

Design Flaws in IP Surveillance Cameras

Exploiting Web Interfaces

IP surveillance cameras are used extensively for monitoring of live targets. However, the inherent design of web interface of the IP surveillance cameras is not robust and is prone to security flaws.

What you will learn...

- Art of testing IP surveillance cameras
- Understanding the design flaws and attacks
- Will prove beneficial in application testing.

What you should know...

- Basic understanding of secure design practices
- Basic understanding of web attacks

IP surveillance cameras are used extensively for monitoring of live targets. However, the inherent design of web interface of the IP surveillance cameras is not robust and is prone to security flaws.

This paper sheds light on the vulnerabilities that exist in the design and deployment of web application interface of IP surveillance cameras. This paper is an outcome of the extensive testing of the deployed IP surveillance cameras in the live environment as a part of the open research.

Overview

IP surveillance cameras are used typically for surveillance purposes in organizations and public places. IP surveillance cameras play a critical role in providing evidence against crimes and unlawful activities. The surveillance technologies have become mature and sophisticated with the passage of time. The IP surveillance cameras are serving as an important defensive tool in today's environment as they help in:

- Enhancing public and employee safety thereby reducing the fear of crime
- Aiding the detection of crime
- Preventing crime by identifying potential criminal activity and anti-social behavior
- Helping police to respond more quickly to incidents



There is no doubt, IP surveillance cameras can serve as one of the best defense against crime. But, IP surveillance cameras are also prone to security vulnerabilities. In the last couple of years, we have noticed an explicit set of network attacks [1] on CCTV cameras such as injecting video streams [2] through ARP spoofing. In this paper, we will discuss the security vulnerabilities which are an outcome of the insecure design of IP surveillance cameras web interface. The insecure design makes them susceptible and hampers the desired functioning of IP surveillance cameras.

We will discuss the top design fallacies in the IP surveillance cameras web interface. We are not going to disclose information about any specific vendor in this paper but our aim is to provide security testing guidelines when a penetration tester is dealing with IP surveillance camera testing.

Cross Domain Image Streaming

Cross Domain Image Streaming (CDIS) is an attack in which an attacker exploits the web interface of IP surveillance camera to render another fake or non legitimate image stream from third party domain in the context of legitimate domain where the IP surveillance camera is hosted. This is based on the concept of *Remote File Inclusion* (RFI). However, as a matter of fact, in this attack a similar motion file is included from

the cross domain. In order to test this vulnerability in web interface of IP surveillance cameras, following steps are required and should be validated:

- The web interface should be completely debugged and HTTP request/ response should be dissected in order to understand the type of motion images that are rendered by the running server. For example: Content-Type HTTP header should be appropriately checked.
- It is required to look for the injection point, basically the image path from where the motion images are included remotely. We have found this vulnerability in one of the major vendors of IP surveillance cameras.
- The image path variable needs to be validated. It has been noticed that a number of web interfaces do not validate the input values which provide the attacker with a chance to include and replace the motion stream of his own choice.
- A typical motion picture recorder and editor are required to construct a similar set of picture as displayed by the IP surveillance camera.

This attack could be dangerous because an attacker might be able to record the non-legitimate motion pictures in the same format as accepted by the web interface and by exploiting the inherent remote file inclusion vulnerability to conduct successful CDIS attacks. These types of issues can hamper the surveillance capabilities because an attacker can leverage the attack to trick the victim to watch a similar non-legitimate motion picture.

Figure 1 shows the web interface showing legitimate motion images. Figure 2 shows non- legitimate images from the attacker domain running on the web interface of IP surveillance cameras.

In the above scenario, this flaw has been exploited using MJPG file format which is the most predominant motion picture format used in IP surveillance cameras.

