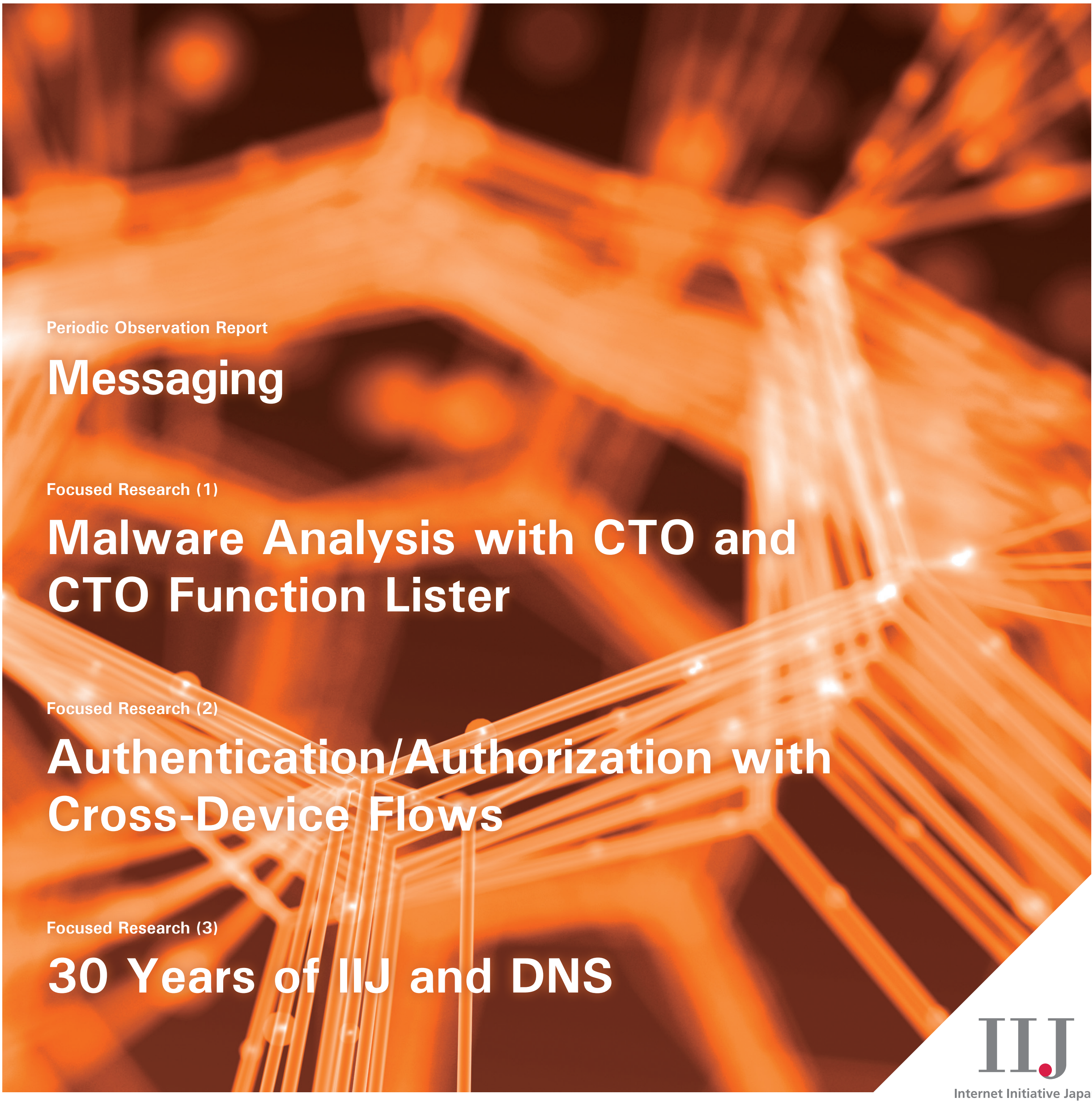
IIR

Internet
Infrastructure
Review

Aug. 2023
Vol. 59



Periodic Observation Report

Messaging

Focused Research (1)

Malware Analysis with CTO and CTO Function Lister

Focused Research (2)

Authentication/Authorization with Cross-Device Flows

Focused Research (3)

30 Years of IIJ and DNS

IIJ
Internet Initiative Japan

Internet Infrastructure Review

August 2023 Vol.59

© Internet Initiative Japan Inc.

Executive Summary

In the previous edition, I touched on OpenAI's ChatGPT. Since then, information about generative AI and large language models has dominated the public consciousness, not only in the IT industry but across society as a whole. One gets the sense that these technologies are taking the world by storm, with more and more companies using them.

It does also seem, however, that people are increasingly also talking about the negative aspects of generative AI. At the G7 Digital and Tech Ministers' Meeting in Takasaki, Gunma, for instance, participants endorsed the G7 Action Plan for Enhancing Global Interoperability of AI Governance, and the G7 Hiroshima Leaders' Communiqué addressed "governance, safeguard of intellectual property rights including copyrights, promotion of transparency, response to foreign information manipulation, including disinformation, and responsible utilization of these technologies."

AI is a technology, and it goes without saying that those who develop and use such cutting-edge technologies must hold themselves to high ethical standards. As a technology engineer myself, I can say that these recent events have given me a renewed awareness of this.

The IIR introduces the wide range of technology that IIJ researches and develops, comprising periodic observation reports that provide an outline of various data IIJ obtains through the daily operation of services, as well as focused research examining specific areas of technology.

Our periodic observation report in Chapter 1 discusses messaging with a focus on email. All sorts of services have been developed on the Internet, but email remains one of the essentials, and IIJ has been providing email services since its founding. While this important service has a long and storied history, improvements are still being made today. Against that backdrop, the report discusses the discontinuation of one of IIJ's email services, DMARC 2.0 as a key M^{3}AAWG topic, and the uptake of sender authentication and STARTTLS.

The focused research report in Chapter 2 introduces malware analysis tools developed by an IIJ employee. Malware is a major threat on the Internet and has caused all sorts of damage. As a malware and forensics analyst at IIJ, the author is engaged in customer incident response and also draws on his experience to develop malware analysis tools. The author implements features into these tools that he deems necessary from the perspective of someone who actually performs the analyses, and the report thus provides a compelling glimpse into real-world malware analysis.

Chapter 3 presents a focused research report on authentication and authorization using cross-device flows. With Internet-based services being part of our social infrastructure, the importance of authentication and authorization when using these services is ever increasing. Cross-device flows facilitate safer, easier-to-use authentication and authorization flows via the smartphones that most people keep on their person. The report discusses a number of device flow specifications, both those that have been standardized and some that are still being drafted, and goes over the differences between them.

