Owowa: the add-on that turns your OWA into a credential stealer and remote access panel

By Paul Rascagneres

Published: 2021-12-14 · Archived: 2026-04-05 16:44:57 UTC

While looking for potentially malicious implants that targeted Microsoft Exchange servers, we identified a suspicious binary that had been submitted to a multiscanner service in late 2020. Analyzing the code, we determined that the previously unknown binary is an IIS module, aimed at stealing credentials and enabling remote command execution from OWA. We named the malicious module 'Owowa', and identified several compromised servers located in Asia.

Meet Owowa, the IIS module you don't want

Owowa is a C#-developed .NET v4.0 assembly that is intended to be loaded as a module within an IIS web server that also exposes Exchange's Outlook Web Access (OWA). When loaded this way, Owowa will steal credentials that are entered by any user in the OWA login page, and will allow a remote operator to run commands on the underlying server.

The malicious module was most likely compiled between late 2020 and April 2021. The assembly default "LegalCopyright" field shows "2020" as a date, and the most recent Owowa sample we could find was detected in April 2021 in our telemetry. The assembly contains a reference to a debugging database (PDB) in its "File" property, and its public key token is set to "b07504c8144c2a49".

We determined that Owowa is intended to be launched as an IIS module because the only relevant code is placed in the class ExtenderControlDesigner, which implements an IIS-specific interface (IHttpModule). Owowa is specifically designed to inspect HTTP requests and responses by hooking the PreSendRequestContent event. This event is supposedly raised when a web application of IIS is about to send content to the client (but according to Microsoft, such an event should never be used in an IHttpModule instance because it is likely to cause an application or server crash).

```
public class ExtenderControlDesigner : IHttpModule
{
    // Token: 0x06000002 RID: 2 RVA: 0x00002057 File Offset: 0x00000257
    public void Dispose()
    {
    }

    // Token: 0x06000003 RID: 3 RVA: 0x00002059 File Offset: 0x00000259
    public void Init(HttpApplication context)
    {
        context.PreSendRequestContent += this.PreSend_RequestContent;
    }

    // Token: 0x06000004 RID: 4 RVA: 0x00002070 File Offset: 0x00000270
    private void PreSend_RequestContent(object sender, EventArgs e)
    {
        HttpContext context = ((HttpApplication)sender).Context;
        HttpRequest request = ((HttpApplication)sender).Request;
        HttpResponse response = ((HttpApplication)sender).Response;
        string text = string.Join(",", request.Params.AllKeys);
        if (request.HttpMethod.ToUpper() == "POST")
```

Malicious HTTP module definition

We determined that Owowa is specifically targeting OWA applications of Exchange servers because its code is purposely ignoring requests from OWA-specific monitoring of account names that start with the HealthMailbox string.

The malicious module is actually designed to log credentials of users that successfully authenticated on the OWA authentication web page. Successful authentication is verified by checking that the OWA application is sending an authentication token back to the user. If that's the case, the username, password, user's IP address and current timestamp are stored in a file at C:\Windows\Temp\af397ef28e484961ba48646a5d38cf54.db.ses. Data are encrypted using the RSA algorithm, with a hardcoded public key stored as an XML blob:

```
<RSAKeyValue>
<Modulus>vTxV8wUJ0PoO2yu/Pm/aICbsT+nFwHXouNo623VIVMl6LY4R96a8cpMTHw92rs0foNcVJB8/SYQvL/6Ko9aOv1K3mm3Txa3Dfe6(
</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>
```

A malicious operator can interact with Owowa by entering specifically crafted commands within the username and password fields in the OWA authentication page of a compromised server. Owowa will respond to these commands through the IIS server, and display the results to the operator, instead of the expected OWA login error messages:

- if the OWA username is jFuLIXpzRdateYHoVwMlfc, Owowa will return the encrypted credentials log, encoded in base64;
- if the OWA username is Fb8v91c6tHiKsWzrulCeqO, the malicious module deletes the content of the encrypted credentials log, and returns the OK string (encrypted using RSA);
- If the OWA username is dEUM3jZXaDiob8BrqSy2PQO1, Owowa executes the command that is typed in the OWA password field using PowerShell on the compromised server. The result of the command is encrypted (as previously described) and returned to the operator.

Owowa contains an empty and unused additional assembly, stored as a compressed resource, as well as an additional AssemblyLoader class from a Costura namespace. These are most likely the result of using the Fody bytecode weaving tool and its Costura add-in from the build chain of Owowa's developer. Fody allows .NET developers to dynamically add features to assemblies at compilation time by weaving, or dynamically modifying the assembly bytecode. In particular, Costura is aimed at packaging dependencies, by adding these dependencies as compressed resources to the assembly. These Costura by-products could either be left-overs from the developer's build chain or an obfuscation attempt that is still being developed, since Owowa's malicious code could potentially be hidden as a compressed resource within a Costura-built assembly.