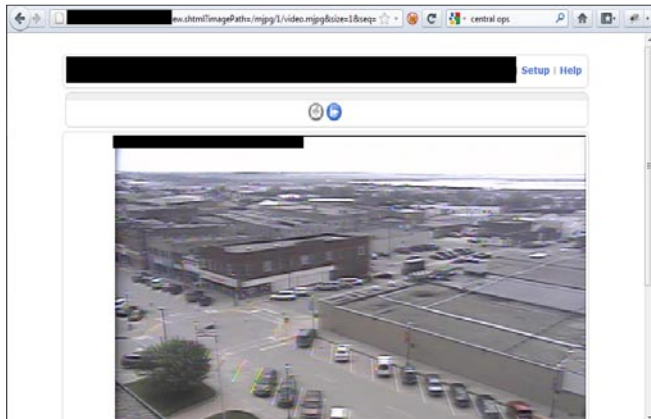


Figure 1. "ImagePath" parameter including legitimate motion images

Listing 1. CGI Scripts used in IP Surveillance Camera Web Interface

```

-/cgi-bin/password.cgi
-/cgi/maker/unittest.cgi?action=
-/cgi/maker/tools.cgi?command=
-/control/click.cgi?list| /img/image.cgi?next_
                                file=main_fs.htm
-/control/rotorcgi?help
-/en/help.cgi?ID=25 | /main_activex.cgi
-/cgi-bin/wg_login-act.cgi
-/CgiStart?page=Login&Language=0
-/cgi/b/users/usrpage/?nm=1
-/cgi-bin/csi_login-act.cgi
    
```

This flaw can be combined with more robust web attacks which makes it more serious in nature.

Insecure CGI Interface

Web interface used in IP surveillance cameras explicitly use the *Common Gateway Interface* (CGI) [3] which interacts with CGI scripts and CGI programs effectively. CGI scripts are used aggressively for executing web server programs dynamically. The deployment of IP surveillance cameras web interface requires content to be handled in a dynamic manner because images are received in a continuous stream of data. Generally, 85%-90% of the IP surveillance camera's web interface is designed in CGI. However, continuous testing has shown the result that CGI scripts used in the IP surveillance platform is not secure and can be used to conduct potential attacks. Remote command execution, authentication bypasses and authorization control manipulation are the common attacks that can be conducted easily on CGI interface. Inappropriate authentication logic such as presence of binary check on the client side has also shown such discrepancies in the real time environment. Listing 1

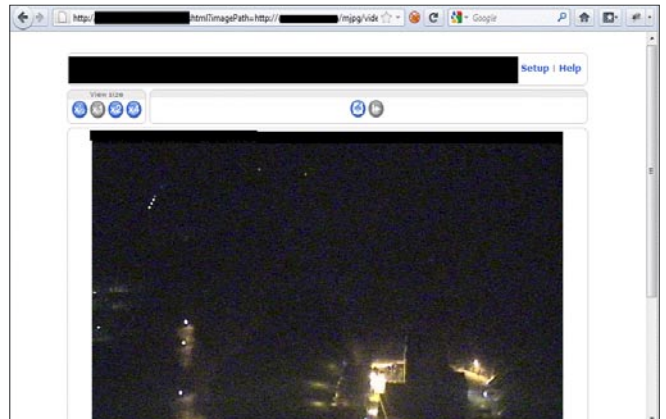


Figure 2. "Image Path" parameter including non-legitimate images from attacker's domain

shows some of the real time CGI scripts used in the IP surveillance camera's web interface.

The CGI scripts discussed in the Listing 1 provides a potential attack surface to execute OS commands on the remote server. Basically, *Server Side Injections* (SSI) can be executed easily.

As discussed earlier, inappropriate authentication and authorization logic are problematic issues. Figure 3 shows that there is binary (Yes/No) control implemented in authentication parameter that asks for basic authentication credentials. Attacker can use the existing knowledge about the web interface and manipulate the binary control logic. Figure 4 shows the final result if this kind of design flaw.

This type of insecurity in CGI design and control logic can be disastrous in a real time environment. The standard CGI exploits have been discussed here [4] which can be useful for testing purposes.

Decompiling and Dissecting Applets

Web interface is used to stream live images from IP surveillance cameras. It has been noticed that Java applets, flash files and Active X Controls are being used heavily to perform this operation. This means client browser has to install applets in order to run live images. This type of design has been followed by a number of IP surveillance camera vendors. As a matter of fact, the applets used in the web interfaces of IP surveillance cameras are not designed in a potential secure manner. As applets interact with client side environment, these applets are required to be decompiled to extract information. Potential testing of IP surveillance cameras web interfaces has resulted in the leakage of sensitive information such as session tokens, credentials and other hard coded information. It also leverages the execution logic of live streaming of images. Infact, it is hard to find that any applet is signed or has a unique signature which is required to preserve the integrity and privacy. While testing, it has

