletowithdrawAlice’sassets. Recommendations Shortterm,takeoneofthefollowingactions: • Correcttheimplementationofthe addSignedDelta functiontoaccountfor underflows. • Usehigh-levelSolioditycodeinsteadofassemblycodetoavoidfurtherissues; assemblycodedoesnotsupportsub256-bittypes. Regardlessofwhichactionistaken,addtestcasestoverifythecorrectnessofthenew implementation;addbothunittestcasesandfuzztestcasestocapturealloftheedge cases.

Longterm,carefullyreviewthecodebasetofindassemblycodeandverifythecorrectness oftheseassemblycodeblocksbyaddingtestcases.Donotrelyoncertainbehaviorof assemblycodewhileworkingwithsub256-bittypesbecausethisbehaviorisnotdefined andcanchangeatanytime. Note:Donotconsiderusingthe lt functioninplaceofthe slt functionbecauseitisnot sufficienttocapturealloftheoverflowandunderflowconditions. hasheye 33PrimitiveSecurityAssessment PUBLIC

4.Riskoftokentheftduetouncheckedtypeconversion Severity:HighDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-4 Target: contracts/Assembly.sol Description Anattackercanstealfundsfromapoolusingafunctionthatfailstorevertonunexpected input. The addSignedDelta functionisusedbythe allocate , unallocate ,and unstake functionstoaltertheliquidityofapool.Thesefunctionslackcheckstoensurethatthe inputvaluesarewithinpermissiblelimits;instead,theyrelyonthe addSignedDelta functiontorevertonunexpectedinputs. Whenthevalueof delta isapositiveinteger,theresultofthe add function(fromtheEVM dialectoftheYULlanguage)willoverflow;however,thecodecannotcapturethisoverflow. Thisisbecausetheargumentsofthefunctionare128-bittypes,buttheassemblycode doesnothavetypesandoperateson256-bitvalues.The add functionreturnsa256-bit valueasitsresult.Theadditionoftwo128-bitintegerscanneveroverflowa256-bitinteger. Forthisreason,theresultofthe add functionwillneverwraparoundthemaximumvalue ofa256-bitinteger,andthe slt functionwillneverfindthe output valuetobelessthan the input value,whichmeansitwillneverreturn 1 toindicatethatanoverflowhas occurred.Asaresult,thecodefailstocapturetheoverflowconditioncorrectly. functionaddSignedDelta(uint128input,int128delta)purereturns(uint128output) { bytesmemoryrevertData=abi.encodeWithSelector(InvalidLiquidity.selector); assembly{ switchslt(delta,0)//delta<0?1:0 //negativedelta case1{ output:=sub(input,add(not(delta),1)) switchslt(output,input)//output<input?1:0 case0{ //notlessthan revert(add(32,revertData),mload(revertData))//0x1fff9681 } } //positiondelta case0{ output:=add(input,delta) hasheye 34PrimitiveSecurityAssessment PUBLIC

switchslt(output,input)//(output<input?1:0)==0?1:0 case1{ //lessthan revert(add(32,revertData),mload(revertData))//0x1fff9681 } } } } Figure4.1:Thevulnerable addSignedDelta functionin Assembly.sol Therearemultiplewaysinwhichanattackercouldusethisissuetowithdrawmore liquiditythantheyhavedepositedinapool. ExploitScenario1 AliceandBoballocate500USDCeachintoaUSDC-ETHpool.Thetotalallocatedliquidityis 1,000USDC.Eve,anattacker,unallocates1,000USDCfromtheentirepool,withdrawing everyone’sassets. ExploitScenario2 Aliceallocates100USDCintoapool.Evecallsthe unstake functiontoincreasethevalueof herownliquiditywithoutdepositinganyassetsandthencalls unallocate towithdrawthe fundsfromthepool. Recommendations Shortterm,takeoneofthefollowingactions: • Correcttheimplementationofthe addSignedDelta functiontoaccountfor overflows. • Usehigh-levelSolidityinsteadofassemblycodetoavoidfurtherissues; assemblycodedoesnotsupportsub256-bittypes. Regardlessofwhichactionistaken,addtestcasestoverifythecorrectnessofthenew implementation;addbothunittestcasesandfuzztestcasestocapturealloftheedge cases. Longterm,carefullyreviewthecodebasetofindassemblycodeandverifythecorrectness oftheseassemblycodeblocksbyaddingtestcases.Donotrelyoncertainbehaviorof assemblycodewhileworkingwithsub256-bittypesbecausethisbehaviorisnotdefined andcanchangeatanytime. hasheye 35PrimitiveSecurityAssessment PUBLIC

5.Userscanswapwithoutpayinganyfees Severity:MediumDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-5 Target: contracts/Hyper.sol Description Whilethe swap functionisbeingprocessed,theinputandoutputvaluesoftheswapare calculatedwithandwithoutthefeeamountassessedfromthetotal.However,iftheuser’s requestedamountexceedsthe maxInput amount,there-additionoftheswapfeeto deltaInput ismissed,allowingtheusertoswapwithoutpayingthefee. The swap functioncallsthe _swapExactIn function,whichcontainslogictosave intermediatevalueswhileprocessingtokenswaps.The deltaInput variableisinitially usedtoderivethe nextIndependent valuewithoutthefeeamountassessed.Thatfee amountshouldbeaddedbackto deltaInput afterwardsothatitcanbeaddedtothe totalamountwithdrawnfromtheuserlaterintheprocess.Thefeeamountisaddedback to deltaInput when _swap.remainder islessthanorequalto maxInput ,butnotif _swap.remainder isgreaterthan maxInput .Ifthefeeisnotadded,thenwhen deltaInput isaddedto _swap.input ,thefeewillnotberepresentedinthatamount, whichmeansthatitwillnotbewithdrawnfromtheuser. if(_swap.remainder>maxInput){ deltaInput=maxInput-_swap.feeAmount; nextIndependent=liveIndependent+deltaInput.divWadDown(_swap.liquidity); _swap.remainder-= (deltaInput+_swap.feeAmount); }else{ deltaInput=_swap.remainder-_swap.feeAmount; nextIndependent=liveIndependent+deltaInput.divWadDown(_swap.liquidity); deltaInput=_swap.remainder;//Swapinputamountincludingthefeepayment.

_swap.remainder=0;//Cleartheremaindertozero,astheorderhasbeen filled. } //Computetheoutputoftheswapbycomputingthedifferencebetweenthe dependentreserves. if(_state.sell)nextDependent=rmm.getYWithX(nextIndependent); elsenextDependent=rmm.getXWithY(nextIndependent); _swap.input+=deltaInput; _swap.output+= (liveDependent-nextDependent); Figure5.1:The _swapExactIn functionin Hyper.sol hasheye 36PrimitiveSecurityAssessment PUBLIC

ExploitScenario Eveexecutesaswapinwhichtheremainderisgreaterthanthemaximuminput.Duetothe calculationsinthe swap function,Evecanswapwithoutpayingthefeeamount. Recommendations Shortterm,fixthefunctiontofactorin feeAmounts whenthevalueof remainder is greaterthanthevalueof maxInput . Longterm,thoroughlyanalyzethesystemtoidentifyinvariantsrelatedtoproperfee assessment.FuzzthoseinvariantsusingEchidnatoensurethatthefunctionsreturnthe expectedvaluesandthattheyareaccurate. hasheye 37PrimitiveSecurityAssessment PUBLIC

6.Swapfunctionreturnsincorrectlyscaledoutputtokenamount Severity:HighDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-6 Target: contracts/Hyper.sol Description The swap function’s output valueisgivenperunitofliquidityinthegivenpool,butitisnot scaledbythepool’stotalliquidity.Asaresult,userswillnotreceivethenumberoftokens thattheyexpectonswaps: functionswap(uint64poolId, boolsellAsset, uintamount, uintlimit)externallockinteractionsreturns(uintoutput,uintremainder){ Figure6.1:Thefunctionsignatureofthe swap functionin Hyper.sol The output tokenvalueiscalculatedusingthedifferencebetweenthe liveDependent and nextDependent variables,bothofwhicharecalculatedusingthereserveamountof theinputtoken.However,theoutputvalueisnotmultipliedbythetotalliquidityvalueof thepool,sotheoutputamountisscaledincorrectly: _swap.output+=(liveDependent-nextDependent); Figure6.2:The output calculationinthe _swapExactIn functionin Hyper.sol Thiscausesthenumberofoutputtokenstobeeithertoofewortoomany,dependingon thecurrentamountofliquidityinthepool. Primitivealsodiscoveredthisissueduringthecodereview. ExploitScenario AliceswapsWETHforUSDCusingHyper.Thepoolhaslessthan1wadofliquidity.The tokenoutputvaluethatisreturnedtoAliceislessthanwhatitshouldbe. Recommendations Shortterm,revisethe swap functionsothatitmultipliesthe output tokenbythetotal liquiditypresentinthepoolinwhichtheswaptakesplace. hasheye 38PrimitiveSecurityAssessment PUBLIC

Longterm,identifyadditionalsysteminvariantsandfuzzthemusingEchidnatoensure thatthefunctionsreturntheexpectedvaluesandthattheyareaccurate. hasheye 39PrimitiveSecurityAssessment PUBLIC

