**基于MININET的CPS仿真项目**

# 基本概况

为支撑配电网CPS仿真的重点研发项目，探讨基于MININET的CPS仿真平台的系统原理、实现方式，以及验证基于MININET的CPS仿真平台的运行情况，特制定本方案。

## 主要背景

配电系统转向以传统电力供应、分布式新能源发电、储能、信息通信设备和各类用电负荷为主的物理系统与信息系统深度耦合的配电网信息物理系统（CPS）。

与传统配电网相比，CPS具有如下显著特征：

1）多源异构、网络复杂

负荷的分散性和新能源发电的间歇性，不仅改变了传统功率单向流动的特点，而且分布式电源和储能的种类繁多，各种设备性能各异；随着电力系统的信息化建设，大量传感器、通信设备加入，导致系统的物理网络拓扑结构愈加复杂，规模愈加庞大，整个系统的动态特性具有明显的非线性、不确定、时变、异构性等特征。

2）交互频繁、易受攻击

随着智能电表、分布式电源和储能设备的大量接入，为了完成海量信息采集、传输、优化控制及需求侧响应等功能，电力流与信息流交互频繁，信息感知的深度、广度及密度大幅提高，使得信息安全的问题也愈发突出，而通信协议的开放化、标准化，也给调度自动化、继电保护和安全装置自动化、负荷控制、用户信息采集等领域带来了新的安全隐患。

3）分层调度、防护困难

单一的集中式、分散式控制方法，无法综合考虑投资成本、控制性能、可靠通信等指标，而分布式控制能较好地满足系统要求，但电网的分层式调度结构，使得系统严重依赖通信网络的可靠性和完备性，增加了优化控制难度和安全防御风险。

电网CPS是一个比较新的概念，在配用电信息的智能同步与采集、监控设备智能化与数字化等方面取得了一些初步成果。但是对于如何分析和理解电力系统与信息系统的交互机理，如何建立综合计及能量流、信息流和业务流的安全防御体系，如何充分利用前沿信息与通信技术增强配电系统的安全性、可靠性、自适应和自愈能力等问题，还没有形成系统的理论体系，现有的研究成果也相当初步。

# 项目整体方案

如下图所示，针对本课题的需求，设计了如下架构将RTLAB、实际设备（及物理仿真）、Mininet进行融合。



图 1. 部署示意图

如上图所示，Mininet可进行相关的配电网拓扑以及节点的仿真。Mininet与RTLAB之间通过网口连接，Mininet可为RTLAB提供配电网通信状态及配置信息，RTLAB可直接下发命令控制Mininet中的任何一个节点。Mininet可外接交换机，通过交换机与相关实际设备（主站、终端）等连接。主站的网口接入交换机，连接通信网络。主站的非网口（电口）接入物理仿真部分。

基于上述部署环境，虚实结合的实验环境可实现如下功能：

* 配电网通信网络的仿真，包含模拟仿真以及与实际设备结合的仿真
* 各种真实攻击的再现式仿真（可使用真实网络中的攻击发生器进行实际攻击，与真实网络几乎无差别）
  + DDOS攻击：可使用真实的DDOS攻击程序进行攻击。
  + 中间人攻击：可实施典型的中间人攻击。
  + 病毒攻击（模拟病毒传播）:模拟病毒去感染相关机器，这些机器发出一些异常包或者瘫痪等过程。也可以模拟伊朗“震网”病毒的传播模式，病毒在某些机器隐藏、及隐秘传播，只等到相关的关键词出现进行攻击。
  + 网络控制中心侵入的攻击：模拟网络中的控制节点被控制，从而修改网络中的路由策略影响整个网络的连通性。整个网络大多数节点无法按照要求发送到预期的目标。
* 网络动态防御功能的仿真
  + 针对DDOS攻击进行网络的动态配置，降低攻击的危害：在发现可能的攻击流之后，对网络边缘的节点进行控制，从而将相关可疑流导入到模拟的安全设备。
  + 基于网络隔离技术：通过配置虚拟网络，实现网络的分片，从而实现多分区隔离网络。在各个隔离网络的节点，无法访问到其他网络的信息，从而实现安全性。

# 系统设计

## 3.1. 系统功能结构

### 3.1.1. 系统功能结构框图

以下为网络仿真系统的模块结构图。如图所示，系统由三层架构组成，展示层、处理层、存储层组成。

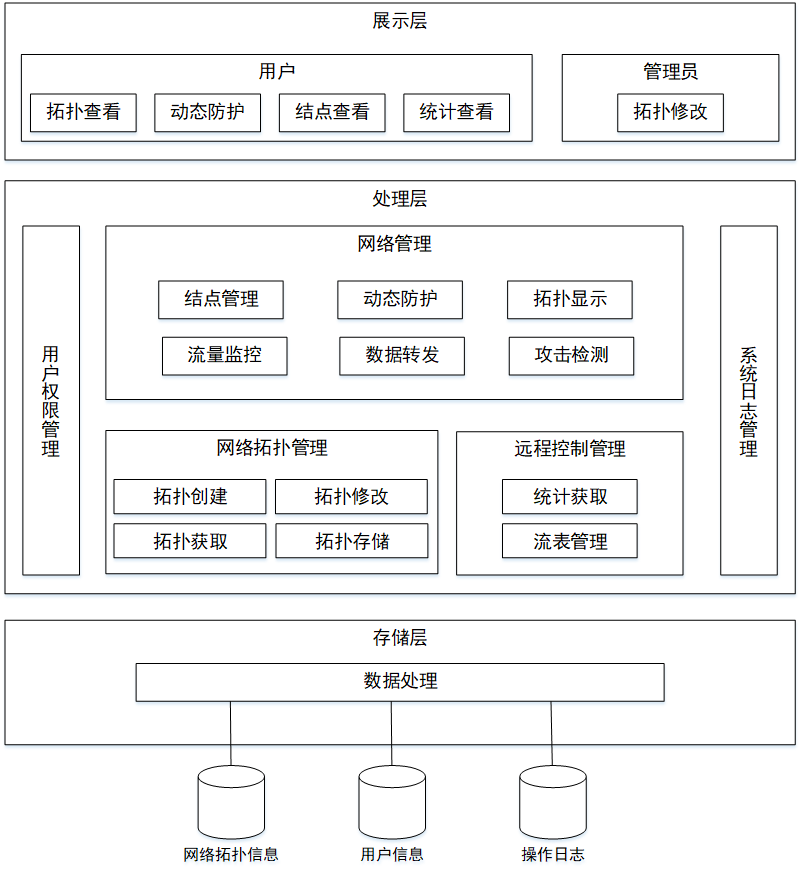


图 2.系统模块架构图

如图所示，最上层的展示层对应系统架构的系统客户端界面，由用户模块和管理员模块组成，主要实现用户对系统操作的界面；中间层的处理层对应系统架构的服务器的处理层，由网络管理模块、网络拓扑管理、远程控制管理模块以及用户权限管理模块和系统日志管理模块组成，主要实现系统对用户需要的任务所进行的具体操作；底层的存储层对应服务器的存储层，由数据处理模块和数据源组成，主要实现系统对数据的可持久化。

### 3.1.2. 系统功能组件定义

下面将对上图中的各个模块的功能和主要实现方法作简要说明。

**数据处理**模块主要实现对持久化数据的一系列操作，网络拓扑信息、用户信息和操作日志的查询，以及针对这些信息的修改。

**网络拓扑管理**模块主要实现仿真系统中网络拓扑的创建以及修改，此外还包含了对拓扑信息的获取和存储功能。网络拓扑管理模块由拓扑创建、拓扑获取、拓扑修改、拓扑存储子模块构成。

**远程控制管理**模块主要实现仿真系统中网络通信中的一系列控制操作和统计信息的获取，具体可获取通信过程中实时流量信息和累计流量信息，以及针对网络中有限制的通信，可通过控制流表的操作实现。远程控制管理模块由统计获取和流表管理子模块构成。

**网络管理**模块主要实现仿真系统中对网络一系列操作，对网络结点信息的管理，网络拓扑的显示，以及针对网络攻击所实施的动态防护功能；此外，还有流量监控功能，控制网络结点数据转发的功能和结点攻击检测的功能。网络管理模块由结点管理、动态防护、拓扑显示、流量监控、数据转发、攻击检测子模块构成。

**用户权限管理**模块主要实现对系统用户信息的操作，该模块仅限管理员操作，可进行用户的增加、删除，修改用户的密码及权限，限制用户是否具有查看拓扑、修改拓扑、查看日志等功能。

**系统日志管理**模块主要实现记录用户的各种操作及发生时间，例如登入、登出、修改拓扑、查询流量信息等。

**拓扑查看**模块主要实现用户查看仿真系统的网络拓扑信息，网络中各类结点个数，结点之间的连接关系通过图的形式展示。

**动态防护**模块主要实现网络拓扑中攻击的识别以及动态防护，并且实时展示防护过程，用户可以选择开启或者关闭入侵检测和网络动态防护以了解动态防护对于网络拓扑系统的保护效果，可以完成对网络拓扑中网络攻击的识别也就是系统入侵检测以及针对于入侵选择合适的防护策略以完成动态防护，并且实时展示防护过程中的信息流或者网络分区变化。

**结点查看**模块主要实现用户查看网络结点详细信息，包括host的类型和ip信息以及资源使用情况，还可查看switch的结点详细信息。

**统计查看**模块主要实现对网络中流量信息的统计功能，实时显示当前各个线路中流量大小，和每条线路中各个时段总流量大小。

**拓扑修改**模块主要实现管理员对网络拓扑中结点的增删和链路的增删操作。

## 3.2. 系统功能实现动态描述

### 3.2.1. 系统与外部交互

从系统整体的层面来看，下图动态地显示了系统与外部对象之间的交互关系：

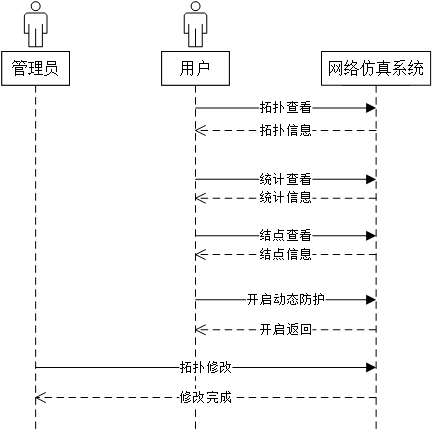


图 3.系统与外部交互

上图隐藏了系统的内部模块划分，重点在于显示出本系统与外部对象的动态消息交互关系，展现了各个消息之间的时序关系。其中，消息交互的序列简述如下：

1. 用户调用系统查看拓扑，系统返回网络拓扑信息，并将拓扑信息依据规定方式以图的形式展示出，并可继续查看拓扑中结点详细信息。
2. 用户调用系统统计查看，系统返回网络中当前时段各个区域流量累计总和，通过折线图形式展示给用户，以及实时返回每条链路的流量信息，在图中以颜色区分。
3. 用户调用系统结点查看，系统依据用户输入的参数，对结点信息进行查询并返回，还可查询结点当前cpu等资源使用情况。
4. 用户调用系统开启动态防护，系统即开启动态防护功能，对网络中的网络攻击进行检测并采取防护措施，返回提示用户功能已开启。
5. 管理员调用系统拓扑修改功能，对网络拓扑的结点和链路进行增删操作，系统修改完成并返回提示用户修改完成。

### 3.2.2. 系统模块间交互

系统模块由网络管理模块、网络拓扑管理模块、远程控制管理模块和数据处理模块组成，网络管理模块由结点管理、动态防护、拓扑显示、流量监控、数据转发、攻击检测子模块构成；网络拓扑管理模块由拓扑创建、拓扑获取、拓扑修改、拓扑存储子模块构成；远程控制管理模块由统计获取和流表管理子模块构成。下面描述这几个模块在各种操作下它们之间的交互：

1. 拓扑创建模块交互

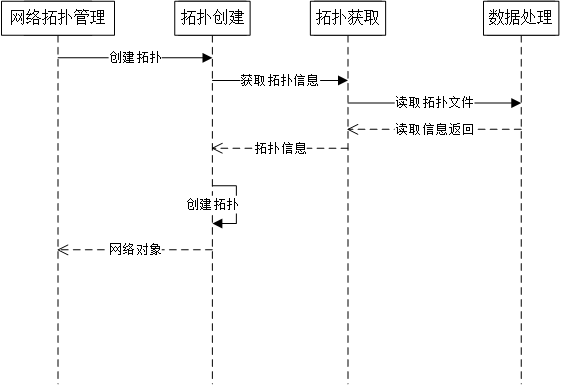


图 4. 拓扑创建模块交互

上图描述了网络拓扑创建时，各系统模块间动态交互的时序；交互时序为：

1. 网络拓扑管理模块调用拓扑创建模块的创建拓扑接口；
2. 拓扑创建模块调用拓扑获取模块的获取拓扑信息；
3. 拓扑获取模块调用数据处理模块读取，读取拓扑文件并返回信息；
4. 拓扑获取模块将返回的拓扑信息返回给拓扑创建模块；
5. 拓扑创建模块获得拓扑信息，调用自身创建拓扑方法；并将创建后的网络对象返回给上级网络管理模块。
6. 拓扑显示模块交互

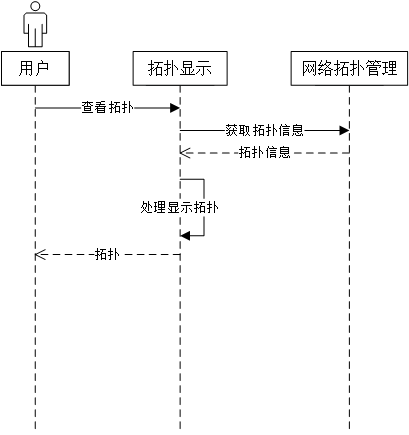


图 5. 拓扑显示模块交互

上图描述了用户调用展示层的拓扑显示模块查看拓扑时，各系统模块间动态交互的时序；交互时序为：

1. 用户调用拓扑显示模块的查看拓扑方法；
2. 拓扑显示模块调用网络管理模块的获取拓扑信息方法，网络管理模块返回拓扑信息；
3. 拓扑显示模块获得拓扑信息，根据拓扑信息处理显示并返回；
4. 用户可在界面上查看到网络拓扑。
5. 结点信息查看模块交互



图 6. 结点信息查看

上图描述了用户利用展示层的结点查看模块查看网络结点信息时，各系统模块间动态交互的时序；交互时序为：

1. 用户调用结点管理的查看结点信息；
2. 结点管理模块调用网络拓扑管理的查看结点信息，网络拓扑管理模块将结点信息返回给上一级；
3. 结点管理模块将结点信息显示返回给用户；
4. 用户调用结点管理模块的查看结点资源信息；
5. 结点管理模块调用网络拓扑管理模块的查看结点资源信息方法；
6. 网络拓扑管理模块调用自身的查看结点资源信息方法，并将返回结果的资源信息返回上级；
7. 结点管理模块获得结点资源信息显示返回给用户。
8. 统计信息查看模块交互

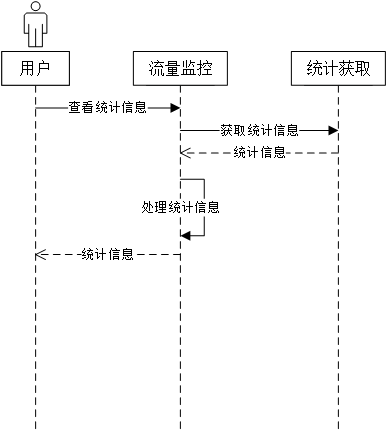


图 7. 统计信息查看模块交互

上图描述了用户利用展示层的统计查看模块查看网络统计信息时，各系统模块间动态交互的时序；交互时序为：

1. 用户调用流量监控模块的查看统计信息；
2. 流量监控模块调用统计获取模块的获取统计信息；
3. 统计获取模块执行方法后将统计信息返回上级；
4. 流量监控模块获取统计信息后，执行处理统计信息的方法，显示并返回给用户查看。
5. 动态防护模块交互

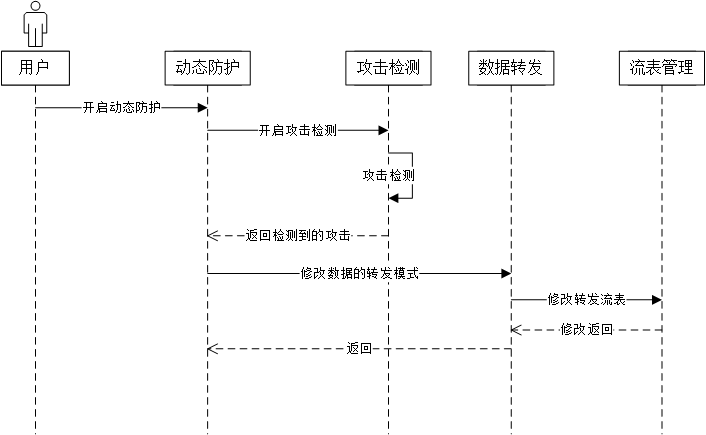


图 8. 动态防护模块交互

上图描述了用户利用展示层的动态防护模块开启动态防护时，各系统模块间动态交互的时序；交互时序为：

1. 用户调用动态防护模块的开启动态防护；
2. 动态防护模块调用攻击检测模块的开启攻击检测方法；
3. 攻击检测模块开启攻击检测的方法，在检测到攻击后返回给动态防护模块；
4. 动态防护模块调用数据转发模块的修改数据的转发模式；
5. 数据转发模块调用流表管理模块的修改转发流表，流表管理模块修改完成后返回；
6. 数据转发模块修改转发模式后返回上级动态防护模块。
7. 拓扑修改模块交互



图 9. 拓扑修改交互

上图描述了管理员利用展示层的拓扑修改模块对网络拓扑进行修改时，各系统模块间动态交互的时序；交互时序为：

1. 管理员调用网络拓扑管理模块的修改拓扑；
2. 网络管理模块调用拓扑修改模块的修改拓扑；
3. 拓扑修改模块修改完毕拓扑后调用拓扑存储模块的处理并存储拓扑方法；
4. 拓扑存储模块对拓扑信息依据规定方式处理拓扑信息，调用数据处理模块的存储拓扑文件；
5. 数据处理模块存储完成后返回上级，各级一次返回。

# 通信方案设计

## 4.1. 通信需求与设计思路

项目的通信需求包括：Mininet可为RTLAB提供配电网通信状态及配置信息，RTLAB可直接下发命令控制Mininet中的任何一个节点，Mininet可外接交换机，通过交换机与相关实际设备（主站、终端）等连接，主站的网口接入交换机，连接通信网络。为了满足上述通信需求，我们采取以下通信方案的设计思路：Mininet中的节点和外部真实网络通信，可利用网桥，将Mininet环境所在主机网口与Mininet中的一个交换机端口绑定，从而实现Mininet内任何节点和RTLAB以及主站进行通信。

