

#sicherheit

**Kritische Infrastrukturen vertrauen auf PSIM.
Wir haben es gehackt!**

BSKI 
Bundesverband für den Schutz
Kritischer Infrastrukturen e.V.


CONCEPTURE



Inhalt

- 1 Was sind KRITIS?
- 2 Sicherheitstechnik & PSIM
- 3 Sicherheitsanalyse Winguard
- 4 Finding 1:
Webserver
Denial of Service
- 5 Finding 2:
Winguard Service-Zugang
Passwort Brute force
Benutzer enumeration
- 6 Zusammenfassung



1

Was sind KRITIS?



Energie



Wasser



Ernährung



Gesundheit



Transport & Verkehr



Abfallentsorgung

Kritische Infrastrukturen



Medien & Kultur



IT & TK



Finanz- & Versicherungswesen



Staat & Verwaltung



Unternehmen mit bes. Bedeutung
(Rüstung, Gefahrstoffe etc.)



2

Sicherheitstechnik & PSIM

Häufig integrierte Systeme



- Videoüberwachung
- Bewegungsmelder
- Glasbruchsensoren
- Kartenleser (Zutrittskontrolle)
- Zaunsicherung (Detektoren)
- ELA / SAA
- Schranken & Poller
- Schließsysteme
- Fluchttürterminal
- Brandmelder
- Intercom
- PNA (Personennotruf)
- Klima, Lüftung, Heizung
- Jalousiesteuerung
- Lichtsteuerung
- Aufzugssteuerung
- Gebäudeleittechnik
- Produktionstechnik
- Maschinen- und Anlagentechnik



2

Sicherheitstechnik & PSIM

Die Sicherheitsleitstelle



Unterschiedlichste
Sicherheitssysteme
werden zentral
überwacht und
gesteuert



2

Sicherheitstechnik & PSIM

Sicherheitsmanagement in der Peripherie



...das kann auch
mal so aussehen





3 Sicherheitsanalyse Winguard



 **winguard**
always retain control ist ein herstellernerutrales Gefahrenmanagementsystem (GMS/PSIM) der Firma Advancis

Als zentrales System für kritische Prozesse, wie z.B. Zutrittskontrollen oder der Betrieb der Brandmeldeanlage hat Winguard einen erhöhten IT-Sicherheitsschutzbedarf

In der Werbebroschüre wird das aufgegriffen, so ist z.B. die Rede von:

- *Eingesetzter Verschlüsselung (TLS und AES)*
- *Möglichkeit der Redundanz*

Um diese Aussagen zur Sicherheit zu prüfen, wurde WinGuard auf mögliche Schwachstellen untersucht.

- *Dabei wurden Sicherheitsprobleme, die nicht praxisrelevant sind ignoriert (z.B. information disclosure <https://cwe.mitre.org/data/definitions/200.html>)*



3 Sicherheitsanalyse Winguard

Versuchsaufbau

Die Herausforderung besteht in einer möglichst realistischen Nachstellung eines Angriffs

- Blackbox Test
 - ▶ Der Angreifer hat keine Kenntnis über den Quellcode
- Nutzen der öffentlich verfügbaren Dokumentation
- Lokale WinGuard Installation
 - ▶ Ein Angreifer wird kein großes Testnetzwerk mit verschiedensten Schnittstellen zur Verfügung haben

Prämisse: Der Angreifer kann mit dem WinGuard Server kommunizieren.