7.Liquidityproviderscanwithdrawtotalfeesearnedbyapool Severity:HighDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-HYPR-7 Target: contracts/Hyper.sol Description The syncPositionFees functionisimplementedincorrectly.Asaresult,liquidity proviederscanwithdrawthetotalfeesearnedbyapoolanddrainassetsfromthecontract reserves. Poolsearnfeesfromswaps,whichshouldbedistributedamongtheliquidityproviders proportionaltotheliquiditytheyprovidedduringtheswaps.However,the Hyper contract insteaddistributesthetotalfeesearnedbythepooltoeveryliquidityprovider,resultingin thedistributionofmoretokensinfeesthanearnedbythepool. Asshowninfigure7.1,the syncPositionFees functionisusedtocomputethefeeearned byaliquidityprovider.Thisfunctionmultipliesthefeeearnedperwadofliquiditybythe liquidity value,providedasanargumenttothefunction. functionsyncPositionFees(HyperPositionstorageself, uintliquidity, uintfeeGrowthAsset, uintfeeGrowthQuote)returns(uintfeeAssetEarned,uintfeeQuoteEarned){ uintcheckpointAsset=Assembly.computeCheckpointDistance(feeGrowthAsset, self.feeGrowthAssetLast); uintcheckpointQuote=Assembly.computeCheckpointDistance(feeGrowthQuote, self.feeGrowthQuoteLast); feeAssetEarned=FixedPointMathLib.mulWadDown(checkpointAsset,liquidity); feeQuoteEarned=FixedPointMathLib.mulWadDown(checkpointQuote,liquidity); self.feeGrowthAssetLast=feeGrowthAsset; self.feeGrowthQuoteLast=feeGrowthQuote; self.tokensOwedAsset+=SafeCastLib.safeCastTo128(feeAssetEarned); self.tokensOwedQuote+=SafeCastLib.safeCastTo128(feeQuoteEarned); } Figure7.1:The syncPositionFees functionin HyperLib.sol hasheye 40PrimitiveSecurityAssessment PUBLIC

The syncPositionFees functionisusedinthe _changeLiquidity and claim functions definedinthe Hyper contract.Inbothlocations,whenthe syncPositionFees functionis called,thevalueofthepool’stotalliquidityisprovidedasthefirstargument,whichisthen multipliedbythefeeearnedperwadofliquidity.Thevalueresultingfromthe multiplicationisthenaddedtothefeeearnedbytheliquidityprovider,whichmeansthe totalfeeearnedbythepoolisaddedtothefeeearnedbytheliquidityprovider. Therearemultiplewaysinwhichanattackercouldusethisissuetowithdrawmorethan whattheyhaveearnedinfees. ExploitScenario1 Eveprovidesminimalliquiditytoapool.Evewaitsforsometimeforsomeswapsto happen.Shecallsthe claim functiontowithdrawthetotalfeeearnedbythepoolduring

theperiodforwhichAliceandotherusershaveprovidedliquidity. ExploitScenario2 Eveprovidesminimalliquiditytoapoolusing10accounts.Evemakessomelargeswapsto accruefeesinthepool.Shethencallsthe claim functionfromall10accountstowithdraw 10timesthetotalfeeshepaidfortheswaps.Sherepeatsthesestepstodrainthecontract reserves. Recommendations Shortterm,revisetherelevantcodesothatthevalueofthesumofaliquidityprovider's liquidity(freeLiquidity summedwith stakedLiquidity)ispassedasanargumentto the syncPositionFees functioninsteadoftheentirepool'sliquidity. Longterm,takethefollowingactions: • Inallfunctions,documenttheirarguments,theirmeanings,andtheirusage. • Addunittestcasesthatcheckallhappyandunhappypaths. • IdentifyadditionalsysteminvariantsandimplementEchidnatocapturebugsrelated tothem. hasheye 41PrimitiveSecurityAssessment PUBLIC

8.AssettokenpricedeviatesfromthepricecurveofthepoolSeverity:UndeterminedDifficulty:Undetermined Type:UndefinedBehaviorFindingID:TOB-HYPR-8 Target: contracts/Hyper.sol Description Theassettokenpricedeviatesfromthepricecurveofthepoolwitheachswapoperation becauseofthepriceadjustmentperformedinthe swap function. Oneachswap,the _swapExactIn functioncomputesnewvirtualreservesofapoolbased ontheuser'sinputandthencomputesthenewpriceoftheassettokenbasedonthepool's newvirtualreserves.Asshowninfigure8.1,afactorof10 11 weiisaddedtothecalculated price. _swap.price= (nextPrice*10_000_001)/10_000_000; Figure8.1:Thepriceadjustmentstatementinthe _swapExactIn functionin Hyper.sol Thisadjustedpriceisthenusedtocalculatetheoutputamountinthenextswapoperation. Thepriceoftheassettokenincreasesbythefactorof10 11 weiwitheachswapoperation, whichcausesthepriceoftheassettokentodeviatefromthepricecurveofthepool. Thisadjustedpriceisalsousedtocalculatethepool'snewvirtualreservesonsubsequent swaps,whichmeansthatthevirtualreservesofthepoolwillbedifferentfromthoseused tocalculatethisprice.Thiscreatesadifferencebetweenthecontract'sbalanceofpool tokensandthepool'svirtualreserves;thisdifferenceincreaseswitheveryswapoperation, causingassetstobecomestuckinthecontract. Recommendations Shortterm,investigatethemaximumamountofdeviationthatcanoccurbetweenthe assettokenpriceandthepricecurveofthepooltoensurethattheerrorfitswithinsafe bounds. Longterm,executethorougheconomicanalysisontheimplicationsofanyupdateto systemvariables.Thisshouldincludemaximumerrorcalculationsforallvariables. hasheye 42PrimitiveSecurityAssessment PUBLIC

9.NewpaircreationcanoverwriteexistingpairsSeverity:HighDifficulty:High Type:UndefinedBehaviorFindingID:TOB-HYPR-9 Target: contracts/Hyper.sol Description Anoverflowofthemaximumvalueof uint24 couldoccurinthe _createPair() function, whichcanbeusedtooverwriteexistingpairs. Asshowninfigure9.1,the _createPair() functionusesanuncheckedblocktocompute andcastthevalueof getPairNonce toassignthevalueof pairId .This pairId isusedas akeyinthe pairs mappingtostoreinformationrelatedtoaspecificpair. unchecked{ pairId=uint24(++getPairNonce); } Figure9.1:The pairId assignmentinthe _createPair() functionin Hyper.sol AccordingtotheSoliditydocumentation,valuesthatundergoexplicittypeconversionare truncatedifthenewtypecannotholdallofthebitsrequiredtorepresentthenewvalue. Here,thetypeof getPairNonce isconvertedfrom uint256 to uint24 ,soifthenewvalue overflowsthemaximum uint24 value,thehigher-orderbitsof getPairNonce willbe truncatedtoanunexpectedvalue.Thismeansthatexistingpoolswillbeoverwrittenby everynewpaircreationoperation,resultinginaninconsistentcontractstatewiththe followingissues: • Twosetsofassetswillstorethesame pairId inthe getPairId mapping. • Thevalueof HyperPair willbeoverwritteninthe pairs mapping. • Allofthepreviouspoolscreatedforthepairwillstillholdthepreviousvalueof HyperPair . Thisoverflowlimitmaynotseemfeasibletoreachbecausethemaximumvalueofthe uint24 typeis16,777,215.Itwouldtakealotoftimeandcostalotofgastocreateso manypairs.However,becauseofthelow-costnatureandhighertransactionthroughputof theL2networks,amalicioususercouldexploitthisissuetoconductaDoSattackonthe protocolwithinsignificantfinancialloss. hasheye 43PrimitiveSecurityAssessment PUBLIC

WealsofoundthatapoolexistencecheckisimplementedIn _createPool function,but aninlinecommentindicatesPrimitive'splanstoremovethischeck.Werecommend keepingthischecktopreventsimilar poolId overflowandoverwritingissues. ExploitScenario EvedeploysnumerousfakeERC-20tokencontracts.ShethenusestheseERC-20tokensto create16,777,215boguspairsinthe Hyper contract.Now,everynewpairoverwrites existingpairs,makingthisinstanceoftheprotocolunusable. Recommendations Shortterm,makethefollowingchanges: 1.Addacheckofthe pairId in _createPair() toensurethatithasnotalready beenusedtopreventexistingpairsfrombeingoverwritten. 2.Increasetheupperboundofthevalueof pairId bychangingitstype. Longterm,carefullyreviewthecodebaseforexplicittypeconversions.Documentscenarios

inwhichtheseexplicittypeconversionscanresultinanoverfloworunderflowoftheresult, anduseEchidnatotestforthesescenariosthroughoutthecodebase. hasheye 44PrimitiveSecurityAssessment PUBLIC

10.ErrorinInvariant.getX Severity:InformationalDifficulty:High Type:UndefinedBehaviorFindingID:TOB-HYPR-10 Target: Invariant.sol Description The getX functioninthe Invariant contractimplementsaformulathatdoesnotalign withtheformulaspecifiedinthewhitepaper. 1*@devComputes`x`in`x=1-Φ(Φ⁻¹((y+k)/K)+σ√τ)`. Figure10.1:TheNatSpeccommentforthe getX() functionin Invariant.sol Thebodyofthefunctionmatchestheformuladescribedintheparenthesesshouldbe (y-k) insteadof (y+k) . Theformulaisderivedbyrearrangingthetermsinthefollowing: y=KΦ(Φ-1(1-x)-σ√τ)+k Bysubtracting k frombothsidesoftheequation,itbecomesclearthattheformulashould read (y-k) . Theactualimpactofthiserrorinthecurrentimplementationislowbecausethefunctionis onlyevercalledwiththeinvariant k=0 . ExploitScenario Inafuturereleaseoftheprotocol,Primitivedecidestousethefunctionwiththeparameter k≠0 .Thisbreaksthedesiredinvariantafterswapsoccur. Recommendations Shortterm,correctthe getX function’scodeandupdatetheformulainthefunction’s NatSpeccomment. Longterm,keeptrackofthederivationsofformulasusedthroughoutthecodebase,and addfuzzteststhatverifythepropertiesofandassumptionsaboutthefunctionsthat implementthem. hasheye 45PrimitiveSecurityAssessment PUBLIC

