A vulnerability assessment of smart vacuums

Abigail Baker, Dakota State University

Mentor: Laura Ann Anderson, Oak Ridge National Laboratory

ABSTRACT

Smart vacuums are commonplace in America. Many homes have one due to their ability to keep the house clean and free of dirt, crumbs, and pet hair with little effort from the user. Part of the Internet of Things (IoT), these devices, while convenient, are often not designed with security measures in place. If security measures are included in the device, these features are often minimal and easily circumvented. This report explores the security of smart vacuums from two companies, iRobot and Eufy, using three different tests. The vacuums used were the iRobot Roomba Combo i5+, iRobot Roomba Vac Essential, and Eufy G30 SES Robovac. To determine the security of each device, a Wireshark capture was performed while the device was setting up, running, and idling overnight. These captures revealed the Vac Essential communicated the most with outside sources while the Robovac communicated the least with outside sources. The next test performed was an Nmap scan. Both a Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) scan were conducted on each device. It was discovered the Vac Essential had three ports open, the Robovac two, and the Combo i5+ one port open. The last test run was performing a deauthentication attack and measuring the time it took for the vacuum to recover. This test revealed the Combo i5+ took the longest to recover and crashed the most often, while the Robovac recovered the quickest. Neither the Vac Essential nor the Robovac crashed throughout the test. These results indicate the Eufy G30 SES Robovac is the most secure smart vacuum of the three tested, and the iRobot Roomba Combo i5+ is the least secure.

INTRODUCTION

Smart vacuums can be found in many homes throughout America. The popularity of these devices has continued to grow due to their ability to keep a house clean of crumbs, pet hair, and dirt with little effort from the user. The prevalence of smart vacuums means a review of their security is necessary. Smart vacuums are part of the Internet of Things (IoT), devices that connect to the internet to complete their work but do not need human interaction to run properly. IoT devices provide ease of use to people, but they often have little to no security on them. This lack of security means the devices can be easily exploited by malicious actors. For example, iRobot, the manufacturer of Roomba vacuums, recently had a privacy breach, which led to photos of consumers being posted online.[1] Events like this one emphasize the need for increased security in these smart home cleaners.

To that end, this paper explores several avenues of exploitation for two brands of smart vacuums, iRobot's Roomba vacuums and Eufy's Robovacs. These vacuums were evaluated for vulnerabilities via three different methods, and the results were examined to determine the security of the device. Challenges are addressed throughout the paper and next steps for research are also provided.

SETUP

For these smart vacuums to work, they need to be run on the 2.4 GHz network and WPA3 encryption (if available) must be turned off on the router. Additionally, the iRobot Home app requires iOS 15.0 and newer or Android OS 9.0 and newer to work properly. Eufy's Clean app requires iOS 10.0 and newer or Android OS 5.0 and newer to run correctly. All the vacuums can be used without the app: each robot has three buttons on the top of itself that can start

cleaning, send the machine home, and spot clean specified areas. However, the app allows the user to create a schedule of cleaning times, locate their machine, and control the machine remotely, amongst other features. This leads most users to use the vacuum in combination with the app, as it is more convenient.

In our setup, we worked with the iRobot Roomba Combo i5+ (nicknamed Annie), the iRobot Roomba Vac Essential (nicknamed Destroyer), and the Eufy Clean G30 SES (nicknamed King Pin). A Kali Linux machine and Windows HP machine were used to perform the research and analyze results. The companion apps were installed on a Samsung Galaxy A12 phone running Android 10. To perform the assessment of the vacuums Wireshark, Nmap, and a deauthentication script were used.

Each app leads the user through the setup process of the vacuum. Interestingly, both King Pin and Annie created their own Wi-Fi access point (AP) during setup. The phone had to connect to these APs to complete the process, but the vacuum returned to client mode after setup was done. The setup process takes several minutes for each machine and requires the user to have physical contact with the vacuum for parts of it. The physical contact provides a layer of protection against a remote user setting up the vacuum without the owner knowing.