3 Sicherheitsanalyse Winguard

Erster Ansatz

Untersuchung der verschiedenen Module, die WinGuard unterstützt, jedoch:

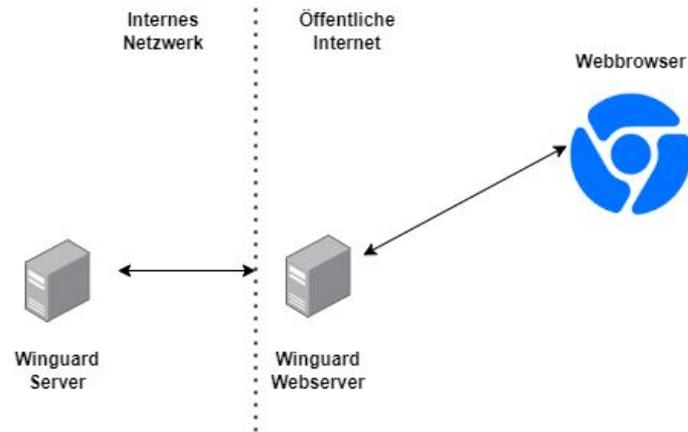
- Mehr als 100 Module für verschiedenste Systeme
 - Genutzte Module variieren stark
 - Ein Angreifer müsste für den Großteil der Module über eine Sicherheitslücke verfügen, um flexible Winguard Installationen anzugreifen
-  Daher Konzentration auf Schnittstellen zur Steuerung von Winguard (Webserver, Serviceschnittstelle etc.)



4 Finding 1: Webserver

Erstes Ziel: Untersuchung des WinGuard Webserver für die WinGuard Webapp
Grund: Webserver sind klassischerweise nach außen exponiert und ein Angreifer könnte daher Schaden anrichten (ohne im internen Netz zu sein).

Versuchsaufbau



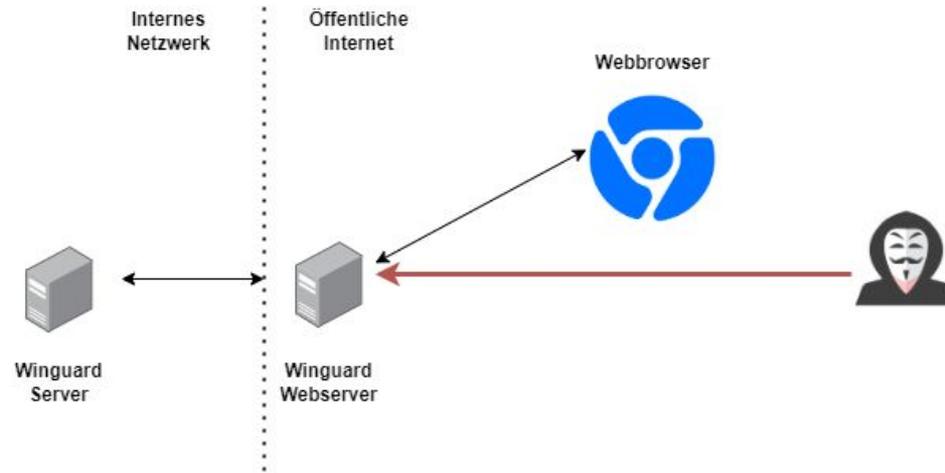
Winguard Server und Winguard Webserver sind voneinander getrennte Prozesse. Diese können z.B. auf getrennten Systemen betrieben werden.



4 Finding 1: Webserver

Vorgehen

Ein Angreifer würde also nicht direkt den Winguard Server angreifen, sondern den Winguard Webserver





4 Finding 1: Webserver



Ziel: Winguard Webserver

- Webserver liefert wie gewöhnlich Dateien für die WebApp aus JS/HTML/CSS...
- Zusätzlich stellt der Webserver einen "Websocket" Server bereit
- Die Webanwendung wird also durch den Webserver ausgeliefert
Anschließend kommuniziert die Webanwendung dann mit dem "Websocket" Server. Beispiel:

```
{"method":"login","params":{"username":"Benuter1",  
"password":"Geheim","language":"de","workstation":  
"Web","type":"web"},"Id":3}
```

The screenshot shows a login interface for Winguard. At the top left is the Winguard logo, a stylized 'W' in a circle, followed by the text 'winguard' and the tagline 'always retain control' below it. Below the logo are four input fields: a text field for the username 'Benuter1', a password field with masked characters '*****', a dropdown menu currently showing 'Webclient', and a checkbox labeled 'Angemeldet bleiben'. At the bottom of the form is a blue button with the text 'Anmelden'.