## 4.2. 部署通信方案

Mininet内部网络需要和外部网络进行通信，所以可以将Mininet所在主机网口和Mininet内部虚拟交换机绑定，如此，可以借助通向外部网络的交换机实现内部结点与外部网络通信。通信方案的部署如下图所示，Mininet所在环境将网口与Mininet交换机switch的端口绑定。Mininet中的一个节点利用eth1通过外部交换机和RTLAB及主站通信。

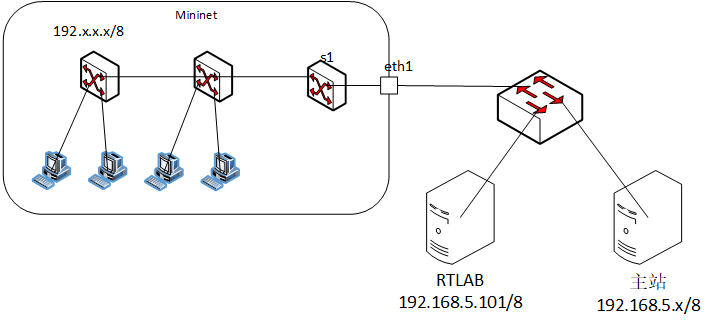


图 10. 网络通信部署图

上述的通信方案中，Mininet网络可以采用不同的子网段，由于Mininet是采用主机的网口和外部网络进行通信，所在主机的IP地址属于同一子网，本方案实验中采用的是192.0.0.0/8该子网，可以支持2^24-2台主机。

## 4.3. 通信模型状态机

通信过程中结点主要有三个状态：未连接状态、连接状态、断开连接状态。通信过程中结点的状态迁移变化如下图所示：

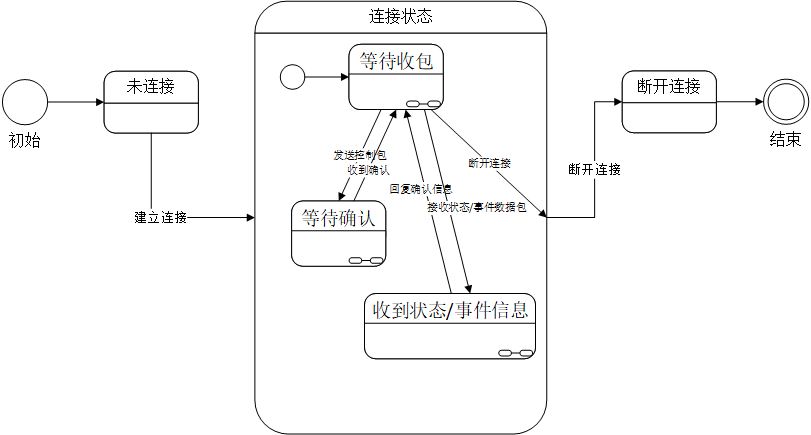


图 11. Mininet内部结点状态迁移图

上图描述了通信过程中Mininet内部结点的状态迁移变化，初始为未连接状态，经过建立连接的操作变为连接状态，在连接状态经过断开连接的操作变为断开连接的状态即结束状态。在连接状态中，初始子状态为等待收包状态，此状态若发送控制包变为等待确认状态，若接收到状态或时间数据包则变为收到状态/事件信息状态，在等待确认状态时只有收到确认信息方可变为等待收包状态，在收到状态/事件信息状态只有回复确认消息方可变为等待收包状态；在等待收包状态若断开连接则变为断开连接状态，结束通信。

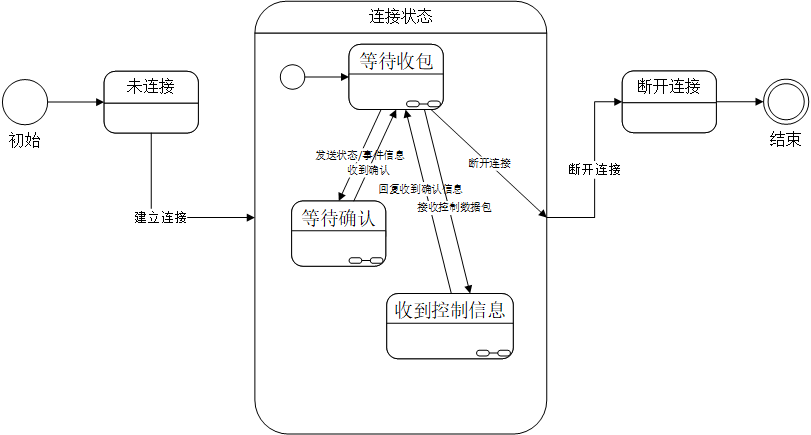


图 12.RTLAB状态迁移图

上图描述了通信过程RTLAB的状态迁移变化，初始为未连接状态，经过建立连接的操作变为连接状态，在连接状态经过断开连接的操作变为断开连接的状态即结束状态。在连接状态中，初始子状态为等待收包状态，此状态若发送状态/事件包变为等待确认状态，若接收到控制数据包则变为收到控制信息状态，在等待确认状态时只有收到确认信息方可变为等待收包状态，在收到控制信息状态只有回复确认消息方可变为等待收包状态；在等待收包状态若断开连接则变为断开连接状态，结束通信。

## 4.4. 通信同步机理

### 4.4.1. 状态同步



图 13.Mininet与RTLAB的状态同步图

Mininet与RTLAB的同步机理体现在它们之间的状态同步上，上图展示了Mininet与RTLAB之间的状态同步，本节通过该状态同步图中两者的状态对应关系来阐述Mininet与RTLAB的同步机理。两者连接的前后，状态相互对应，即两者的未连接状态和断开连接状态相互对应。在连接状态中，Mininet与RTLAB的等待收包状态相互对应，该状态下即可由RTLAB向Mininet主动发送消息，也可由Mininet向RTLAB主动发送消息。当Mininet向RTLAB发送控制数据包后，由等待收包状态迁移到等待确认状态，与之同步的，RTLAB由等待收包状态迁移到收到控制信息状态，在状态同步图中体现为Mininet的等待确认状态对应RTLAB收到控制信息状态。同理，RTLAB当向Mininet发送事件信息后，由等待收包状态迁移到等待确认状态，与之同步的，Mininet由等待收包状态迁移到收到事件信息状态，在状态同步图中体现为RTLAB的等待确认状态对应Mininet收到控制信息状态。通过这种收发数据的状态同步与回复确认数据包的方式来实现Mininet与RTLAB的通信过程中的同步机理。

### 4.4.2. 时间戳同步

Mininet与RTLAB的同步机理还体现在它们之间通信过程中所附带的时间戳的同步，利用两者的时间戳的同步来有效减少通信过程的产生的时延导致Mininet和RTLAB之间不同步的现象。



图 14.时钟同步请求应答机制实现过程

通信的双方为了保持时钟的同步需要区分双方的时钟节点的主从关系，对于一对需要相互同步的一对时钟节点来说，存在如下主从关系：发布同步时间的节点称为主节点，而接收同步时间的节点则称为从节点；主节点上的时钟称为主时钟，而从节点上的时钟则称为从时钟。在本系统中，Mininet作为主节点，其时钟作为主时钟，RTLAB作为从节点，其时钟作为从时钟。上图示意了时钟同步请求应答机制的实现过程：

(1) 主时钟向从时钟发送Sync报文，并记录发送时间t1；从时钟收到该报文后，记录接收时间t2。

(2) 主时钟发送Sync报文之后，紧接着发送一个携带有t1的Follow\_Up报文。

(3) 从时钟向主时钟发送Delay\_Req报文，用于发起反向传输延时的计算，并记录发送时间t3；主时钟收到该报文后，记录接收时间t4。

(4) 主时钟收到Delay\_Req报文之后，回复一个携带有t4的Delay\_Resp报文。

此时，从时钟便拥有了t1～t4这四个时间戳，由此可计算出主、从时钟间的往返总延时为[(t2–t1) + (t4–t3)]，由于网络是对称的，所以主、从时钟间的单向延时为[(t2–t1) + (t4–t3)] / 2。因此，从时钟相对于主时钟的时钟偏差为：

经过上述过程，可以计算出RTLAB相对于Mininet的时钟偏差Offset，想要计算Mininet与RTLAB的通信的时延还需计算Mininet的内部节点的平均通信时延。经过测试得到如下的时延测量表，其中src为测试的源点IP，dst为测试的终点IP，min、max、avg分别为最小往返时延、最大往返时延和平均往返时延，单位为毫秒。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| src (ip) | dst (ip) | min(ms) | max(ms) | avg(ms) |
| 10.0.1.1(out) | 10.0.25.104 | 1 | 114 | 28 |
| 10.0.1.1(out) | 10.0.25.16 | 1 | 145 | 35 |
| 10.0.1.1(out) | 10.0.11.102 | 1 | 123 | 31 |
| 10.0.25.104 | 10.0.1.1(out) | 0.380 | 208 | 52.703 |
| 10.0.25.16 | 10.0.1.1(out) | 0.335 | 131 | 26.732 |
| 10.0.11.102 | 10.0.1.1(out) | 0.325 | 139 | 20.289 |
| 10.0.11.102 | 10.0.25.16 | 0.055 | 131.846 | 33.274 |
| 10.0.11.102 | 10.0.12.104 | 0.088 | 250.328 | 63.036 |
| 10.0.25.16 | 10.0.12.104 | 0.089 | 291.992 | 73.550 |

表 1.时延测试统计表

可以看到，Mininet内部host和外部host通信平均时延为32.28733 ms；Mininet内部host之间通信平均时延 56.62 ms。这里我们重点考虑Mininet内部host之间的通信往返时延56.62，将该值除以2即为Mininet向RTLAB通信过程中，消耗在Mininet内部通信的时延部分。

将RTLAB与Mininet的主从时钟偏差加上Mininet内部host的时延56.62 / 2（ms）即为两者的通信时延。Mininet与RTLAB通信过程中，将该通信时延附带发送给RTLAB，以此调整时间戳从而实现Mininet与RTLAB之间的时间戳同步。

## 4.5. 通信协议设计

### 4.5.1. 状态数据包

状态数据包负责传输各电力设备当前各类属性的状态信息，包括电压、电流、功率、相别和相角等电力设备的重要状态指标。状态数据包分两类，一类是RTLAB发送给Mininet的，另一类是Mininet向RTLAB的回复。

|  |  |
| --- | --- |
| 单位（bytes） | 说明 |
| 8 | 报头（保留，后续使用） |
| 2 | 包头(，0xAA，0x55) |
| 2 | 状态字 |
| 8 | 状态包序号（浮点数） |
| 8 | 设备种类(浮点数) |
| 8 | 设备编号(浮点数) |
| 8 | 电压 (伏，浮点数) |
| 8 | 电流 (安，浮点数) |
| 8 | 有功功率(千瓦，浮点数) |
| 8 | 无功功率(千瓦，浮点数) |
| 8 | 相角（浮点数） |
| 8 | 相别（浮点数） |
| 8 | 工作状态（浮点数） |
| 8 | 闭合状态（浮点数） |

表 2. RTLAB向Mininet发送状态数据包的格式

状态字的定义如下：

1. 状态数据包表示字段 1为状态包 0无效
2. 状态数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
3. 事件数据包表示字段 1 为事件包 0无效
4. 事件数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
5. 控制数据包表示字段 1 为控制包 0 无效
6. 控制数据包发送方向字段 0 RTLAB发出 1 RTLAB接收

|  |  |
| --- | --- |
| 单位（bytes） | 说明 |
| 8 | 报头（保留，后续使用） |
| 2 | 包头(0xAA，0x55) |
| 2 | 状态字 |
| 8 | 状态包序号（浮点数） |

表 3.Mininet向RTLAB回复状态数据包的格式

状态字的定义如下：

1. 状态数据包表示字段 1为状态包 0无效
2. 状态数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
3. 事件数据包表示字段 1 为事件包 0无效
4. 事件数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
5. 控制数据包表示字段 1 为控制包 0 无效
6. 控制数据包发送方向字段 0 RTLAB发出 1 RTLAB接收

### 4.5.2. 事件数据包

事件数据包负责传输各电力设备当前发生的各类事件信息。事件数据包分两类，一类是RTLAB发送给Mininet的，另一类是Mininet向RTLAB的回复。

|  |  |
| --- | --- |
| 单位（bytes） | 说明 |
| 8 | 报头（保留，后续使用） |
| 2 | 包头(0xAA，0x55) |
| 2 | 状态字 |
| 8 | 事件包序号（浮点数） |
| 8 | 设备种类（浮点数） |
| 8 | 设备编号（浮点数） |
| 8 | 事件种类（浮点数） |
| 8 | 事件内容（浮点数） |

表 4. RTLAB向Mininet发送事件数据包的格式

状态字定义如下：

1. 状态数据包表示字段 1为状态包 0无效
2. 状态数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
3. 事件数据包表示字段 1 为事件包 0无效
4. 事件数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
5. 控制数据包表示字段 1 为控制包 0 无效
6. 控制数据包发送方向字段 0 RTLAB发出 1 RTLAB接收

|  |  |
| --- | --- |
| 单位（bytes） | 说明 |
| 8 | 报头（保留，后续使用） |
| 2 | 包头(0xAA，0x55) |
| 2 | 状态字 |
| 8 | 事件包序号（浮点数） |

表 5. Mininet向RTLAB回复事件数据包的格式

状态字定义如下：

1. 状态数据包表示字段 1为状态包 0无效
2. 状态数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
3. 事件数据包表示字段 1 为事件包 0无效
4. 事件数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
5. 控制数据包表示字段 1 为控制包 0 无效
6. 控制数据包发送方向字段 0 RTLAB发出 1 RTLAB接收

### 4.5.3. 控制数据包

控制数据包负责传输对各电力设备的控制命令信息。控制数据包分两类，一类是RTLAB发送给Mininet的，另一类是Mininet向RTLAB的回复。

|  |  |
| --- | --- |
| 单位（bytes） | 说明 |
| 8 | 报头（保留，后续使用） |
| 2 | 包头(0xAA，0x55) |
| 2 | 状态字 |
| 8 | 控制包序号（浮点数） |
| 8 | 设备种类（浮点数） |
| 8 | 设备编号（浮点数） |
| 8 | 控制类型（浮点数) |
| 8 | 控制命令（浮点数) |

表 6. RTLAB向Mininet发送控制数据包的格式

状态字定义如下：

1. 状态数据包表示字段 1为状态包 0无效
2. 状态数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
3. 事件数据包表示字段 1 为事件包 0无效
4. 事件数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
5. 控制数据包表示字段 1 为控制包 0 无效
6. 控制数据包发送方向字段 0 RTLAB发出 1 RTLAB接收

|  |  |
| --- | --- |
| 单位（bytes） | 说明 |
| 8 | 报头（保留，后续使用） |
| 2 | 包头(0xAA，0x55) |
| 2 | 状态字 |
| 8 | 控制包序号（浮点数） |

表 7. Mininet向RTLAB回复控制数据包的格式

状态字定义如下：

1. 状态数据包表示字段 1为状态包 0无效
2. 状态数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
3. 事件数据包表示字段 1 为事件包 0无效
4. 事件数据包发送方向字段 0 RTLAB发出 1 RTLAB接收
5. 控制数据包表示字段 1 为控制包 0 无效
6. 控制数据包发送方向字段 0 RTLAB发出 1 RTLAB接收

## 4.6. 通信网建模

### 4.6.1. 通信元件模型

1. 交换机模型

通信网中的交换机模型采用Mininet中的OVSSwitch模型，如下图所示，OVSSwitch交换机的两个网口eth0和eth1分别连接两台主机host1和host2。虽然OVS是个虚拟交换机，但是它拥有物理交换机的功能，一般在LINUX内核中进行设计。在物理交换机里面，数据流从网卡出发的数据包，经过层层解析，最后到达用户层。而在虚拟交换机里数据流从网卡出发的数据包需要首先建立一个网桥，然后将网桥与网卡绑定起来。所以数据包从eth0出来后，通过vPort进入OpenvSwitch中，然后根据键值进行流表匹配，如果匹配成功，则根据流表所对应的action来进行处理。

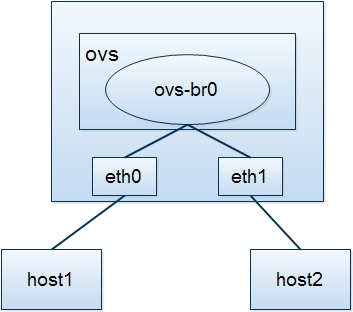


图 15. OVSSwitch交换机模型

1. 路由器模型

通信网中的路由器在Mininet中使用Switch模型代替，配合ONOS控制器下发流表来实现路由器的路由转发功能，在系统中的路由器模型主要由以下五部分组成：进程、内核、报文缓冲、设备驱动程序、交换控制软件。

进程：由执行特定任务的独立线程和相关的数据组成，如系统配置维护的telnet守护进程、客户端进程，FTP进程、TFTP进程，SNMP进程，各种协议进程：IP、TCP、UDP、RIP、OSPF、BGP、ARP、ICMP、IGMP，其它有加解密进程、报文过滤进程、NAT进程等。

内核：为系统的其它部分提供基本的系统服务，如存储器管理、进程调度、定时器和时钟管理。它为进程提供了硬件（CPU和存储器）资源的管理。

报文缓冲：用来存放将要被交换的报文。

设备驱动程序：控制网络接口硬件设备及其它外围设备（如Flash）。设备驱动程序接口位于进程、内核、硬件之间，同时与交换控制软件有接口。

交换控制软件：根据转发方式控制报文的交换，在高端线速路由器中该部分功能由硬件实现。

1. 终端模型

通信网中的终端模型采用Mininet中的Host模型，它在Mininet中类的继承关系如下图所示，Host模型在Mininet网络中充当虚拟主机，它作为通信网的终端节点，可以像真实的电脑一样工作，可以使用ssh登录，启动应用程序，程序可以向以太网端口发送数据包，数据包会被交换机、路由器接收并处理。

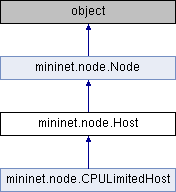


图 16.Mininet中Host类的继承关系

1. 链路模型

通信网中的链路模型采用Mininet中的Link模型，它在Mininet中的继承关系如下图所示，Link模型作为通信网的链路可以仿造真实世界中的绝大部分链路模型，在Mininet创建拓扑中添加链路时可以设置链路的属性值。

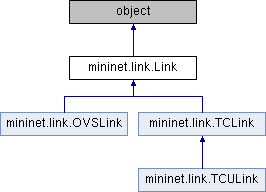


图 17.Mininet中的Link类继承关系

如下图所示，Mininet中的链路模型存在以下属性值，包括带宽bw，延迟delay，丢失率loss，最大队列大小max\_queue\_size等，通过设置上述链路属性值来实现不同的链路特性，从而在通信网中创建诸如光纤链路模型、无线链路模型、SDH链路模型等。



图 18.Mininet中Link模型的属性

### 4.6.2. 拓扑结构模型

1. 星型拓扑

星型拓扑是由中央节点和通过点到点通信链路接到中央节点的各个站点组成。中央节点执行集中式通信控制策略,因此中央节点相当复杂,而各个站点的通信处理负担都很小。这种结构一旦建立了通道连接,就可以无延迟地在连通的两个站点之间传送数据。

