# 链网融合逼真灵活攻防演练靶标自动化构建技术组件

## 区块链网络靶标自动化构建测试

### 区块链网络靶标自动化构建模块

128 区块链网络靶标配置自动化生成

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链网络靶标配置自动化生成（以太坊） |
| **测试目的** | 验证提出链网场景快速构建方法，实现基于网络靶场的区块链网络靶标场景配置及导入 |
| **测试环境** | 1. 网络靶场 2. 区块链靶标配置构建平台 |
| **必选/可选** | 必选 |
| **前置条件** | 无 |
| **测试流程** | 1. 在首页输入用户名及密码，登录配置构建平台； 2. 进入系统后点击“区块链节点配置”。配置以太坊网络场景参数，包括客户端版本号、启动节点配额、出块节点配额等，点击“确认”生成以太坊网络配置； 3. 点击“基础网络拓扑配置”，输入基础网络构建参数，包含网络拓扑规模和节点镜像参数，并点击“确认”； 4. 预览并下载基础网络拓扑配置文件； 5. 输入基础链网场景参数，包含客户端版本号、启动节点数量、出块节点数量等参数，并确认； 6. 预览并下载链网场景配置文件。 |
| **测试判据** | 1. 验证成功登陆受测系统 2. 验证受测系统成功生成基础网络拓扑配置文件 3. 验证受测系统成功生成链网场景配置文件 |
| **测试结果** |  |
| **测试结论** |  |

129 区块链网络靶标自动化构建测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链网络靶标自动化构建测试（以太坊） |
| **测试目的** | 验证在网络靶场上构建支持区块链系统部署的网络靶标一套，区块链节点不少于500个 |
| **测试环境** | 网络靶场 |
| **必选/可选** | 必选 |
| **前置条件** | 完成区块链网络靶标配置自动化生成（以太坊） |
| **测试流程** | 1. 登录网络靶场，在首页输入用户名及密码； 2. 进入网络靶场并创建工程； 3. 进入已创建的工程，导入已生成的500节点基础网络拓扑文件； 4. 选择部署上述工程，等待工程部署完成； 5. 选择路由下发，下发路由协议； 6. 选择配置导入，导入已生成的500节点链网场景配置文件，包括： 7. 开启导调，等待导调完成； 8. 进入设备管理，通过SSH方式进入节点，查看其客户端配置详情； 9. 在任一个节点客户端查看区块同步情况，验证区块链网络可用性。 |
| **测试判据** | 1. 验证成功登陆网络靶场平台 2. 验证成功创建工程 3. 验证成功导入受测系统生成的500节点基础网络拓扑文件至工程 4. 验证成功部署上述工程并完成路由下发 5. 验证成功导入受测系统生成的500节点链网场景配置文件至工程 6. 验证测试后配置下发后的节点客户端能够完成区块链业务功能 |
| **测试结果** |  |
| **测试结论** |  |

## 逼真灵活攻防演练靶标攻防演练测试

### 逼真灵活攻防演练靶标攻防演练模块

130 区块链网络靶场DDoS攻击技术复现测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链网络靶场DDoS攻击技术复现测试 |
| **测试目的** | 验证在网络靶场上实现DDoS攻击技术复现；区块链节点不少于500个 |
| **测试环境** | 网络靶场 |
| **必选/可选** | 必选 |
| **前置条件** | 区块链网络靶标自动化构建（以太坊） |
| **测试流程** | 1. 登录网络靶场，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在矿工节点中，执行挖矿命令，进行挖矿； 4. 在背景节点，攻击节点中分别创建8个账户。 5. 矿工节点向所有的账户进行转帐操作； 6. 在观察节点更新观察脚本，更新脚本中的账户地址为攻击节点中生成的地址vi web3/getTx\_multi.js，然后在/root目录下运行观察脚本 bash getTx.sh； 7. 然后在背景节点中运行执行交易的脚本，发送交易：bash Back\_Start.sh； 8. 在攻击节点 /root目录下执行攻击脚本：bash att.sh； 9. 返回交易观察节点，观察交易池状态； |
| **测试判据** | 1. 验证矿工节点转账成功 2. 验证各脚本执行成功 3. 验证交易池被堵塞 |
| **测试结果** |  |
| **测试结论** |  |

131 区块链网络靶场重入攻击技术复现测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链网络靶场日蚀攻击技术复现测试 |
| **测试目的** | 验证在网络靶场上实现日蚀攻击技术复现；区块链节点不少于500个 |
| **测试环境** | 网络靶场 |
| **必选/可选** | 必选 |
| **前置条件** | 区块链网络靶标自动化构建（以太坊） |
| **测试流程** | 1. 登录网络靶场，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点；在节点上，执行geth attach geth.ipc命令，进入geth 3. 执行personal.newAccount("password")命令生成一个新的账户； 4. 然后退出交互环境，在keystore中找到对应地址的文件，复制到Replay.js中,以及在Replay.js中更新新建账户的密码； 5. 执行重入脚本：node Replay.js； 6. 查看被攻击合约的资金是否被恶意窃取； 7. 通过多次随机重复测试，多次测试后的得出平均结果作为最终测试结果。 |
| **测试判据** | 1. 验证成功执行重入攻击脚本 2. 验证恶意交易执行成功 3. 验证受害合约资金被恶意窃取 |
| **测试结果** |  |
| **测试结论** |  |