4 Finding 1: Webserver - Denial of Service

Denial of Service über Websocket

- Ist der Webserver nicht erreichbar erhält der Bediener keinen Zugriff auf Winguard
- “Klassischer” Ansatz
 - ▶ Angreifer sendet Millionen von nutzlosen Anfragen
 - ▶ Web Server ist ausgelastet, kein Zugriff mehr für reguläre Benutzer möglich.

Dabei gibt es für den Angreifer aber folgende Probleme:

- Der Angreifer muss über eine sehr gute Netzwerkanbindung verfügen (besser als das angegriffene System)
- IP des Angreifers kann einfach gesperrt werden (z.B. in der Firewall)
 - ▶ Auslastung geht sofort wieder auf 0 und der Webserver ist für Benutzer wieder normal erreichbar



4 Finding 1: Webserver - Denial of Service



Denial of Service über Websocket

- Es findet sich jedoch ein Fehler in der Websocket Funktion:
{"method":"GetSettings","params":{"keys":[]...
- Der Winguard Server führt den Befehl wie folgt aus:

```
foreach (string key1 in keys)
{
    string key = key1;
    AppSetting appSetting = this.WebCfg.AppSettings.Where<AppSetting>((Func<AppSetting, bool>) (g => g.Name.ToLower() == key.ToLower())).FirstOrDefault<App
    if (appSetting != null && (!appSetting.AuthorizationRequired || this.IsAuthorized))
```

- Das Problem ist hier, dass kein Length-Check des Parameters "Keys" vorgenommen wird. Dadurch ist ein Denial of Service möglich.



4 Finding 1: Webserver - Denial of Service



Denial of Service über Websocket

- Sendet der Angreife viele leere Keys werden diese ohne Überprüfung durch WinGuard in den Speicher geladen und dort vorgehalten
 - Aufgrund der Winguard Speicherstruktur genügt es, wenn der Angreifer wenige MB Daten sendet um den Speicher komplett zu füllen
- Die anfänglich beschriebenen “klassischen” Probleme eines Denial of Service werden aus der Sicht des Angreifers damit gelöst
 - Netzwerkanbindung des Angreifers spielt nun keine Rolle mehr (Die wenigen MB Daten lassen sich auch z.B. über das Mobilfunknetz übertragen)
 - IP Sperrung des Angreifers löst das Problem nicht, da die Daten einmalig gesendet wurden, folglich bleibt das WinGuard System ausgelastet.



4 Finding 1: Webserver - Denial of Service



Denial of Service über WebSocket, praktisches Beispiel

Auslastung im Normalbetrieb:

Name	Status	CPU	Arbeits...	Datenträ...	Netzwerk	GPU	GPU-Modul	Stromverbrauch	Stromverbrau...
> Webserver		0%	15,4 MB	0 MB/s	0 MBit/s	0%		Sehr niedrig	Sehr niedrig

- Sehr geringe Systemauslastung, nur wenige MB Arbeitsspeicher (Auf dem Testsystem stehen ~ 4000mb zur Verfügung)
- Kaum CPU Last

Nach einem Angriff:

Name	Status	CPU	Arbeits...	Datenträ...	Netzwerk	GPU	GPU-Modul	Stromverbrauch	Stromverbrau...
> Webserver		71,6%	4.172,3 MB	0 MB/s	0 MBit/s	0%		Sehr hoch	Hoch

- Angreifer sendete einmalig weniger als 300mb Daten, System ist einige Minuten sehr ausgelastet



5 Finding 2: WinGuard Service Zugang

WinGuard Service Zugang

Neben dem Webserver, gibt es noch einen Service-Zugang, der während der Suche nach Schwachstellen aufgefallen ist.