And following on from our piece on the IIJ backbone network in the previous edition, the focused research report in Chapter 4 looks at IIJ's efforts with DNS. It goes without saying that DNS is a cornerstone of the Internet's foundations, and IIJ has been engaged with DNS in all sorts of ways since its founding. The report looks back on 30 years of IIJ and DNS from the perspective of services and technology, and also discusses the relationship between DNS and society at large. It also provides a picture of DNS back at the dawn of the commercial Internet era, and I think you will find it an intriguing read.

Through activities such as these, IIJ strives to improve and develop its services on a daily basis while maintaining the stability of the Internet. We will continue to provide a variety of services and solutions that our customers can take full advantage of as infrastructure for their corporate activities.

Junichi Shimagami

Mr. Shimagami is a Senior Executive Officer and the CTO of IIJ. His interest in the Internet led to him joining IIJ in September 1996. After engaging in the design and construction of the A-Bone Asia region network spearheaded by IIJ, as well as IIJ's backbone network, he was put in charge of IIJ network services. Since 2015, he has been responsible for network, cloud, and security technology across the board as CTO. In April 2017, he became chairman of the Telecom Services Association of Japan's MVNO Council, stepping down from that post in May 2023. In June 2021, he also became a vice-chairman of the association.

Messaging

1.1 30 Years with Email

IIJ celebrated its 30th anniversary last year, and over those years we have created and released many services. Among Internet infrastructure services, email in particular has one of the longest histories. Many new services continue to come to life, but where there is a beginning, there is also eventually an end. Safely bringing live services to a close with minimal impact on customers can be far more difficult than creating new services, a fact that is often underappreciated.

In the first half of this chapter, we reflect on the IIJ Post Office Service, which IIJ retired last year after 24 years in operation. In the second half, we report on recent debate around sender authentication technology DMARC and on email route encryption observed on IIJ's email services.

1.2 IIJ Post Office Service

The IIJ Post Office Service is an email hosting service for business customers that allows you to send and receive emails via your own domain name.

To start using the service, all a customer does is point the MX record(s) on their DNS server to IIJ. This sort of functionality is something that all hosting providers offer these days, but according to our records, we launched the service in July 1998[*1][*2]. The service provided unlimited mail storage capacity, with the ability to save received email for 14 days. Being directly connected to IIJ's high-quality backbone, the service offered stability. It was naturally well received by customers, and with email becoming an essential tool for businesses, the service also became well known within IIJ as one that sales reps had no trouble marketing.

In the years following its launch, we continued to add functionality and related services (Table 1).

Date	Press release
July 2001	Virus protection (antivirus) functionality added[*3]
December 2002	IIJ Mail Gateway Service started. Email audit option added[*4]
March 2003	MailViewer (webmail feature) offered as standard[*5][*6]
October 2004	Spam filter option added[*7]
October 2006	IIJ Secure MX Service started[*8]
January 2010	Support for sender authentication technology DKIM[*9]
June 2010	Support for IPv6 as standard

Table 1: History of the IIJ Post Office Service

1.2.1 Challenges of a Long-running Service

All companies naturally want to build and develop their services in such a way as to generate as much revenue and profit as possible, and to continue providing those services for a long time. This is precisely how the IIJ Post Office Service has evolved, making it one of the services that have contributed greatly to IIJ's growth.

Yet with long-running services like this, the following three challenges sooner or later present themselves.

■ (1) The software cannot support the latest technology

Software innovation is constantly evolving. This will no doubt be all too familiar to you if you're a software engineer. While something may have represented the latest technology when first developed, jump forward just a few years and you'll find newer technologies and better frameworks being developed.

Adding revisions to old code and playing catch up with the latest trends is quite a challenging task under such circumstances, and it requires a high level of motivation, not to mention skill.

[*1] IIJ, "Launch of the IIJ Post Office Service" (https://www.iij.ad.jp/news/pressrelease/1998/pdf/postoffice.pdf, in Japanese).

[*2] Microsoft released Windows 98 the same month, July 1998.

[*3] IIJ, "IIJ to launch antivirus service on July 1" (https://www.iij.ad.jp/news/pressrelease/2001/pdf/po-virusprotection.pdf, in Japanese).

[*4] IIJ, "IIJ launches the IIJ Mail Gateway Service to help medium-sized enterprises stop information breaches" (https://www.iij.ad.jp/news/pressrelease/2002/pdf/iij-mgw.pdf, in Japanese).

[*5] IIJ, "IIJ adds the new MailViewer feature to the IIJ Post Office Service" (https://www.iij.ad.jp/news/pressrelease/2003/pdf/0327.pdf, in Japanese).

[*6] The iconic Gmail webmail service was launched in 2004 on an invitation-only basis.

[*7] IIJ, "IIJ bolsters the anti-spam features of its business email outsourcing service" (https://www.iij.ad.jp/news/pressrelease/2004/pdf/0928.pdf, in Japanese).

[*8] IIJ, "IIJ Launches the IIJ Secure MX Service for comprehensive email risk management" (https://www.iij.ad.jp/news/pressrelease/2006/pdf/0905.pdf, in Japanese).

[*9] IIJ, "IIJ adds support for DKIM sender authentication to the IIJ Post Office Service" (https://www.iij.ad.jp/news/pressrelease/2010/pdf/po_dkim_2.pdf, in Japanese).

For a service with a lengthy history, it becomes difficult to respond to the ever-changing Internet security requirements as well as customer demands for new features and improvements.

■ (2) Efforts to address vulnerabilities come up against limits

Your task isn't over once you've developed a piece of software. You then need to address the seemingly daily reports of vulnerabilities as well as the need for ongoing software maintenance to deal with eventual middleware end-of-life. It's also important to note that with service release cycles in general, software maintenance is a far longer-term undertaking than initial development.

For example, we continued to provide the IIJ Post Office service through six generations and seven types of OSs, replacing the OS each time support ended. This sort of maintenance and development is crucial when it comes to maintaining service quality. But the truth is that it is also a rather unglamorous undertaking. It does not yield as many visible changes as new feature development, and it also carries the risk of introducing new bugs because you're modifying something that already works, and these aspects can be difficult to convey to customers, sales reps, and management.

■ (3) The development history and background may be unclear

With older IIJ services, it was not uncommon for the team that developed the service to be responsible for running it as well. While an advantage of this approach is that the service is run by those who are most familiar with it, making it possible to recover quickly in the event of failure, the disadvantage here is that it does not incentivize documentation and it impedes the transfer of skills to new team members. The IIJ Post Office Service fell into this category.

So as time passes and the number of people from the original development team still running the service decreases, the development and operation of the service ends up being handled by people who were not involved in the planning and launch phase. And when they encounter any undocumented parts of the service, these newer team members have no option but to guess at the original development motivations. They increasingly find themselves asking, "Why is it like this?", which raises the barriers to maintenance development and maintenance itself considerably.

1.2.2 Service Termination Decision and Roadmap

We continued to push ahead with the service against this backdrop, but we eventually came up against technological issues that would prevent us from extending the service's life any further.