132 区块链网络靶场交易抢跑攻击技术复现测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链网络靶场交易抢跑攻击技术复现测试 |
| **测试目的** | 验证在网络靶场上实现交易抢跑攻击技术复现；区块链节点不少于500个 |
| **测试环境** | 网络靶场 |
| **必选/可选** | 必选 |
| **前置条件** | 区块链网络靶标自动化构建（以太坊） |
| **测试流程** | 1. 登录网络靶场，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在矿工节点上，执行geth attach geth.ipc命令，进入geth，然后进行挖矿； 4. 等到区块高度为500时，向其他账户转账； 5. 然后在矿工节点上，执行：node Deploy.js来部署受害者合约，并记录合约地址； 6. 在攻击节点更新攻击脚本：Attack.js，把里面合约地址改为受害者合约地址； 7. 在矿工节点上，执行脚本：node Deploytest.js,来部署模拟背景流量的合约，并记录背景合约地址 8. 继续修改矿工节点上的Ready.js,把脚本里面的合约地址改为背景合约地址； 9. 在拦截节点，执行抢跑交易脚本，来进行监控链上合约调用情况； 10. 在矿工节点，执行背景流量脚本:Ready.js； 11. 在攻击节点上执行攻击脚本:node Attack.js； 12. 在矿工节点，查看账户地址，观察抢跑交易是否拦截攻击成功   通过多次随机重复测试，多次测试后的得出平均结果作为最终测试结果。 |
| **测试判据** | 1. 验证成功执行区块监控脚本及攻击脚本 2. 验证在矿工节点的区块监控脚本日志中，攻击交易抢先执行成功 |
| **测试结果** |  |
| **测试结论** |  |

# 链网融合区块链网络技术验证平台

## 链网融合场景资源验证功能测试

### 链网融合场景资源验证模块

133 在开放公有的网络环境下，平台支持部署不少于5000个区块链系统节点。

|  |  |
| --- | --- |
| **测试编号** | 在受测系统中测试开放公有环境下的靶标构建能力 |
| **测试项目** | 开放公有区块链网络场景构建接口功能测试 |
| **测试目的** | 验证在开放公有的网络环境下，受测系统支持场景构建接口功能 |
| **测试环境** | 1、网络靶场  2、链网融合平台  3、BSN平台 |
| **必选/可选** | 必选 |
| **前置条件** | 无 |
| **测试流程** | 1. 运行大规模一体化链网协同技术验证平台后端； 2. 运行大规模一体化链网协同技术验证平台前端； 3. 测试平台后端所在服务器的网络带宽； 4. 在平台首页输入用户名及密码，登录链网融合平台； 5. 查询项目可用资源，点击“链网一体化资源查询”; 6. 查看链网一体化资源状态。 |
| **测试判据** | 1. 验证成功登录链网融合平台；的网络带宽不小于40Mbps； 2. 验证项目可用资源查询成功。 3. 平台支持的可部署节点总数不少于5000。 |
| **测试结果** |  |
| **测试结论** |  |

134 开放公有区块链网络部署功能测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 开放公有区块链网络部署功能测试 |
| **测试目的** | 验证在开放公有的网络环境下，验证平台支持区块链系统节点部署 |
| **测试环境** | 网络靶场 |
| **必选/可选** | 必选 |
| **前置条件** | 开放公有区块链网络场景资源已确认 |
| **测试流程** | 1. 在平台首页输入用户名及密码，登录链网融合平台； 2. 查询项目可用资源，并确认总数不少于5000节点； 3. 点击“链网一体化场景构建”。输入将要构建的以太坊网络Id和共识节点数量与全节点数量，点击“确认”进行以太坊网络创建； |
| **测试判据** | 1. 验证成功登录链网融合平台 2. 验证项目可用资源查询成功 3. 验证以太坊网络创建事件执行成功 |
| **测试结果** |  |
| **测试结论** |  |

135 开放公有区块链网络环境验证功能测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 开放公有区块链网络环境验证功能测试 |
| **测试目的** | 验证在开放公有的网络环境下，验证平台支持对创建的大规模区块链网络（不少于5000节点，不少于3个境外节点）环境进行验证 |
| **测试环境** | 1. 网络靶场 2. 一体化验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 区块链网络创建事件执行成功 |
| **测试流程** | 1. 在平台首页输入用户名及密码，登录大规模一体化链网协同技术验证平台； 2. 点击链网一体化场景查询，根据以太坊网络Id查询创建结果； 3. 根据查询结果内已部署节点IP，验证不少于3个境外节点； 4. 执行脚本，验证5000节点网络同步正常。 |
| **测试判据** | 1. 验证成功登录一体化链网协同技术验证平台 2. 验证区块链网络节点部署结果查询成功，包含不少于5000节点，其中不少于3个境外节点 3. 5000节点区块正常同步，区块高度差异不超过3个区块。 |
| **测试结果** |  |
| **测试结论** |  |