11.Poolswithoverflowingmaturitydatescanbecreated Severity:LowDifficulty:High Type:DataValidationFindingID:TOB-HYPR-11 Target: Hyper.sol Description Userscouldcreatepoolswithmaturitydatetimestamps(whichareoftype uint32)that aretooclosetowhen uint32 timestampsoverflow;poolswithoverflowingtimestamps wouldbecomeunusable. 326functionmaturity(HyperCurvememoryself)viewreturns(uint32endTimestamp) { 327return(Assembly.convertDaysToSeconds(self.duration)+ self.createdAt).safeCastTo32(); 328} Figure11.1:The maturity() functionin HyperLib.sol Maturitydatesarecurrentlylimitedtofiveyearsinthefuture,andthedatewhen uint32 timestampswilloverflowisSeptember25,2104. Thematuritydateisnotvalidatedonpoolcreation,sopoolsthatwillbeunusablewhenthe year2104approachescouldbecreated.Thematuritydateisalsonotvalidatedinthe checkParameters() functionofthe HyperLib contract,whichcouldallowanattackerto setthetimestampparameterofthepooltoanoverflowingtimestamp. ExploitScenario Inafutureversionoftheprotocol,Primitiveremovesthefive-yearlimitonpoolmaturity dates.Alice,thecontrollerofapool,decidestosetthematuritydatepasttheyear2104 andisabletotrapallofthefundsofthepool’sliquidityproviders. Recommendations Shortterm,addachecktothe checkParameters() functionin HyperLib toensurethat pools’maturitydateswillnotoverflow. Longterm,thoroughlydocumentalloftheassumptionsthataremadeonthecodebase’s variables.Wheretypesarelimitedinsize,usemodulararithmeticoralargerdatatypeto handleanyissuesthatcouldoccur. hasheye 46PrimitiveSecurityAssessment PUBLIC

12.MintingfundstotheHypercontractarbitrarilyincreasesthenextcaller’s balance Severity:InformationalDifficulty:High Type:ConfigurationFindingID:TOB-HYPR-12 Target: Invariant.sol Description Whenausermintsfundstothe Hyper contract,thecontractreliesonacalculationofthe differencebetweenitsphysicalbalanceanditsvirtualbalanceofthegiventoken.However, thiscalculationincreasesthe Hyper contract’sreservesandthenextcaller’sbalance. Toaddtokensintothesystem,userscallthe fund function;thisfunctionusesthe _settlement() function,whichcallsthe settle() function.Thisfunctionusesthe getNetBalance() function,whichcalculatesthedifferencebetweenthereturnvalueof the token.balanceOf functionandthe Hyper contract’sreservesofthetoken. functiongetNetBalance(AccountSystemstorageself,addresstoken,addressaccount) viewreturns(int256net){ uint256internalBalance=self.reserves[token]; uint256physicalBalance=__balanceOf__(token,account); net=int256(physicalBalance)-int256(internalBalance); } Figure12.1:The getNetBalance() functionin Hyper.sol ExploitScenario Eve,anattacker,createsaDRPtoken.Withhermintingrights,shemints1millionDRPto the Hyper contract.Asaresult,whenAlicefundsheraccount,the Hyper contract’sreserve balanceandAlice’strackedtokenbalanceincreaseby1millionDRP,alongwiththetokens sheintendedtofund. Recommendations Shortterm,documentthefactthattheHyperreservesandtherespectiveuser’sbalance fortheairdroppedtokenwillbeattributedtothenextuser. Longterm,clearlyidentifytheexpectedandunexpectedflowsinthecontracttoensure thatusersareawareofexpectedbehavior. hasheye 47PrimitiveSecurityAssessment PUBLIC

13.Poolstrikepricecouldbezeroduetolackoflowerboundcheckon maxTick Severity:HighDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-13 Target: Hyper.sol Description The maxTick

variableisusedtoapproximatethestrikepriceofapool.However,thecode doesnotvalidatethelowerboundof maxTick ,whichmeansthatthestrikepriceofapool canbe 0 ,causingthepool’sassetstobemispriced. The maxTick variableisprovidedbypoolcreatorsandisvalidatedonpoolcreationbythe checkParameters function,whichchecksthatthe volatility , maxTick , duration , jit ,and priorityFee valuesarewithinsafebounds.However,forthe maxTick parameter,thevalidationfunctionchecksonlyitsupperbound: /**@devInvalidparametersshouldrevert.*/ functioncheckParameters(HyperCurvememoryself)viewreturns(bool,bytesmemory){ if(!Assembly.isBetween(self.volatility,MIN_VOLATILITY,MAX_VOLATILITY)) return(false,abi.encodeWithSelector(InvalidVolatility.selector, self.volatility)); if(!Assembly.isBetween(self.duration,MIN_DURATION,MAX_DURATION)) return(false,abi.encodeWithSelector(InvalidDuration.selector, self.duration)); if(self.maxTick ⩾ MAX_TICK)return(false, abi.encodeWithSelector(InvalidTick.selector,self.maxTick));//todo:fix,mintick check? if(self.jit>JUST_IN_TIME_MAX)return(false, abi.encodeWithSelector(InvalidJit.selector,self.jit)); if(!Assembly.isBetween(self.fee,MIN_FEE,MAX_FEE)) return(false,abi.encodeWithSelector(InvalidFee.selector,self.fee)); //0priorityfee==nocontroller,impossibletosettozerounlessdefault fromnoncontrolledpools. if(!Assembly.isBetween(self.priorityFee,0,self.fee)) return(false,abi.encodeWithSelector(InvalidFee.selector, self.priorityFee)); return(true,""); } Figure13.1:The checkParameters() functionin HyperLib.sol The maxTick valueisusedtocalculateprices,includingthestrikepriceofanasset: hasheye 48PrimitiveSecurityAssessment PUBLIC

functionstrike(HyperCurvememoryself)viewreturns(uint){ returnPrice.computePriceWithTick(self.maxTick); } Figure13.2:The strike() functionin HyperLib.sol However,becausethelowerboundof maxTick isnotchecked,the computePriceWithTick functioncouldreturna 0 valuefortheprice,whichwouldcause thesystemtousetheincorrectvalueforstrikeprices. /** *@devComputesapricevaluefromatickkey. * *@custom:mathprice=e^(ln(1.0001)*tick) * *@paramtickKeyofaslotinaprice/liquiditygrid. *@returnpriceWADValueonakey(tick)valuepairofapricegrid. */ functioncomputePriceWithTick(int24tick)internalpurereturns(uint256price){ int256tickWad=int256(tick)*int256(FixedPointMathLib.WAD); price=uint256(FixedPointMathLib.powWad(TICK_BASE,tickWad)); } Figure13.3:The computePriceWithTick() functionin HyperLib.sol ExploitScenario Alicecreatesapoolwitha maxTick valueof -887272 .Uponcalculatingthestrikepriceat maturity,the tickWad and price valuesarecalculatedasfollows: ▯▯▯ 256 ▯▯▯▯▯▯▯ = -887272 * 1▯18 = -7.2▯19 ▯▯▯▯▯ = 1_0001▯14 ^ (-7.2▯19) = 0 Recommendations Shortterm,bound maxTick toalowerboundthatwillnotallowstrikepricestoconverge to 0 ,andhavestrikepricesrounduptoensurethattheycanneverbe 0 . Longterm,clearlydocumenttheexpectedandunexpectedflowsinthecontracttoensure thatusersareawareofexpectedbehavior. hasheye 49PrimitiveSecurityAssessment PUBLIC

14.Roundingerrorallowsliquiditytobeaddedwithoutdepositingtokens Severity:HighDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-14 Target: HyperLib.sol Description Inpoolsthathaveanassettokenofsixdecimals,smallallocationsofthosetokenswillbe scaledbythenumberofdecimalsandthenrounded down.A deltaAsset valueof 0 can bereturnedbythe getLiquidityDeltas function,evenifanonzero deltaLiquidity argumentisprovided,allowinganattackertoaddliquiditywithouttransferringanytokens. functiongetLiquidityDeltas(HyperPoolmemoryself, int128deltaLiquidity)viewreturns(uint128deltaAsset,uint128deltaQuote){ if(deltaLiquidity==0)return(deltaAsset,deltaQuote); (uintamountAsset,uintamountQuote)=self.getAmounts(); uintdelta; if(deltaLiquidity>0){ delta=uint128(deltaLiquidity); deltaAsset=amountAsset.mulWadUp(delta).safeCastTo128(); deltaQuote=amountQuote.mulWadUp(delta).safeCastTo128(); }else{ delta=uint128(-deltaLiquidity); deltaAsset=amountAsset.mulWadDown(delta).safeCastTo128(); deltaQuote=amountQuote.mulWadDown(delta).safeCastTo128(); } } /**@devDecimalamountsperWADofliquidity,roundeddown...*/ functiongetAmounts(HyperPoolmemoryself)viewreturns(uintamountAssetDec,uint amountQuoteDec){ (uintamountAssetWad,uintamountQuoteWad)=self.getAmountsWad(); amountAssetDec=amountAssetWad.scaleFromWadDown(self.pair.decimalsAsset); amountQuoteDec=amountQuoteWad.scaleFromWadDown(self.pair.decimalsQuote); } Figure14.1:The getLiquidityDeltas and getAmounts functionsin HyperLib.sol Inthecalculationof amountAssetDec inthe getAmounts function,theamountofthe assettokeninwadunits(1e18)isscaledtoavaluerepresentativeofthattoken’sdecimals andthenroundeddown.If amountAssetWad isasmallvalue, amountAssetDec is hasheye 50PrimitiveSecurityAssessment PUBLIC