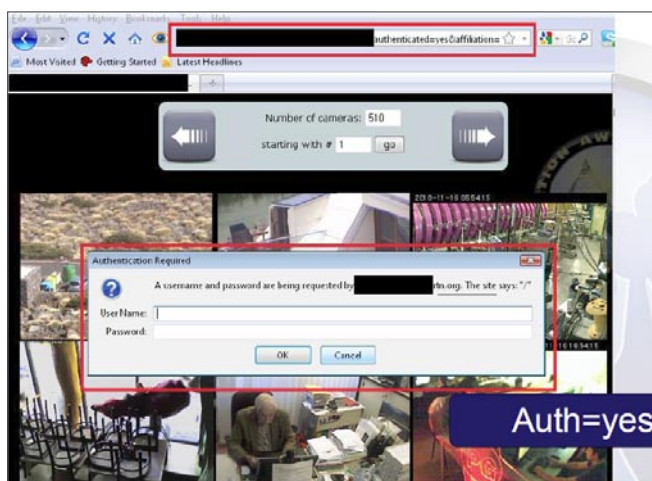


Figure 3. Insecure authentication and authorization logic

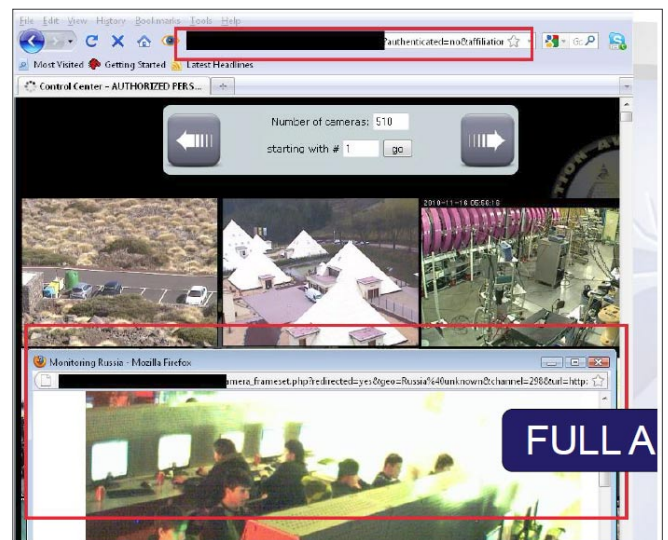


Figure 4. Complete access to web interface

been detected that a number of applets reflect back the credentials in the hidden field on the client side. On successfully decompiling the applets, the attacker can extract the credentials to perform non legitimate operations. Further, it is also possible to use the cached applets to run the same set of operations. The point of insecurity is the design of applets which might result in execution of different attacks. Figure 5 shows the critical information on client side.

Factory Defaults and Undocumented Credentials

Factory default credentials are used to provide a first time administrative control of the online devices. The same case is applied for IP surveillance camera administrative interfaces. Apart from web interfaces, the other administrative interfaces are telnet and ftp which are used to manage the remote firmware and OS running on devices. The credentials are not interface specific but work unanimously across all the administrative interfaces. Factory default credentials are easy and can be found on the respective IP surveillance camera's documents and on the internet. There is no doubt that the credentials are required to



Figure 5. Credentials reflected back on client side

be changed after the first installation and every time when an IP surveillance camera is configured. It has been noticed that it is possible to access the interfaces using factory default credentials in a number of deployed IP surveillance cameras in the real time environment. One of the most basic flaws that have been encountered during analysis is the fact that guest account credentials are stored as hidden elements on the client side and when ever user accesses the web interface form, auto complete is used to fill the *username* and *password* input boxes automatically. Figure 6 shows this kind of behavior by one of the IP surveillance camera web interface.

A number of issues have already been disclosed publicly about the presence of undocumented accounts [5] that are present on the operating systems used to run the content of IP surveillance camera through an interface. This design practice requires a complete change or certain standard guidelines in order to deploy the factory defaults principle in a robust manner.