## 高性能指标测试

### 高性能指标模块

136 区块链验证平台链网传输延迟测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网传输延迟测试 |
| **测试目的** | 验证在验证平台上实现链网传输延迟测试 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在普通节点1上运行python receive.py 4. 修改send.py中ip为第一个节点，即接受节点的ip 5. 在普通节点2上运行python send.py 6. 在普通节点2上执行geth attach geth.ipc命令，进入geth，然后解锁账号并发送交易，eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:100}) 7. 查看普通节点1 |
| **测试判据** | 1. 验证成功执行监控脚本，测出时间延迟 |
| **测试结果** |  |
| **测试结论** |  |

137 区块链验证平台链网传输时延抖动测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网传输时延抖动测试 |
| **测试目的** | 验证在验证平台上实现链网传输时延抖动测试；区块链节点不少于500个 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在其他节点上运行python receive\_node.py 4. 修改send.py中ip为第一个节点，即接受节点的ip 5. 在普通节点2上运行python send\_node.py 6. 在普通节点2上执行geth attach geth.ipc命令，进入geth，然后解锁账号并发送交易，eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:100}) 7. 查看普通节点2 |
| **测试判据** | 1. 验证成功执行区块监控脚本，链网传输时延抖动测试完成 |
| **测试结果** |  |
| **测试结论** |  |

138 区块链验证平台节点带宽利用率测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台节点带宽利用率测试 |
| **测试目的** | 验证在验证平台上实现节点带宽利用率测试 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点上运行python BU.py； |
| **测试判据** | 1. 验证成功执行脚本，测出节点带宽利用率 |
| **测试结果** |  |
| **测试结论** |  |

139 区块链验证平台链网带宽均衡性测试

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网带宽均衡性测试 |
| **测试目的** | 验证在验证平台上实现链网带宽均衡性测试 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在普通节点1上运行python server.py； 4. 在其他节点上运行python BUi.py |
| **测试判据** | 1. 验证server脚本执行 2. 验证多节点带宽均衡性 |
| **测试结果** |  |
| **测试结论** |  |

140 区块链验证平台网络连通性

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台网络连通性 |
| **测试目的** | 验证在验证平台上实现网络连通性测试；区块链节点不少于500个 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python PCu.py 命令 |
| **测试判据** | 1. 验证成功执行脚本，网络连通性指标正常 |
| **测试结果** |  |
| **测试结论** |  |

141 区块链验证平台网络连通均衡性

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台网络连通均衡性 |
| **测试目的** | 验证在验证平台上实现网络连通均衡性测试；区块链节点不少于500个 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python ANC.py 命令 |
| **测试判据** | 1. 验证成功执行脚本，网络连通均衡性指标正常 |
| **测试结果** |  |
| **测试结论** |  |

142 区块链验证平台基础资源利用率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台基础资源利用率 |
| **测试目的** | 验证在验证平台上实现基础资源利用率指标检测；区块链节点不少于500个 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python3 TPC.py命令 |
| **测试判据** | 1. 验证成功执行脚本，基础资源利用率指标检测正常 |
| **测试结果** |  |
| **测试结论** |  |

143 区块链验证平台链网节点发现率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点发现率 |
| **测试目的** | 验证在验证平台上节点被添加到网络中的速度。 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python PDR.py命令 |
| **测试判据** | 1. 验证成功执行脚本，链网节点发现率指标检测正常 |
| **测试结果** |  |
| **测试结论** |  |

144 区块链验证平台链网RPC响应率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网RPC响应率 |
| **测试目的** | 验证在验证平台上实现RPC响应率检测 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python RRR\_u.py命令 |
| **测试判据** | 1. 验证成功执行脚本，RPC响应率指标正常 |
| **测试结果** |  |
| **测试结论** |  |

145 区块链验证平台链网全主动冗余

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网全主动冗余 |
| **测试目的** | 验证在验证平台上备份完整状态副本的节点数与节点总数量的比值；区块链节点不少于500个 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python RRR.py命令 |
| **测试判据** | 1. 验证成功执行脚本 |
| **测试结果** |  |
| **测试结论** |  |

146 区块链验证平台链网节点异步性

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点异步性 |
| **测试目的** | 验证在验证平台上链网中节点区块高度与最新区块高度的差值 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python NA.py命令 |
| **测试判据** | 1. 验证成功执行脚本 |
| **测试结果** |  |
| **测试结论** |  |

147 区块链验证平台链网交易挂起率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易挂起率 |
| **测试目的** | 验证在验证平台上链网节点交易池中处于挂起状态的交易数量 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点下运行python3 TPR.py命令 |
| **测试判据** | 1. 验证成功执行脚本，链网交易被成功挂起 |
| **测试结果** |  |
| **测试结论** |  |