roundeddownto 0 andreturned,trickingthesystemintothinkingzeroassettokensare requiredtofulfillliquidityallocation. ExploitScenario Eve,anattacker,findsorcreatesapoolwheretheassettokenhassixdecimals.Shecalls the allocate functionandaddsthesmallestpossibleunit(1)astheamounttothatpool. The getAmounts functionscalesthevaluetosixdecimals,roundsdown,andreturns 0 for amountAssetDec ,whichisthenmultipliedby deltaLiquidity ;asaresult, 0 isreturned fortherequired deltaAsset .Theparametersofthepoolarechangedandthe _increaseReserves functioniscalledwiththecorrect deltaQuote valuebut 0 forthe deltaAsset value. Recommendations Shortterm,makeoneofthefollowingchanges: • Haveamountsoftokenallocationsroundeduptothenearestdecimalunit dependingonthetoken’sassigneddecimalvalue(e.g.,foratokenwithsixdecimals, amountsshouldrounduptothenexttokendecimalpoint,whichwouldbe 1e12= 1e(18-6)). • Addazero-valuecheckonthereturnvaluesof getAmounts . Becarefultoconsiderthedownstreamimplicationsofanyshort-termfixesimplemented forthisissue,asthe getAmounts functionisusedincriticalsystemoperations. Longterm,continuetoaddunitteststhatconsidertheexpectedoutcomesofawidearray ofinputsandscenarios.Documentalloftheassumptionswithinthesystemand implementfuzztestingforthemwithEchidnainordertocatchedgecaseslikethisthat mightbreakassumptionsthatarenotreadilyapparent. hasheye 51PrimitiveSecurityAssessment PUBLIC

15.AttackerscansandwichchangeParameterscallstostealfunds Severity:HighDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-HYPR-15 Target: HyperLib.sol Description The changeParameters functiondoesnotadjustthereservesaccordingtothenew parameters,whichresultsinadiscrepancybetweenthevirtualreservesofapoolandthe reservesofthetokensinthe Hyper contract. Anattackercancreateanewcontrolledpoolwiththetokentheywanttostealandafake token.Theycanthenuseasequenceofoperations— allocate , changeParameters ,and unallocate — tostealtokensfromthesharedreservesofthe Hyper contract.Inthe worst-casescenario,anattackercandrainallofthefundsfromthe Hyper contract. The Hyper contractdoesnotstorethereservesoftokensinapool.Thevirtualreservesof thepoolarecomputedwiththelastpriceofthegivenassettokenandthecurve parameters.Below,wediscusstheimpactthatachangeinparameterscouldhaveon variousoperations: AllocatingandUnallocating Ausercanaddliquiditytoorremoveliquidityfromapoolusingthe allocate and unallocate functions.Bothofthesefunctionsusethe getLiquidityDeltas functionto computetheamountoftheassettokenandtheamountofthequotetokenrequiredto changetheliquiditybythedesiredamount.The getLiquidityDeltas functioncallsthe getAmountsWad functiontocomputetheamountofreservesrequiredforaddingoneunit ofliquiditytothepool.The getAmountsWad functionusesthelastpriceoftheassettoken (self . lastPrice),strikeprice,timetomaturity,andimpliedvolatilitytocomputethe amountofreservesperliquidity. functiongetAmountsWad(HyperPoolmemoryself)viewreturns(uintamountAssetWad, uintamountQuoteWad){ Price.RMMmemoryrmm=self.getRMM(); amountAssetWad=rmm.getXWithPrice(self.lastPrice); amountQuoteWad=rmm.getYWithX(amountAssetWad); } Figure15.1:The getAmountsWad functionin HyperLib.sol hasheye 52PrimitiveSecurityAssessment PUBLIC

Whenacontrollercallsthe changeParameters function,thefunctionupdatesthepool parametervaluesstoredinthe Hyper contracttotheprovidedvalues.Thesenewvalues arethenusedinthenextexecutionofthe getAmountsWad functionalongwiththevalueof pool . lastPrice ,whichwascomputedwiththepreviouspoolparameters.If getAmountsWad usestheprevious pool.lastPrice valuewithnewpoolparameters,it willreturnnewvaluesforthereservesofthepool;however,the Hyper contract’stoken balanceswillmatchthosecomputedwiththepreviouspoolparameters. Whenausercallsthe unallocate functionafterapool’sparametershavebeenupdated, thenewcomputedreserveamountsareusedtotransfertokenstotheuser.Becausethe Hyper contractusessharedreservesoftokensforallofthepools,ausercanstillwithdraw thetokens,allowingthemtostealtokensfromotherpools. Swapping The swap functioncomputesthepriceoftheassettokenusingthe _computeSyncedPrice function.Thisfunctionusesthepoolparameterstocomputethepriceoftheassettoken.It uses pool . lastPrice asthecurrentpriceoftheassettoken,asshowninfigure15.2.It thencallsthe computePriceChangeWithTime functionwiththepoolparametersandthe timeelapsedsincethelastswapoperation. function_computeSyncedPrice(uint64poolId)internalviewreturns(uint256price, int24tick,uintupdatedTau){ HyperPoolmemorypool=pools[poolId]; if(!pool.exists())revertNonExistentPool(poolId); (price,tick,updatedTau)= (pool.lastPrice,pool.lastTick, pool.tau(_blockTimestamp())); uintpassed=getTimePassed(poolId); if(passed>0){ uint256lastTau=pool.lastTau();//pool.params.maturity()- pool.lastTimestamp. (price,tick)=pool.computePriceChangeWithTime(lastTau,passed); } } Figure15.2:The _computeSyncedPrice() functionin Hyper.sol The computePriceChangeWithTime functioncomputesthestrikepriceofthepooland thencallsthe Price . computeWithChangeInTau functiontogetthecurrentpriceofthe assettoken. functioncomputePriceChangeWithTime(

HyperPoolmemoryself, uinttimeRemaining, uintepsilon)purereturns(uintprice,int24tick){ hasheye
53PrimitiveSecurityAssessment PUBLIC

uintmaxPrice=Price.computePriceWithTick(self.params.maxTick);
price=Price.computePriceWithChangeInTau(maxPrice,self.params.volatility,
self.lastPrice,timeRemaining,epsilon); tick=Price.computeTickWithPrice(price); } Figure15.3:The
computePriceChangeWithTime() functionin HyperLib.sol The computePriceWithChangeInTau()
usesaformulathatisderivedunderthe
assumptionthattheimpliedvolatilityandstrikepriceofthepoolremainconstantduring
theepsilonperiod.Thisepsilonperiodisthetimeelapsedsincethelastswapoperation.
Theproblemariseswhenthecontrollerofthepoolchangesthepool’sparameters.The formulausedinthe
computePriceWithChangeInTau functionthenbecomesinvalidifthe epsilonperiodisgreaterthanzero.
Ifauserswapstokensafterthecontrollerhasupdatedthecurveparameters,thenthe
wrongpricecomputedbytheswapfunctionwillresultinunexpectedbehavior.Thisissue
canbeusedbythecontrollerofthepooltoswapatadiscountedrate.
Therearemultiplewaysinwhichanattackercouldusethisissuetostealfundsfromthe Hyper contract.
ExploitScenario1 Evecreatesanewcontrolledpoolwith WETHasanassettokenandafaketokenasaquote
token.Eveallocates1e18wadofliquidityinthenewpoolbydepositingXamountofWETH
andYamountofthefaketoken.Evethendoublesthestrikepriceofthepoolbycalling changeParameters()
.Thischangeinthestrikepricechangesthevirtualreservesofthe
pool;specifically,itincreasesthenumberofassettokensanddecreasesthenumberof
quotetokensrequiredtoallocate1e18wadofliquidity.Evethenunallocates1e18wad
liquidityandwithdrawsX1amountofWETHandY1amountofthefaketoken.Thevalueof
X1ishigherthanXbecauseofthechangeinthestrikeprice.ThisallowsEvetowithdraw moreWETHthanshedeposited.
ExploitScenario2 Evecreatesacontrolledpoolfortwopopulartokens.Otherusersaddliquiditytothepool.
Evethenchangesthepool’sparameterstochangethestrikepricetoavalueinherfavor
andexecutesalargeswaptobenefitfromtheliquidityaddedtothepool.Theotherusers
seeEve’sactionsasarbitrageandlosethevalueoftheirprovidedliquidity.Evehas
effectivelyswappedtokensatadiscountedrateandstolenfundsfromthepool’sliquidity providers.
Recommendations Shortterm,modifythe changeParameters functionsothatitcomputesnewtoken
reserveamountsforpoolsbasedonupdatedpoolparametersandtransferstokenstoor
fromthecontrollertoalignthe Hyper contractreserveswiththenewpoolreserves. hasheye
54PrimitiveSecurityAssessment PUBLIC

Longterm,carefullyreviewthecodebasetofindinstancesinwhichtheassumptionsused
informulasbecomeinvalidbecauseofuseractions;resolveissuesarisingfromtheuseof
invalidformulas.Addfuzzteststocapturesuchinstancesinthecodebase. hasheye
55PrimitiveSecurityAssessment PUBLIC