Insecure Front End Design

Insecure front end design is one of the worst design practices that have been noticed in the web interfaces of IP surveillance cameras. Considering the general principle of security, a user should not be allowed to access any resource on the remote server until appropriate authentication is done and required authorization control is processed. This is not an applicable case in the real time environment because front end design is not secure in the case of IP surveillance cameras. For example: a user found a specific IP surveillance camera web interface running on port 80 on specific IP address and connecting to that IP address, one can easily access the running images. This has been noticed in a number of different IP surveillance cameras. On the other note, despite the open access to the running images, certain administrative operations are restricted using basic

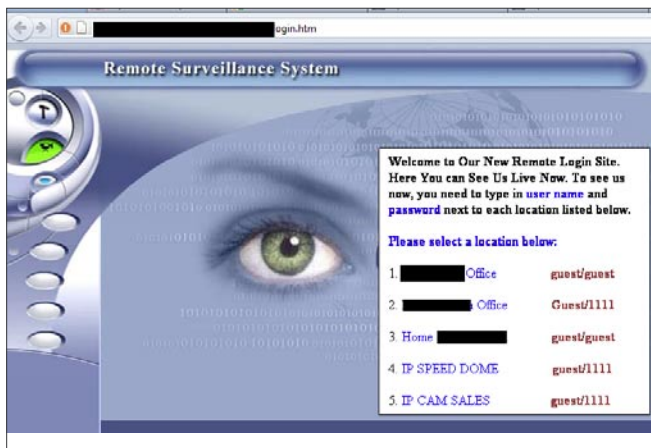


Figure 6. Weak and bad design practices

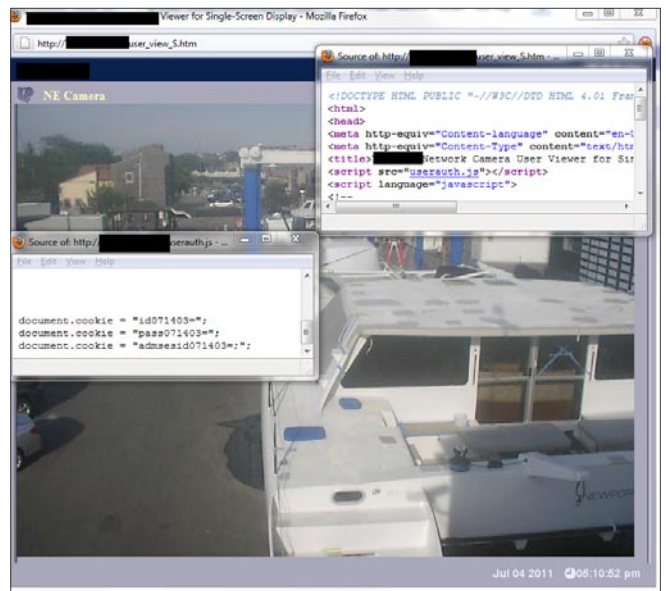


Figure 7. IP surveillance camera serving live images without authentication

authentication. Further, No IP restrictions are applied to determine the identity of the end point connection. Hence, a number of IP surveillance cameras are accessed out of the LAN and are publicly accessible. Overall, this design is not fully secured. This can lead to privacy breach and even if attacker is not able to perform administrative operations, it is still possible to monitor or spy on the organization, public and private assets. Figure 7 shows the potential bad front end design.

Invalidated Entry Points

Insecure design of web interface of IP surveillance cameras leads to potential web attacks such as *Cross Site Scripting (XSS)* and *Cross Site Request Forging (CSRF)* [6, 7 and 8]. These issues persist due to presence of entry points that are not validated. The user supplied data is not filtered which results in the