Ist diese Schnittstelle in der Konfiguration aktiviert, ergaben sich mehrere Sicherheitsprobleme

Service-Zugang	
Aktivieren	Ja
Port	13377



5 Finding 2: WinGuard Service Zugang - Passwort Brute Force



Passwort Brute Force

- Über WinGuard Einstellungen lässt sich ein Schutz vor Brute-Force-Angriffen (Durchprobieren von Passwörtern <https://cwe.mitre.org/data/definitions/307.html>) einstellen
- Service Zugang scheint auf den ersten Blick uninteressant für Angreifer
 - Monitoring Interface
 - Ob man sich als Admin oder User anmeldet macht keinen Unterschied
 - Einstellungen können nicht geändert werden etc.
 - Wahrscheinlich daher keine großen Schutzmaßnahmen wie beim Client Login



5 Finding 2: WinGuard Service Zugang - Passwort Brute Force



Passwort Brute Force

- Jedoch **keine** speziellen Nutzer für den Servicezugang!
- Daher die Idee:

Brute Force der Passwörter über den Servicezugang



Sicherheitsproblem!

Brute-Force-Schutz greift **nicht** beim Servicezugang!



5 Finding 2: WinGuard Service Zugang - Passwort Brute Force



Winguard Service Zugang - Protokoll

- Der Service Zugang nutzt ein eigenes Protokoll, welches nicht dokumentiert ist
- Bitweise Analyse der übertragenen Daten:
 - Keine Verschlüsselung eingesetzt!
 - Protokoll wurde soweit nachvollzogen, bis ein Login versuch möglich war

The screenshot shows a Wireshark interface with a packet capture of a login attempt. The packet list pane shows a frame of 208 bytes on wire. The packet details pane shows the data field containing a hex string: 00e304009c00000469a47020100480400000004b0203004c0201004d0201004e020000... [Length: 164]

```
0000 02 00 00 00 45 00 00 cc 9a 6e 40 00 80 06 00 00  ....E...ng.....
0010 7f 00 00 01 7f 00 00 01 34 41 10 c1 e9 b9 c4 9b  ....p...4A.....
0020 6d ea e7 f0 50 18 27 f9 5f dc 00 00 00 e3 04 00  m...p...H.....
0030 5c 00 00 00 46 9a 47 02 01 03 03 04 00 00 00 00  ....F...H.....
0040 4b 02 03 00 4c 02 01 00 4d 02 01 00 4e 02 00 00  K...L...H...N...
0050 4f 02 01 00 49 5e 43 00 3a 00 5c 00 50 00 72 00  O...I...C...:..P...
0060 6f 00 67 00 72 00 61 00 6d 00 20 00 46 00 69 00  g...r...a...m...f...i...
0070 6c 00 65 00 73 00 5c 00 42 00 64 00 76 00 61 00  I...e...s...:..A...d...r...a...s...
0080 6e 00 63 00 69 00 73 00 5c 00 57 00 69 00 6e 00  n...c...i...s...:..W...h...i...n...
0090 47 00 75 00 61 00 72 00 64 00 5c 00 57 00 69 00  G...u...a...r...d...:..e...
00a0 6e 00 47 00 75 00 61 00 72 00 64 00 2e 00 65 00  n...G...u...a...r...d...:..e...
00b0 78 00 63 00 4a 1a 4c 00 53 00 4f 00 45 00 4b 00  x...e...r...i...:..S...D...E...K...
00c0 45 00 46 00 45 00 4c 00 44 00 2d 00 4e 00 42 00  E...F...E...L...D...:..N...B...
```