148 区块链验证平台链网交易验证失败率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易验证失败率 |
| **测试目的** | 验证在验证平台上链网中交易不能被节点验证通过的频率 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python3 TVFR.py |
| **测试判据** | 1. 验证成功执行脚本，验证交易失败率 |
| **测试结果** |  |
| **测试结论** |  |

149 区块链验证平台链网区块验证成功率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网区块验证成功率 |
| **测试目的** | 验证在验证平台上链网中区块被节点验证通过的频率 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python BVSR.py |
| **测试判据** | 1. 验证成功执行脚本，区块打包上链成功被脚本检测 |
| **测试结果** |  |
| **测试结论** |  |

150 区块链验证平台链网系统响应比

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网系统响应比 |
| **测试目的** | 验证在验证平台上检测链网节点在处理本地RPC请求时，花费在处理请求的实际时间与请求等待时间之和的比率 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点上中运命令python3 RTR.py，持续运行脚本 4. 在普通节点上，执行geth attach geth.ipc命令，进入geth控制台 5. 在geth控制台中运行调用RPC的命令 6. 脚本输出系统响应比指标 |
| **测试判据** | 1. 验证Geth成功处理RPC请求 2. 验证成功执行系统响应比检测脚本 |
| **测试结果** |  |
| **测试结论** |  |

## 高通量指标测试

### 高通量指标模块

151 区块链验证平台链网交易吞吐率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易吞吐率 |
| **测试目的** | 验证在验证平台上实现测量链网在给定时间段内平均每秒成功处理的交易数量； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在B节点上，利用for循环var from = eth.accounts[0];var to = "转账地址";var count = 1000; var amount = web3.toWei(0.000000000000000010, "ether"); for (var i = 0; i < count; i++) {var tx = eth.sendTransaction({from: from, to: to, value: amount});console.log("第 " + (i+1) + " 笔交易已发送，TxHash: " + tx);}向A节点持续发送交易 4. 在A节点上运行python 1TPS.py，监控交易 |
| **测试判据** | 1. 验证链网交易可执行性，并测出TPS指标 |
| **测试结果** |  |
| **测试结论** |  |

152 区块链验证平台交易传输开销

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易传输开销 |
| **测试目的** | 验证在验证平台上实现检测链网中节点将交易从本地交易池广播到其它节点交易池接收到交易的时间与带宽使用开销； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 进入监控节点，在监控节点执行脚本2ABCI.py,开启监听； 4. 进入A节点，使用命令personal. unlockAccount ( eth . accounts [0] , "123" , 10000)解锁账户； 5. 从A节点向B节点持续发送交易； |
| **测试判据** | 1. 验证成功执行脚本，检测交易传输开销 |
| **测试结果** |  |
| **测试结论** |  |

153 区块链验证平台链网分叉率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网分叉率 |
| **测试目的** | 验证在验证平台上实现检测链网中发生分叉事件的频率； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 选择两个节点运行下列命令，使其与链网中其他节点分离   sudo iptables -A INPUT -p tcp --dport 30303 -j DROP  sudo iptables -A INPUT -p udp --dport 30303 -j DROP  sudo iptables -A OUTPUT -p tcp --dport 30303 -j DROP  sudo iptables -A OUTPUT -p udp --dport 30303 -j DROP   1. 等待链网中各个节点分别挖矿 2. 在选择的节点中执行python 3FR.py，运行检测脚本 |
| **测试判据** | 1. 验证成功执行脚本，得到分叉率 |
| **测试结果** |  |
| **测试结论** |  |

154 区块链验证平台链网共识成本时间

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网共识成本时间 |
| **测试目的** | 验证在验证平台上实现检测链网中一笔交易从成为待处理状态到其所在区块与最新区块的高度差为12的时间开销； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在普通节点上，执行geth attach geth.ipc命令，进入geth 4. 解锁账号后，利用for循环var from = eth.accounts[0];var to = "转账地址";var count = 20; var amount = web3.toWei(0.000000000000000010, "ether"); for (var i = 0; i < count; i++) {var tx = eth.sendTransaction({from: from, to: to, value: amount});console.log("第 " + (i+1) + " 笔交易已发送，TxHash: " + tx);}向外发送20笔交易 5. 在终端执行python 4CCT.py |
| **测试判据** | 1. 验证成功执行脚本，检测到交易共识时间 |
| **测试结果** |  |
| **测试结论** |  |

155 区块链验证平台链网挖矿CPU占用比MCR

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网挖矿CPU占用比MCR |
| **测试目的** | 验证在验证平台上实现检测Geth 客户端处于挖矿状态与空闲状态时的 CPU 占用率之比； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python 5MCR.py |
| **测试判据** | 1. 验证成功执行脚本，检测出挖矿资源占比 |
| **测试结果** |  |
| **测试结论** |  |