We explained this situation to the Steering Committee, and at an internal meeting attended by people from the operation, development, and support departments in 2018, the decision was made to discontinue the IIJ Post Office service in four years' time. We laid out the following action plan.

- Set the service end date
- Establish teams to support the transition to the successor service (IIJ Secure MX Service)
- Develop and implement migration support functionality for the IIJ Secure MX Service
- Internal announcement
- Announcement to customers
- Check on individual progress with sales reps

To minimize the impact of the service termination on customers' businesses, we contacted every single one of our sales reps to confirm progress. This is a fairly involved, hands-on task, but the decision to discontinue the service was our own after all, so after a careful preliminary investigation, roughly a year before the actual service termination date, we began working with the sales team and got the cross-departmental process underway.

1.2.3 Calling Curtains on 24 Years of History

On September 30, 2022, the IIJ Post Office Service was quietly retired. We apologize to our customers for any inconvenience caused by the termination of this service.

As a member of a technology department, I don't usually have the opportunity to thank customers directly, so I would like to take this opportunity to express my sincere gratitude to everyone who has used the IIJ Post Office Service over the years. Thank you very much for your support.

The IIJ Secure MX Service is now available as the successor to the IIJ Post Office Service. We humbly ask for your continued support of IIJ's services.

1.2.4 To Customers Who Were Using the IIJ Post Office Service

We have a final request for customers who were using the IIJ Post Office Service.

If the TXT record for your domain contains
include:spf.po.2iij.net
please be sure to delete this.

We are currently performing a post-service cleanup and will delete this SPF record soon.

Any customers who inadvertently leave the IIJ Post Office Service include tag ("include:spf.po.2iij.net") in their email domain's SPF record are likely to experience sender authentication failures (permerror) at email destinations once we have deleted this record. This could hobble your domain's spoofing countermeasures.

We have contacted those customers who are reachable via our sales reps, but if you're reading this, please take this opportunity to check on your end.

1.3 DMARC 2.0 (M^{3}AAWG Topic)

The first in-person M^{3}AAWG meeting in three years took place in San Francisco in February 2023. M^{3}AAWG (the Messaging, Malware, and Mobile Anti-Abuse Working Group), established in 2004, is an organization that facilitates discussion with a focus on email and other messaging technologies. In recent years, the focus of discussion has broadened beyond email, with a number of companies and academic institutions engaged in areas such as SMS and social media messaging also getting involved. M^{3}AAWG participants include MSPs (mailbox service providers) like IIJ, ESPs (email service providers), server hosting providers, academic institutions, DNS Federation, and security vendors that offer antispam/antivirus engines.

International M^{3}AAWG meetings are held three times a year, typically in San Francisco around February, somewhere in Europe around June, and in a North American city around October.

The February meeting covered a range of themes, with the session on DMARC 2.0 offering a particularly lively discussion. In this section, I summarize some key information on DMARC 2.0[*10] as of April 2023.

M^{3}AAWG meetings are private, so I am unable to disclose details of what was discussed. The information here is based solely on what is available publicly. Also note that while IETF documents also use the term DMARC-bis, here I refer to DMARC 2.0 throughout for consistency.

DMARC is a sender authentication technology currently used on the Internet and is defined as an international specification in RFC 7489[*11]. A number of changes are under consideration for DMARC 2.0. The main ones are as follows.

- While RFC 7489 is classified into the Informational category, the aim is to make DMARC 2.0 a standard.
- Use of DNS Tree Walk instead of the Public Suffix List for DMARC policy discovery.
- Removal of some tags and addition of new tags.

1.3.1 Public Suffix List and DNS Walk Tree

The Public Suffix List[*12] is a list maintained by volunteers who manage domains called eTLDs (Effective TLDs). It was once maintained by Mozilla, known for products like Firefox and Thunderbird, and is now handled by volunteers.

[*10] IETF, Datatracker (https://datatracker.ietf.org/doc/draft-ietf-dmarc-dmarcbis/).

[*11] IETF, Datatracker (https://datatracker.ietf.org/doc/html/rfc7489).

[*12] GitHub, publicsuffix/list (https://github.com/publicsuffix/list).

© Internet Initiative Japan Inc.

Common domains for Japan on this list include co.jp and ne.jp, and domains used by local governments are also listed.

RFC 7489 notes the problem of not being able to search for or determine organizational domains for domains not registered on the Public Suffix List. DMARC 2.0 solves this problem by using DNS Tree Walk to determine the organizational domain and search for DMARC policies when DMARC evaluation is performed.

Domain owners who register and publish DMARC records need not make any changes. Meanwhile, for IIJ and other mailbox providers that provide services like the IIJ Secure MX service, which receives email and evaluates DMARC records, it may be necessary to modify the program used when evaluating domains.

1.3.2 Removal of Tags and Addition of New Tags

Table 2 shows the planned tag removals and additions for DMARC 2.0. Here, a point of concern for operators like IIJ that receive DMARC reports or filter using DMARC records is when exactly to discontinue/commence support for the old/new tags. The DMARC record for each domain is implemented by the organization that manages that domain, and each organization can update its record when it sees fit. Hence, we will need to carefully determine exactly when to end support for tags slated for removal and when to commence support for tags slated to be added.

1.4 Report on the Uptake of Sender Authentication and STARTTLS

1.4.1 Sender Authentication Data

As in IIR Vol. 55, here we chart (Figure 1) the proportion of emails received by the IIJ Secure MX service for which sender authentication was supported.

Looking at the SPF verification results, the proportions are almost the same as last time, but DKIM pass and DMARC pass have each increased by around 8 points each. It is evident that even in Japan the number of companies implementing DKIM signatures and setting DMARC policies as a means of combating email spoofing is rising, albeit gradually. In February 2023, Japan's Ministry of Internal Affairs and Communications asked credit card companies to adopt DMARC to bolster their phishing email defenses[*13].