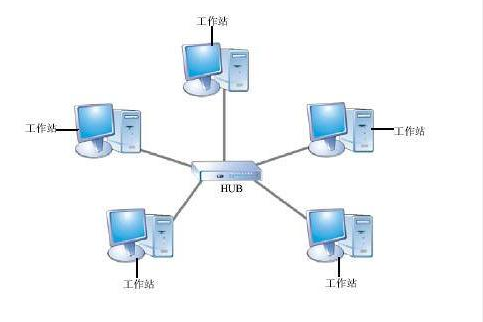


图 19.星型拓扑结构

星型拓扑结构的优点：

1. 结构简单，连接方便，管理和维护都相对容易，而且扩展性强。
2. 网络延迟时间较小，传输误差低。
3. 在同一网段内支持多种传输介质，除非中央节点故障，否则网络不会轻易瘫痪。
4. 每个节点直接连到中央节点，故障容易检测和隔离，可以很方便地排除有故障的节点。

星型拓扑结构的缺点：

1. 安装和维护的费用较高
2. 共享资源的能力较差
3. 一条通信线路只被该线路上的中央节点和边缘节点使用，通信线路利用率不高
4. 对中央节点要求相当高，一旦中央节点出现故障，则整个网络将瘫痪。
5. 总线拓扑

总线拓扑结构采用一个信道作为传输媒体,所有站点都通过相应的硬件接口直接连到这一公共传输媒体上,该公共传输媒体即称为总线。任何一个站发送的信号都沿着传输媒体传播,而且能被所有其它站所接收。因为所有站点共享一条公用的传输信道,所以一次只能由一个设备传输信号。通常采用分布式控制策略来确定哪个站点可以发送时,发送站将报文分成分组,然后逐个依次发送这些分组,有时还要与其它站来的分组交替地在媒体上传输。当分组经过各站时,其中的目的站会识别到分组所携带的目的地址,然后复制下这些分组的内容。

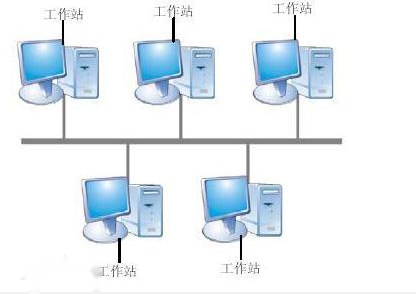


图 20.总线拓扑结构

总线拓扑结构的优点：

1. 总线结构所需要的电缆数量少，线缆长度短，易于布线和维护。
2. 总线结构简单,又是元源工作,有较高的可靠性。传输速率高，可达1~100Mbps。
3. 易于扩充,增加或减少用户比较方便，结构简单，组网容易，网络扩展方便
4. 多个节点共用一条传输信道，信道利用率高。

总线拓扑的缺点：

1. 总线的传输距离有限,通信范围受到限制。
2. 故障诊断和隔离较困难。
3. 分布式协议不能保证信息的及时传送，不具有实时功能。站点必须是智能的，要有媒体访问控制功能,从而增加了站点的硬件和软件开销。
4. 环型拓扑

在环型拓扑中各节点通过环路接口连在一条首尾相连的闭合环型通信线路中，环路上任何节点均可以请求发送信息。请求一旦被批准，便可以向环路发送信息。环型网中的数据可以是单向也可是双向传输。由于环线公用，一个节点发出的信息必须穿越环中所有的环路接口，信息流中目的地址与环上某节点地址相符时，信息被该节点的环路接口所接收，而后信息继续流向下一环路接口，一直流回到发送该信息的环路接口节点为止。

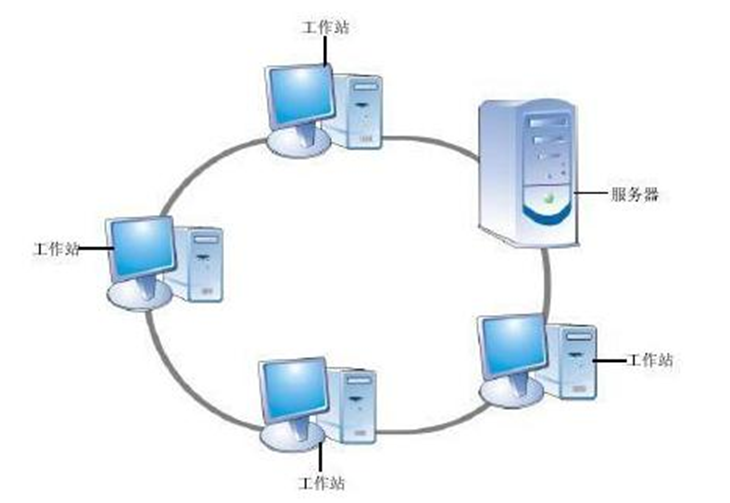


图 21.环型拓扑结构

环型拓扑的优点：

1. 电缆长度短。环型拓扑网络所需的电缆长度和总线拓扑网络相似，但比星形拓扑网络要短得多。
2. 增加或减少工作站时，仅需简单的连接操作。
3. 可使用光纤。光纤的传输速率很高，十分适合于环型拓扑的单方向传输。

环型拓扑的缺点：

1. 节点的故障会引起全网故障。这是因为环上的数据传输要通过接在环上的每一个节点,一旦环中某一节点发生故障就会引起全网的故障。
2. 故障检测困难。这与总线拓扑相似,因为不是集中控制,故障检测需在网上各个节点进行,因此就不很容易。
3. 环型拓扑结构的媒体访问控制协议都采用令牌传递的方式,在负载很轻时,信道利用率相对来说就比较低。
4. 树型拓扑

树型拓扑可以认为是多级星型结构组成的，只不过这种多级星型结构自上而下呈三角形分布的，就像一颗树一样，最顶端的枝叶少些，中间的多些，而最下面的枝叶最多。树的最下端相当于网络中的边缘层，树的中间部分相当于网络中的汇聚层，而树的顶端则相当于网络中的核心层。它采用分级的集中控制方式，其传输介质可有多条分支，但不形成闭合回路，每条通信线路都必须支持双向传输。

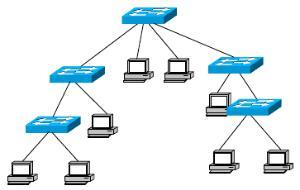


图 22.树型拓扑结构

树型拓扑的优点：

1. 易于扩展。这种结构可以延伸出很多分支和子分支,这些新节点和新分支都能容易地加入网内。
2. 故障隔离较容易。如果某一分支的节点或线路发生故障,很容易将故障分支与整个系统隔离开来。

树型拓扑的缺点：

1. 各个节点对根的依赖性太大，如果根发生故障，则全网不能正常工作。从这一点来看，树型拓扑结构的可靠性有点类似于星型拓扑结构。
2. 混合型拓扑

混合型拓扑是将两种单一拓扑结构混合起来，取两者的优点构成的拓扑。

一种是星型拓扑和环型拓扑混合成的"星-环"拓扑，另一种是星型拓扑和总线拓扑混合成的"星-总"拓扑。这两种混合型结构有相似之处，如果将总线拓扑的两个端点连在一起也就变成了环型拓扑。在混合型拓扑结构中，汇聚层设备组成环型或总线型拓扑，汇聚层设备和接入层设备组成星型拓扑。

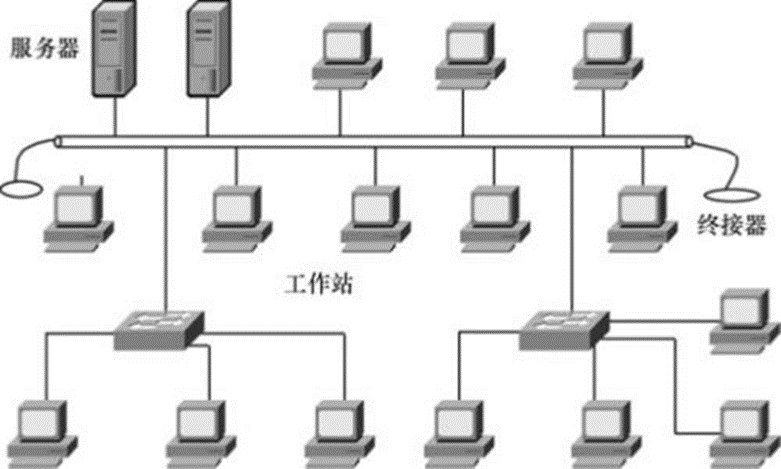


图 23.混合型拓扑结构

混合型拓扑的优点：

1. 故障诊断和隔离较为方便。一旦网络发生故障，只要诊断出哪个网络设备有故障，将该网络设备和全网隔离即可。
2. 易于扩展。要扩展用户时，可以加入新的网络设备，也可在设计时，在每个网络设备中留出一些备用的可插入新站点的连接口。
3. 安装方便。网络的主链路只要连通汇聚层设备，然后再通过分支链路连通汇聚层设备和接入层设备。

混合型拓扑的缺点：

1. 需要选用智能网络设备，实现网络故障自动诊断和故障节点的隔离，网络建设成本比较高。
2. 像星型拓扑结构一样，汇聚层设备到接入层设备的线缆安装长度会增加较多。

### 4.6.3. 物理节点与通信节点映射关系

在Mininet中通信节点与真实世界中的物理节点是一一映射的关系，映射的过程如下图所示，在系统的创建拓扑的过程中，数据处理模块读取拓扑文件，后由拓扑获取模块对拓扑数据进行解析并处理生成通信网中的通信节点所需的数据结构，最后由拓扑结构根据生成的通信节点数据结构创建出相应的Mininet拓扑。

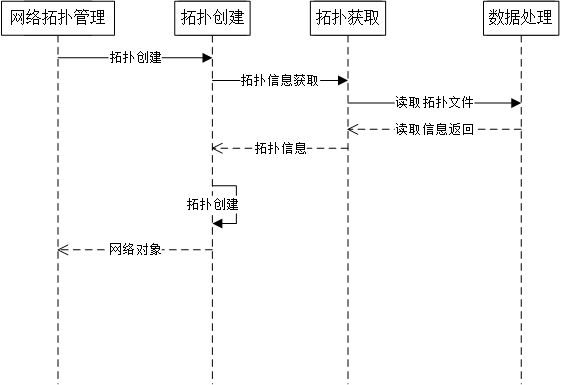


图 24.创建拓扑的序列图

数据处理模块读取的拓扑文件中拓扑数据如下表所示，拓扑文件中每条数据代表一个物理节点，每条数据包含馈线名称，设备名称，终端种类，编号和地理位置。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 10kV馈线名称 | 设备名称 | 终端种类 | 编号 | 地理位置 |
| 赤岗II路627 | 赤岗II路627开关 | S | 49627 | 119.173046,25.474525 |

表格 1.物理节点的数据格式

当拓扑获取模块读取到上述物理节点的拓扑数据后将该数据转换成如下表的通信节点的JSON数据格式，内容包含ip地址，mac地址，节点类型type，节点名称name，节点坐标location。可以发现通信节点的数据结构中name字段即为物理节点的终端类型以及编号的组合，以此实现了物理节点与通信节点的一一映射。

表格 2.通信节点的数据格式

{

"ip": "10.0.11.250",

"mac": "10:00:00:00:00:14",

"type": "host",

"name": "s49627",

"location": [

119.173046,

25.474525

]

}

# 动态防护策略

## 5.1. 病毒攻击防护流程

### 5.1.1. 病毒攻击检测防护简介

（1）**安全环境下的网络拓扑：**

（PC为绿色代表处于可信环境中，未受攻击）

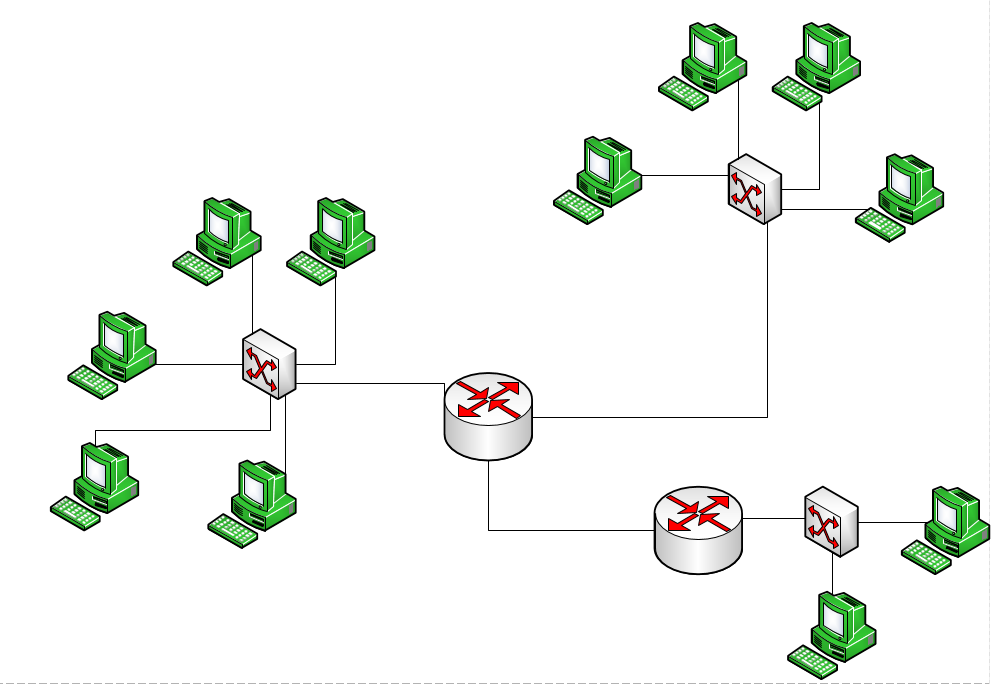


图 25. 安全环境下的网络拓扑

（2）**某台计算机受到外部网络中的病毒程序攻击：**

（PC为红色代表受到病毒程序攻击）

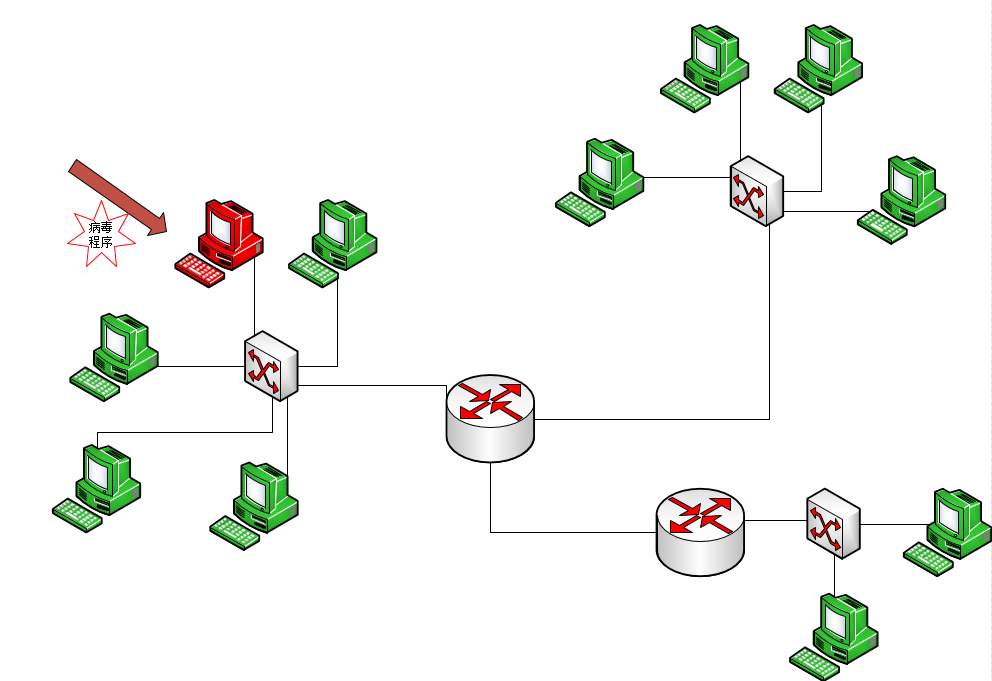
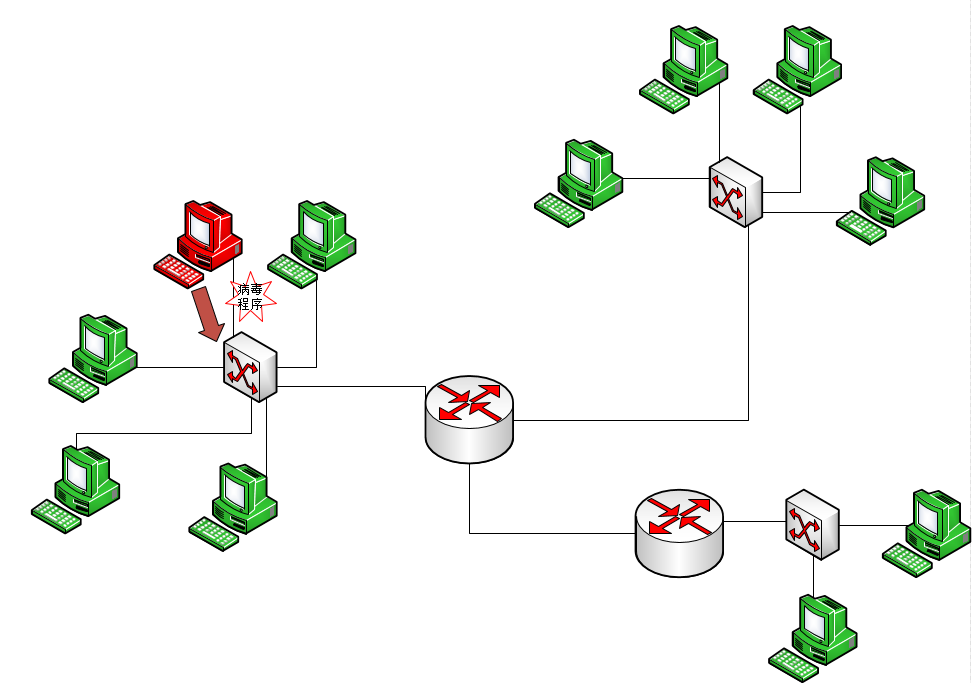