156 区块链验证平台链网节点出块间隔

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点出块间隔 |
| **测试目的** | 验证在验证平台上实现检测链网中矿工节点生成新区块的时间间隔； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 进入终端执行tail -f geth.log检测节点是否在出块； 4. 执行python 6ABI.py 5. 等待100s，脚本输出节点出块间隔 |
| **测试判据** | 1. 验证成功执行脚本，检测出节点出块间隔 |
| **测试结果** |  |
| **测试结论** |  |

157 区块链验证平台链网节点请求回复比NDRR

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点请求回复比NDRR |
| **测试目的** | 验证在验证平台上实现； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 执行命令curl -X POST -H "Content-Type: application/json" \   --data '{"jsonrpc":"2.0","method":"debug\_verbosity","params":[5],"id":1}' \  http://127.0.0.1:8545/ ，对日志进行提权   1. 执行命令 python 7NDRR.py，运行检测脚本 |
| **测试判据** | 1. 验证成功执行脚本，得到链网节点请求回复比NDRR |
| **测试结果** |  |
| **测试结论** |  |

158 区块链验证平台链网交易花费平均值ATC

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易花费平均值ATC |
| **测试目的** | 验证在验证平台上实现检测链网中一个区块内所有交易的gas使用量平均值； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在普通节点上，执行geth attach geth.ipc命令，进入geth 4. 解锁账号后，使用for循环var from = eth.accounts[0];var to = "转账地址";var count = 20; var amount = web3.toWei(0.000000000000000010, "ether"); for (var i = 0; i < count; i++) {var tx = eth.sendTransaction({from: from, to: to, value: amount});console.log("第 " + (i+1) + " 笔交易已发送，TxHash: " + tx);}向外发送20笔交易 5. 在节点中运行python 8ATC.py，检测区块中的gas费 |
| **测试判据** | 1. 验证成功执行脚本，检测出区块中的gas费 |
| **测试结果** |  |
| **测试结论** |  |

159 区块链验证平台链网丢包率PL

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网丢包率PL |
| **测试目的** | 验证在验证平台上实现检测链网节点传输数据过程中丢失的数据包比例； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python 9PL.py命令 |
| **测试判据** | 1. 验证成功执行脚本，检测出丢包率 |
| **测试结果** |  |
| **测试结论** |  |

160 区块链验证平台链网交易确认时间ATT

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易确认时间ATT |
| **测试目的** | 验证在验证平台上实现链网中一笔交易从提交请求到该笔交易被同步上链的时间； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在普通节点上，执行geth attach geth.ipc命令，进入geth 4. 解锁账号后，使用eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:100})命令发送交易 5. 在节点中运行python 10ATT.py |
| **测试判据** | 1. 验证成功执行脚本，检测出交易确认时间指标 |
| **测试结果** |  |
| **测试结论** |  |

161 区块链验证平台链网交易等待时间WT

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易等待时间WT |
| **测试目的** | 验证在验证平台上实现检测交易在上链前留存在交易池中的时间； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在普通节点上，执行geth attach geth.ipc命令，进入geth 4. 解锁账号后，使用eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:100})命令发送交易 5. 运行python 11WT.py |
| **测试判据** | 1. 验证成功执行脚本，检测出交易等待时间指标 |
| **测试结果** |  |
| **测试结论** |  |

162 区块链验证平台链网节点交易吞吐量TTP

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易吞吐量TTP |
| **测试目的** | 验证在验证平台上测试节点在单位时间内处理完成的交易数量； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在A节点执行geth attach geth.ipc命令，进入geth解锁账号后，使用for循环var from = eth.accounts[0];var to = "转账地址";var count = 20; var amount = web3.toWei(0.000000000000000010, "ether"); for (var i = 0; i < count; i++) {var tx = eth.sendTransaction({from: from, to: to, value: amount});console.log("第 " + (i+1) + " 笔交易已发送，TxHash: " + tx);}向监控节点发送20笔交易 4. 在监控节点中运行python 12TTP.py,执行脚本，查看脚本执行结果 |
| **测试判据** | 1. 验证成功执行脚本，检测出测试节点在单位时间内处理完成的交易数量 |
| **测试结果** |  |
| **测试结论** |  |

## 高安全指标测试

### 高安全指标模块

163 区块链验证平台链网交易丢弃率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易丢弃率 |
| **测试目的** | 验证在验证平台上链网中节点运行期间，检测由于交易池容量已满而拒绝接收新交易的频率 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 进入普通节点2 运行python calculate\_TPTR.py |
| **测试判据** | 1. 验证成功执行脚本，得到链网交易丢弃率 |
| **测试结果** |  |
| **测试结论** |  |

164 区块链验证平台链网节点连接失败率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点连接失败率 |
| **测试目的** | 验证在验证平台上实现链网中节点与其它节点的P2P连接断开、拒绝的频率； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python calculate\_ncfr.py |
| **测试判据** | 1. 验证成功执行脚本，得到链网中节点与其它节点的P2P连接断开、拒绝的频率 |
| **测试结果** |  |
| **测试结论** |  |