Once connected to the app and home network, the vacuum is ready to run. Both Annie and King Pin need to complete a mapping run which allows them to create a map of the area. This initial run can take multiple hours, depending on the size of the space. During this time, the robot does not clean. Instead, it runs around the area to find walls, stairs, and other obstacles to make an internal map that will then guide it during future use. Upon completion of the mapping run, the robot returns to its home base to charge and save the map. The saving

process can take a few minutes, during which the robot is unavailable to use. Destroyer does not save a map of the space, so this step is not required.

Before the vulnerability assessment could begin, the Media Access Control (MAC) address of each vacuum needed to be discovered. The Eufy app provided the MAC address for King Pin, and Annie's address was easily discoverable via Wireshark captures. However, Destroyer's MAC address was much more difficult to discover. When looking for Destroyer's address, there were several possible addresses that could have been his. Using a proprietary tool to monitor network traffic, a potential Destroyer address was isolated via a deauthentication attack, and commands were sent to the vacuum to see if the chosen address was correct. However, each attempt to isolate the address and confirm its owner resulted in failure. When sending the malicious packets (encapsulated data sent over the network), the robot continued to work properly. This led us to believe Destroyer may be immune to deauthentication attacks. After many days of trial and error, a colleague was able to determine the correct address for Destroyer using the tool mentioned above. During this same time, we discovered Destroyer was sending hundreds of packets over the network, even when he was not running.

VULNERABILITY ASSESSMENT

Wireshark captures

After all three addresses were confirmed, the vacuums were sent on cleaning runs. During each run, a Wireshark capture was performed using Kali Linux running on a ThinkPad X1 Carbon. In order to capture the traffic, the computer's wireless card needed to be put into "Monitor" mode. This was achieved via:

```
sudo service NetworkManager stop
sudo ifconfig wlan0 down
sudo iwconfig wlan0 mode Monitor
sudo ifconfig wlan0 up
```

Monitor mode allows the wireless card to pick up all traffic on the network, regardless of the destination address. Once completed, Wireshark can then capture traffic sent from the vacuum and to the vacuum.

Annie

A Wireshark capture was run overnight (approximately 18 hours), and it was discovered Annie reached out to 27 Internet Protocol (IP) addresses and received replies from 25 IP addresses (Figure 1). All the replies were from IP addresses Annie had reached out to first. She sent out 2,913 packets and received 2,013 packets back. Most IP address Annie reached out to were Amazon related, including Amazon Technologies. However, there were four IP address associated with non-Amazon entities: Ovh Us LLC, Choopa LLC, Weber State University, and IONOS Inc.

Packets From Outside Sources to Annie					
From	Packets	Bytes	From	Packets	Bytes
3.162.174.24	77	50	54.172.206.226	112	51
3.162.174.80	15	11	54.175.181.213	116	52
3.162.174.103	39	25	54.175.183.244	125	53
3.162.174.121	14	10	54.230.202.65	14	9,213
51.81.226.229	67	11	108.61.73.243	66	11
52.7.220.131	122	53	108.159.227.18	36	21
52.23.126.227	101	44	108.159.227.58	42	28
52.86.130.167	145	60	108.159.227.110	56	36
52.207.135.141	114	51	137.190.2.4	65	11
54.157.110.53	110	48	192.168.8.1	98	33
54.160.183.37	107	50	198.71.50.75	66	11
54.161.243.21	51	9,347			
54.161.252.77	124	53			
54.164.245.77	131	54			

Packets From Annie to Outside Sources					
To	Packets	Bytes	To	Packets	Bytes
3.162.174.24	92	19	54.164.245.77	194	40
3.162.174.80	15	3,150	54.172.206.226	184	38
3.162.174.103	45	9,450	54.175.181.213	198	41
3.162.174.121	16	3,821	54.175.183.244	195	37
51.81.226.229	71	12	54.230.202.65	18	6,081
52.7.220.131	202	42	108.61.73.243	74	13
52.23.126.227	163	35	108.159.227.18	50	18
52.86.130.167	198	41	108.159.227.58	51	17
52.207.135.141	191	40	108.159.227.110	68	23
54.47.150.40	1	185	137.190.2.4	70	12
54.157.110.53	188	39	192.168.8.1	91	18
54.160.183.37	185	39	198.71.50.75	74	13
54.161.243.21	70	13	255.255.255.255	5	2,215
54.161.252.77	204	42			

Figure 1. Conversations Annie participated in, including number of packets and number of bytes sent

Destroyer

A Wireshark capture was run overnight (approximately 15 hours), and it was discovered Destroyer reached out to 58 IP addresses and received replies from 56 IP addresses (Figure 2). All the replies were from IP addresses Destroyer had reached out to first. He sent out 5,853 packets and received 3,822 packets back. The majority of IP address Destroyer reached out to were Amazon related, including Amazon Technologies and Amazon Data Services NoVa. However, 18 of the addresses were not related to Amazon, including Comcast Cable Communications LLC, Akamai Technologies Inc, Oracle Corporation, and Cloudflare Inc.