Add/remove	Tag	Description
Add	np	Flag specifying policy for non-existent subdomains (taken from RFC 9091)
Add	psd	Flag indicating whether Public Suffix Domain or not
Add	t	Flag retaining some of the functionality of the pct flag (see below)
Remove	pct	Flag declaring the percentage of messages to which the DMARC policy is applied
Remove	rf	Format to be used for DMARC failure reports
Remove	ri	Interval requested between aggregate DMARC reports (defaults to once a day, higher values to be accommodated on a best-effort basis)

Table 2: Tags Slated to be Added/Removed in DMARC 2.0

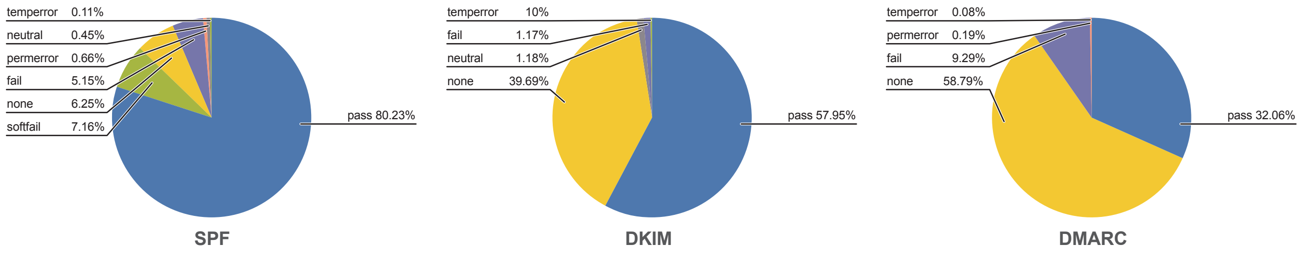


Figure 1: Proportion of Sender Authentication Support for Emails Received by the IIJ Secure MX Service

[*13] Ministry of Internal Affairs and Communications, "Call for credit card companies and the like to bolster anti-phishing measures" (https://www.soumu.go.jp/menu_news/s-news/01kiban18_01000184.html, in Japanese).

1.4.2 Inbound/Outgoing STARTTLS data

For several years now, the idea of bringing an end to PPAP (colloquial Japanese term for the practice of sending encrypted, password-protected ZIP files via email) has frequently been raised when it comes to discussing security issues around email.

There have been media reports about some companies taking steps to end PPAP in response to the spread of the Emotet virus, which we have covered in previous editions of the IIR, but PPAP continues to pop up all the time as an Internet security issue in Japan.

IIR Vol. 55, published a year ago, explained that IIJ was blocking PPAP, but here I would like to approach these issues from the perspective of encrypting communication routes instead of encrypting email attachments.

For companies, PPAP provides a way of encrypting attachments in a form that is easily recognized by employees who send email and by humans working with the email transmission system, but if the emails to which files are attached can themselves be encrypted, this would probably reduce the risk of information breaches. So here we take a look at what proportion of emails received from the Internet and emails sent out over the Internet on the IIJ Secure MX Service were exchanged over TLS over the twelve months from April 2022 to April 2023.

The IIJ Secure MX Service supports the encryption of routes when sending and receiving emails[*14].

The SMTP email transfer protocol uses the STARTTLS protocol extension for TLS transfers. Once a connection with the other server is established, subsequent communications are sent over TLS, with the Envelope From, Envelope To, and DATA (email header and body) fields sent using the supported TLS version and encryption method. If the other server does not support TLS, the email is sent as plain text[*15].

Figure 2 graphs the proportion of emails received on the IIJ Secure MX Service that used STARTTLS over the twelve months from April 2022 to April 2023.

The IIJ Secure MX Service is used by a wide range of business customers, so we observe connections coming in from all sorts of servers on the Internet.

Depending on the day, we observe attacks targeting specific customers and phishing emails that are more or less broadcast to many different recipients.

These emails come from a whole range of servers on the Internet, and it seems that many of the servers send emails without encrypting the SMTP communications using STARTTLS.

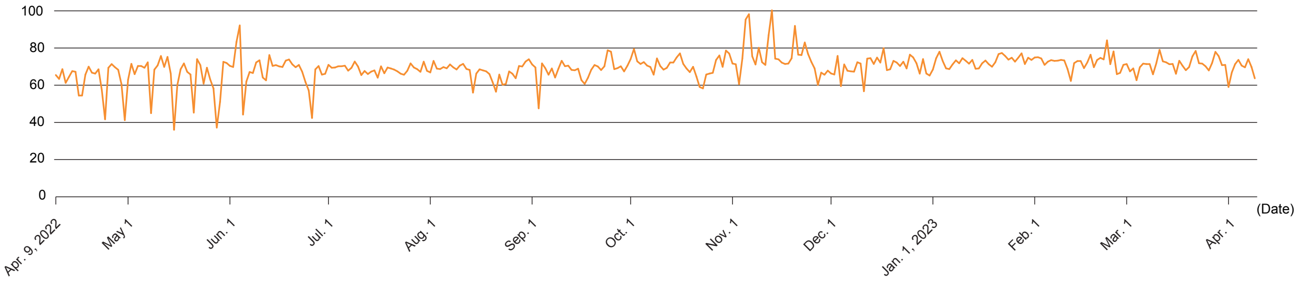


Figure 2: Proportion of Emails Received on the IIJ Secure MX Service Using STARTTLS (Apr. 2022 – Apr. 2023)

[*14] IIJ, "Route encryption" (https://www.iij.ad.jp/en/biz/smx/other.html#anc_02).

[*15] ietf.org, "SMTP Service Extension for Secure SMTP over Transport Layer Security" (https://www.ietf.org/rfc/rfc3207.txt).

The wide fluctuations in the STARTTLS readings from April to May 2022 probably reflect the fact that Emotet was proliferating at the time and thus sending itself out from a lot of Internet-connected servers.

Figure 3 graphs STARTTLS usage for emails sent out onto the Internet from IIJ Secure MX Service servers.

Route encryption is used for outgoing emails from the IIJ Secure MX Service unless the destination server does not support STARTTLS, so as is evident, a higher proportion of these communications are encrypted than is the case with received emails.

For many years, IIJ has also provided an automatic attachment encryption feature, but we plan to discontinue this within the next few years as a means of combating viruses that exploit encrypted ZIP files like Emotet.

While route encryption and the encryption of email attachment contents are not conceptually all that comparable, our hope is that these data will prove useful in efforts to break free from PPAP.

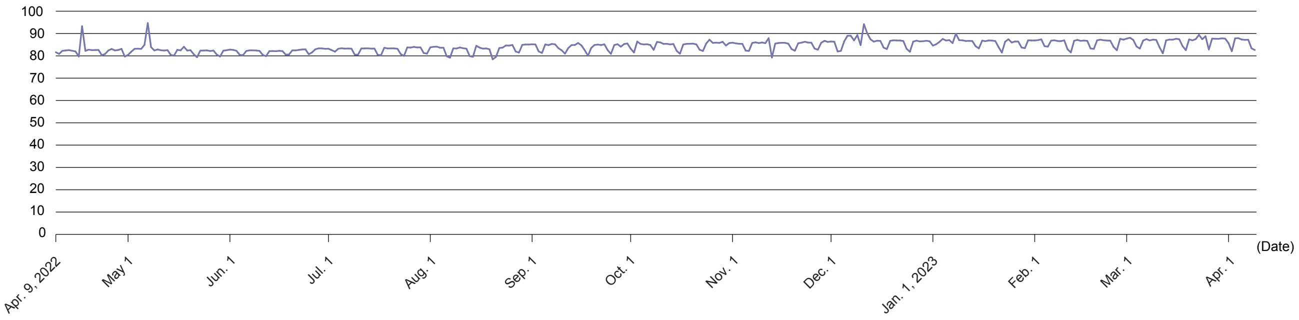


Figure 3: Proportion of Emails Sent on the IIJ Secure MX Service Using STARTTLS

1.1 30 Years with Email, 1.2 IIJ Post Office Service
Isamu Koga
Manager, Operation & Engineering Section, Application Service Department, Network Division & (concurrently) Member of the President's Office, IIJ
Mr. Koga joined IIJ in 2007. He is engaged in the operation of email services and investigates email-related trends in the wild. To keep customers' email boxes safe, he communicates information about the latest attack methods, trends in spam, and countermeasures. He is also involved in a wide range of community activities, including M^{3}AAWG, WIDE Project, and openSUSE.