图 26. 某台计算机受到外部网络中的病毒程序攻击

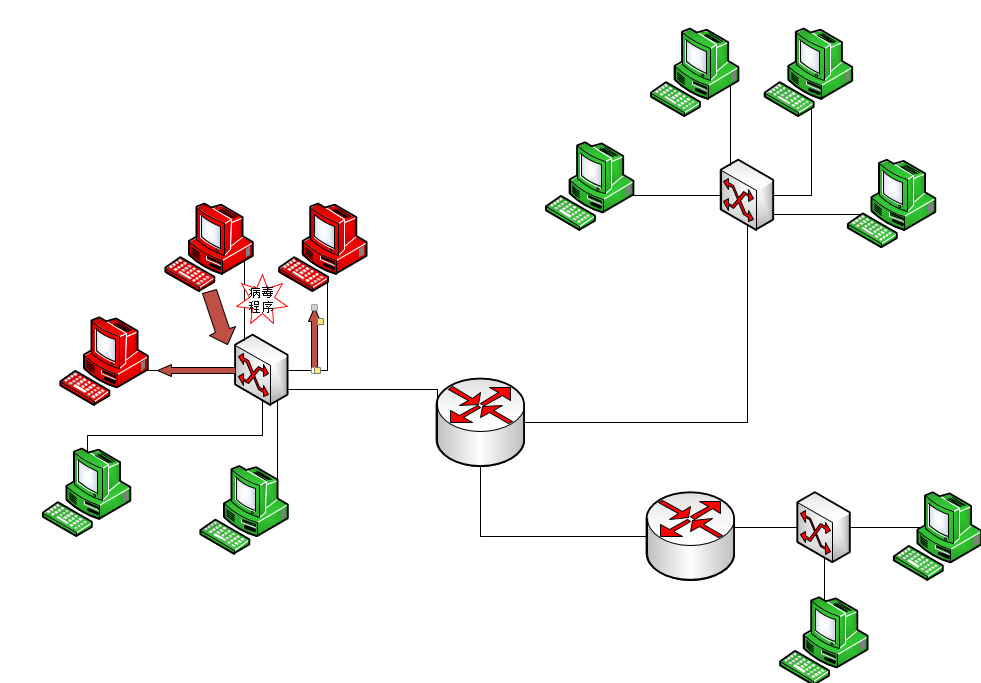
**（3）病毒程序通过局域网或者互联网中的网络设备传播到其他网络环境中：**

受到病毒攻击的主机向外传播病毒，通过网络设备使得更多的主机受到病毒攻击，导致网络中主机大范围出现被病毒程序攻击的情况。

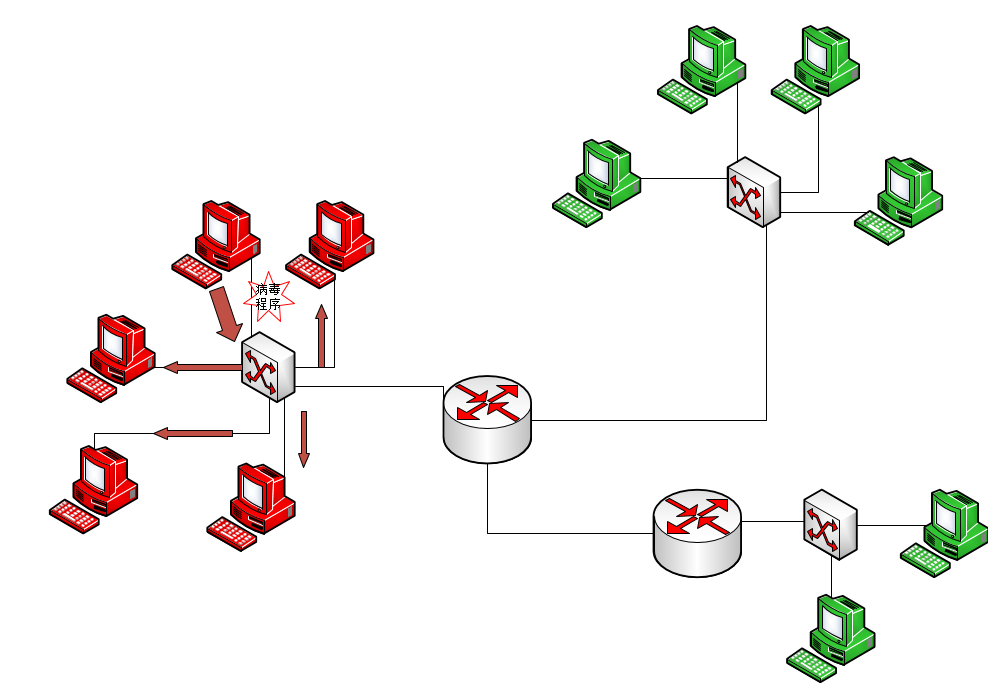
①



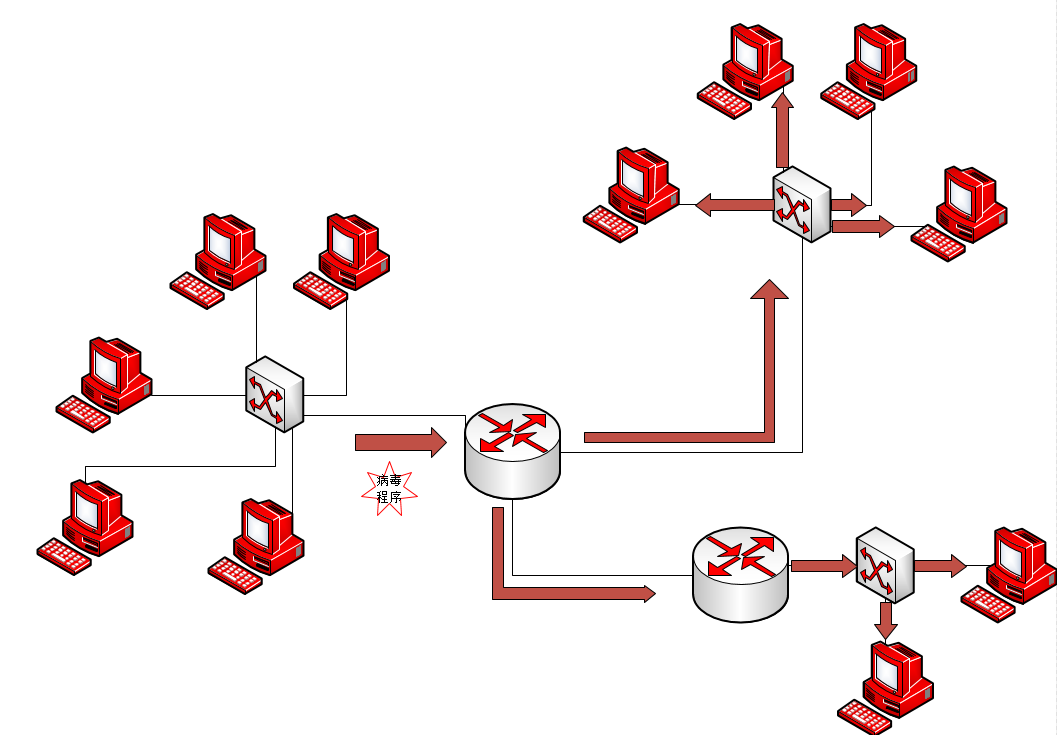
②



③

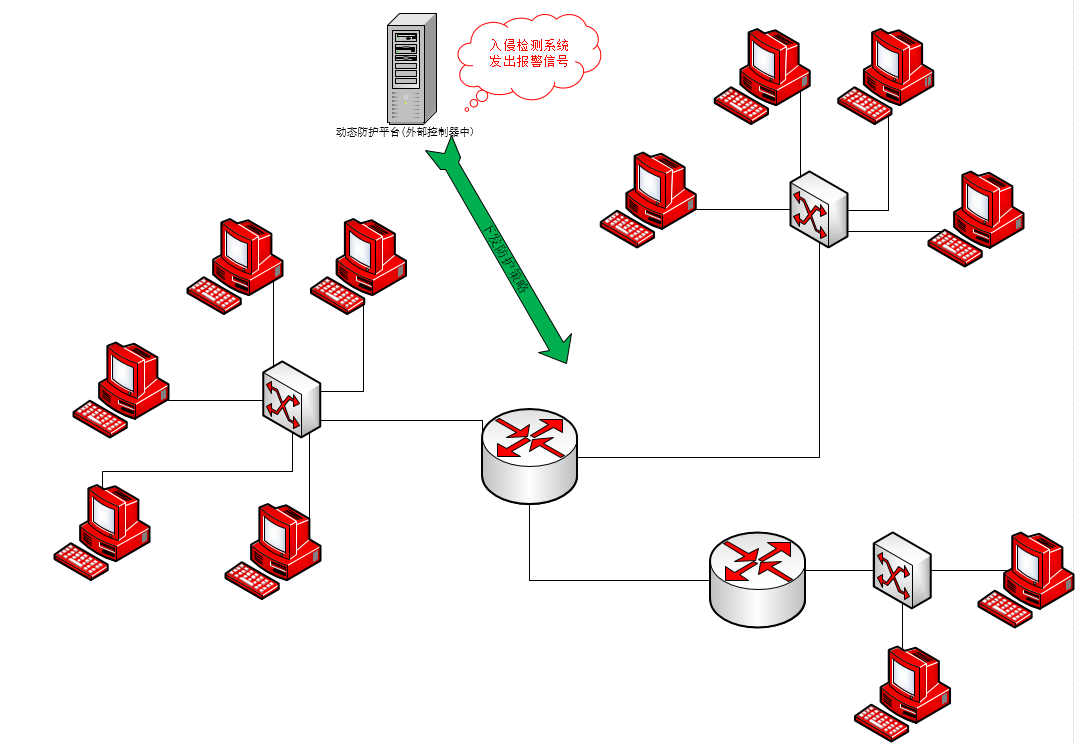


④

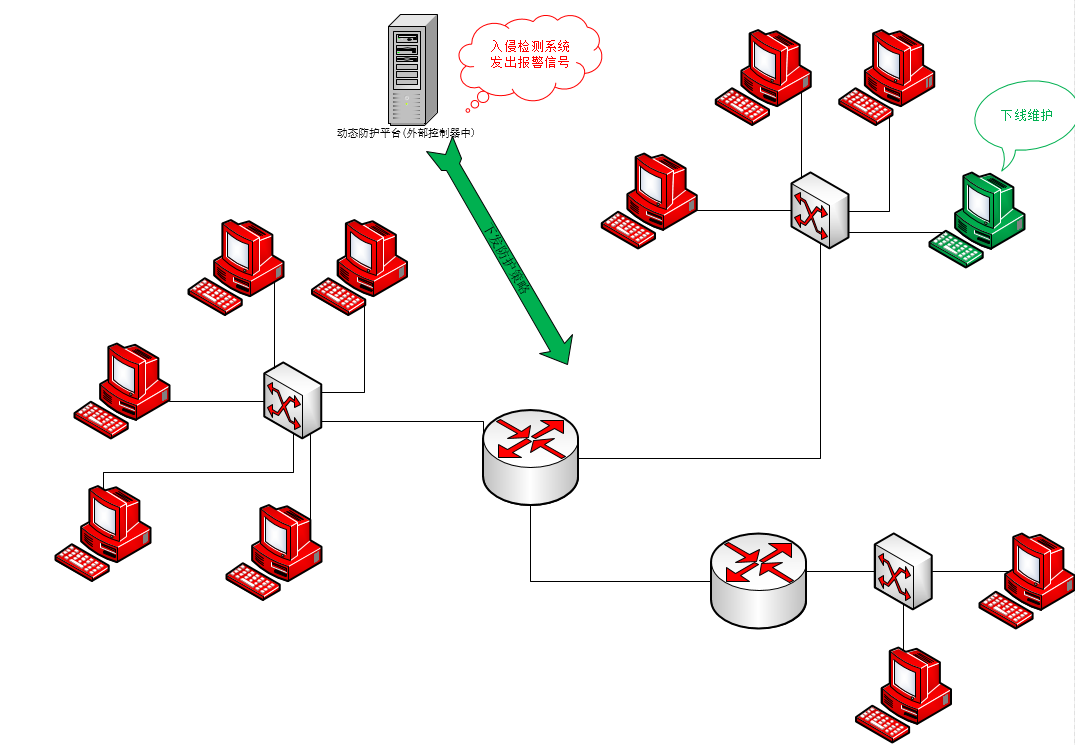


**（4）防护平台执行入侵检测后发出报警信号并向网络下发防护策略进行动态防护**：

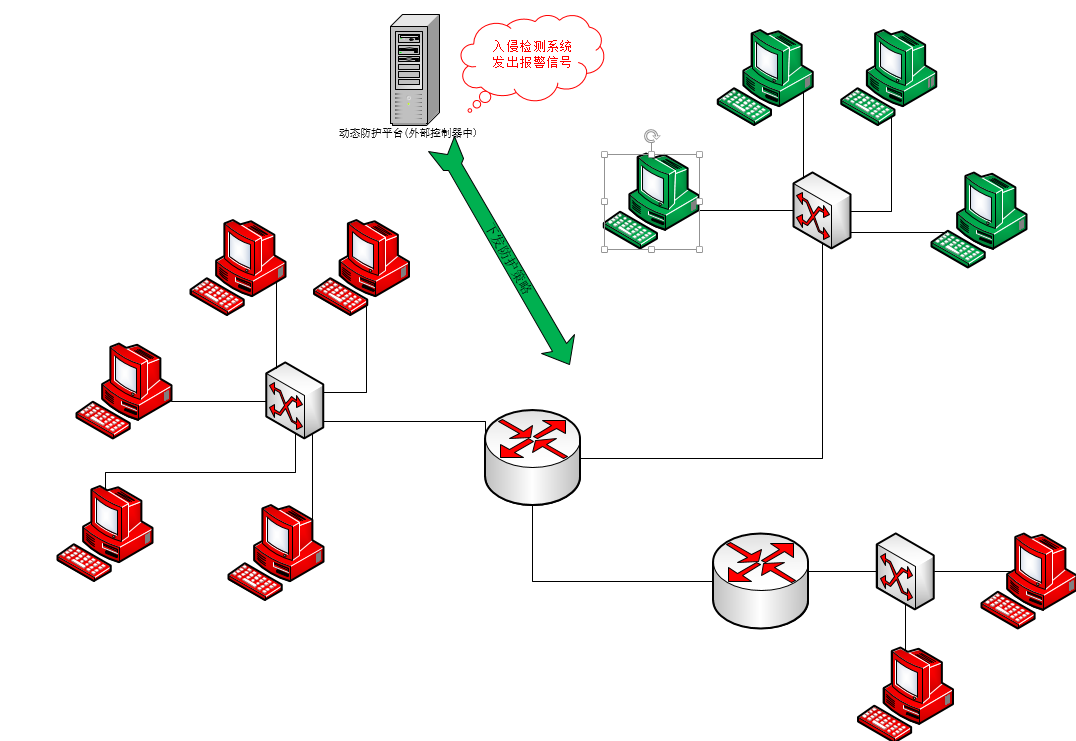
位于网络外部控制器上的防护平台在执行攻击检测算法过程中发现受攻击主机节点，发出警报并报告受攻击节点，随后下发防护策略，如主机下线维护清除病毒程序。待主机维护完成之后，重新上线。所有主机维护完成后，防护平台警报清除。



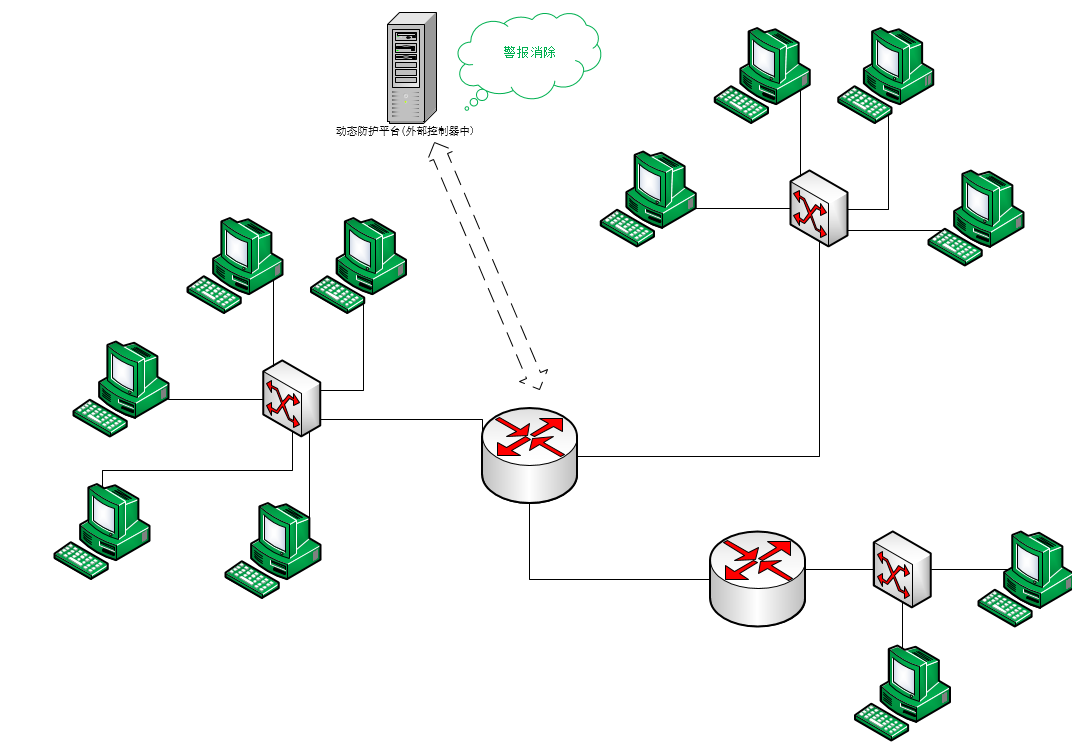
②



③



④



### 5.1.2. 攻击检测流程

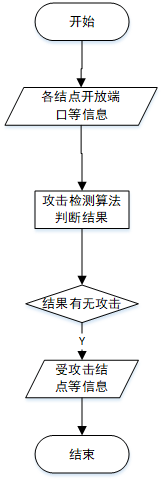


图 27.病毒攻击检测流程图

上图描述了病毒攻击的检测流程，接收各个结点开放端口等信息，交由攻击检测算法判断结果，攻击检测算法主要关注结点开放的端口信息中不常用的端口，以及这些端口接收和发送的信息，根据这些判断该主机是否做异常行为，判断是否受到病毒攻击，若受到攻击输出受攻击结点的信息。

对于病毒攻击检测的初步构思为基于数据包检测的实时分析：运行一个监听进程，获取发送至主机的所有数据包。我们实现根据攻击定义一个规则集，包含一些预定义的规则，例如在一个连续时间内，收到一定频率以上的SYN或者FIN请求（TCP），我们就可以认为检测到了SYN/FIN扫描攻击，对于其他攻击也类似。通过一个匹配算法，将收到的数据包与规则集中的规则进行匹配，进而确定属于哪种攻击类型，然后输出攻击信息。

### 5.1.3 攻击检测算法设计与实现

攻击检测的主要方法是根据当网络系统中，病毒攻击主机后也是通过开放某些主机端口，然后向这些端口发放特定指令，受攻击主机可以检测相关端口及数据信息检测判断是否受病毒攻击。