16.Limitedprecisioninstrikepricesduetofixedtickspacing Severity:LowDifficulty:Low
Type:DataValidationFindingID:TOB-HYPR-16 Target: Price.sol Description
Thestrikepriceisallowedtotakeonvaluesonlyfromapredefinedsetofvalues.Thisisa
fixedpricinggridwithlimitedprecision,whichmeansthatthestrikepricecandeviatefrom adesiredprice.
ThestrikepriceinHyperissuppliedthroughan int24 tickvaluethatmapstoapriceinthe
pricinggrid,whichwascomputedfromthetickbasevalueusingtheexponentialfunction (price=TICK_BASE^tick
).Asaresult,thepricevaluesarespacedoutexponentially fromeachother. /**
*@devComputesapricevaluefromatickkey. * *@custom:mathprice=e^(ln(1.0001)*tick) *
*@paramtickKeyofaslotinaprice/liquiditygrid.
*@returnpriceWADValueonakey(tick)valuepairofapricegrid. */
functioncomputePriceWithTick(int24tick)internalpurereturns(uint256price){
int256tickWad=int256(tick)*int256(FixedPointMathLib.WAD);
price=uint256(FixedPointMathLib.powWad(TICK_BASE,tickWad)); } Figure16.1:The computePriceWithTick()
functionin Price.sol Theriskofpricedeviationisevidentwhenlookingattheresultingvaluesfromonetickto
another.Givenapriceintherangeof30,000perquotetoken(e.g.,BTC/USD),theprice
differencefromonetick toanotherisapproximately3USD.Thedifferencebetweenthe
ticksgrowsexponentiallybyonepartperthousand.
Furtherimprecisionscouldcompoundwhencomputingthenexttickfromthepriceaftera
swap.However,thisisnotanexploitableissue,asthenexttickisnotactuallyusedto
derivethenextpriceintheprotocol. hasheye 56PrimitiveSecurityAssessment PUBLIC

ExploitScenario AliceopensapoolwithaUSD/BTCpairandwantstosetapriceof30,003USD/BTC.Dueto
thelimitedprecisionintheticks,thestrikepriceendsupbeingsetto30,000USD/BTC. Recommendations
Shortterm,documentwhetherthisisdesiredbehavioranddescribethelimitationsand
roundingissuesthatcouldresultfromit.
Longterm,considerwhetherafixedpricegridisnecessary;ifitisnot,considerusingthe
decimalsrepresentationforstoringprices. hasheye 57PrimitiveSecurityAssessment PUBLIC

17.Functionsthatroundbyadding1resultinunexpectedbehavior Severity:InformationalDifficulty:Undetermined Type:UndefinedBehaviorFindingID:TOB-HYPR-17 Target: Assembly.sol Description Unitconversionfunctionsinthe Assembly contractadd 1 totheresultsoftheirdivision operationstoroundthemup,whichresultsinunexpectedroundingeffects. The scaleFromWadUp() functionisusedtoconverttheinputforswapoperationsfroma wadunittoatokendecimalunit.Thefunctionalwaysadds 1 totheresultofthedivision operation,eveniftheinputamountwouldbeawholenumberintokendecimals.Asa result,theuserwilltransfermoretokensthanexpected. functionscaleFromWadUp(uintamountWad,uintdecimals)purereturns(uintoutputDec) { uintfactor=computeScalar(decimals); assembly{ outputDec:=add(div(amountWad,factor),1) } } Figure17.1:The scaleFromWadUp() functionin Assembly.sol The scaleFromWadUpSigned() alsoadds 1 totheresultofthedivisionoperationto roundupthereturnvalue. functionscaleFromWadUpSigned(intamountWad,uintdecimals)purereturns(int outputDec){ uintfactor=computeScalar(decimals); assembly{ outputDec:=add(sdiv(amountWad,factor),1) } } Figure17.2:The scaleFromWadUpSigned() functionin Assembly.sol Recommendations Shortterm,usetheformula((a-1)/b)+1 inthe scaleFromWadUp() and scaleFromWadUpSigned() functionstocomputetheroundedupresultofthedivision operation a/b . hasheye 58PrimitiveSecurityAssessment PUBLIC

Longterm,reviewtheentirecodebaseforfunctionsthatroundtoensurethattheydonot add 1 unconditionallytoresults.Addfuzzingtestcasestofindedgecasesthatcouldcause unexpectedroundingissues. References • NumberLogic hasheye 59PrimitiveSecurityAssessment PUBLIC

18.Solidityсompileroptimizationscanbeproblematic Severity:InformationalDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-HYPR-18 Target: solstat/foundry.toml Description TheHypercontractshaveenabledoptionalcompileroptimizationsinSolidity. Therehavebeenseveraloptimizationbugswithsecurityimplications.Moreover, optimizationsareactivelybeingdeveloped.Solidityсompileroptimizationsaredisabledby default,anditisunclearhowmanycontractsinthewildactuallyusethem.Therefore,itis unclearhowwelltheyarebeingtestedandexercised. High-severitysecurityissuesduetooptimizationbugshaveoccurredinthepast.A high-severitybuginthe emscripten -generated solc-js compilerusedbyTruffleand Remixpersisteduntillate2018.Thefixforthisbugwasnotreportedintheсolidity CHANGELOG .Anotherhigh-severityoptimizationbugresultinginincorrectbitshiftresults waspatchedinSolidity 0.5.6.Morerecently,anotherbugduetotheincorrectcachingof keccak256 wasreported. AcompilerauditofSolidityfromNovember2018concludedthattheoptionaloptimizations maynotbesafe. Itislikelythatthereарelatentbugsrelatedtooptimizationandthatnewbugswillbe introducedduetofutureoptimizations. ExploitScenario AlatentorfuturebuginSoliditycompileroptimizations—orintheEmscriptentranspilation to solc-js — opensupasecurityvulnerabilityintheSolstatcontracts. Recommendations Shortterm,measurethegassavingsfromoptimizationsandcarefullyweighthemagainst thepossibilityofanoptimization-relatedbug. Longterm,monitorthedevelopmentandadoptionofSoliditycompileroptimizationsto assesstheirmaturity. hasheye 60PrimitiveSecurityAssessment PUBLIC

19.getAmountOutreturnsincorrectvaluewhencalledbycontroller Severity:LowDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-19 Target: HyperLib.sol Description Whenacontrollercontractcallsthe getAmountOut function,the feeAmount shouldbe calculatedusingthe priorityFee value;however,thefunctionincorrectlyusesthe fee value.Thiscausesthefunctiontoreturnanincorrect output value. data.feeAmount= ((data.remainder>maxInput?maxInput:data.remainder)* self.params.fee)/10_000; Figure19.1:The getAmountOut functionin HyperLib.sol Whenacontrollercontractswapstokens,thefeeassessedinthetransactioniscalculated usingthe priorityFee parameter. _state.fee=msg.sender==pool.controller?pool.params.priorityFee: uint(pool.params.fee); Figure19.2:The _swap functionin Hyper.sol Thepurposeofthe getAmountOut functionistoreturntheexpectedtokenamountthat theuserwouldreceiveafterexecutingaswap.Usingthewrong feeAmount willskewthe output calculation,causingadiscrepancybetweenwhattheuserexpectstoreceiveand whattheyactuallyreceiveafterexecutingtheswap. ExploitScenario Alicewantstoswaptwotokensfromapoolthathasacontrollercontractset.Shequeries thecontrollerwiththeproposedswapparameters,andthecontrollercontractcallsthe getAmountOut functionwiththosesameparameters.The getAmountOut functionreturns anoutputamountoffivetokens.Alicesendsaswaptransactiontothecontroller,which callsthe swap functiononthe Hyper contract.Onlyfourtokensarereturnedto Alice insteadoftheexpectedfive. Recommendations Shortterm,refactorthe getAmountOut functiontoaccuratelymirrorcalculationsmadein the swap functionandtouse priorityFee whenacontrollercalls getAmountOut . hasheye 61PrimitiveSecurityAssessment PUBLIC

Longterm,expandthecurrentunittestsuitetoensurethatdatareturnedbyview functionsisaccurateanduptodate. hasheye 62PrimitiveSecurityAssessment PUBLIC

20.Mismatchedbaseunitcomparisoncaninflatelimittolerance Severity:MediumDifficulty:Low Type:DataValidationFindingID:TOB-HYPR-20 Target: Hyper.sol Description Whenauserswapstokens,thecodeenforcestheuser’s limitPrice value,denominated inthetoken’sdecimalunits,bycomparingittothevalueofthepool’s lastPrice value, denominatedinwadunits.Thediscrepancybetweentheseunitscouldpreventauser’s intendedpricelimitfrombeingenforced,resultinginaswapatamarketratethattheuser didnotintendtoswapat. Whenausercallsthe swap function,the limit argumentisusedinthe _swapExactIn functionasthe limitPrice value(figure20.1);the limitPrice valuedetermineswhether theuser’sintendedlimitpricehasbeenexceeded.Theuser’sassumptionisthatthisvalue isdenominatedinthequotetoken’sdecimalunits.The limitPrice valueiscomparedto the nextPrice value,takenfromthepool’s lastPrice value,andiftheuser’spricelimit hasbeenmetorexceeded,theswapreverts. uintnextPrice=pools[args.poolId].lastPrice; if(!sellAsset&&nextPrice>limitPrice)revertSwapLimitReached(); if(sellAsset&&limitPrice>nextPrice)revertSwapLimitReached(); Figure20.1:The _swapExactIn functionin Hyper.sol The lastPrice variableisdenominatedinwadunits,soithas18decimalsofprecision.If thequotetokenusedtodenominate limitPrice hasanyfewerthan18decimals,thenit willalwaysbesmallerthanintendedcomparedto nextPrice .Asaresult,aswap transactionsubmittedbyauserwhohasmetorexceededtheirpricelimitwillnotrevertas expected. ExploitScenario Aliceswapstokensinapoolinwhichbothtokenshavesixdecimals.Shesetsapricelimit ofsevenAtokensforoneBtoken.Thetradecausesthepricingformulatoswingtonine A tokensforoneBtoken.The _swapExactIn functionchecks whether 7e6 is greater than 9e18, which it is not. No SwapLimitReached erroristhrownasaresult,andtheswapis allowedtocompletedespitethesurpassedpricelimit. hasheye 63PrimitiveSecurityAssessment PUBLIC