IIS modules management: loading, finding and getting rid of Owowa

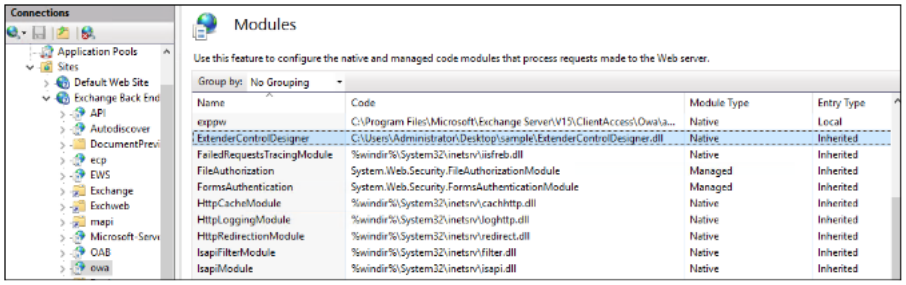
Owowa is loaded (for all compatible applications run by a given IIS server, including OWA) by the following PowerShell script:

```
[System.Reflection.Assembly]::Load('System.EnterpriseServices, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a');

$publish = New-Object System.EnterpriseServices.Internal.Publish;

$name = (Get-Item PATH\ExtenderControlDesigner.dll).FullName;

$publish.GacInstall($name);

$type = 'System.Web.Extensions.Resource.ExtenderControlDesigner,' +
[System.Reflection.AssemblyName]::GetAssemblyName($name).FullName;

Appcmd.exe add module /name:ExtenderControlDesigner /type:"$type"
```

The module is first registered in the global assembly cache, and can then be loaded by the IIS server that is running the OWA application. This setup technique is very reminiscent of one previously used by an unknown threat actor and described by RSA in March 2020 as part of an incident investigation that also involved malicious HTTP modules.

Malicious IIS modules, and Owowa in particular, can be identified by using the command appcmd.exe or the IIS configuration tool, which lists all the loaded modules on a given IIS server instance:

```
C:\Windows\System32\inetsrv>appcmd.exe list modules | findstr ExtenderControl

MODULE "ExtenderControlDesigner" (
type:System.Web.Extensions.Resource.ExtenderControlDesigner,ExtenderControlDesigner, Version=1.0.0.0,
Culture=neutral, PublicKeyToken=b07504c8144c2a49, preCondition: )
```



Malicious module in the IIS configuration manager

Owowa victims

We identified a cluster of targets in Asia with compromised servers in Malaysia, Mongolia, Indonesia and the Philippines. Most of them belong to government organizations, except for one that belongs to a government-owned transportation

company.

Geography of Owowa targets

While we did not discover further compromised servers, we assess with medium to high confidence that additional organizations may also have been targeted in Europe, based on additional data that we provided to customers of our threat intelligence services and that we cannot publicly disclose.

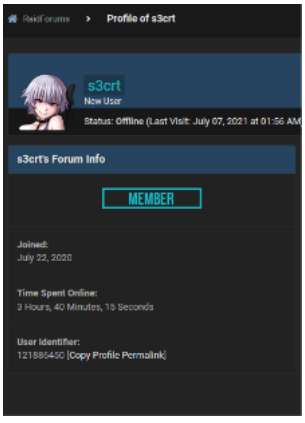
Attribution

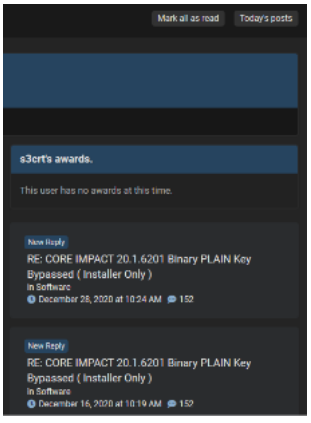
We couldn't find any link between Owowa and any known threat actor, due to insufficient data regarding Owowa's deployment. However, the developer behind Owowa failed to remove the PDB paths in the two identified samples, which both start with C:\Users\S3crt\source\repos\ClassLibrary2\, suggesting a specific username.

Searching for potentially related resources, we identified a Keybase account sharing the same user name – s3crt – with the aforementioned PDB paths. Notably, it shares offensive tools, such as Cobalt Strike and Core Impact:

s3crt Keybase account

The same username also exists as an account on RAID Forums, demonstrating an interest in Core Impact, a popular penetration testing software suite:





s3crt RAID Forums account

Finally, we identified a blog profile on CSDN displaying both the s3crt and z7ys as usernames (the blog title is "z7ys'_s3crt_CSDN博客-XSS领域博主"). The user shows an interest in hacking techniques and distributes files that

supposedly contain leaked Cobalt Strike source code, dating to November 2018:

***s3crt CSDN account*[1]**

Leveraging these clues, the PDB paths and corresponding username, we identified several additional malicious binary files that may have been developed or packaged by the same developer:

- A binary loader (MD5: D4BDFB90D9AA6D573F3FF3A755E2E630) containing a PDB path that shares a common root with Owowa's: C:\Users\S3crt\source\repos\Shellcode_inject\Release\artifact32.pdb.