165 区块链验证平台链网交易转发覆盖率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网交易转发覆盖率 |
| **测试目的** | 验证在验证平台上实现链网中一笔交易广播期间，参与广播行为的节点数量与节点总数量的比值； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点一中运行python tfcr\_server\_strict.py 4. 在其他节点中分别运行python tfcr\_client\_0.2.py和python tfcr\_client\_0.3.py 5. 在节点中在普通节点上，执行geth attach geth.ipc命令，进入geth 6. 在geth中运行miner. start (1)开启挖矿 7. 解锁账号后，使用eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:100})命令发送交易 8. 检查脚本输出 |
| **测试判据** | 1. 验证成功执行脚本，检测出交易转发覆盖率 |
| **测试结果** |  |
| **测试结论** |  |

166 区块链验证平台链网节点SYN请求承受能力

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点SYN请求承受能力 |
| **测试目的** | 验证在验证平台上实现检测链网节点理论上承受TCP SYN请求数据包的最大数量； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 进入节点，运行python calculate\_NSHC.py命令 |
| **测试判据** | 1. 验证成功执行脚本，检测出链网节点SYN请求承受能力 |
| **测试结果** |  |
| **测试结论** |  |

167 区块链验证平台链网ICMP回复请求比

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网ICMP回复请求比 |
| **测试目的** | 验证在验证平台上实现检测链网节点一段时间内收到的ICMP回复与发起ICMP请求的数据包数量比。 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python calculate\_IRRR.py |
| **测试判据** | 1. 验证成功执行脚本，检测出指标内容 |
| **测试结果** |  |
| **测试结论** |  |

168 区块链验证平台链网剩余带宽承受能力

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网剩余带宽承受能力 |
| **测试目的** | 验证在验证平台上实现链网节点运行期间理论最大带宽容量与峰值带宽的差值； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python calculate\_rbc.py |
| **测试判据** | 1. 验证成功执行脚本，成功获得剩余带宽承受能力指标 |
| **测试结果** |  |
| **测试结论** |  |

169 区块链验证平台链网节点剩余连接能力

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点剩余连接能力 |
| **测试目的** | 验证在验证平台上实现链网节点设置的最大TCP监听数量减去当前处于已建立连接状态的TCP连接数量； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python calculate\_nrc.py命令 |
| **测试判据** | 验证成功执行脚本，获得节点剩余连接能力指标 |
| **测试结果** |  |
| **测试结论** |  |

170 区块链验证平台链网节点在线率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网节点在线率 |
| **测试目的** | 验证在验证平台上实现检测在线的节点在网络中总节点数量中的占比； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python calculate\_node\_online\_rate.py |
| **测试判据** | 验证成功执行脚本，获得节点在线率指标 |
| **测试结果** |  |
| **测试结论** |  |

171 区块链验证平台链网DDOS承受能力

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网DDOS承受能力 |
| **测试目的** | 验证在验证平台上实现检测链网节点被DDOS攻击期间可承受的最大数据包接收速率 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python calculate\_DRC.py |
| **测试判据** | 验证成功执行脚本，检测出DDOS承受能力 |
| **测试结果** |  |
| **测试结论** |  |

172 区块链验证平台链网

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网 |
| **测试目的** | 验证在验证平台上实现模拟攻击强度X的DDoS攻击，检测区块链网络中节点对外部交易的接收与转发能力 |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程; 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行python trc\_server.py命令，执行脚本 4. 在另外多个节点中运行python trc\_client.py命令，执行脚本 5. 在一个节点中运行curl -X POST [http://202.118.1.10:8091/start](http://192.168.0.2:8091/start) 来启动命令 6. 在普通节点上，执行geth attach geth.ipc命令，进入geth 7. 在geth中运行miner. start (1)开启挖矿 8. 解锁账号后，使用eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:100})命令不断发送交易 9. 等待脚本输出结果 |
| **测试判据** | 验证成功执行脚本，获得DDoS攻击交易转发抵抗能力 |
| **测试结果** |  |
| **测试结论** |  |

173 区块链验证平台链网DDOS攻击交易可识别性

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网DDOS攻击交易可识别性 |
| **测试目的** | 验证在验证平台上实现检测网络在识别和报告DDOS攻击事件的能力； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 确认受害节点，在攻击节点上运行命令./send\_sh\_tcp.sh，执行攻击脚本 4. 在受害节点上，运行python calculate\_AR.py |
| **测试判据** | 验证成功执行脚本，获得网络在识别和报告DDOS攻击事件的能力 |
| **测试结果** |  |
| **测试结论** |  |