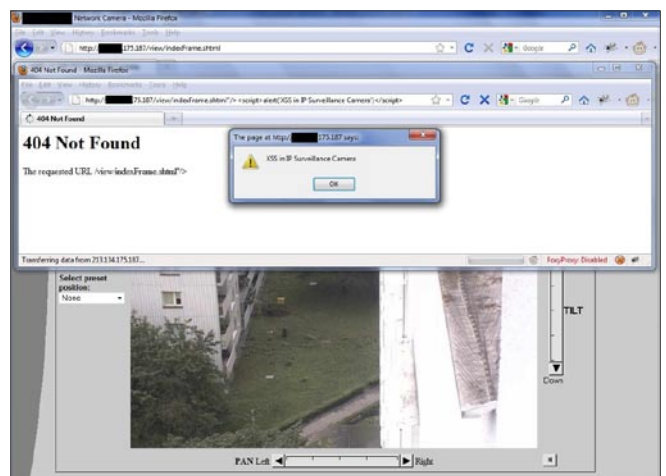


Figure 8. Successful XSS attack in web interface of IP surveillance camera

References

- Hacking CCTV's, http://events.ccc.de/congress/2005/fahrplan/attachments/692-slides_cctv_hacking.pdf [1]
- Security Camera Hack Conceals Heists Behind Dummy Video, <http://www.wired.com/threatlevel/2009/07/video-hijack/> [2]
- Dynamic Content with CGI, <http://httpd.apache.org/docs/1.3/howto/cgi.html> [3]
- CGI Abuses, http://www.donkboy.com/html/cgi_exploits.htm [4]
- Camtron CMNC-200 IP Camera Undocumented Default Accounts, <http://www.exploit-db.com/exploits/15507/> [5]
- Introduction to Cross Site Request Forgery, http://www.isecpartners.com/files/CSRF_Paper.pdf [6]
- Robust Defenses for Cross-Site Request Forgery, <http://seclab.stanford.edu/websec/csrf/csrf.pdf> [7]
- CSRF – It Still Works? <https://www.defcon.org/images/defcon-17/dc-17-presentations/defcon-17-bailey-mcree-csrf.pdf> [8]
- Owing Big Brother, http://www.procheckup.com/vulnerability_manager/documents/document_1258758669/Vulnerability_Axis_2100_research.pdf [9]

execution of non legitimate scripts in the context of web interface. The default design of web interface used in IP surveillance cameras does not validate the requests initiated by client and fails to append any unique identifier token information with the requests send by the users; this exposes the web interface of IP surveillance cameras to cross site request forgery attacks. A legitimate user or admin can be tricked easily to perform malicious actions unknowingly. Some of these vulnerabilities have already been reported by *ProCheckup* [9] which shows the successful execution of XSS and CSRF in the Axis cameras. These common attacks can be identified across many vendors that manufacture the IP surveillance cameras. Such web based vulnerabilities are easy to exploit and could lead to data and privacy breaches. Further, XSS can also be used to spread malware which is quite inappropriate considering the functionality of IP surveillance cameras. Figure 8 shows the successful XSS attack in web interface of the IP surveillance camera.

Unencrypted Parameters and Process Memory Dumping

Live process memory dumping is always a good testing strategy because it is possible to search sensitive information on the basis of general string patterns. Process dumping becomes more rustic when memory dump is large. It is a general design practice to send administrative credentials or software setup credentials hard coded in the installers. Usually, these are hard to find in the static installer but there is always a possibility of finding administrative credentials (generic strings such as *admin*, *guest* etc) from the active installer process memory. This is because a number of IP surveillance cameras do not use appropriate encryption modules which enable the credentials to flow in a clear text in the respective process memory.

In addition to this, IP surveillance cameras use basic authentication which is nothing but a base 64 encoded string and very few cases are implemented over HTTPS.

Conclusion

In this paper, we have presented the top notch design flaws in the IP surveillance cameras. The aim of discussing these issues is to raise a point against the insecure design practices that are followed by the vendors in their products such as IP surveillance cameras. The design bugs lead to security vulnerabilities. In order to secure the environment, the design and deployment of product should be executed in a secure manner. Every design should undergo regression testing and we must not deploy the security products in a default state, otherwise the risk is all yours.

ADITYA K SOOD

Aditya K Sood is a security researcher and PhD candidate at Michigan State University. He has already worked in the security domain for Armorize, COSEINC and KPMG. He is also a founder of SecNiche Security Labs, an independent security research arena for cutting edge computer security research. At SecNiche, he also acts as an independent researcher and security practitioner for providing services including software security and malware analysis. He has been an active speaker at industry conferences and already spoken at RSA, HackInTheBox, ToorCon, HackerHalted, Source, TRISC, AAVAR, EuSecwest, XCON, Troopers, OWASP AppSec USA, FOSS, CERT-IN, etc. He has written content for HITB Ezine, Hakin9, ISSA, ISACA, CrossTalk, Usenix Login, and Elsevier Journals such as NESE and CFS. He is also a co author for debugged magazine.

BIPIN GAJBHIYE

Bipin Gajbhiye is a security consultant at Cigital. He holds MS in Security Informatics from Johns Hopkins University. At Cigital, his job responsibilities include web application security assessments, thick client testing and source code reviews. His research interests include network security and developing new attack vectors for conducting efficient web application pentesting.



Harden your Network from the Inside Out



Network Access Control



Asset Vulnerability Management



Compliance Auditing and Reporting



www.netclarity.net

Available through Partners Worldwide