1.3 DMARC 2.0 (M^{3}AAWG Topic), 1.4 Report on the Uptake of Sender Authentication and STARTTLS
Yusuke Imamura
Lead Engineer, Operation & Engineering Section, Application Service Department, Network Division, IIJ
Mr. Imamura joined IIJ in 2015. He is engaged in the operation of email services. His past experience working at IIJ Europe benefits him in fulfilling his global role.

Malware Analysis with CTO and CTO Function Lister

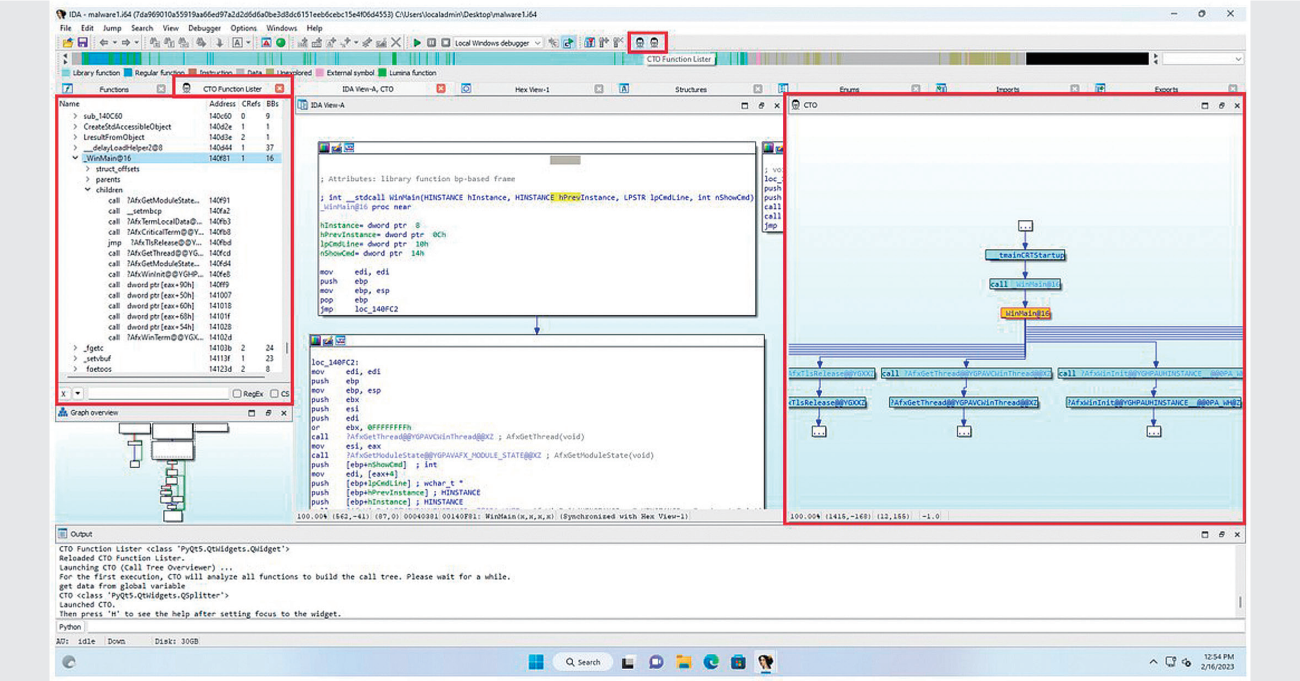
At the Virus Bulletin conference in 2021 (VB2021 localhost), I presented tools called CTO and CTO Function Lister[*1]. I have continued to improve the tools since then by adding new functionality. In this article, I explain what sort of malware analysis tasks these tools are applicable to, along with an in-the-wild malware sample.

The malware sample I use here is selfmake3, which downloads and executes a RAT called SpiderPig, and it has been used in targeted attacks. The SHA256 hash value appears below.

```
7DA969010A55919AA66ED97A2D2D6D6A0BE3D8DC6151EEB6CEBC15E4F06D4553
```

2.1 Startup and Initial Windows

Both CTO and CTO Function Lister are IDA Pro[*2] plugins. They can be launched from Plugins in the Edit menu, toolbar buttons, or shortcut keys. In Figure 1, you will see icons that look like a middle-aged man on the far right of the IDA window toolbar. These are the CTO and CTO Function Lister icons. When clicked, CTO Function Lister appears on the left side of the window, and CTO on the right. The CTO tool is mainly used for visualizing function call parent-child relationships. The main purpose of CTO Function Lister is to extract and retain a list of functions and notable characteristics of each function, and to search for the information via filters. In the figure, you can see that each tool is synchronized with the address of the "_WinMain" function (more precisely, the MFC AfxWinMain function) displayed in IDA's disassembly view (IDA View-A) and is displaying information for that address.