174 区块链验证平台链网DDOS攻击交易溯源能力

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网DDOS攻击交易溯源能力 |
| **测试目的** | 验证在验证平台上实现检测衡量网络在攻击发生时追踪攻击源的能力； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在攻击节点上运行./send\_sh\_tcp.sh 4. 在受害节点上，执行geth attach geth.ipc命令，进入geth 5. 在geth中运行miner. start (1)开启挖矿 6. 解锁账号后，使用eth.sendTransaction({from:eth.accounts[0],to:eth.accounts[1],value:100})命令不断发送交易 7. 在受害节点上，运行python ts\_client\_integrated.py |
| **测试判据** | 验证成功执行脚本，得到DDOS攻击交易溯源能力 |
| **测试结果** |  |
| **测试结论** |  |

175 区块链验证平台链网DDOS溯源准确度

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网DDOS溯源准确度 |
| **测试目的** | 验证在验证平台上实现检测衡量网络在攻击发生时追踪攻击源的能力； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在攻击节点上运行./send\_sh\_tcp.sh 4. 在受害节点上运行python traceability\_accuracy\_ip.py |
| **测试判据** | 验证成功执行脚本，获得DDOS溯源准确度 |
| **测试结果** |  |
| **测试结论** |  |

176 区块链验证平台链网DDOS攻击流量传播范围

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网DDOS攻击流量传播范围 |
| **测试目的** | 验证在验证平台上DDoS攻击流量传播范围； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程； 2. 点击并进入设备管理，点击并通过SSH进入控制节点； 3. 在节点中运行./send\_sh\_tcp.sh 4. 在第二个节点中运行python spread\_ratio.py 5. 在第三个节点中运行python spr\_client.py |
| **测试判据** | 验证成功执行脚本，得到DDOS攻击流量传播范围 |
| **测试结果** |  |
| **测试结论** |  |

177 区块链验证平台链网DDOS交易广播转发抵抗能力

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网DDOS交易广播转发抵抗能力 |
| **测试目的** | 验证在验证平台上实现检测网络对恶意交易转发行为的识别和阻止能力； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 进入节点终端，运行python3 trigger\_ddos\_attack.py，运行触发脚本      1. 打开第二个节点终端，运行python3 ddos\_detector\_standard.py --loop --interval 5，运行检测脚本 |
| **测试判据** | 验证成功执行脚本，获得DDOS交易广播转发抵抗能力指标 |
| **测试结果** |  |
| **测试结论** |  |

178 区块链验证平台链网叉链交易重复性

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网叉链交易重复性 |
| **测试目的** | 验证在验证平台上实现链网分叉，检测在侧链与主链的暂时分叉区块中都存在交易的发起账户数量； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 运行下述命令使其分叉   sudo iptables -A INPUT -p tcp --dport 30303 -j DROP  sudo iptables -A INPUT -p udp --dport 30303 -j DROP  sudo iptables -A OUTPUT -p tcp --dport 30303 -j DROP  sudo iptables -A OUTPUT -p udp --dport 30303 -j DROP     1. 运行分叉触发脚本，python3 trigger\_fork\_scenarios.py \   --local-ipc /root/Work/PraviteChain/geth.ipc \  --remote-rpc http://192.168.0.22:8545 \  --private-key 3751e1372eb387ac84a06cc0b0c5c1b44f0a2879170325c18d05a053a20a3cda \  --to 0x12345D8DaF196e9eAe6a92EdDAa04b265cA9Be35 \  --value 0.01 \  --gas-limit 21000 \  --miner-threads 1 \  --tx-interval 3 \  --broadcast-local \  --p2p-port 30303 \  --wait-after-reconnect 15     1. 运行检测脚本，python3 simple\_fork\_detector.py --rpc-a http://127.0.0.1:8545 --rpc-b http://192.168.0.22:8545 --mode once |
| **测试判据** | 1. 验证成功执行脚本，得到叉链交易重复性 |
| **测试结果** |  |
| **测试结论** |  |

179 区块链验证平台链网恶意交易可识别性

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网恶意交易可识别性 |
| **测试目的** | 验证在验证平台上实现恶意交易可识别性； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 运行命令python3 trigger\_ddos\_attack.py，执行触发脚本      1. 打开节点新的一个终端，运行命令python3 ./run\_4\_2\_3\_17\_identifiability.py --total-attacks 1 --normal-events 0，执行检测脚本 |
| **测试判据** | 验证成功执行脚本，得到恶意交易可识别性指标 |
| **测试结果** |  |
| **测试结论** |  |

180 区块链验证平台链网恶意交易轨迹追踪能力

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网恶意交易轨迹追踪能力 |
| **测试目的** | 验证在验证平台上实现检测链网在恶意交易攻击发生时追踪攻击源的溯源速率和溯源准确度； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 运行部署脚本python deploy\_revert\_contract.py \   --rpc-url http://127.0.0.1:8545 \  --private-key 3751e1372eb387ac84a06cc0b0c5c1b44f0a2879170325c18d05a053a20a3cda     1. 运行触发脚本python trigger\_failing\_transactions.py \   --rpc http://127.0.0.1:8545 \  --rpc http://192.168.0.22:8545 \  --private-key 3751e1372eb387ac84a06cc0b0c5c1b44f0a2879170325c18d05a053a20a3cda \  --count 20 --label scenario-demo     1. 运行检测脚本python detect\_4\_2\_3\_18\_traceability.py \   --ground-truth data/ground\_truth\_18\_19.jsonl \  --attack-log data/rpc\_attack\_log.jsonl |
| **测试判据** | 1.验证成功执行脚本，得到链网恶意交易轨迹追踪能力 |
| **测试结果** |  |
| **测试结论** |  |