Packets From Outside Sources to Destroyer											
From	Packets	Bytes	From	Packets	Bytes	From	Packets	Bytes	From	Packets	Bytes
3.162.112.67	22	17	52.44.111.78	51	8,473	54.165.245.245	156	38	108.157.150.114	32	23
3.162.174.24	133	129	52.44.250.222	56	9,483	54.174.171.67	77	21	108.159.227.18	184	151
3.162.174.80	28	24	52.55.35.207	149	45	54.197.98.38	54	11	108.159.227.30	40	34
3.162.174.103	60	51	52.86.41.150	157	28	54.204.123.219	144	33	108.159.227.58	98	83
3.162.174.121	148	146	52.86.78.2	78	18	54.205.123.132	102	25	108.159.227.110	35	33
5.161.111.190	1	176	52.86.124.109	144	37	54.208.76.39	140	25	142.202.190.19	1	176
18.164.96.73	26	24	52.206.4.112	68	10	54.210.234.40	151	39	148.135.68.31	2	352
18.238.55.66	21	17	54.144.10.161	140	42	54.230.202.9	21	15	152.70.159.102	3	528
18.238.55.88	24	20	54.144.200.28	64	10	54.235.60.4	50	11	162.159.200.123	4	704
44.221.224.84	145	26	54.146.50.140	173	50	65.8.228.70	16	10	192.168.8.1	314	129
45.63.54.13	2	352	54.147.63.38	76	18	65.100.46.166	1	176	198.71.50.75	1	176
50.205.57.38	2	352	54.152.27.128	40	13	69.164.213.136	1	176	198.137.202.32	2	352
51.81.226.229	1	176	54.152.192.87	50	12	74.6.168.72	2	352	204.2.134.163	1	176
52.23.12.202	180	42	54.157.29.131	149	32	74.208.117.38	1	176	205.233.73.201	1	176

Packets Sent From Destroyer to Outside Sources											
To	Packets	Bytes	To	Packets	Bytes	To	Packets	Bytes	To	Packets	Bytes
3.162.112.67	18	7,849	52.44.250.222	81	13	54.197.98.38	161	38	108.159.227.30	34	15
3.162.174.24	65	12	52.55.35.207	167	38	54.204.123.219	169	39	108.159.227.58	69	30
3.162.174.80	23	4,634	52.86.41.150	181	40	54.205.123.132	174	42	108.159.227.110	52	23
3.162.174.103	35	6,638	52.86.78.2	170	39	54.208.76.39	166	39	142.202.190.19	1	176
3.162.174.121	80	15	52.86.124.109	178	40	54.210.234.40	181	40	148.135.68.31	3	528
5.161.111.190	1	176	52.206.4.112	164	41	54.230.202.9	17	7,709	152.70.159.102	2	354
18.164.96.73	16	7,571	54.144.10.161	130	36	54.235.60.4	171	39	157.245.125.229	2	352
18.238.55.66	12	2,258	54.144.200.28	167	38	65.8.228.70	12	2,258	162.159.200.123	6	1,056
18.238.55.88	22	4,236	54.146.50.140	173	40	65.100.46.166	1	176	192.168.8.1	1,285	225
44.221.224.84	173	40	54.147.63.38	174	40	69.164.213.136	1	176	198.71.50.75	2	352
45.63.54.13	3	528	54.152.27.128	188	45	74.6.168.72	3	528	198.137.202.32	4	704
50.205.57.38	2	352	54.152.192.87	166	36	74.208.117.38	2	352	204.2.134.163	2	352
51.81.226.229	1	176	54.157.29.131	173	39	104.194.8.227	1	176	205.233.73.201	1	176
52.23.12.202	178	40	54.165.245.245	175	39	108.157.150.114	17	7,709			
52.44.111.78	74	12	54.174.171.67	169	38	108.159.227.18	155	72			