### 5.1.4. 攻击防护流程

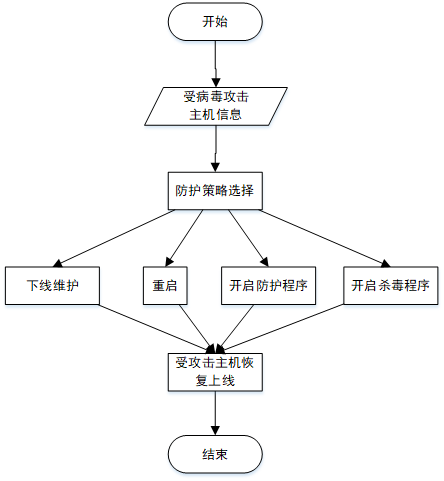


图 28.病毒攻击防护流程图

上图描述了主机受到病毒攻击后的防护流程，当系统接收到受病毒攻击的主机的信息，并经过用户对防护策略的选择后，可对受害主机进行下线维护、重启、开启防护程序和开启杀毒程序等防护措施，直到受害主机恢复正常后再重新上线。

### 5.1.5. 攻击防护设计与实现



## 5.2. DDos拒绝服务攻击防护流程

### 5.2.1. DDoS拒绝服务攻击检测防护简介

该攻击识别防护系统中当用户选择开启入侵检测和网络动态防护以了解动态防护对于网络拓扑系统的保护效果，当动态防护功能开启，系统将不断对网络拓扑中各节点间的流量信息进行统计，并根据流量信息统计结果判定当前受到攻击的主机，接着通过Mininet接口获取受到攻击主机的相关信息，将主机信息传递给动态防护系统，由防护系统根据用户选择的防护策略完成对受到攻击主机的防护。在这个过程中，我们应该能在有网络中有主机受到攻击时在仿真拓扑中看到主机节点受到攻击的信号提示，前端展示的模拟主机图标变为红色。随后动态防护系统根据攻击识别模块传递来的受到攻击主机的信息以及攻击源的相关信息通过对ONOS流表下发的操作完成对应防护功能，防护过程完成后，主机受攻击的警告信息应该消失，也就是说主机应该会被攻击识别模块判定为当前未受到攻击，并且模拟主机的图标应该变为未受到攻击的图标。

攻击防护过程描述如下：

**（1）网络中服务器提供某服务处于正常状态中：**

局域网或者通过路由器连接至因特网中的主机可以通过网络连接访问该服务器上的服务。

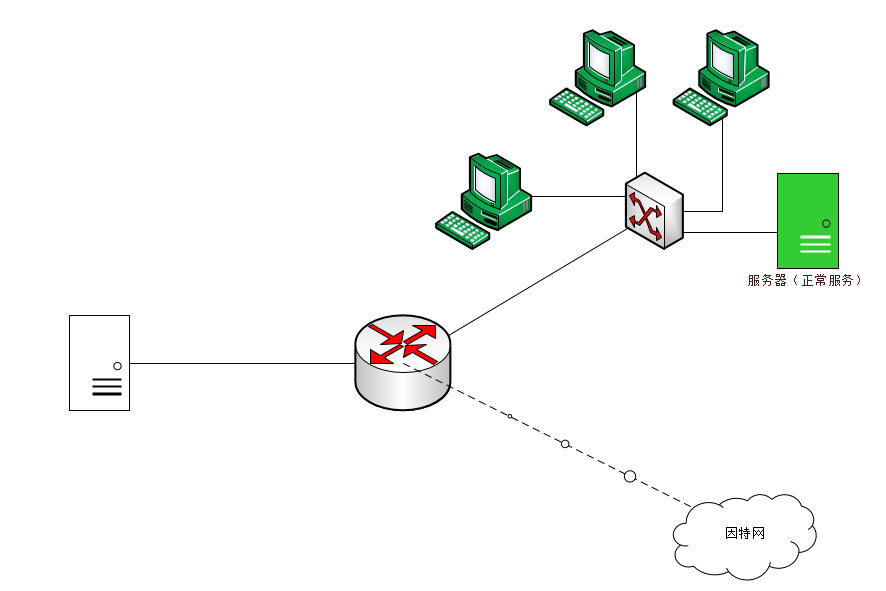


图 29. 网络中服务器提供某服务处于正常状态

**（2）服务器受到因特网中DDos攻击导致服务资源被占用，拒绝服务：**

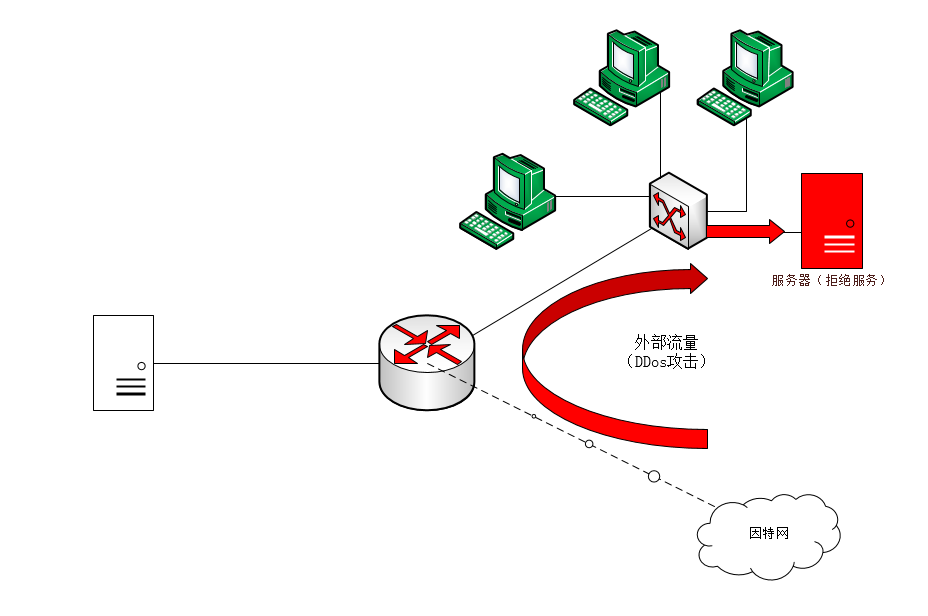


图 30. 服务器受到因特网中DDos攻击导致服务资源被占用

**（3）动态防护平台通过流量检测发现服务器流量异常，发出警报：**

防护平台经过流量统计监控发现该服务器流量异常，发出警报并报告攻击节点。然后通过下发流表等方式将来自外部网络的DDos攻击流导向网络中特定服务器处理，使得服务能够正常进行。

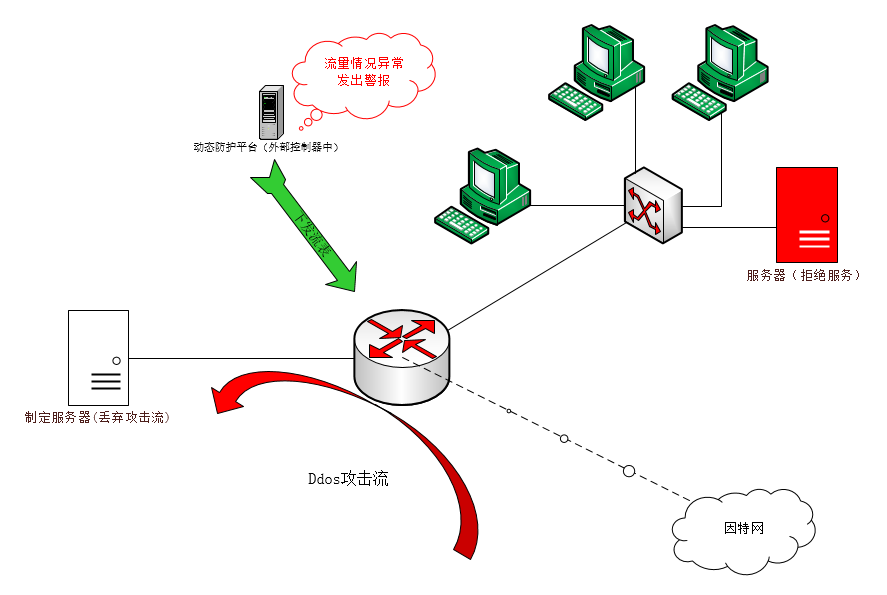


图 31. 动态防护平台检测到流量异常发出警报

**（4）服务器恢复正常服务之后，防护平台警报消除：**

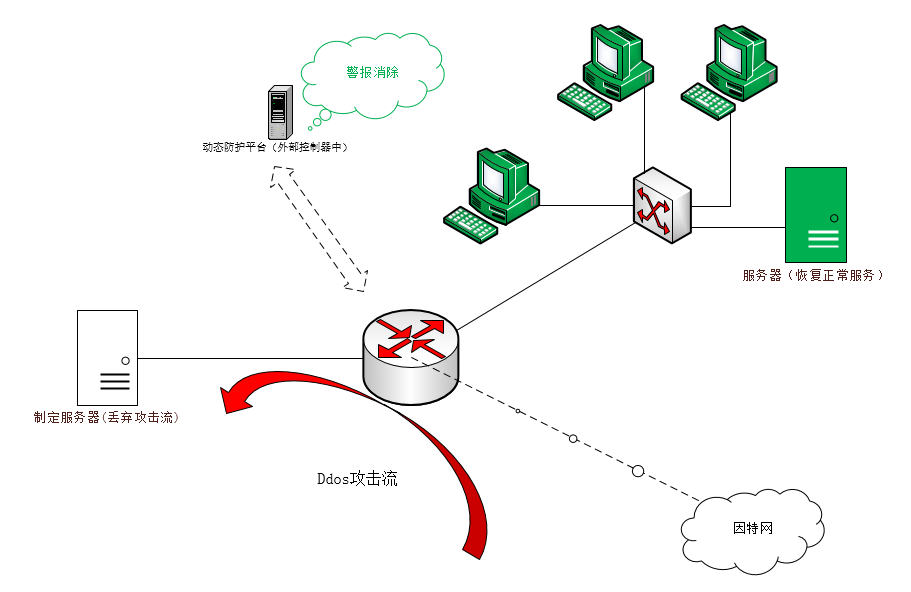


图 32. 服务器恢复正常服务

**服务器流量随时间变化图：**

时间

Ddos攻击

开始

图 33. 服务器流量曲线图

我们可以从此示意图中看到服务器流量在受到DDos攻击时呈一个突然上升趋势，当防护平台检测到受到攻击后下发流表将攻击流导向至特定服务器节点后受攻击服务器流量开始下降最后趋于稳定。

### 5.2.2. 攻击检测流程

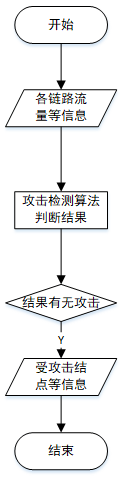


图 34 .DDoS攻击检测流程图

上图描述了DDoS攻击的检测流程，系统接收各链路流量等信息，交由攻击检测算法判断结果，攻击检测算法主要关注各个结点单位时间段某个端口收到的数据包数量，根据这些判断该主机是否异常，判断是否受到DDoS攻击，若受到攻击输出受攻击结点的信息。

### 5.2.3. 攻击检测算法设计与实现

攻击检测的主要方法是根据当网络系统中主机受到DDos攻击时，主机流量信息会出现异常增大的情况，或者病毒攻击主机后也是通过开放某些主机端口，然后向这些端口发放特定指令，攻击识别系统主要根据Mininet下仿真拓扑连接到ONOS控制器，而ONOS开放了返回Mininet拓扑下节点流量信息的接口，系统可以根据检测流量信息判定主机是否受到攻击，随后从Mininet中获取受到攻击主机的信息。

攻击检测模块的交互过程如下图所示，系统通过攻击检测模块启用攻击检测算法，攻击检测模块调用流量检测模块来获取各个主机的收发流量，流量监控模块从统计获取模块获取到各个链路的流量统计，由此攻击检测模块获取到各个主机的收发流量数据，对这些数据进行分析判断出主机是否正遭受DDos攻击，对于正遭受攻击的主机，攻击检测模块将其主机id记录下来，当每个时刻的攻击检测算法执行完以后一起返回给系统。

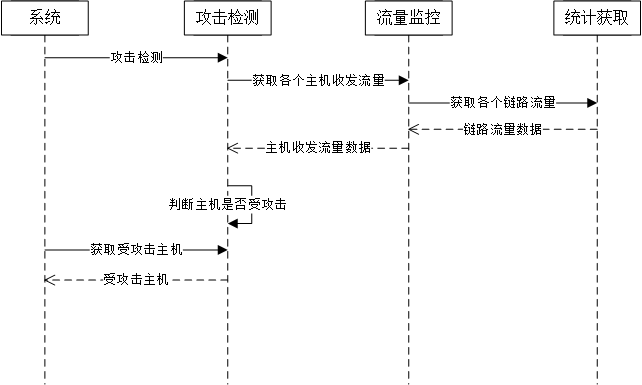


图 35.攻击检测模块交互过程

上图的攻击检测模块执行攻击检测算法来判断当前主机是否正遭受DDos攻击，详细的攻击检测算法流程如下图所示，算法的输入是所有节点的流量信息，输出是检测出来受攻击的主机的列表。算法拿到所有节点的流量信息以后计算所有节点当前间隔时间内的流量数据，之后判断每个节点的流量数据是否超过系统给定的阈值，如果超过则认为该节点流量异常，即遭受DDos攻击。将所有被判断出流量数据超过阈值的主机节点加入到受攻击的主机列表中，在算法结束前输出该列表。



图 36.攻击检测算法流程图

### 5.2.4. 攻击防护流程

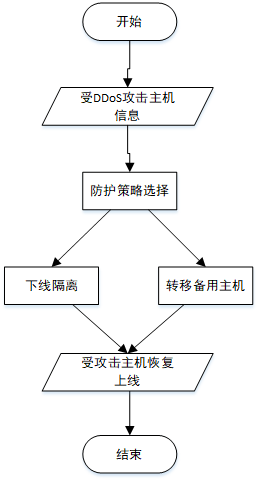


图 37. DDoS攻击防护流程图

上图描述了主机受到DDoS攻击后的防护流程，系统接收DDoS攻击主机的信息，经过防护策略的选择，可对其进行下线隔离和转移备用主机等防护措施，直到受害主机恢复后再重新上线。

### 5.2.5. 攻击防护设计与实现

动态防护过程中，需要根据攻击识别模块中提供的主机信息以及攻击信息，修改SDN流表，使得攻击流不能再到达受攻击主机或者将主机进行下线隔离处理。流表下发主要采用ONOS Rest API请求的方式，交由流表管理模块实现。

攻击防护模块交互过程如下图所示，对于那些受攻击的主机，系统选择相应的防护策略，攻击防护模块从数据转发模块获取主机的转发信息后，调用流表管理模块执行下发流表的操作，流表管理模块修改流表后返回流表下发结果。

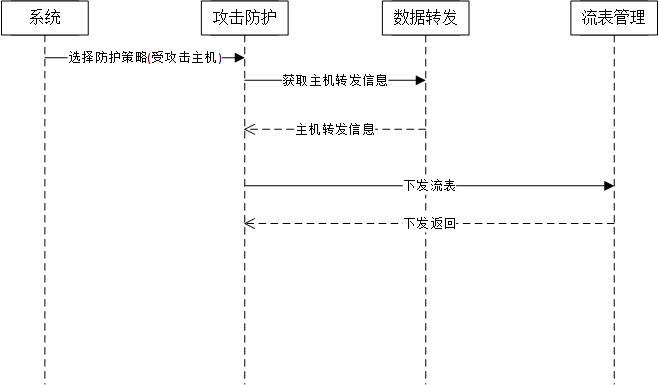


图 38.攻击防护模块的交互过程

系统请求执行防护策略后，攻击防护模块执行相应的攻击防护算法来对受攻击主机采取防护措施，具体的攻击防护算法流程如下图所示。算法的输入是受害主机，开始时先获取受害主机的数据转发时所经过的交换机及端口的信息，根据这些信息并结合系统所请求选择的防护策略来执行不同的防护措施。对于下线隔离策略，算法处理受害主机的数据转发时所经过的交换机及端口的信息，通过改变流表的方式将受害主机与该交换机的通信链路隔绝并下发流表，以此实现受害主机的下线隔离。对于转移备用主机策略，算法获取受害主机的攻击源的IP地址，通过修改流表的方式，将这些攻击源到受害主机路由修改转移到一个备用主机上并下发该流表，以此实现对攻击的转移。算法执行完上述的防护措施后，即完成了防护策略，算法结束。

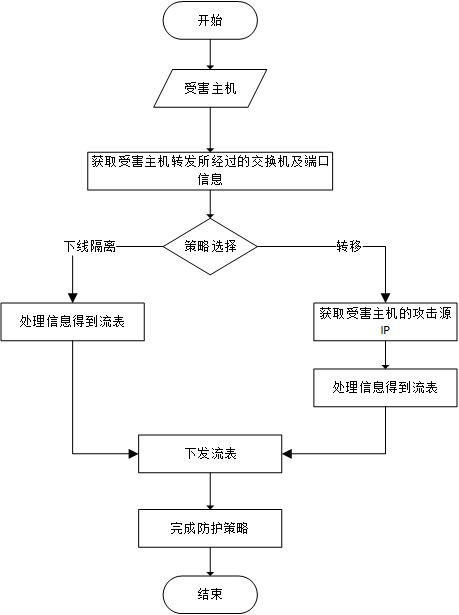


图 39.攻击防护算法流程

# 系统具体实现

## 功能展示

### 6.1.1. 拓扑展示

****

图 40.系统主界面

上图为系统的主界面，界面左侧是系统的控制面板，主要提供基本信息的统计和通信网络拓扑管理和操作等功能，主要包括数据统计表和状态统计表的展示，创建网络拓扑，互联互通测试，动态防护策略的选择。界面右侧展示可视化的Mininet拓扑，当前通信网拓扑是以莆田市电力通信网为原型而搭建的，拓扑之下的背景为莆田市的地图，拓扑的每个节点按照实际的地理位置映射到地图的相应位置上。