Recommendations Shortterm,modifytheassociatedcodesothateitherthe limitPrice inputvalueisscaled towadunitsorthe lastPrice valueisscaledtohowevermanydecimalsthequotetoken has.Clearlydocumentforusersthedenominationtheyshoulduseforthepricelimit. Longterm,expandthecurrentunittestsuitetoconsidertokenpoolswithallrangesof decimals;foreachscenario,ensurethatthe swap functionwillrevertwhenapricelimitis reached. hasheye 64PrimitiveSecurityAssessment PUBLIC

21.IncorrectimplementationofedgecasesingetYfunction Severity:LowDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-HYPR-21 Target: solstat/src/Invariant.sol Description The getY functioninthe Invariant contractdeviatesfromtheintendedbehaviorwhen thevalueof R_x isequivalentto WAD and 0 . if(R_x==WAD)returnuint256(int256(stk)+inv);//For`ppf(0)`case,because1-R_x==0,and`y=K*1+k`simplifiesto`y=K+k` if(R_x==0)returnuint256(inv);//For`ppf(1)`case,because1-0==1,and`y =K*0+k`simplifiesto`y=k`. Figure21.1:Thetwoincorrect if statementsinthe getY functionin Invariant.sol Thefirst if statementseemstobecheckingforthe ppf(1) case,evincedbythe comparisonof R_x==WAD ,with WAD representingoneunit.Thefunctionshouldreturnthe invariantinthiscase,butitreturnsthe stk valuesummedwiththeinvariant. y=KΦ(Φ⁻¹(1-1)-σ√τ)+k y=KΦ(Φ⁻¹(0)-σ√τ)+k y=KΦ(negativeinfinite-σ√τ)+k y=KΦ(negativeinfinite)+k y=K*0+k y=k Figure21.2:The getY derivationfromthewhitepaperwhen ppf approachesnegativeinfinity Similarly,when R_x is 0 ,thereturnvalueshouldbe stk summedwiththeinvariant,butit issimplytheinvariant. y=KΦ(Φ⁻¹(1-0)-σ√τ)+k y=KΦ(Φ⁻¹(1)-σ√τ)+k y=KΦ(positiveinfinite-σ√τ)+k y=KΦ(positiveinfinite)+k y=K*1+k y=K+k Figure21.3:The getY derivationfromthewhitepaperwhenapproachingpositiveinfinity Therefore,thereturnvaluesforthetwobranchesareincorrect. hasheye 65PrimitiveSecurityAssessment PUBLIC

ExploitScenario Aliceattemptstoexecuteaswap,forwhichthe R_x valueisequivalentto 0 .Thefunction returnstheinvariantratherthan stk summedwiththeinvariant.Thisresultsinfurther miscalculationsintheswaps. Recommendations Shortterm,switchthereturnvaluestatementsinthetwoaffectedbranchesofthe getY function:if R_x is WAD ,thefunctionshouldreturntheinvariant,andif R_x is 0 ,thefunction shouldreturn stk summedwiththeinvariant. Longterm,thoroughlydocumentalloftheexpectededgecasesofinputsandcheckthat theseedgecasesarehandled. hasheye 66PrimitiveSecurityAssessment PUBLIC

22.LackofproperboundhandlingforsolstatfunctionsSeverity:UndeterminedDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-HYPR-22 Target: solstat/Gaussian.sol Description Theuseofuncheckedassemblyacrossthesystemcombinedwithalackofdatavalidation meansthatobscurebugsthataredifficulttotrackdownmaybeprevalentinthesystem. Oneexampleofuncheckedassemblythatcouldresultinbugsisthe getY function.Dueto assemblyroundingissuesinthecodebase,theinverseofthefunction’stwovariablesdoes nothold.Thisfunctionusestargetedfunctionsinthe Gaussian code.Wewroteafuzztest andranitonthe getY functiontoensurethatthereturnvaluesaremonotonically

decreasing.However,thefuzzingcampaignfoundcasesinwhichthe getY functionreturns alowervaluethanitshould: Logs: BoundResult999999989999999997 BoundResult10000000003 BoundResult10000000000 BoundResult10000000000 BoundResult86400 Error:a⩽bnotsatisfied[uint] Valuea:10000000000 Valueb:99 Figure22.1:Resultsoffuzztestingthe getY function Thesolstat Gaussian contractcontainsthecumulativedistributionfunction(cdf),which reliesonthe erfc function.The erfc function,however,hasmultipleissues,indicatedby itsmanybreakinginvariants;forexample,itisnotmonotonic,itreturnshard-codedvalues outsideofitsinputdomain,ithasinconsistentroundingdirections,anditismissing overflowprotection(furtherdescribedbelow).Thismeansthatthe cdf function's assumptionthatitalwaysreturnsamaximumerrorof 1.2e-7 maynotholdunderall conditions: /** *@noticeApproximationoftheCumulativeDistributionFunction. * *@devEqualto`D(x)=0.5[1+erf((x-μ)/ σ√2)]`. *Onlycomputescdfofadistributionwithμ=0andσ=1. hasheye 67PrimitiveSecurityAssessment PUBLIC

* *@custom:errorMaximumerrorof1.2e-7. *@custom:sourcehttps://mathworld.wolfram.com/NormalDistribution.html. */ functioncdf(int256x)internalpurereturns(int256z){ int256negated; assembly{ letres:=sdiv(mul(x,ONE),SQRT2) negated:=add(not(res),1) } int256_erfc=erfc(negated); assembly{ z:=sdiv(mul(ONE,_erfc),TWO) } } Figure22.2:The cdf() functionin Gaussian.sol The erfc function(andrelatedfunctions)areusedthroughoutthe Hyper contractto computethecontract'sreserves,prices,andtheinvariants.Thisfunctioncontainsafew issues,describedbelowthefigure. functionerfc(int256input)internalpurereturns(int256output){ uint256z=abs(input); int256t; int256step; int256k; assembly{ letquo:=sdiv(mul(z,ONE),TWO) letden:=add(ONE,quo) t:=sdiv(SCALAR_SQRD,den) //[...] } } Figure22.3:The erfc() functionin Gaussian.sol LackofOverflowChecks The erfc functiondoesnotcontainoverflowchecks.Intheaboveassemblyblock,thefirst multiplicationoperationdoesnotcheckforoverflow.Operationsperformedinanassembly blockuseuncheckedarithmeticbydefault.If z ,theabsolutevalueoftheinput,islarger than ⌊ type(int256).max/1e18⌋ (roundedup),themultiplicationoperationwillresultin anoverflow. UseofsdivInsteadofdiv Additionally,the erfc functionusesthe sdiv functioninsteadofthe div functiononthe resultofthemultiplicationoperation.The div functionshouldbeusedinsteadbecause z andtheproductarepositive.Eveniftheresultofthepreviousmultiplicationoperation hasheye 68PrimitiveSecurityAssessment PUBLIC

doesnotoverflow,iftheresultislargerthan type(int256).max ,thenitwillbeincorrectly interpretedasanegativenumberduetotheuseof sdiv . Becauseoftheseissues,theoutputvaluescouldliewellbeyondtheintendedoutput domainofthefunction, [0,2] .Forexample, erfc(x)≈1e57 isapossibleoutputvalue. Otherfunctions—andthosethatrelyon erfc ,suchas pdf , ierfc , cdf , ppf , getX ,and getY — aresimilarlyaffectedandcouldproduceunexpectedresults.Someoftheseissues arefurtheroutlinedinappendixE. Duetothehighcomplexityanduseofthefunctionthroughoutthiscodebase,theexact implicationsofanincorrectboundonthefunctionareunclear.Wespecifyfurtherareas thatrequireinvestigationinappendixC;however,Primitiveshouldconductadditional analysisontheprecisionlossandspecificityofsolstatfunctions. ExploitScenario Anattackerseesthatunderacertainswapconfiguration,theoutputamountin Hyper 's swap functionwillresultinasignificantadvantagefortheattacker. Recommendations Shortterm,rewritealloftheaffectedcodeinhigh-levelSolidditywithnativeoverflow protectionenabled. Longterm,setupsufficientinvarianttestsusingEchidnathatcandetecttheseissuesinthe code.Forallfunctions,performthoroughanalysisonthevalidinputrange,documentall assumptions,andensurethatallfunctionsrevertiftheassumptionsontheinputsdonot hold. hasheye 69PrimitiveSecurityAssessment PUBLIC

23.Attackerscanstealfundsbyswappinginbothdirections Severity:HighDifficulty:Low Type:UndefinedBehaviorFindingID:TOB-HYPR-23 Target: contracts/Hyper.sol Description Unexpectedbehaviorinthe swap functionallowsuserstoprofitwhenexecutingaswapin onedirectionandthenexecutingthesameswapintheotherdirection. Inafuzztest,weidentifiedacaseinwhichanattackerisabletocreateapoolwithacertain configurationthatallowsthemtoswapin1weioftheassettokenandthentoswapback outahighnumberofassettokens.Thiswouldallowtheattackertodrainthepool.This issuewasfoundtowardtheendoftheaudit,sowewereunabletolocatetherootcause. Thisisthepathtaken:Swapping:1asset→1quote→10000000001asset CreatingPool: controller0x7FA9385bE102ac3EAc297483Dd6233D62b3e1496 priorityFee1 fee1 volatility100 duration272 jit0 stk3 price3 Allocatingliquidity:171859515069719386357 Selling,thenbuyingasset Swapping:1asset->1quote→10000000001asset SWAPdir0(selling):asset→quote SWAPdir1(buying):quote→asset balasset10000000000 balquote0 Figure23.1:The swap outputdepictingunexpectedbehavior Recommendations Shortterm,analyzethe swap functiontoidentifytherootcauseofthevulnerability.This

issuestillpersistsafterfixingoverflowissuesandunitconversionsbyreplacingassembly codewithhigh-levelcode. hasheye 70PrimitiveSecurityAssessment PUBLIC