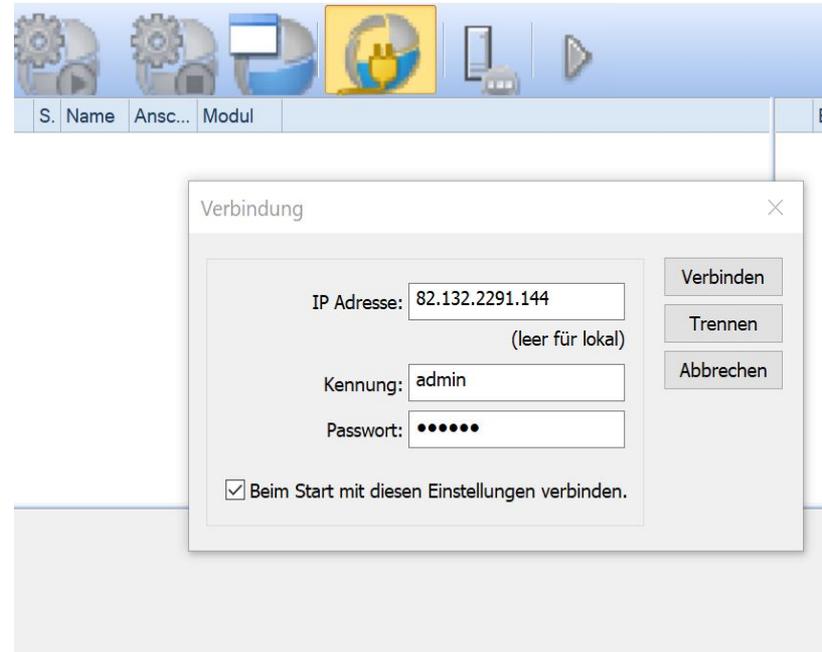


5 Finding 2: WinGuard Service Zugang - Passwort Brute Force



Passwort Brute Force

- Fehlende Verschlüsselung ermöglicht es einem Angreifer Passwörter im Netzwerk einfach mitzulesen!
- Die Verschlüsselung lässt sich auch nicht aktivieren (ist nicht vorgesehen, siehe Screenshot der Service Panel Oberfläche)
- Verbreitete Netzwerk Sniffer wie z.B. Wireshark genügen um das Passwort im Klartext zu erhalten!





5 Finding 2: WinGuard Service Zugang - Passwort Brute Force



Passwort Brute Force

- Auf Basis der Protokollanalyse, wurde ein kleines Programm entwickelt, welches Passwörter für einen Benutzer-Account ausprobiert
- Pro Sekunde sind einige tausend Passwort-Versuche möglich
 - Vier Gleichzeitige Verbindungen werden zum Server aufgebaut, je nach Netzwerkverbindung kann diese Zahl noch erhöht werden
 - Abhängig von der Position des Angreifers im Netzwerk
- Wortlisten mit Passwörtern können mit bekannten Programmen auf das

Ziel zugeschnitten werden

- crunch
- cewl

```
C:\Users\I.sökefeld\Desktop\WinGuard>python3 Service_Zugang_Bru
6407 versuche pro Sekunde Von Wordlist: 5%
6626 versuche pro Sekunde Von Wordlist: 9%
6562 versuche pro Sekunde Von Wordlist: 14%
7025 versuche pro Sekunde Von Wordlist: 19%
6918 versuche pro Sekunde Von Wordlist: 24%
7233 versuche pro Sekunde Von Wordlist: 29%
7451 versuche pro Sekunde Von Wordlist: 34%
7168 versuche pro Sekunde Von Wordlist: 40%
```



5 Finding 2: WinGuard Service Zugang - User Enumeration

Servicezugang: User Enumeration

Effektives Passwörter raten über den Servicezugang ist nun möglich



Doch was wenn der angegebene Benutzer nicht existiert?