IDA - malware1.i64 (7da969010a55919aa66ed97a2d2d6d6a0be3d8dc6151eeb6cebc15e4f06d4553) C:\Users\localadmin\Desktop\malware1.i64

```
Name                            Address  CRefs  BBs
 > sub_140C60                   140c60   0      9
 > CreateStdAccessibleObject    140d2e   1      1
 > LresultFromObject            140d3e   2      1
 > __delayLoadHelper2@8         140d44   1      37
 v _WinMain@16                  140f81   1      16
    > struct_offsets
    > parents
    v children
        call  ?AfxGetModuleState...   140f91
        call  __setmbcp               140fa2
        call  ?AfxTermLocalData@...   140fb3
        call  ?AfxCriticalTerm@@Y...  140fb8
        jmp   ?AfxTlsRelease@@Y...    140fbd
        call  ?AfxGetThread@@YG...    140fcd
        call  ?AfxGetModuleState...   140fd4
        call  ?AfxWinInit@@YGHP...    140fe8
        call  dword ptr [eax+90h]     140ff9
        call  dword ptr [eax+50h]     141007
        call  dword ptr [eax+60h]     141018
        call  dword ptr [eax+68h]     14101f
        call  dword ptr [eax+54h]     141028
        call  ?AfxWinTerm@@YGX...     14102d
 > _fgetc                       14103b   2      24
 > _setvbuf                     14113f   1      23
 > foetoos                      14123d   2      8
X ▾                              □ RegEx □ CS
```

```
; Attributes: library function bp-based frame

; int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nShowCmd)
_WinMain@16 proc near

hInstance= dword ptr  8
hPrevInstance= dword ptr  0Ch
lpCmdLine= dword ptr  10h
nShowCmd= dword ptr  14h

mov     edi, edi
push    ebp
mov     ebp, esp
pop     ebp
jmp     loc_140FC2
```

```
loc_140FC2:
mov     edi, edi
push    ebp
mov     ebp, esp
push    ebx
push    esi
push    edi
or      ebx, 0FFFFFFFFh
call    ?AfxGetThread@@YGPAVCWinThread@@XZ ; AfxGetThread(void)
mov     esi, eax
call    ?AfxGetModuleState@@YGPAVAFX_MODULE_STATE@@XZ ; AfxGetModuleState(void)
push    [ebp+nShowCmd]  ; int
mov     edi, [eax+4]
push    [ebp+lpCmdLine] ; wchar_t *
push    [ebp+hPrevInstance] ; HINSTANCE
push    [ebp+hInstance] ; HINSTANCE
```

```
100.00% (562,-41) (87,0) 00040381 00140F81: WinMain(x,x,x,x) (Synchronized with Hex View-1)
```

__tmainCRTStartup
call _WinMain@16
_WinMain@16
AfxTlsRelease@@YGXXZ
call ?AfxGetThread@@YGPAVCWinThread@@XZ
call ?AfxWinInit@@YGHPAUHINSTANCE__@@0PA_W
xTlsRelease@@YGXXZ
?AfxGetThread@@YGPAVCWinThread@@XZ
?AfxWinInit@@YGHPAUHINSTANCE__@@0PA_WH@Z
...
100.00% (1416,-168) (12,155) -1.0

```
CTO Function Lister <class 'PyQt5.QtWidgets.QWidget'>
Reloaded CTO Function Lister.
Launching CTO (Call Tree Overviewer) ...
For the first execution, CTO will analyze all functions to build the call tree. Please wait for a while.
get data from global variable
CTO <class 'PyQt5.QtWidgets.QSplitter'>
Launched CTO.
Then press 'H' to see the help after setting focus to the widget.
```

```
AU:  idle   Down     Disk: 30GB
```

Figure 1: The CTO and CTO Function Lister Launch Buttons and Display Panels

[*1] The presentation given at VB2021 localhost is available at the following URL. CTO (Call Tree Overviewer) yet another function call tree viewer (https://vblocalhost.com/conference/presentations/cto-call-tree-overviewer-yet-another-function-call-tree-viewer/). CTO and CTO Function Lister are also published on my GitHub repository (https://github.com/herosi/CTO).

[*2] IDA Pro (https://hex-rays.com/ida-pro/) is a disassembler and decompiler, essential tools for malware analysts. CTO and CTO Function Lister were written using the IDAPython API.

2.2 Detecting Encryption/Decryption and Encoding/Decoding Routines

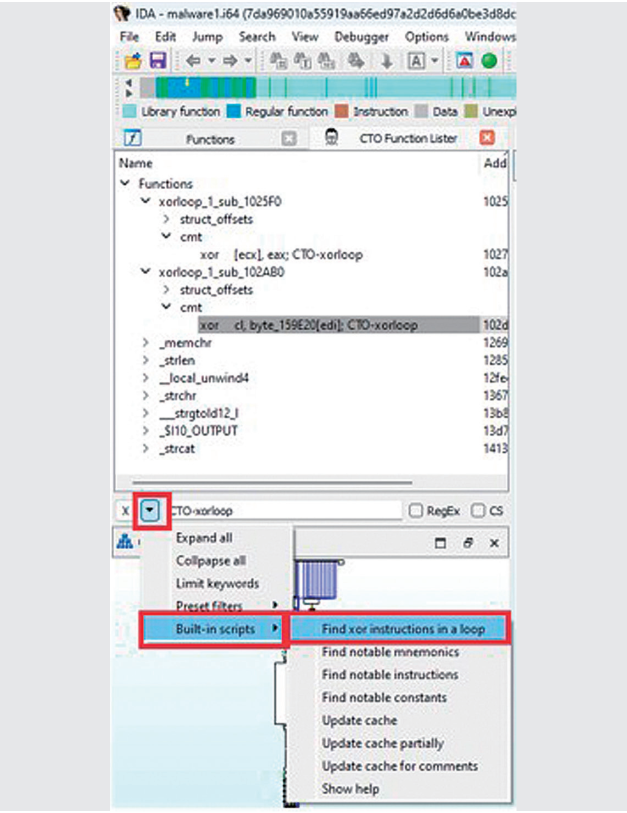
As you no doubt know, malware authors often encrypt or encode communications and config data to make them difficult to detect. In some cases, the authors use existing encryption algorithms such as AES and RC4, and in others, they simply use xor instructions to create custom encodings. In addition to custom encodings that explicitly use xor, many known cryptographic algorithms, including the aforementioned, also include xor instructions. Further, loop structures are inevitably needed when cryptographically processing data that is longer than the CPU registers. So, CTO has a built-in command that traverses the functions known to IDA, finds xor instructions, checks if they are in loops, and displays the results. If a function name in the results has not been changed from the default, it is renamed by appending “xorloop_” so that it can easily be found via the function name.

Figure 2 shows how to execute that command. It can be executed from CTO via a shortcut also, but here I show how to execute it from the CTO Function Lister menu.

First, click the dropdown menu button and select “Built-in scripts”, then select “Find xor instructions in a loop”. It depends on the size of the program being analyzed, but with the sample malware (code section size of around 280KB), the command completed in around 2–3 seconds.

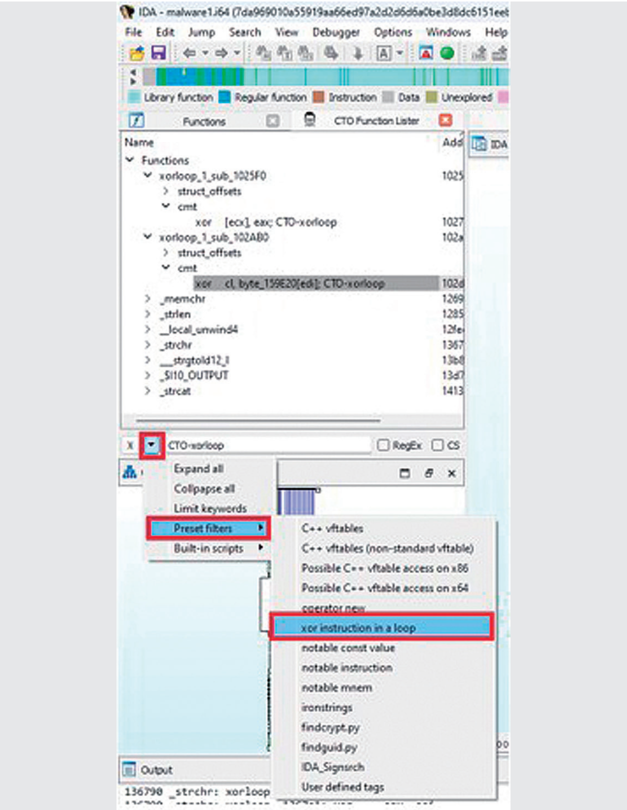
The results are displayed in the Output window, and it can also filter and display only the relevant functions in CTO Function Lister. To do this, open the dropdown menu again and select “Preset filters” and then “xor instruction in a loop” as shown in Figure 3.