Longterm,addfuzztestcasestofindedgecasescausingissueswithunexpectedrounding behavior. hasheye 71PrimitiveSecurityAssessment PUBLIC

A.VulnerabilityCategories
Thefollowingtablesdescribethevulnerabilitycategories,severitylevels,anddifficulty levelsusedinthisdocument. VulnerabilityCategories CategoryDescription
AccessControlsInsufficientauthorizationorassessmentofrights
AuditingandLoggingInsufficientauditingofactionsorloggingofproblems
AuthenticationImproperidentificationofusers
ConfigurationMisconfiguredservers,devices,orsoftwarecomponents
CryptographyAbreachofsystemconfidentialityorintegrity DataExposureExposureofsensitiveinformation
DataValidationImproperrelianceonthestructureorvaluesofdata
DenialofServiceAsystemfailurewithanavailabilityimpact
ErrorReportingInsecureorinsufficientreportingoferrorconditions
PatchingUseofanoutdatedsoftwarepackageorlibrary
SessionManagementImproperidentificationofauthenticatedusers
TestingInsufficienttestmethodologyortestcoverage TimingRaceconditionsorotherorder-of-operationsflaws UndefinedBehaviorUndefinedbehaviortriggeredwithinthesystem hasheye 72PrimitiveSecurityAssessment PUBLIC

SeverityLevels SeverityDescription
InformationalTheissuedoesnotposeanimmediateriskbutisrelevanttosecuritybest practices.
UndeterminedTheextentoftheriskwasnotdeterminedduringthisengagement.
LowTheriskissmallorisnotonetheclienthasindicatedisimportant.
MediumUserinformationisatrisk;exploitationcouldposereputational,legal,or moderatefinancialrisks.
HighTheflawcouldaffectnumeroususersandhaveseriousreputational,legal, orfinancialimplications.
DifficultyLevels DifficultyDescription
UndeterminedThedifficultyofexploitationwasnotdeterminedduringthisengagement.
LowTheflawiswellknown;publictoolsforitsexploitationexistorcanbe scripted.
MediumAnattackermustwriteanexploitorwillneedin-depthknowledgeofthe system.
HighAnattackermusthaveprivilegedaccesstothesystem,mayneedtoknow complextechnicaldetails,ormustdiscoverotherweaknessestoexploitthis issue. hasheye 73PrimitiveSecurityAssessment PUBLIC

B.CodeMaturityCategories
Thefollowingtablesdescribethecodematuritycategoriesandratingcriteriausedinthis document. CodeMaturityCategories CategoryDescription
ArithmeticTheproperuseofmathematicaloperationsandsemantics
AuditingTheuseofeventauditingandloggingtosupportmonitoring Authentication/ AccessControls
Theuseofrobustaccesscontrolstohandleidentificationand authorizationandtoensuresafeinteractionswiththesystem Complexity Management
Thepresenceofclearstructuresdesignedtomanagesystemcomplexity, includingtheseparationofsystemlogicintoclearlydefinedfunctions Cryptographyand KeyManagement
Thesafeuseofcryptographicprimitivesandfunctions,alongwiththe presenceofrobustmechanismsforkeygenerationanddistribution
DecentralizationThepresenceofadecentralizedgovernancestructureformitigating insiderthreatsandmanagingrisksposedbycontractupgrades
DocumentationThepresenceofcomprehensiveandreadablecodebasedocumentation Transaction ReorderingRisks
Thesystem’sresistancetofront-runningattacks Low-Level Manipulation
Thejustifieduseofinlineassemblyandlow-levelcalls Testingand Verification
Thepresenceofrobusttestingprocedures(e.g.,unittests,integration tests,andverificationmethods)andsufficienttestcoverage RatingCriteria RatingDescription
StrongNoissueswerefound,andthesystemexceedsindustrystandards.
SatisfactoryMinorissueswerefound,butthesystemiscompliantwithbestpractices.
ModerateSomeissuesthatmayaffectsystemsafetywerefound. hasheye 74PrimitiveSecurityAssessment PUBLIC

WeakManyissuesthataffectsystemsafetywerefound.
MissingArequiredcomponentismissing,significantlyaffectingsystemsafety.
NotApplicableThecategoryisnotapplicabletothisreview.
NotConsideredThecategorywasnotconsideredinthisreview. Further Investigation Required
Furtherinvestigationisrequiredtoreachameaningfulconclusion. hasheye 75PrimitiveSecurityAssessment PUBLIC

C.RoundingRecommendations Primitiveusesfixed-pointarithmetic.Thecurrentstrategyhasresultedinincorrect rounding,allowingattackerstobenefitfromdustandmispricesandtostealassets (TOB-HYPR-3,TOB-HYPR-4,TOB-HYPR-5,TOB-HYPR-6,TOB-HYPR-7,TOB-HYPR-14, TOB-HYPR-16,TOB-HYPR-19,TOB-HYPR-20,TOB-HYPR-22,andTOB-HYPR-23) . Theseissues pointtoagreaterneedtotestthesysteminmoredepth.Werecommendensuringthat roundingdirectionsalwaysbenefitthepool. DeterminingRoundingDirections Todeterminehowtoapplyrounding(whetherupordown),considertheresultofthe expectedoutput. Forexample,theformulaforaswapoftokenxfortokenycalculateshowmuchoftokenx mustbesenttothecontracttoreceivey. ⍰' = ⍰φ(φ -1 (1-⍰-στ)+ ⍰ Inordertobenefitthepool,y’musttendtowardalowervalue()tominimizetheamount ⍰ paidout.Asaresult,thefollowingshouldhold: • mustround ⍰φ(φ -1 (1-⍰))⍰ • mustround στ⍰ ○ σ⍰τ⍰ • mustround ⍰⍰ Therefore,themathematicsintheformulashouldperformthischeck: ⍰'⍰ = ⍰⍰φ⍰(φ -1 (1-⍰ -σ⍰τ⍰)+ ⍰⍰ Similarroundingtechniquescanbeappliedinallofthesystem’sformulastoensurethat roundingalwaysoccursinthedirectionthatbenefitsPrimitive. RoundingResults • Whenfundsleavethepool,thesevaluesshouldrounddowntofavortheprotocol overtheuser.Roundingthesevaluesupallowsattackerstoprofitfromtherounding directionbyreceivingmorethanintendedonpoolinteractions. • Whenfundsenterthepool,thesevaluesshouldalwaysrounduptomaximizethe numberoftokensapoolreceives.Roundingthesevaluesdowncanresultin hasheye 76PrimitiveSecurityAssessment PUBLIC

near-zerovalues,whichallowattackerstoprofitfromtheroundingdirectionby receivingheavilydiscountedfunds.Roundingdowntozerocanallowattackersto stealfunds. • Whenfeesarecalculated,theamountattributedtothefeebucketshouldalways rounduptomaximizetheamounttheprotocolreceives.Roundingdownmeans residualdustmaybesenttousersinsteadoftheprotocol. RecommendationsforFurtherInvestigation • Analyzeallinstancesinwhichinvariantsonvariableapproximationsareused, becauseafterroundingandscaling,theoriginalstateoftheunscaledvariablemay notbecorrectlybounded. • Implementthoroughhappyandunhappypathtestingthroughoutthecodebase. hasheye 77PrimitiveSecurityAssessment PUBLIC

D.RiskswithArbitraryTokensandThird-Party Controllers Primitiveaimstoallowthird-partyuserstocreatetheirowntokenpairsandpools.These user-createdpairsandpoolscouldintroduceproblemsthatcouldallowattackerstosteal funds.Werecommendthatusersreviewthetokensandvetthird-partycontrollersto ensurethatpoolsdonotbehaveunexpectedly. Ensurethatusersfollowtheseguidelineswhencreatingtokenpairsandpools: • Poolsshouldneverbeupgradeable.Upgradeablepoolshaveinherentrisksthat maynotbeapparentwithdifferentversions. • Tokensshouldnothaveaself-destructcapability.Destructibletokenshave inherentrisks,includingmaliciousupgradesthrough create2 . • Usersshouldnotbeabletochangetokendecimals.Adjustingatoken’sdecimals toeitherlessthansixorgreaterthan18willbreakthetoken’scomposabilitywith thearithmeticinthepool.AnexampleisshowninfigureD.1. pair_decimals_never_exceed_bounds(uint256):failed!⍰ Callsequence: create_pair_with_safe_preconditions(1,0) setDecimals(0) pair_decimals_never_exceed_bounds(0) FigureD.1:AnEchidnafailureonapairwhosetokendecimalschangedaftercreation • Tokensshouldnotbeinterestbearingorre-adjusting.Theformulaicderivation fortheAMMreliesonarisk-freerateofreturnfortheassettoken.Anyformofa wrappertokenthatpaysfeestoliquidityprovidersposesriskstothecodebase. hasheye 78PrimitiveSecurityAssessment PUBLIC