5 Finding 2: WinGuard Service Zugang - User Enumeration



Beobachtung

- WinGuard verhält sich unterschiedlich bei Paketen mit gültigen und ungültigem Benutzernamen
- Beispiel:
 - 1. Packet [Benutzer: root , Passwort: dummy] → Antwort: 24 Bytes
 - 2. Packet [Benutzer: toor, Passwort: dummy] → Antwort: 24 Bytes
 - 3. Packet [Benutzer: admin , Passwort: dummy] → Antwort: 1024 Bytes
- Bei gültigem Benutzer (Hier admin) ist die Antwort länger als bei nicht vorhanden Benutzern (root und toor)
- Auf der Winguard Seite scheint Zeit ein Faktor zu sein, daher ist das vorgehen nicht komplett deterministisch
 - Bei Tests mit einigen 100 Benutzernamen gab es jedoch eine Treffsicherheit > 90%



5 Finding 2: WinGuard Service Zugang - User Enumeration

Entwicklung eines Programms zur User enumeration

Programm bekommt Benutzernamen übergeben und prüft ob dieser vorhanden ist

- kann prüfen, ob z.B. der Benutzer: admin existiert
- kann mit einer Wordlist Benutzernamen herausfinden

```
C:\Users\l.sökefeld\Desktop\WinGuard>python3 user_enumeration.py  
Benutzer: 'root' nicht vorhanden!  
Benutzer: 'toor' nicht vorhanden!  
Benutzer: 'admin' vorhanden!
```



6 Zusammenfassung

- Die Analyse hat gezeigt, dass für einen Angreifer praxisrelevante Sicherheitslücken in WinGuard gibt.
- Abhängig von der genutzten Konfiguration (Wird der Webserver genutzt? Ist der Service Zugang aktiviert?) lässt sich entweder der Betrieb von WinGuard massiv stören (Denial of Service) oder schlimmer noch der Angreifer kann Login Daten erhalten (Entweder durch mitlesen des Netzwerkverkehrs oder durch ausprobieren).



Als sofortige Reaktion, darauf sollte die aktuell WinGuard Version eingespielt werden



6 Zusammenfassung



Was ist dann passiert?

- Schnelle und professionelle Reaktion des Herstellers
- Behebung der Lücken in der aktuellen Version

The screenshot shows the Advision release notes interface. At the top, there is a search bar with the Advision logo on the left. The search criteria include: Module: (empty), Typ: Alle, Text: (empty), von: X4 B16.5 (16.08.2022), bis: (empty), Einträge: (empty), and Objekt: (empty). Below the search bar, the title "Release Notes (8.4.16.5 - *)" is displayed. The main content area is divided into sections:

- WinGuard**
 - Feature**
 - 8.4.17.6: Bei wiederholt fehlerhaftem Login wird der Zugang gesperrt.
 - 8.4.17.3: Von Schnittstellen geänderte Properties von Objekten werden nun auch in der Datenbank persistiert,
 - 8.4.17: Die Performanz der UI der automatischen Projektierung für AOP-Schnittstellen wurde verbessert.
 - 8.4.17: Die tschechische Sprachdatei wurde überarbeitet und aktualisiert.
 - 8.4.17: Eine Option zur Bereinigung der Vererbung bei Meldern wurde bei der Wartung hinzugefügt.
- Addons**
 - Feature**
 - WebServer 8.4.17.1: Der Schutz vor DoS-Attacken wurde verbessert.
 - ServicePanel 8.4.17.1: Die Verschlüsselung der Kommunikation via TLS wird nun unterstützt.
 - Localizer 8.4.17.1: Eine Validierung der Id-Spalte wurde hinzugefügt.
 - BACnetStack 8.4.17.1: Es können nun mehrere BACnetStack-Instanzen gleichzeitig gestartet werden.
- SST**
 - Feature**
 - MIPS DK 8.4.17.3: Ab XProtect 2021 R2 ist es möglich, alle Daten über eine verschlüsselte Verbindung zu übertragen.
 - MIPS DK 8.4.17.2: SDK 2021 R2 wird nun unterstützt.

Fragen?