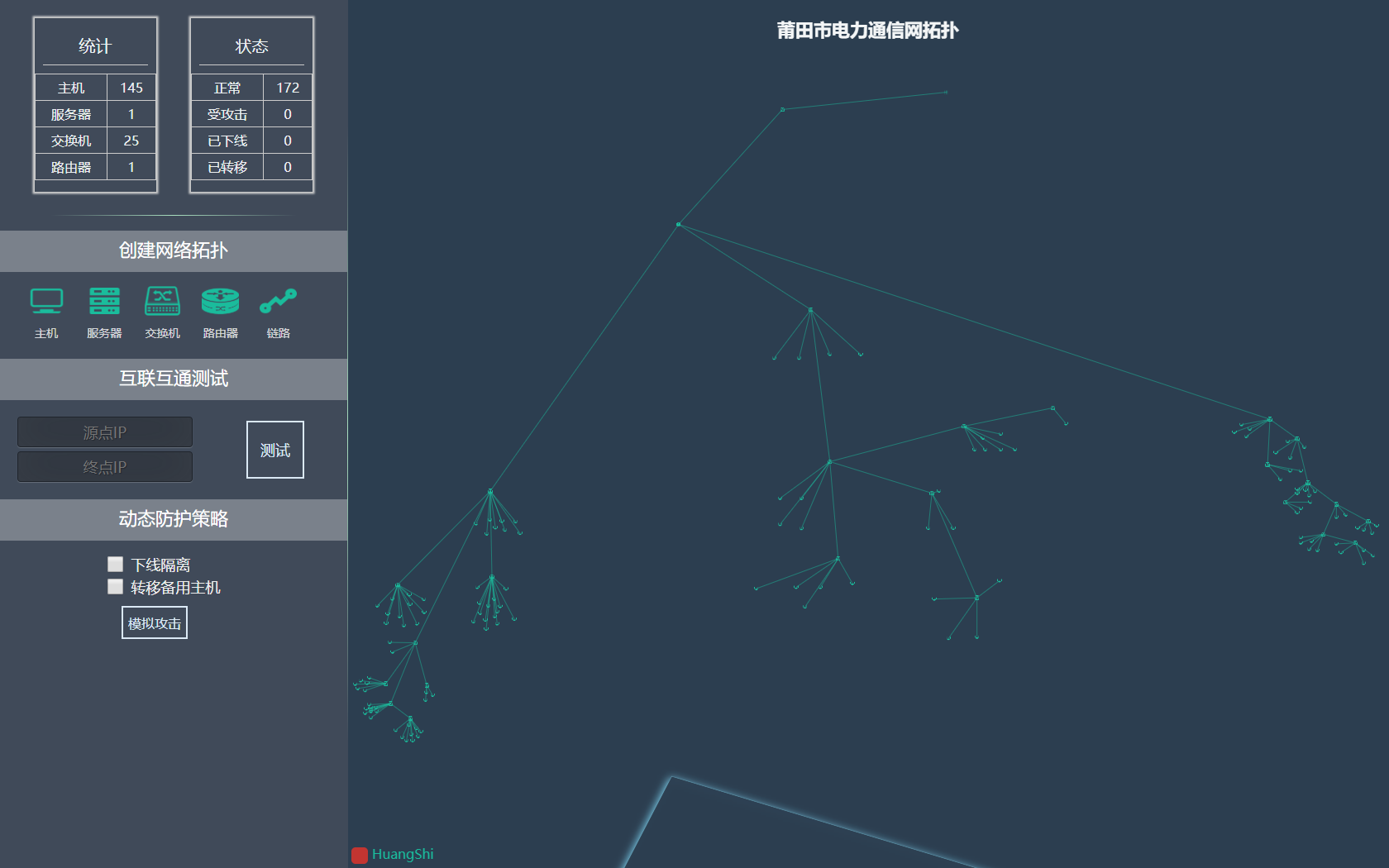


图 41.黄石线路的拓扑

上图拓扑展示的是莆田市电力通信网的黄石线路部分，通过区域缩放可以看到该线路上各部分的详细拓扑结构，可以看到该线路主要分为三个部分，每个部分主要由交换机连接若干主机形成的簇组成。拓扑上的节点分为四种类型：路由器、交换机、服务器和主机，其中每个线路一般存在一个路由器和若干交换机，每个交换机再向下连接若干主机。

### 6.1.2. 流量变化检测

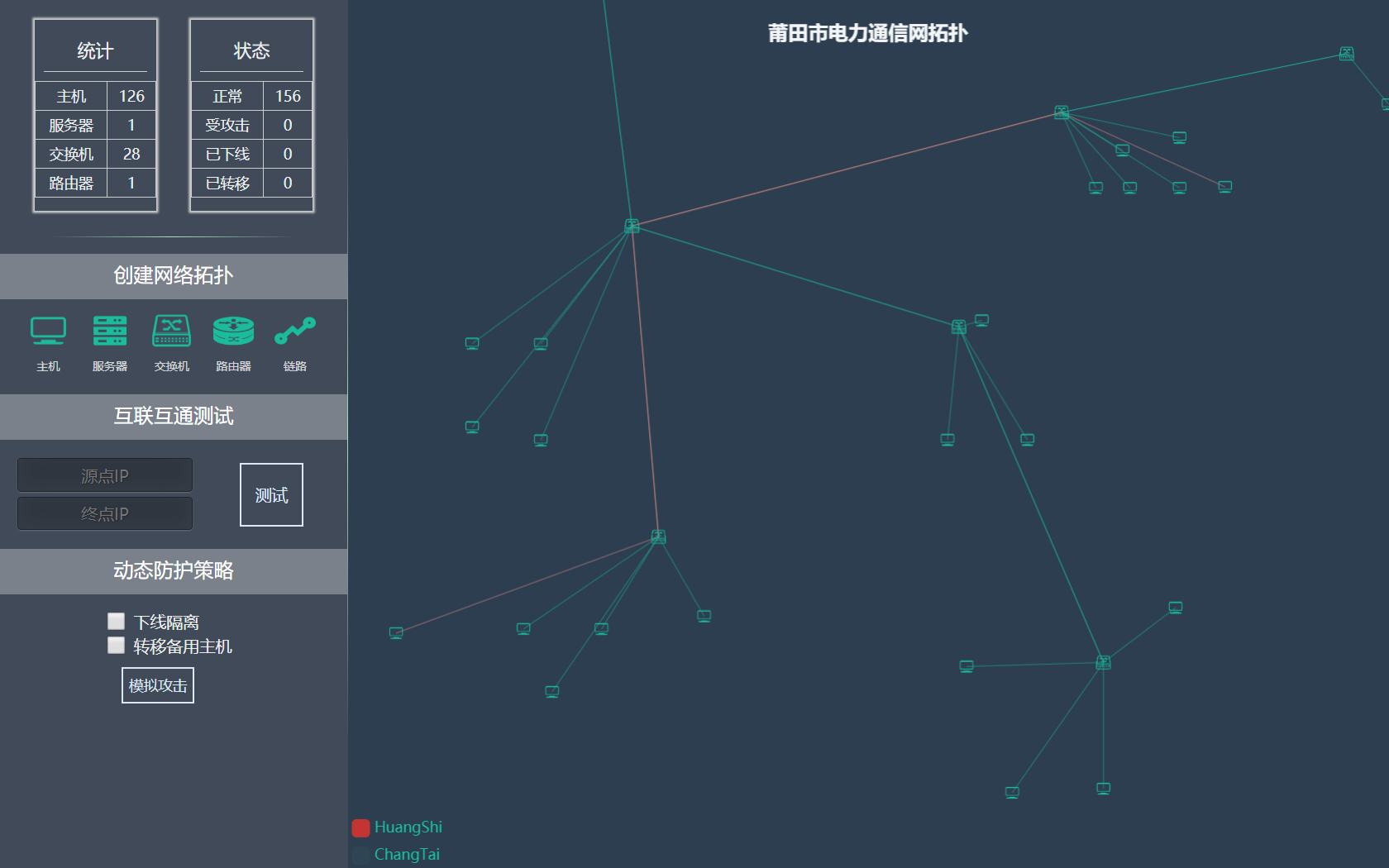


图 42.链路的流量变化

系统实时检测拓扑中各链路的流量的变化，当某条链路出现流量变化时，拓扑上对应的链路通过改变颜色来以示该链路产生流量，链路的颜色从原来的绿色变成橘色并且颜色深度随着流量的增加而增加。上图展示了通信网中两台主机互相发送数据包时，拓扑界面的展示效果：两台主机之间的通信链路的颜色变成橘色。

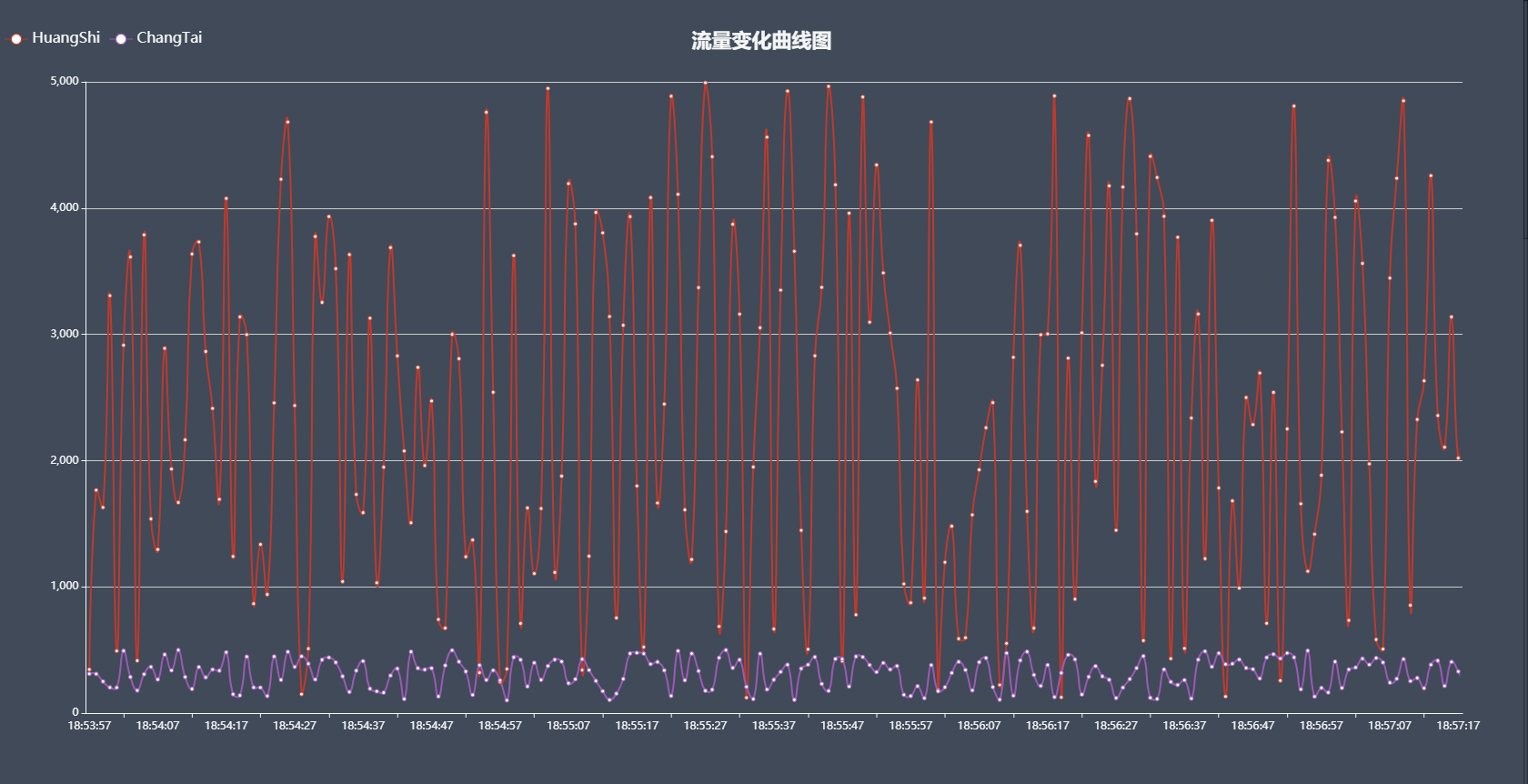


图 43.线路流量变化曲线图

在系统的统计页面可查看当前拓扑上各线路实时变化的流量曲线图，横坐标为时间轴，纵坐标为流量大小，曲线图每隔3秒更新一次。上图展示了黄石线路和常太线路的实时流量变化，其中红线代表黄石线路，紫色代表常太线路。

### 6.1.3. 查看节点信息

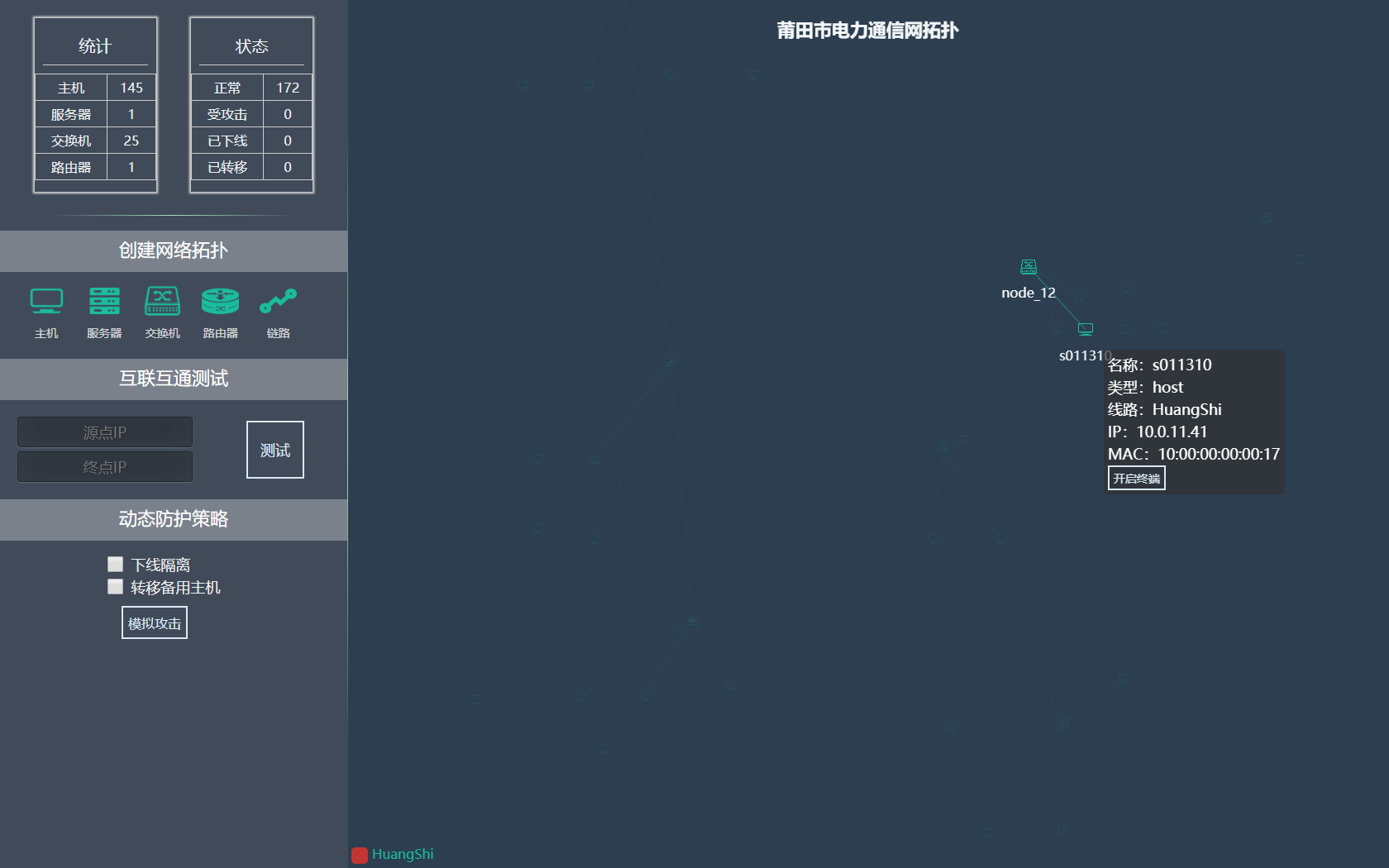


图 44.查看节点信息

鼠标单点拓扑上的节点可查看该节点的对应信息，如上图所示，鼠标单击某个节点后弹出一个提示框，内容包括节点名称，节点类型，节点所属线路，IP地址和MAC地址。



图 45.主机状态信息饼图

通过鼠标双击主机节点可查看对应主机的当前状态信息，包括进程，内存和CPU三类。每一类数据按照各部分的组成以饼图的形式呈现，其中进程状态信息包括运行进程、僵尸进程、停止进程和睡眠进程；内存状态信息包括空闲内存、已用内存和内核缓存；CPU状态信息包括空闲CPU、用户空间CPU、内核空间CPU、进程分配CPU和读写等待CPU。