  This binary was submitted to a multiscanner service in September 2021, but was first spotted in the wild in August 2020. It is designed to decode (XOR) and execute an embedded shellcode. The shellcode was downloading a malicious payload from the IP 150.109.111[.]208 in August 2020. The server was not serving such a payload at the time of our investigation, though based on our telemetry we assume with high confidence it was related to Cobalt Strike;

- We identified another similar binary loader (MD5: 3C5654DDD7998AE39717F7E3D079BD93), first spotted in August 2020, that supposedly also loaded a Cobalt Strike-like payload from 150.109.111[.]208 in August 2020;

- Finally, we identified an additional binary loader (MD5: 3DB7101794CB166CA672814728F4E8D7) that was detected in March 2021 connecting to the domain s3crt[.]biz that also triggered execution of Cobalt Strike payloads. The loader's PDB is C:\Users\Administrator\source\repos\Artifact\x64\Aritfact_big\Artifact.pdb, which is similar in structure to those involving s3crt.

It should be noted that the s3crt username is a simple derivation of the English word "secret" and could very well be used by multiple individuals. Hence, we cannot be certain that the identified accounts and files are actually linked to the developer of Owowa or related to each other. However, the combination of corresponding usernames, PDB paths, projects names and interests in malicious tools or tactics are notable.

Conclusion

The malicious module described in this post represents an effective option for attackers to gain a strong foothold in targeted networks by persisting inside an Exchange server. For malicious operators there are several benefits:

- An IIS module stays persistent on a compromised system even to a Exchange software update;
- Malicious capabilities can easily be triggered by directly sending seemingly innocuous requests to exposed web services – in this case, authentication requests to OWA. Malicious requests like this can be difficult to detect with network monitoring;
- IIS modules are not a common format for backdoors, especially when compared to typical web application threats like web shells and can therefore easily be missed during standard file monitoring efforts;
- The attacker can leverage the module to passively steal credentials from users that legitimately access web services, which presents a stealthier alternative to sending phishing emails.

Unfortunately, we were unable to retrieve enough data to associate the discovered malicious module with any infection chain or post-infection activities. Earlier this year, the ProxyLogon vulnerabilities demonstrated the impact of Exchange server compromise, as well as how quickly threat actors are able to jump on the bandwagon to leverage critical flaws and pursue their goals. It's possible that malicious operators leveraged these server vulnerabilities to initially deploy Owowa.

While showing creativity in Owowa's development, the creator ignored explicit warnings from Microsoft regarding several risky development practices for HTTP modules, which may result in server crashes. Moreover, sensitive information on the development environment (PDB paths, Fody by-products) remained in publicly available samples. Further samples or online profiles can be weakly linked to this kind of information.

The operators behind Owowa demonstrated an interest in government organizations in Asia and specifically South-East Asia. Such targeting may fit a threat actor seeking to gather intelligence on ASEAN's agenda and member states' foreign policies. However, the practices exhibited by what is likely an inexperienced developer don't appear to correspond with such strategic targeting.

Indicators of Compromise

Owowa:
af6507e03e032294822e4157134c9909
ea26bed30da01f5d81c3d96af59424acf2fbb14b
8e1e0ddeb249b9f8331b1562498d2cbd9138ec5e00c55a521d489e65b7ef447d

Possibly related malicious loaders:
d4bdfb90d9aa6d573f3ff3a755e2e630
2e5a752f8d1c3b0ba819381c4539006d40692ee9
dac4c2e5318bf0feca535b2116bd48e72d8f36ff7ec8f3bd176fd7e57bd37fc1

3c5654ddd7998ae39717f7e3d079bd93
8429a32acfed3f010502a5b88199cc0367f92fd7
54fecd3227a435c17463f543eacdb7482fc7b2fde4db1d12d16aab94dfdf4085

3db7101794cb166ca672814728f4e8d7
f8b5d7370b56e127449760701b97bf8f43d16852
f167279c692a14fee15bb1f8eb8a9b6edd43cf74d2b590b27129fd69e6b3de88

PDB Paths:
C:\Users\S3crt\source\repos\ClassLibrary2\obj\Release\ExtenderControlDesigner.pdb
C:\Users\S3crt\source\repos\ClassLibrary2\obj\Release\ClassLibrary2.pdb
C:\Users\S3crt\source\repos\Shellcode_inject\Release\artifact32.pdb
C:\Users\Administrator\source\repos\Artifact\x64\Aritfact_big\Artifact.pdb

Possibly related Cobalt Strike C2:
150.109.111[.]208

Possibly related suspicious domain:
s3crt[.]biz

[1] Note that the picture used in the profile is of a martial art practitioner and is unrelated.