Figure 2. Conversations Destroyer participated in, including number of packets and number of bytes sent

King Pin

A Wireshark capture was run overnight (approximately 16 hours), and it was discovered King Pin reached out to ten IP addresses and received replies from nine IP addresses (Figure 3). All the replies were from IP addresses King Pin had reached out to first. He sent out 13,522 packets and received 4,034 packets back. All the IP addresses King Pin reached out to were Amazon related, including Amazon Technologies. Interestingly, King Pin sent 9,908 packets to the broadcast address, and most of these packets were NAT information sent using the SKYPE protocol to port 6667. This might indicate King Pin was looking for a specific IP address to transmit data to but was unable to find the correct device.

Packets From Outside Sources to King Pin			Packets From King Pin to Outside Sources		
From	Packets	Bytes	To	Packets	Bytes
34.218.107.222	242	64	34.218.107.222	252	42
35.164.57.66	237	61	35.164.57.66	308	50
44.227.85.32	118	32	44.227.85.32	52	13
44.236.174.94	770	159	44.236.174.94	710	134
50.112.167.160	501	103	50.112.167.160	385	72
52.25.223.91	1,401	289	52.25.223.91	1,107	209
52.35.54.49	160	37	52.35.54.49	243	38
54.190.215.197	196	50	54.190.215.197	78	19
192.168.8.1	409	101	192.168.8.1	479	77
			255.255.255.255	9,908	3,388

Figure 3. Conversations King Pin participated in, including number of packets and number of bytes sent

Nmap scan

Another potential vulnerability we explored was the open ports on each vacuum. To explore the open ports, Nmap was used on the Kali Linux box to enumerate the TCP ports (first command below) and UPD ports (second command below):

```
nmap -Pn -p 1-65535 -oN <output file name> <device ip address>
nmap -sU -Pn -p 1-65535 -oN <output file name> <device ip address>
```

The parameter -Pn means the scan will run without pinging any of the ports; this can allow for faster scans and allows Nmap to scan a target even if it is not active. The -p parameter is used to specify the ports scanned; in this case, all ports were scanned. The -oN parameter specifies output should be type "normal" and will be sent to the specified file.

Annie

The Nmap scan revealed that Annie has TCP port 8883 open. This port is used for the Secure-MQTT protocol. Message Queuing Telemetry Transport (MQTT) protocol is used by many IoT devices due to its lightness and ability to work machine to machine.[2] Port 8883 is not associated with any known exploits at the time of writing.[3] A UDP scan of the ports on Annie revealed no ports were open.

Destroyer

The Nmap scan revealed that Destroyer has TCP port 8883 open, explored above. Additionally, the UDP scan of Destroyer revealed ports 5678 and 58362 are open. Port 5678 is used by the Remote Replication Agent Connection (RRAC) and is associated with several trojans and backdoors, some of which lead to remote code execution.[4] Port 58362 is not associated with any services or exploits at the time of writing.[5]

King Pin

The Nmap scan revealed that King Pin has TCP port 6668 open. This port is used by the Internet Relay Chat (IRC) protocol and is associated with many trojans and backdoors.[6] Several of the associated exploits can lead to remote code execution or infection of other systems. The UDP scan of King Pin revealed port 51322 is open. This port is not associated with any known services or exploits at the time of writing.[7]

Deauthentication attack

Due to the issues mentioned earlier with Destroyer not responding to deauthentication packets, it was decided that this test would focus on the speed at which the packets were sent. This would allow us to determine if Destroyer ignored all deauthentication packets, or only packets that were sent fast enough to possibly be an attack. For each robot, packets were sent at ten speeds (1, 2, 4, 8, 16, 32, 64, 128, 256, and 512 packets per second [p/s]), with ten attacks performed at each speed. The packets were sent via a custom script based on a similar attack script written by a colleague.