### 6.1.4. 开启主机终端

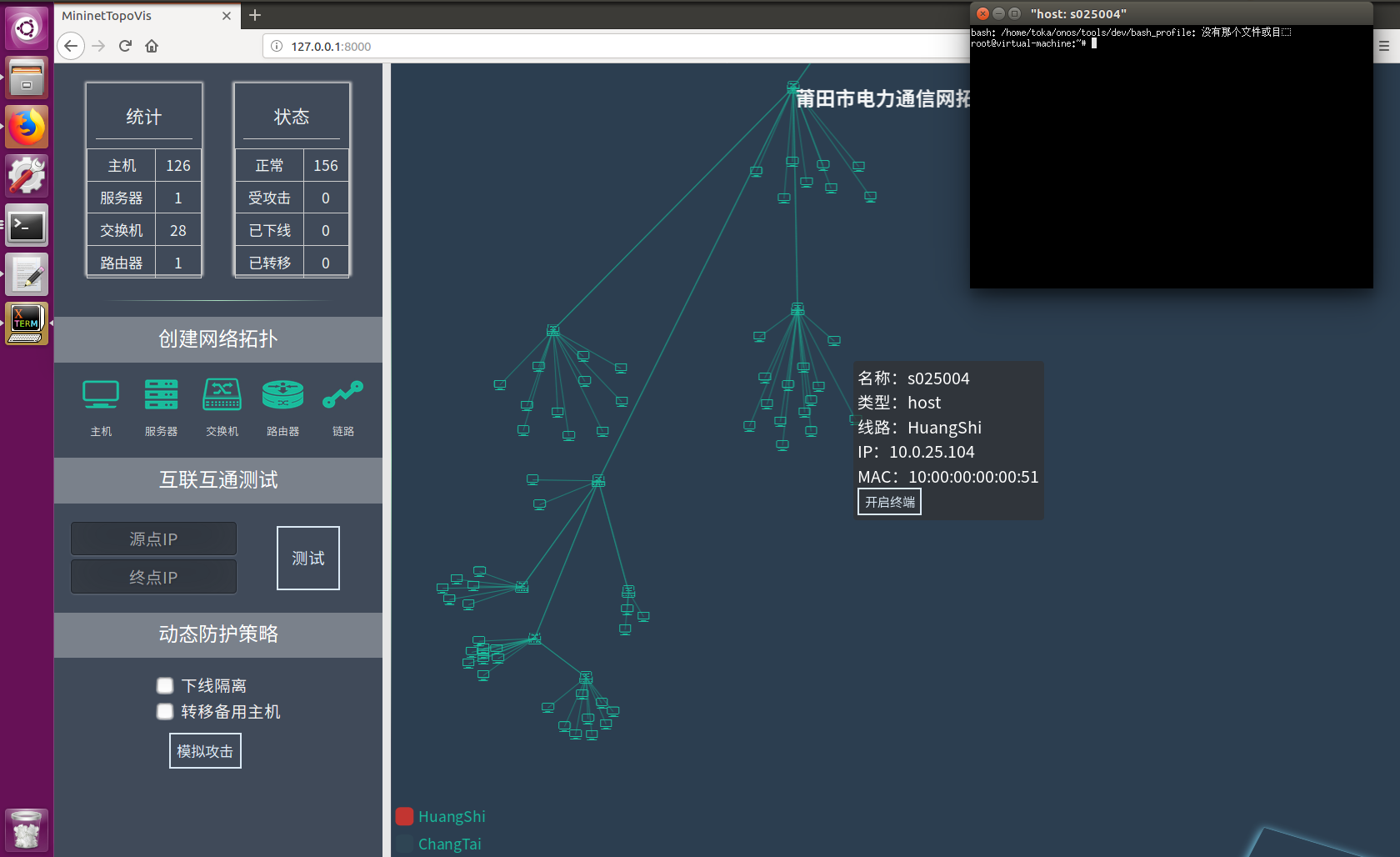


图 46.主机开启终端

在主机节点上可开启其主机终端通过输入命令行进行内部操作。

### 6.1.5. 启用动态防护

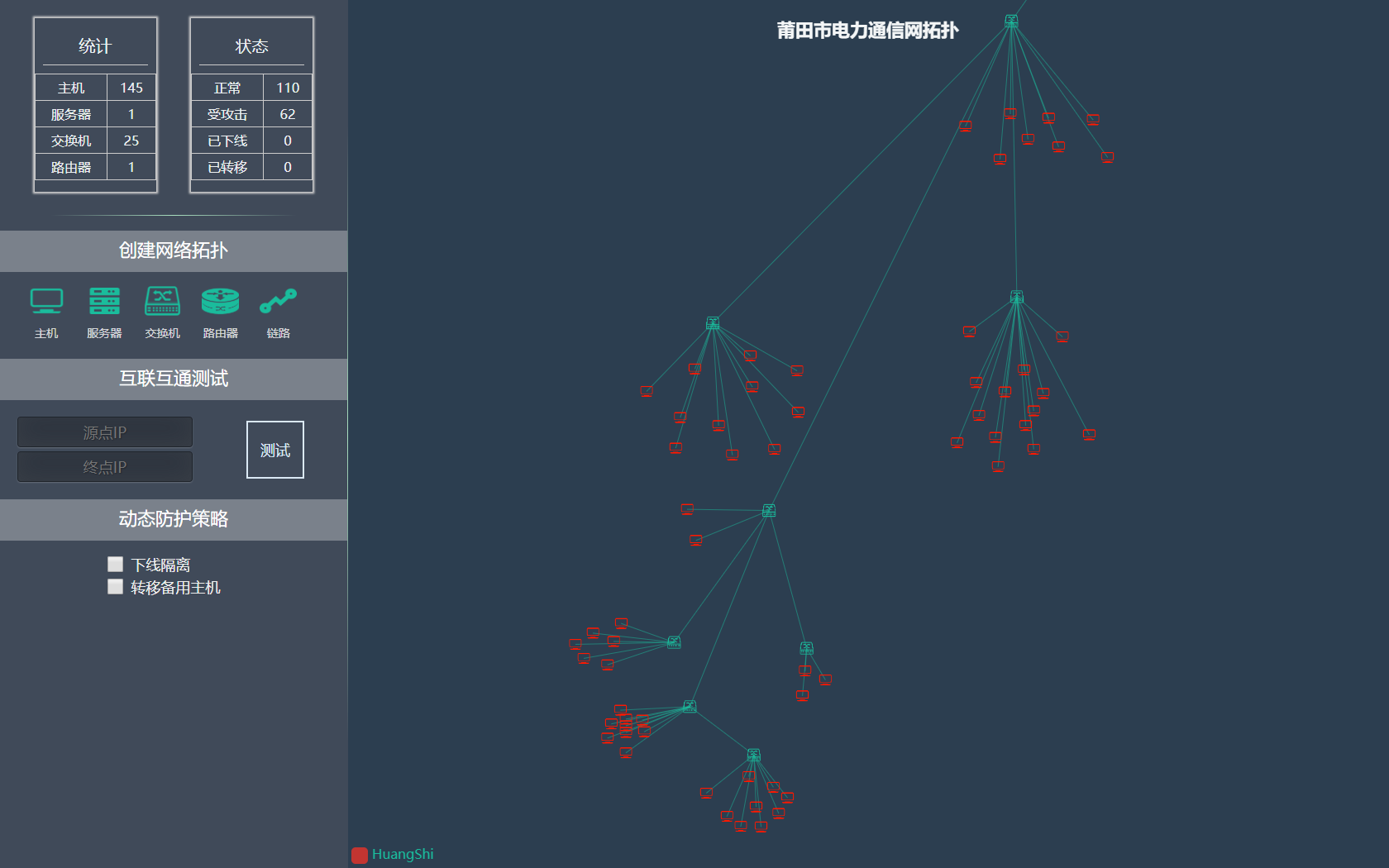


图 47.主机遭受攻击

系统开启攻击检测服务以后，当通信网的主机遭受攻击时系统通过在拓扑界面将受害主机标记为红色作为警示。

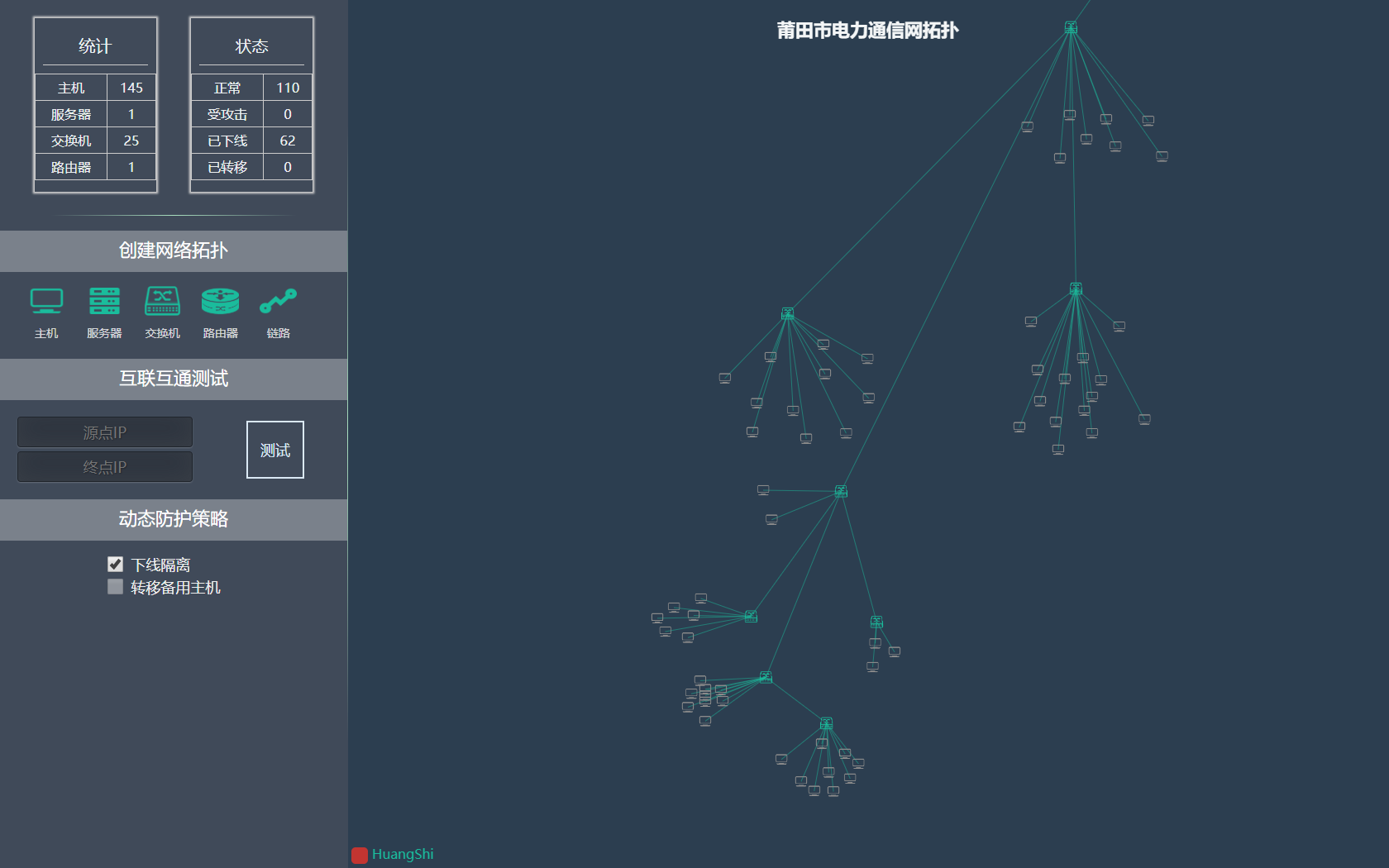


图 48.启用下线隔离防护策略

当主机遭到攻击时，用户可选择动态防护策略中的下线隔离策略，系统通过对受害主机进行下线处理来避免主机遭受侵害进而避免整个通信网受到更大的影响。如上图所示，系统在拓扑界面上对下线处理的主机标记为灰色以示下线状态。

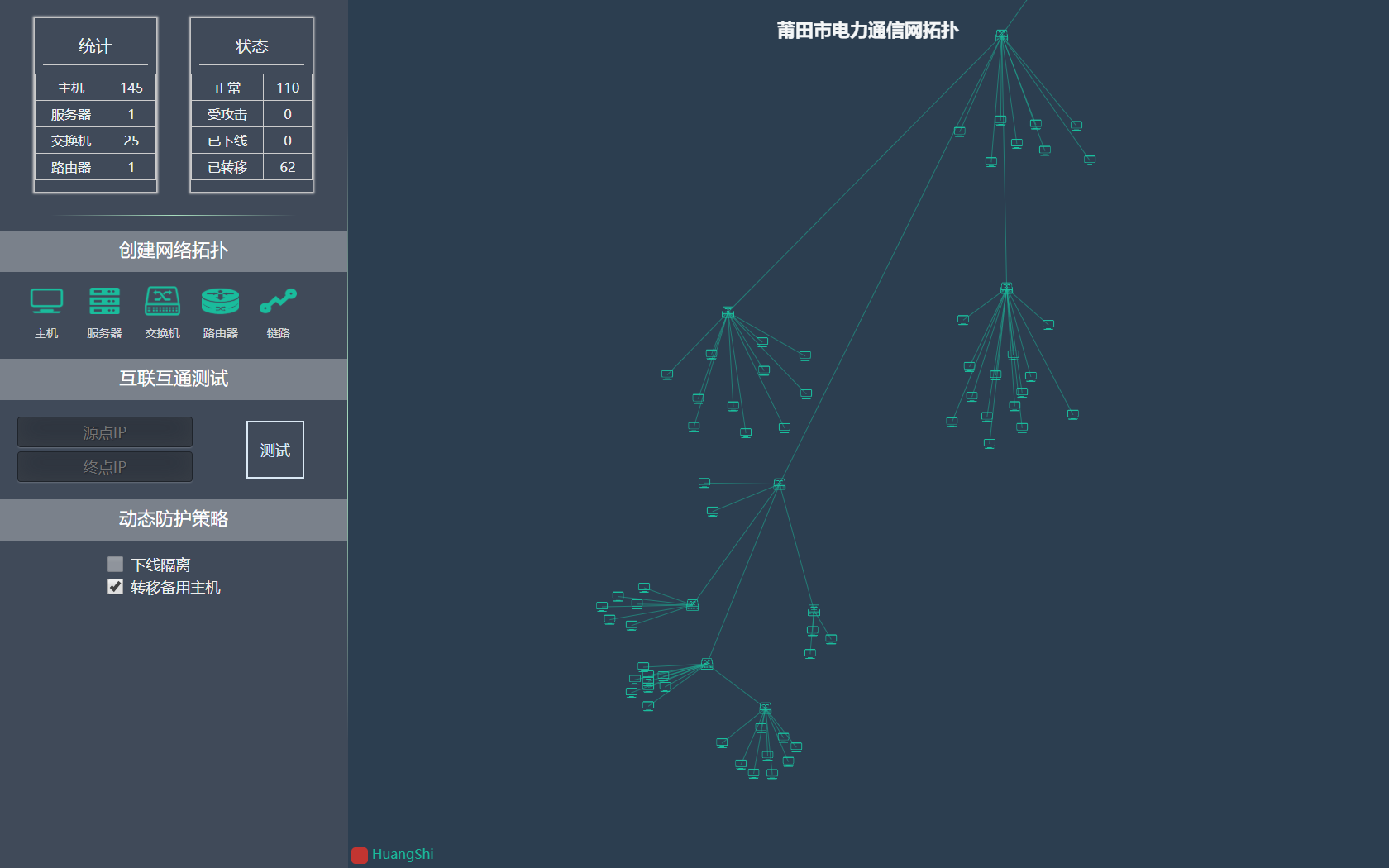


图 49.启用转移备用主机防护策略

当主机遭到攻击时，用户可选择动态防护策略中的转移备用主机策略，系统通过修改流表的方式将携带病毒的数据包转移到备用主机，以此保护目的主机不受病毒侵害。如上图所示，系统在拓扑界面上对受保护的主机恢复正常显示。

### 6.1.6. 查看统计数据

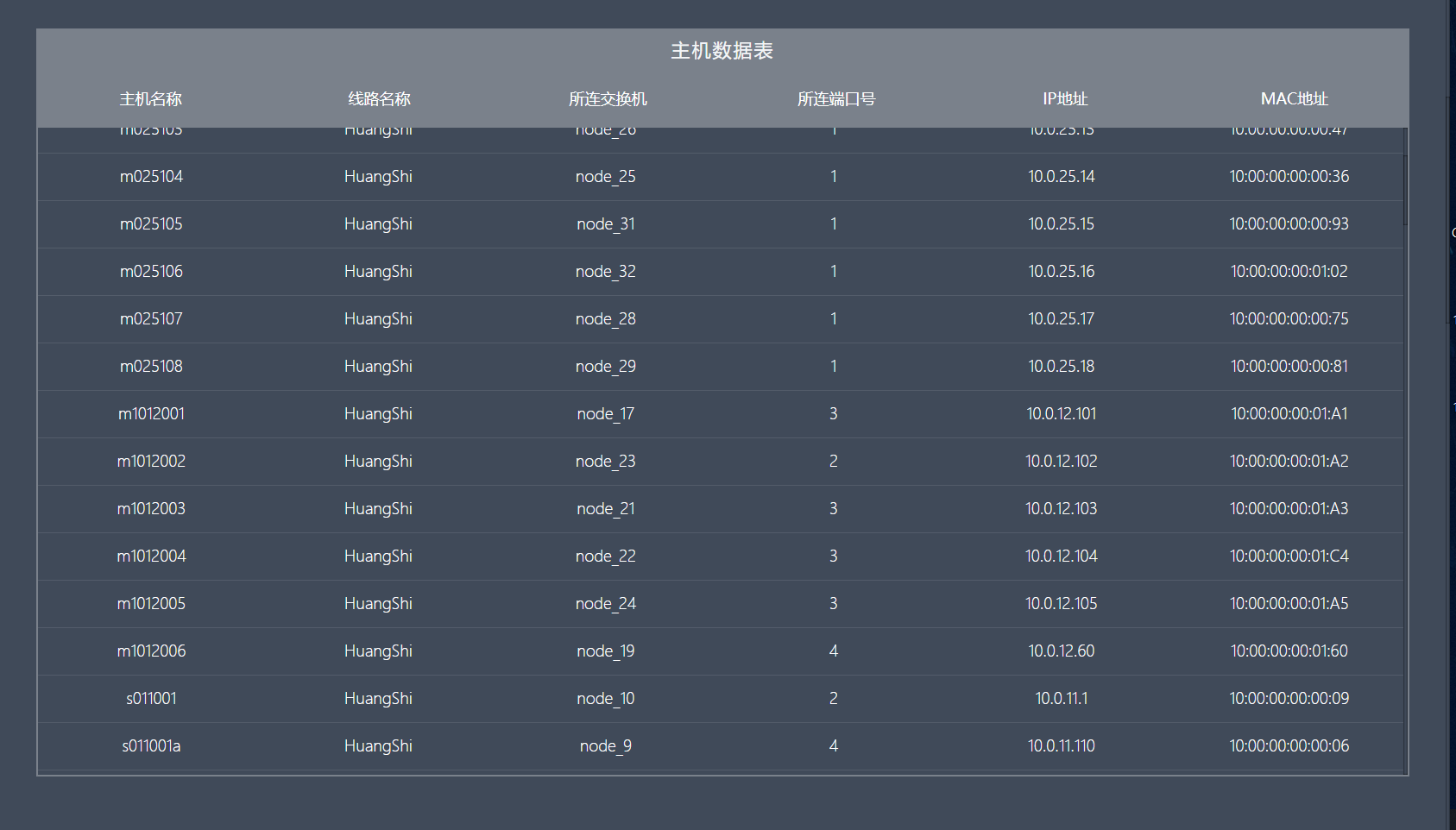


图 50.主机数据统计表

在统计页面可查看当前拓扑上的所有主机的数据统计表，如上图所示，表中的每一项代表一个主机，每项的内容包括主机名称、所属的线路名称、所连交换机、所连端口号、IP地址和MAC地址。



图 51.设备数据表

在统计页面可查看当前拓扑上的所有设备（交换机，路由器）的数据统计表，表中的每一项代表一个交换机或一个路由器，每项的内容包括设备名称、所属线路名称、端口数量和对应端口及其所连设备。

## 场景展示

### 互联互通测试场景

系统通过对Mininet中所有的节点对其他的节点发送数据包，以此实现通信网络的连通性测试。

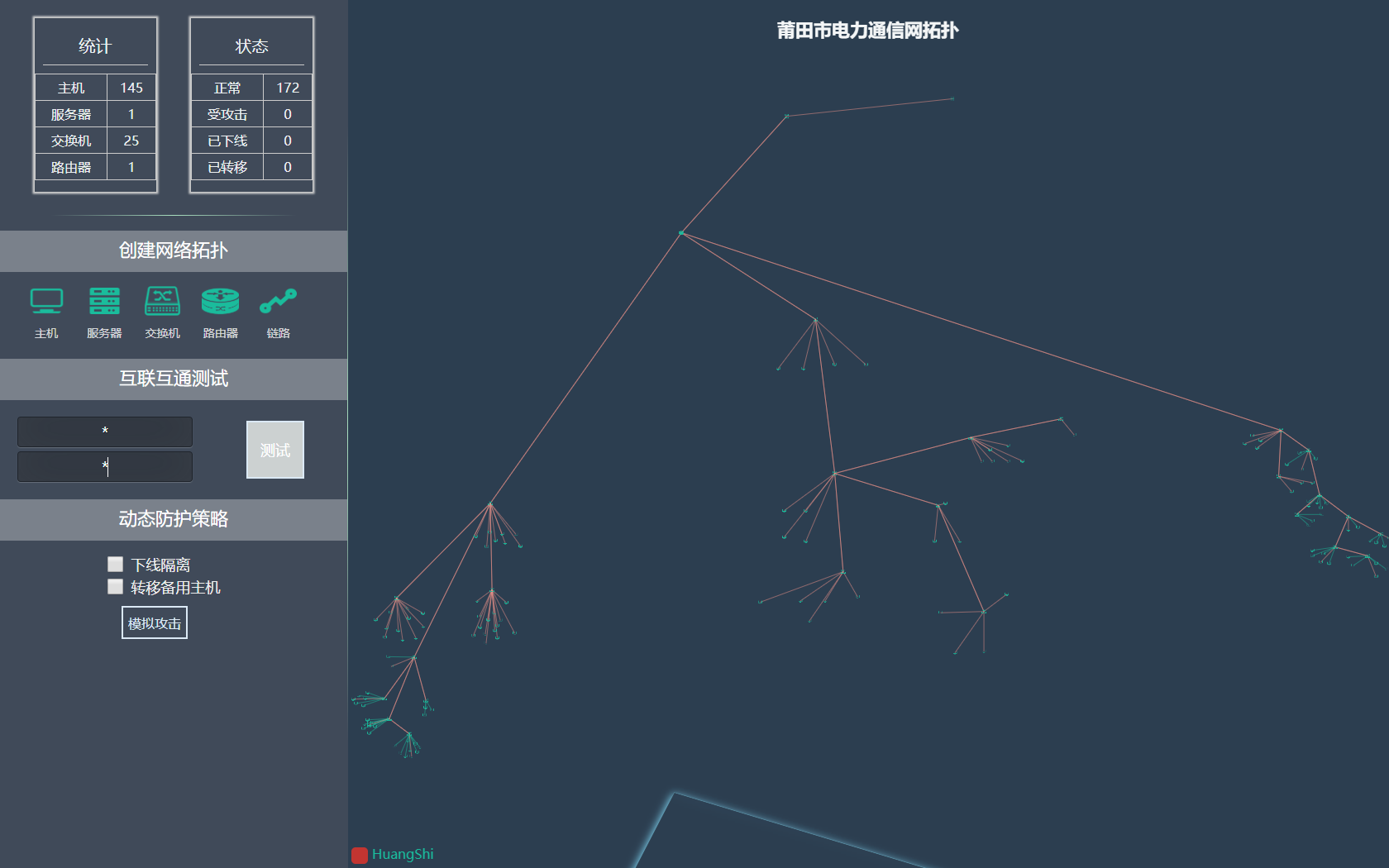


图 52.互联互通测试

如上图所示，在左侧面板中填入指定信息后点击测试，每个节点都对除它之外的其他所有节点发送数据包验证两节点之间是否互联，经过一段时间后，拓扑上的所有的链路都改变颜色说明了每条链路的联通性都是完好的。