This will result in only functions that have an xor instruction inside a loop being listed, as Figure 4 shows. With FLIRT



```
IDA - malware1.i64 (7da969010a55919aa66ed97a2d2d6d6a0be3d8dc
Name                                                    Add
Functions
  xorloop_1_sub_1025F0                                  1025
    struct_offsets
    cmt
        xor    [ecx], eax; CTO-xorloop                  1027
  xorloop_1_sub_102AB0                                  102a
    struct_offsets
    cmt
        xor    cl, byte_159E20[edi]; CTO-xorloop        102d
  _memchr                                               1269
  _strlen                                               1285
  __local_unwind4                                       12fe
  _strchr                                               1367
  ___strgtold12_l                                       13b8
  _$I10_OUTPUT                                          13d7
  _strcat                                               1413
```

Figure 2: Executing the “Find xor instructions in a loop” command



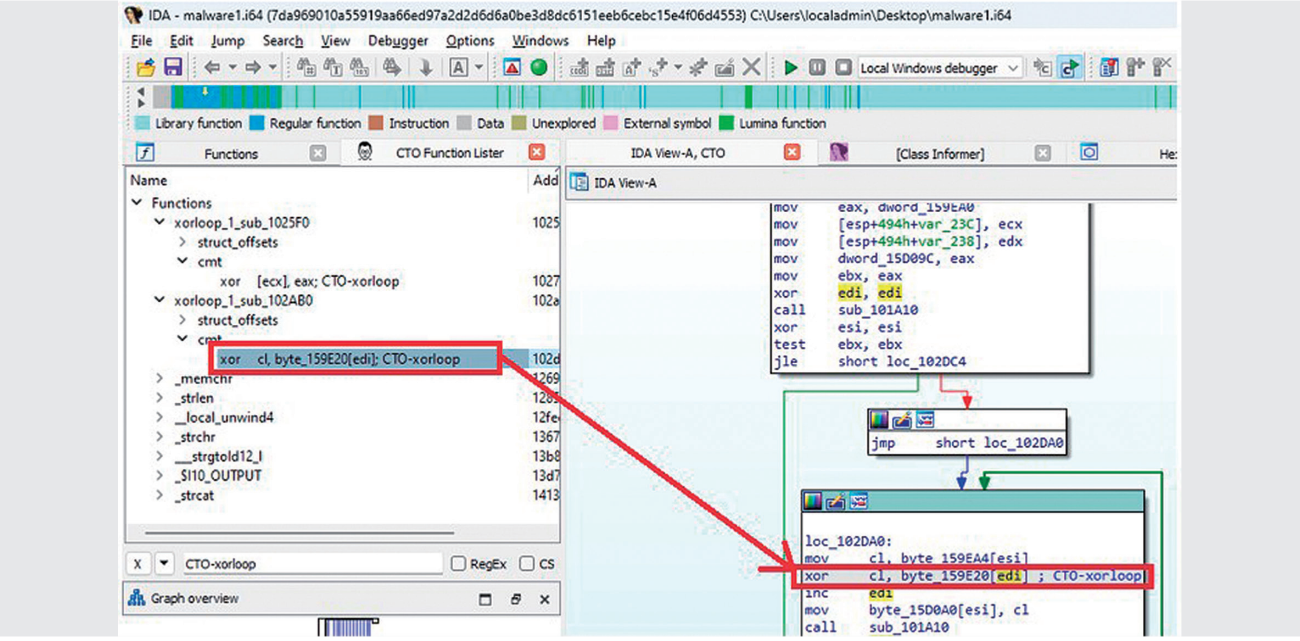
```
IDA - malware1.i64 (7da969010a55919aa66ed97a2d2d6d6a0be3d8dc6151eet
Name                                                    Add
Functions
  xorloop_1_sub_1025F0                                  1025
    struct_offsets
    cmt
        xor    [ecx], eax; CTO-xorloop                  1027
  xorloop_1_sub_102AB0                                  102a
    struct_offsets
    cmt
        xor    cl, byte_159E20[edi]; CTO-xorloop        102d
  _memchr                                               1269
  _strlen                                               1285
  __local_unwind4                                       12fe
  _strchr                                               1367
  ___strgtold12_l                                       13b8
  _$I10_OUTPUT                                          13d7
  _strcat                                               1413

Output
136790 _strchr: xorloop
```

Figure 3: Displaying “xor instruction in a loop”

and Lumina[3], IDA will rename statically linked C (and other language) library functions to an extent. As you can see, all but the first two functions have been named. So, we need to start by looking at the first two functions (sub_1025F0 and sub_102AB0). The aforementioned command adds the comment "CTO-xorloop" to corresponding xor instructions, and these are displayed in CTO Function Lister using a filter. They are in the cmt subtrees. By clicking a line in CTO Function Lister, you can jump to the corresponding address in IDA's disassembly view to inspect the surrounding code.

Looking at the code surrounding the two functions obtained using the command above, one is a routine used to decrypt the payload downloaded from a malicious server, and the other is a routine used to decrypt C&C config data (host name, IP address, etc.) that is hard-coded into the sample. You can find key code blocks like this instantly with these tools.

Here we looked at custom xor-based encodings as an example, but encryption algorithms such as AES and hash algorithms such as SHA256 and MD5 often have characteristic magic values[4] and tables. Third-party scripts and plugins for detecting such characteristics, such as findcrypt[5] and IDA Signsrch[6], have been released. CTO Function Lister recognizes the results of these as well and can filter and display results based on them. Using these tools in combination lets you efficiently discover encryption/decryption and encoding/decoding routines, and quickly check the surrounding code.