Before sending deauthentication packets, proper functionality of the robot was confirmed by sending start and stop commands via the app. Once functionality was confirmed, the script was run, and commands were sent via the app to confirm the robot no longer responded to them. Following completion of the script, a timer was started, and commands were sent via the app to the robot until it became responsive again, beginning three to five seconds after the script was completed. The iRobot's app interface took approximately 15 seconds to send a command, so they were sent every 20-25 seconds. The Eufy app interface took approximately five seconds to send a command, so they were sent every 10-15 seconds. As iRobot takes longer to refresh the app interface, it is possible waiting 15-20 seconds after the attack ends before sending commands may result in a shorter recovery time. This may result in less time waiting for the interface to refresh.

Annie

Annie did not recover well from the deauthentication attacks. Of the 100 attacks, 27 of them required at least one reboot in order for Annie to recover (Figure 4). During several of the

attacks, Annie would follow the command, but the app interface would not switch to the new state, resulting in many interface mix-ups. After completing five of the speed tests, Annie was factory reset to see if that helped with the crashes. This did not help, and Annie continued to crash every few attacks. On average, it took Annie 57 seconds to recover from an attack, with a standard deviation of 53.8 seconds. She recovered quickest from the 16 p/s attack, taking 33.1 seconds with a 31.2 second standard deviation. She recovered slowest from the 32 p/s attack, recovering in 91 seconds with a standard deviation of 66.2 seconds.

Annie											
Recovery											
Packets Per Second	1	2	3	4	5	6	7	8	9	10	Mean
1	29	88	32	43	34	45	32	47	29	30	40.9
2	50	28	31	30	30	14	14	228	26	16	46.7
4	237	13	242	9	35	13	19	12	4	250	83.4
8	16	28	21	13	75	46	26	18	242	21	50.6
16	6	8	116	11	35	33	35	29	30	28	33.1
32	89	37	40	59	92	140	98	34	254	67	91.0
64	56	53	60	56	58	43	55	81	37	83	58.2
128	64	62	39	57	112	38	100	64	63	90	68.9
256	53	35	53	52	55	76	85	75	35	58	57.7
512	46	41	27	48	49	42	29	27	47	27	38.3

Required Reboot

Figure 4. Time it took Annie to recover from each deauthentication attack (in seconds)

Destroyer

Destroyer responded to all the attacks, regardless of speed. This disproved the theory that Destroyer was recognizing attacks and ignoring them. Destroyer recovered well from the deauthentication attacks. Of the 100 attacks performed, five of them required at least one reboot for Destroyer to become functional (Figure 5). Several of the attacks at one p/s and two p/s had one or two commands go through while the script was running before the robot fully stopped responding to commands. Destroyer did not require any factory resets to complete the

testing. On average, Destroyer recovered from the attacks in 33.4 seconds, with a standard deviation of 50.8 seconds. He recovered quickest from the 512 p/s attacks, recovering in 23.4 seconds and having a standard deviation of 2.3 seconds. It took Destroyer the longest to recover from the one p/s attacks, taking 67.2 seconds to recover with a standard deviation of 122 seconds.

Destroyer											
Recovery											
Packets Per Second	1	2	3	4	5	6	7	8	9	10	Mean
1	22	33	30	30	18	25	52	413	36	13	67.2
2	27	17	79	26	29	26	15	33	27	36	31.5
4	31	28	28	22	27	34	15	27	26	27	26.5
8	33	26	25	26	25	25	28	26	25	27	26.6
16	27	25	25	28	25	25	26	26	25	25	25.7
32	24	27	47	26	26	25	24	351	26	17	59.3
64	30	26	25	17	25	26	26	24	25	26	25.0
128	25	27	26	25	26	25	13	26	25	25	24.3
256	24	25	24	26	25	24	25	25	25	26	24.9
512	17	24	24	24	24	24	24	24	25	24	23.4

Required Reboot

Figure 5. Time it took Destroyer to recover from each deauthentication attack (in seconds)

King Pin

King Pin recovered very well from the deauthentication attacks. Eufy's app does not have a reboot options, so none of the attacks resulted in a reboot. However, none of the attacks resulted in down time that would have required a reboot (Figure 6). On average, King Pin recovered from the attacks in 23 seconds, with a standard deviation of 22.9 seconds. He recovered quickest from the four p/s attack, recovering in 15.1 seconds and having a standard deviation of three seconds. His slowest recovery occurred during the 16 p/s attack and the 256 p/s attack, each taking 29.5 seconds to recover. The 16 p/s attack had a standard deviation of 35.6 seconds, and the 256 p/s attack had a standard deviation of 23.1 seconds.