181 区块链验证平台链网恶意交易攻击流量传播范围

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网恶意交易攻击流量传播范围 |
| **测试目的** | 验证在验证平台上实现检测链网中恶意交易攻击流量在总流量中的占比； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 运行部署脚本python deploy\_revert\_contract.py \   --rpc-url http://127.0.0.1:8545 \  --private-key 3751e1372eb387ac84a06cc0b0c5c1b44f0a2879170325c18d05a053a20a3cda     1. 运行触发脚本python trigger\_failing\_transactions.py \   --rpc http://127.0.0.1:8545 \  --rpc http://192.168.0.22:8545 \  --private-key 3751e1372eb387ac84a06cc0b0c5c1b44f0a2879170325c18d05a053a20a3cda \  --count 20 --label scenario-demo     1. 运行检测脚本python detect\_4\_2\_3\_19\_traceability.py \   --ground-truth data/ground\_truth\_18\_19.jsonl \  --attack-log data/rpc\_attack\_log.jsonl |
| **测试判据** | 1. 验证成功执行脚本，成功获得链网恶意交易攻击流量传播范围 |
| **测试结果** |  |
| **测试结论** |  |

182 区块链验证平台链网抢跑交易识别率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网抢跑交易识别率 |
| **测试目的** | 验证在验证平台上实现链网在测量时间内真实发生的抢跑事件中，被系统识别为抢跑的比例； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 运行命令python3 ./trigger\_front\_run\_instant\_mining.py      1. 新开一个节点终端运行 python3 metric\_4\_2\_3\_29\_detection\_coverage.py --truth-file ./front\_run\_output/front\_run\_truth.json –loop |
| **测试判据** | 1. 验证成功执行脚本， |
| **测试结果** |  |
| **测试结论** |  |

183 区块链验证平台链网抢跑成功率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网抢跑成功率 |
| **测试目的** | 验证在验证平台上实现对识别为恶意/可疑目标发起的抢跑尝试中，实际成功插队并达成预期效果的比例； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 运行命令python3 trigger\_front\_run\_instant\_mining      1. 打开另一个终端运行python3 ./metric\_4\_2\_3\_21\_success\_rate.py --loop --interval 2 |
| **测试判据** | 1. 验证成功执行脚本，得到链网抢跑成功率 |
| **测试结果** |  |
| **测试结论** |  |

184 区块链验证平台链网热点合约抢跑率

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网热点合约抢跑率 |
| **测试目的** | 验证在验证平台上实现识别出的抢跑尝试中，目标合约属于热点名单的比例； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 运行python3 front\_run\_lightweight/trigger\_front\_run\_events.py --victim-key 0x1fa0d6956dbdc7b7f88832a85477cf4cc8b639032adbcbdb1b77533530ad1b85 --frontrunner-key 0x3751e1372eb387ac84a06cc0b0c5c1b44f0a2879170325c18d05a053a20a3cda，执行触发脚本      1. 随后执行python3 front\_run\_lightweight/collect\_events.py --rpc http://127.0.0.1:8545 --event-dir data，执行记录脚本      1. 随后运行python3 front\_run\_lightweight/metric\_4\_2\_3\_22\_hot\_contract\_rate.py --config config.json，执行检测脚本 |
| **测试判据** | 1. 验证成功执行脚本，检测出热点合约 |
| **测试结果** |  |
| **测试结论** |  |

185 区块链验证平台链网抢跑平均 gas 溢价

|  |  |
| --- | --- |
| **测试编号** |  |
| **测试项目** | 区块链验证平台链网抢跑平均 gas 溢价 |
| **测试目的** | 验证在验证平台上实现抢跑交易相对其目标受害交易所付出的平均 gas 单价溢价； |
| **测试环境** | 验证平台 |
| **必选/可选** | 必选 |
| **前置条件** | 验证平台构建完成以太坊网络 |
| **测试流程** | 1. 登录验证平台，进入区块链靶标自动化构建工程;      1. 点击并进入设备管理，点击并通过SSH进入控制节点；        1. 执行命令python3 metric\_4\_2\_3\_23\_avg\_gas\_premium.py --loop --interval 5，运行检测脚本      1. 打开节点的另一个终端，执行命令python3 ./trigger\_front\_run\_multi.py，运行触发脚本      1. 检查检测脚本的输出 |
| **测试判据** | 1. 验证成功执行脚本，得到抢跑平均 gas 溢价 |
| **测试结果** |  |
| **测试结论** |  |