### RTLAB发送状态数据包场景

RTLAB依据报文协议周期性地发送带有序号的状态数据包给Mininet，Mininet收到后将序号和状态数据包转发给主站，主站收到后回复确认收到的序号，Mininet将收到的确认序号依据报文协议转发给RTLAB。上述场景描述了系统中Mininet对RTLAB周期性发送状态数据包给主站的交互过程。

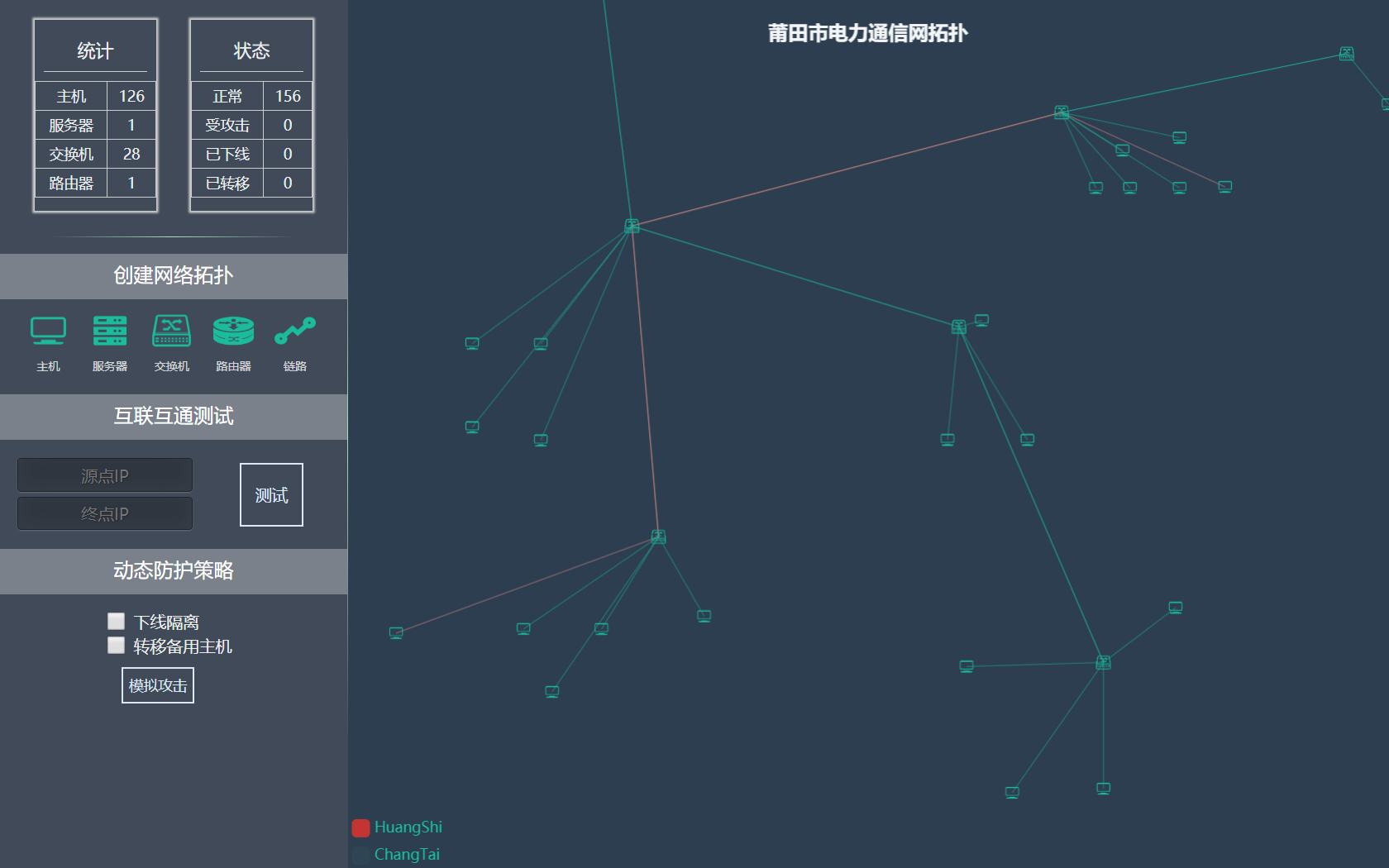


图 53.RTLAB发送状态数据包给Mininet主机

如上图所示，RTLAB发送状态数据包给指定的Mininet主机时，状态数据包按照流表发送到某个交换机后转发到指定的主机，状态数据包所经过的链路被系统检测到流量变化后改变颜色以标记出状态数据表途经的路径。

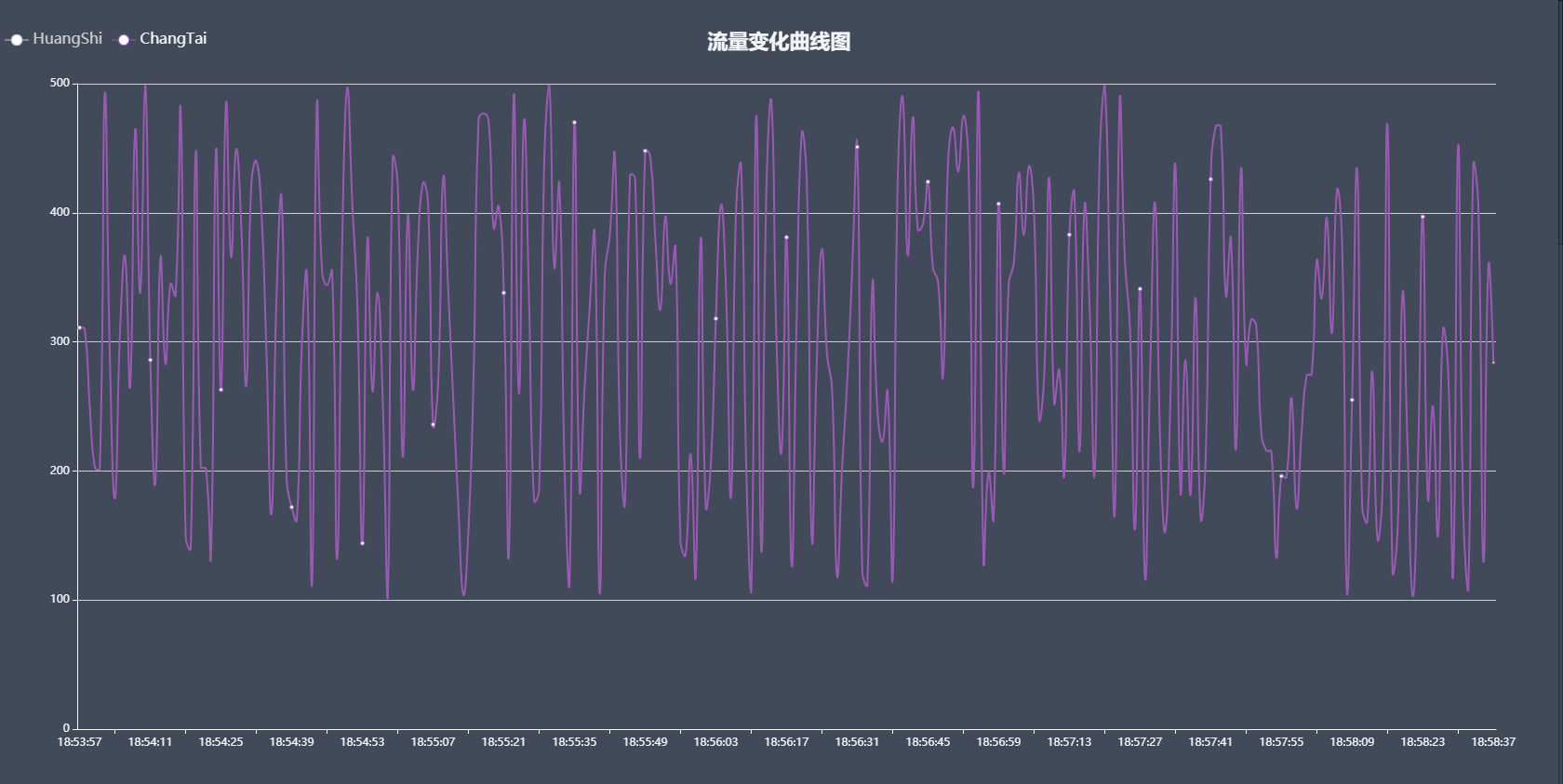


图 54.RTLAB发送状态数据包产生的流量变化

RTLAB发送状态数据包给Mininet主机时在相应线路上产生的流量变化如上图所示，可以看到一段时间内RTLAB周期性地发送状态数据包给Mininet主机。

### 攻击防护场景

当通信网络中的主机遭到病毒攻击时，由主机的攻击检测程序检测出当前主机正在遭受病毒的侵害并上报给系统请求防护，系统对拓扑界面发出警示，此时用户可选择相应的防护策略对受害主机进行防护，在此场景中我们选择下线隔离策略对所有受害主机进行下线处理以避免病毒程序在整个通信网络的蔓延并对通信网络造成更大的影响。

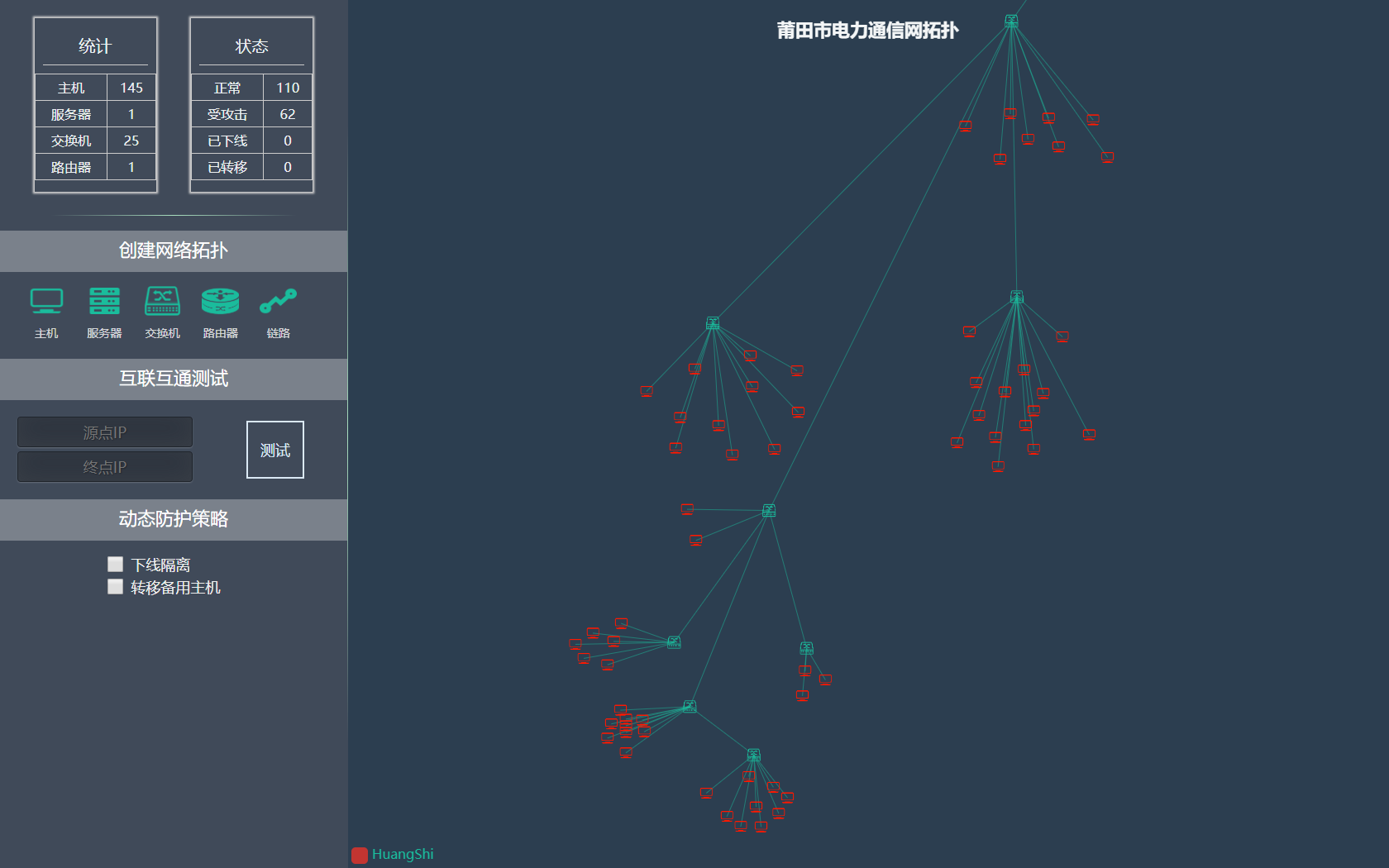


图 55.主机遭受病毒攻击

如上图所示，当通信网中的主机遭到病毒攻击时，系统对拓扑中相对应的主机节点标记为红色以此作为警示，随着病毒程序在通信网中的蔓延，系统会实时地标记出被检测出的受害主机。

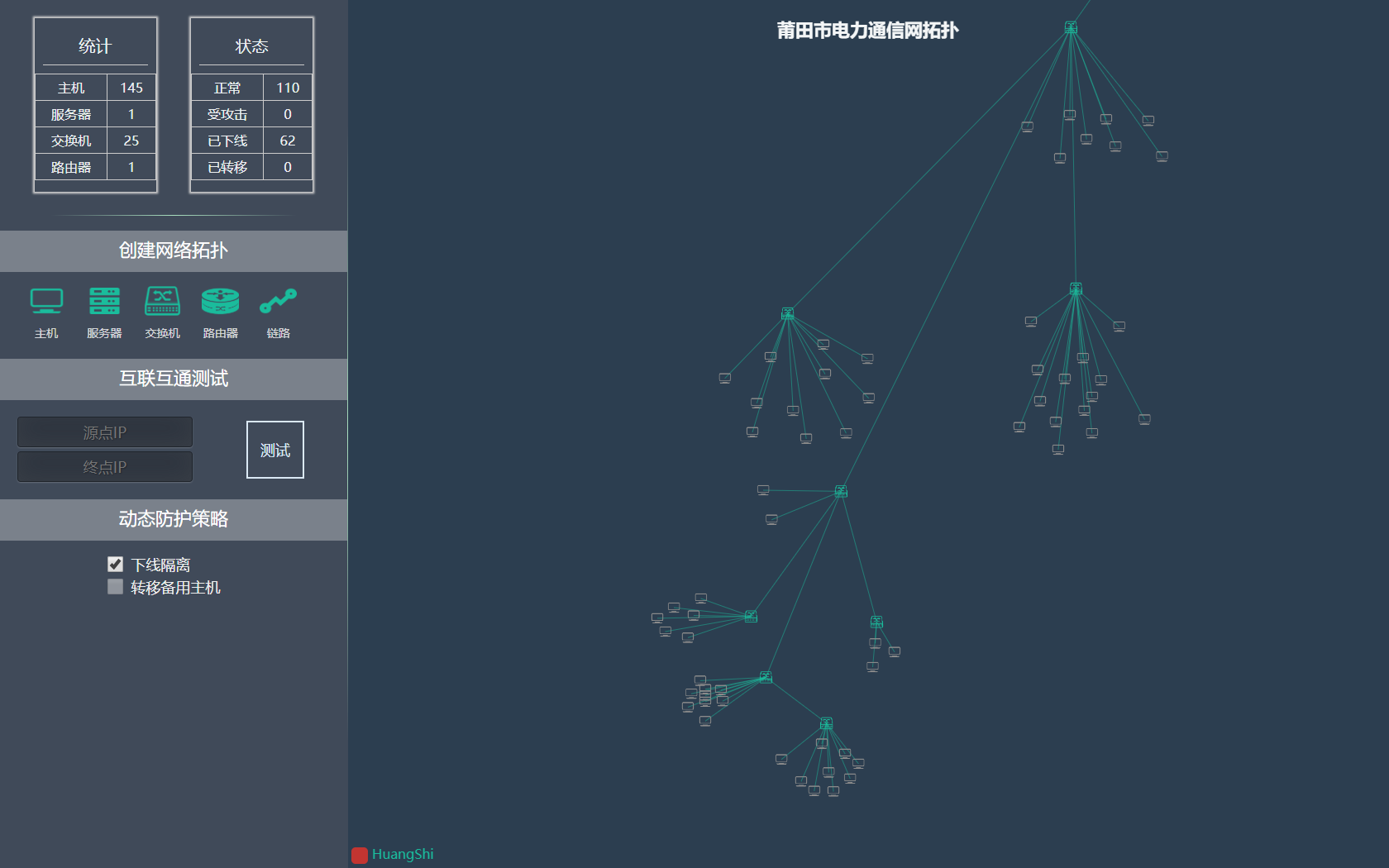


图 56.对受害主机进行下线隔离处理

如上图所示，用户通过启用下线隔离策略对所有受害主机进行下线隔离处理，拓扑界面上相应的主机则变成灰色，以示主机当前的下线状态。