E.RecommendationsforOverflowandUnderflow AnalysisinAssemblyBlocks Inthisappendix,weprovideрecommendationsforimprovingtheassemblyblocksinthe PrimitiveHypercodebase. Operationsinassemblyblockscanbeproblematicifthecodedoesnotcheckforoverflows orunderflows,andifcertainassumptionsabouttheoperations’inputsarenot documentedorchecked. functionpdf(int256x)internalpurereturns(int256z){ int256e; assembly{ e:=sdiv(mul(add(not(x),1),x),TWO)//(-x*x)/2. } e=FixedPointMathLib.expWad(e); assembly{ z:=sdiv(mul(e,ONE),SQRT_2PI) } } FigureE.1:The pdf() functionin Gaussian.sol Forexample,thenegationoperationinthecodeabove(using mul , not ,and add),doesnot checkwhether x equals type(int256).min ,apossibleinputtothefunction.Callingthe functionwith type(int256).min wouldcauseanoverflowintheadditionoperation, preventingtheresultfrombeingnegated. Additionally,themultiplicationoperation(-x*x) couldunderflowiftheresultislessthan type(int256).min. InthenextassemblyblockinfigureE.1, sdiv isused,where div wouldbeappropriate. Thesolmatefunction expWad ’smaximumoutputissuchthatitcouldbemultipliedbyone wad(1e18)withoutoverflowingin int256 .Using sdiv insteadof div impliesthatthe resultisinterpretedasasignedinteger.However, expWad alwaysoutputspositive numbers;therefore, div

shouldbeusedinstead. Becauseofthewaysolmaterestricts expWad
’soutput,overflowisnotanissueinthiscase. Nonetheless,multiplying e byanythinglargerthan 1e18
couldresultinanoverflow, causing sdiv
tomisinterpretanunsignedvalueasasignedvalue.Theseassumptionsmust
becarefullycheckedanddocumentedwhenusinginlineassembly.
Overflowandunderflowcheckscanbeomittedviauncheckedblockswhereappropriate
analysisisperformedandheavyoptimizationisrequired.Anexampleofsuchanalysisis hasheye
79PrimitiveSecurityAssessment PUBLIC

showninfigureE.2.However,westronglyrecommendthatoverflowchecksalwaysbe
includedbecauseitcanbecomehardtokeeptrackoftheassumptionswhenthecode evolves.
functionpdf_checks_overflow(int256x)internalpurereturns(int256z){
uint256absX=abs(x);//Revertsfor`x=type(int256).min`.
uint256xSquared=absX*absX;//Overflowcheckinuint256isrequired. unchecked{
//Wecansafelycasttheresultofthedivisiontoint256,since
//dividing`xSquared`by`2e18`ensuresthattheresultislessthan `type(int256).max`.
//Theresultispositive,whichmeansthatacheckfor`type(int256).min`
//canbeomittedwhennegatingtheresult. x=-int256(xSquared/2e18); }
int256e=FixedPointMathLib.expWad(k); unchecked{ //Theoutputof`expWad`issuchthatitcanbesafely
//multipliedby`1e18`withoutcausinganoverflowinint256. z=e*ONE/SQRT_2PI; } } FigureE.2:Anexample
pdf_checks_overflow functionwithuncheckedblocksthatcontains
enoughanalysistojustifyomittingoverflowchecks
Althoughoverflowchecksandappropriateanalysisinassemblyblockscanimprove
otherwiseuncheckedcode,abetterpracticeistousehigh-levelSolidityversionsinstead,as
exemplifiedinfigureE.3.Usinghigh-levelSoliditywouldimprovethegivenfunction’s
protectionagainstoverflowandunderflow(whennativeoverflowandunderflowprotection isenabledunder
pragma^0.8.0)andimprovetheauditabilityofthecode.We recommendthatPrimitiveconsiderusingthehigher-
levelimplementationsoffunctionsto allowfortheuseofnativein-builtprotection.
functionerfc_checks_overflow(int256input)internalpurereturns(int256output){
uint256z=abs(input);//Revertsfor`x=type(int256).min`.
//Wecansafelycasttheresultofthedivisiontoint256,
//becauseitispositiveandlessthan`type(int256).max`. int256t=int256(1e36/(1e18+z/2));
int256step=ERFC_J; step=ERFC_I+(t*step/1e18); step=ERFC_H+(t*step/1e18); step=ERFC_G+(t*step/1e18);
step=ERFC_F+(t*step/1e18); step=ERFC_E+(t*step/1e18); step=ERFC_D+(t*step/1e18); step=ERFC_C+
(t*step/1e18); hasheye 80PrimitiveSecurityAssessment PUBLIC

step=ERFC_B+(t*step/1e18); step=-ERFC_A+(t*step/1e18);
//Wecansafelycasttheresultofthedivisiontoint256,
//becauseitispositiveandlessthan`type(int256).max`,
//ifthemultiplicationdoesnotrevertduetooverflow. int256k=step-int256(z*z/1e18);
int256expWad=FixedPointMathLib.expWad(k); int256r=t*expWad/1e18; output=input<0?TWO-r:r; }
FigureE.3:Anexample erfc_checks_overflow functioninhigh-levelSolidity hasheye
81PrimitiveSecurityAssessment PUBLIC

F.StakingIssues Althoughthestaking-relatedcodewasconsideredoutofscopeforthisaudit,wegave
thesecontractsabest-effortreviewandidentifiedthefollowingissues.Becausetheywere
notthoroughlyinvestigated,werecommendthatPrimitivecheckthefollowingareas: •
Whenaswapoccurs,thepriorityfeeamountiscomputedforthetotalpoolliquidity
andtransferredfromthecontroller.Thismeansthatanattackermaybeableto
profitfromexecutingaswapagainstthetotalpoolliquidity. • The unstakeTimestamp
valueisnotupdatedafterpoolparametersareupdated. Thismayresultinundesiredbehavior. • Inthe claim and
_changeStake functions,everyliquidityprovidergetsafee
paymentforthetotalstakedliquidity.Itisunclearwhetherthisbehaviorisintended;
ifitis,itshouldbethoroughlyandclearlydocumented. •
UserscannotwithdrawliquidityafterunstakingduetotheJITrestriction. hasheye
82PrimitiveSecurityAssessment PUBLIC

G.CodeQualityRecommendations
Thefollowingrecommendationsarenotassociatedwithspecificvulnerabilities.However,
theyenhancecodereadabilityandmaypreventtheintroductionofvulnerabilitiesinthe future. •
Useconsistentnamingconventionsthroughoutthecodebase.Choose
conventionsfornamingvariablesandusethoseconventionsconsistently
throughoutthecodebase.FigureG.1showsanexampleofvariablesthatuseleading
underscoresandonethatdoesnot. uint256privatelocked=1; Payment[]private_payments;
SwapStateprivate_state; FigureG.1: private variabledeclarationsin Hyper.sol •
Bewareofpotentialoverflowsinassemblyblocks.Informationcontainedinan

arbitrarilylargebytesarraycouldoverflowwhenloadedintomemory.
functiontoBytes32(bytesmemoryraw)purereturns(bytes32data){ assembly{ data:=mload(add(raw,32))
letshift:=mul(sub(32,mload(raw)),8) data:=shr(shift,data) } } FigureG.2:The toBytes32 functionin
Assembly.sol • Ensurethatvariables'higher-orderbitsareclearedbeforetheyareaccessedin
assemblyblocksiftheirtypesarelessthan256bits.Accessingvariablesoftypes
thatarelessthan256bitsinassemblyblocksdoesnotguaranteethathigher-order
bitswillbezeroedout.SeetheSoliditydocumentationformoredetailsonthisissue.
functionaddSignedDelta(uint128input,int128delta)purereturns(uint128output){
bytesmemoryrevertData=abi.encodeWithSelector(InvalidLiquidity.selector); assembly{
switchslt(delta,0)//delta<0?1:0 //negativedelta case1{ output:=sub(input,add(not(delta),1))
switchslt(output,input)//output<input?1:0 case0{ //notlessthan
revert(add(32,revertData),mload(revertData))//0x1fff9681 } hasheye 83PrimitiveSecurityAssessment
PUBLIC

} //positiondelta case0{ output:=add(input,delta) switchslt(output,input)//(output<input?1:0)==0?
1:0 case1{ //lessthan revert(add(32,revertData),mload(revertData))//0x1fff9681 } } } }
FigureG.3:The addSignedDelta functionin Assembly.sol •
Avoidcastingdowninputs.Acceptinginputsofonetypeonlytocastthemtoa
differenttypeinthecodecanbeconfusingtoendusersandcanmakeitdifficultto
reasonabouthowthesystemwillbehave. functionallocate(uint64poolId, uintamount
)externallockinteractionsreturns(uintdeltaAsset,uintdeltaQuote){ booluseMax=amount==type(uint).max;
(deltaAsset,deltaQuote)=_allocate(useMax,poolId,(useMax?1: amount).safeCastTo128()); }
FigureG.4:The allocate functionin Hyper.sol • Standardizetheuseof uint and uint256
throughoutthecode. uint isanalias of uint256 ,andthetwocanbeusedinterchangeablywithoutalteringthe
underlyingtype.However,itisbestpracticetocommittousingoneortheother throughoutacodebase. •
Followconsistentconventionsforoutputsreturnedintuples.Doingsocan
improvethecodebase'sreadability.Forexample,thecodeinfigureG.5returnsthe
quoteamountfirstfollowedbytheassetamount;however,thecodeinfigureG.6 outputstheminreverse.
functioncomputeReserves(RMMmemoryargs,uintprc)internalpurereturns(uintR_y, uintR_x){ FigureG.5:The
computeReserves functionfrom Price.sol returnsthequoteamountand thentheassetamount.
functiongetLiquidityDeltas(HyperPoolmemoryself, int128deltaLiquidity hasheye
84PrimitiveSecurityAssessment PUBLIC

)viewreturns(uint128deltaAsset,uint128deltaQuote){ FigureG.6:The getLiquidityDeltas functionfrom
HyperLib.sol returnstheassetamount andthenthequoteamount. •
Revisecertainerrormessagesthatlackdetail.Insomepartsofthecodebase,
calculationsandroundingcouldtriggerunderflowanddivide-by-zerorevertsthatdo
notreturnhelpfulerrormessages.Enduserscouldbeconfusedaboutwhyexactly
theirapparentlyvalidtransactionshavenotsucceeded.Anexampleofthisissuecan
befoundincalculationsmadebythe computePriceWithChangeInTau functionof the Price library. hasheye
85PrimitiveSecurityAssessment PUBLIC