King Pin											
Recovery											
Packets Per Second	1	2	3	4	5	6	7	8	9	10	Mean
1	11	7	17	20	14	12	15	5	54	21	17.6
2	19	16	28	23	15	12	81	14	14	14	23.6
4	12	10	16	17	16	16	13	13	18	20	15.1
8	12	16	20	14	22	10	17	25	16	12	16.4
16	25	15	16	14	128	13	8	39	19	18	29.5
32	9	16	15	23	11	16	14	13	16	21	15.4
64	14	28	21	34	19	18	23	6	14	148	32.5
128	18	60	17	7	10	7	6	8	79	18	23.0
256	11	10	43	23	10	34	70	22	7	65	29.5
512	25	9	19	18	76	59	16	15	25	9	27.1

Device Status Listed as Offline

Figure 6. Time it took King Pin to recover from each deauthentication attack (in seconds)

RESULTS

Of the three vacuums tested, Eufy's G30 SES (King Pin) appears the most secure. The only port open on him is port 6668. While this port is associated with exploits, his lack of communication with outside devices lends itself to more security than the other vacuums. Additionally, King Pin recovered the best from the deauthentication attacks. His quick recovery means a deauthentication attack would be least effective on King Pin, lending itself to a better security score for him.

The next most secure vacuum tested is iRobot's Roomba Combo i5+ (Annie). Annie has only port 8883 open and has no known exploit associated with this port. This robot was not overly communicative with outside sources, communicating with mostly Amazon servers. Annie did take the longest to recover from the deauthentication attack. However, this does not pose a security risk to the user as much as it is a nuisance. It is possible the user would simply buy a new vacuum if the current one kept crashing, mitigating any previous attacks or access someone might have had.

Coming in as the least secure of the tested vacuums is iRobot's Roomba Vac Essential (Destroyer). Destroyer has three ports open, one of which is used for an unknown service and one of which is associated with many known exploits. Additionally, Destroyer was very communicative with outside sources, contacting the greatest number of devices total and the greatest number of non-Amazon devices. Destroyer's recovery time from deauthentication attacks was between the two other vacuums. Destroyer recovered faster than Annie and crashed far fewer times, but still required some reboots, unlike King Pin.

FINAL THOUGHTS AND FURTHER RESEARCH

Overall, the three tested vacuums seem to have some security measures in place. They all recover from deauthentication attacks and do not appear to have many unnecessary ports open. Further research should explore the chipsets in each device. Destroyer's chipset is designed for an AI voice emulator and has been adapted to work in the vacuum. All the chipsets may have additional features that are unavailable in the current configuration but could be accessed with more digging and research. Additionally, all the vacuums can only run on the 2.4 GHz Wi-Fi network and do not work with WPA3 security. These lend themselves toward more potential vulnerabilities, as the devices cannot use the newest security measures available to other devices. All these possibilities require further exploration.

ACKNOWLEDGEMENTS

[1] Austin Albright provided the base of the script used for the deauthentication attack.

[2] This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internships program.

REFERENCES

[1] E. Guo, *A Roomba recorded a woman on the toilet. How did screenshots end up on Facebook?*, WWW document, (https://www.technologyreview.com/2022/12/19/1065306/roomba-irobot-robot-vacuums-artificial-intelligence-training-data-privacy/).

[2] MQTT, *MQTT: The standard for IoT messaging*, WWW document, (https://mqtt.org/).

[3] Speed Guide, *Port 8883*, WWW document, (https://www.speedguide.net/port.php?port=8883).

[4] Speed Guide, *Port 5678*, WWW document, (https://www.speedguide.net/port.php?port=5678).

[5] Speed Guide, *Port 58362*, WWW document, (https://www.speedguide.net/port.php?port=58362).

[6] Speed Guide, *Port 6668*, WWW document, (https://www.speedguide.net/port.php?port=6668).

[7] Speed Guide, *Port 51322*, WWW document, (https://www.speedguide.net/port.php?port=51322).