IDA - malware1.i64 (7da969010a55919aa66ed97a2d2d6d6a0be3d8dc6151eeb6cebc15e4f06d4553) C:\Users\localadmin\Desktop\malware1.i64

```
Name                                              Add
Functions
  xorloop_1_sub_1025F0                            1025
    struct_offsets
    cmt
      xor    [ecx], eax; CTO-xorloop              1027
  xorloop_1_sub_102AB0                            102a
    struct_offsets
    cmt
      xor    cl, byte_159E20[edi]; CTO-xorloop    102d
  _memchr                                         1269
  _strlen                                         128
  __local_unwind4                                 12fe
  _strchr                                         1367
  ___strgtold12_l                                 13b8
  _SI10_OUTPUT                                    13d7
  _strcat                                         1413
```

Filter: CTO-xorloop

```
mov     eax, dword_159EA0
mov     [esp+494h+var_23C], ecx
mov     [esp+494h+var_238], edx
mov     dword_15D09C, eax
mov     ebx, eax
xor     edi, edi
call    sub_101A10
xor     esi, esi
test    ebx, ebx
jle     short loc_102DC4

jmp     short loc_102DA0

loc_102DA0:
mov     cl, byte_159EA4[esi]
xor     cl, byte_159E20[edi] ; CTO-xorloop
inc     edi
mov     byte_15D0A0[esi], cl
call    sub_101A10
```

Figure 4: Results of "xor instruction in a loop" and the Surrounding Code

[3] FLIRT and Lumina are part of IDA's functionality. Both can use pattern matching to detect and rename known functions. FLIRT refers to a local database, while Lumina refers to a cloud-based database. With Lumina, IDA users push function information to the cloud first. After that, when a function with the same pattern is found in a binary analyzed by another user, the function name on the Lumina server is applied to it.

[4] A magic value is a specific string or numerical value that is used as a marker to uniquely identify a header or footer of a certain format.

[5] FindCrypt (https://github.com/you0708/ida/tree/master/idapython_tools/findcrypt) is a third-party script for IDA Pro that uses pattern matching to detect tables and magic values of well-known encryption and hash algorithms. Several implementations of FindCrypt exist. The one referred to here is implemented in Python and is easy to extend, so this is the one I use.

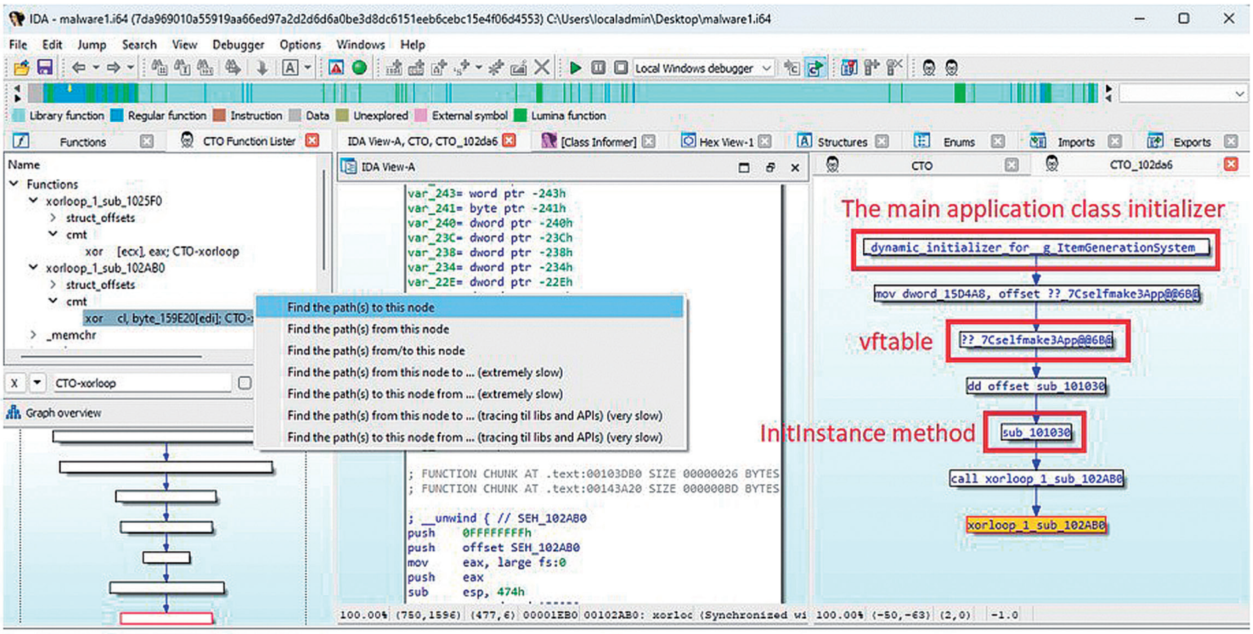
[6] IDA Signsrch (https://sourceforge.net/projects/idasignsrch/) is a third-party plugin for IDA Pro that is used to detect cryptographic and hash algorithms in the same way as FindCrypt. While it is similar to FindCrypt, both tools sometimes differ in scope, so more than one of them is used at times.

2.3 Path Exploration

CTO can display paths to or from an address. Figure 5 shows the result of right-clicking an xor instruction found using CTO Function Lister and selecting “Find the path(s) to this node”. The results appear in a call tree graph on the right side of the window.

Although this graph shows the relationships between each function address and the code and data that refer to it, this is, unfortunately, not a perfect execution path. The reason for this is that even if a function contains a function pointer, it is not necessarily called right away. For example, the function pointer might be stored in a register or on a heap chunk, with the function called much later on. Indirect calls are often used in mechanisms like C++ vftables. To find exactly where a function is executed, you have to track down the class instance, find all the code that refers to it, and find all the code that retrieves a function pointer from the vftable and executes it. It makes the code quite complicated. So, whenever code accesses a function pointer, CTO extracts the address and builds a parent-child relationship graph like this. This is still useful enough, though.

In the example here, a function called dynamic initializer is the first node of the path. This function is processed by the initterm[*7] function in the CRT (C-Runtime). Reading the code reveals that this malware is written using MFC. MFC applications must declare their main application class as a global variable. This declaration causes initterm to call the constructor of the main application class, encapsulated in a dynamic initializer, and the class instance is stored in a


IDA - malware1.i64 (7da969010a55919aa66ed97a2d2d6d6a0be3d8dc6151eeb6cebc15e4f06d4553) C:\Users\localadmin\Desktop\malware1.i64
File
Edit
Jump
Search
View
Debugger
Options
Windows
Help
Local Windows debugger
Library function
Regular function
Instruction
Data
Unexplored
External symbol
Lumina function
Functions
CTO Function Lister
IDA View-A, CTO, CTO_102da6
[Class Informer]
Hex View-1
Structures
Enums
Imports
Exports
IDA View-A
CTO
CTO_102da6
Name
Find the path(s) to this node
Find the path(s) from this node
Find the path(s) from/to this node
Find the path(s) from this node to ... (extremely slow)
Find the path(s) to this node from ... (extremely slow)
Find the path(s) from this node to ... (tracing til libs and APIs) (very slow)
Find the path(s) to this node from ... (tracing til libs and APIs) (very slow)
X
CTO-xorloop
Graph overview

```
Functions
  xorloop_1_sub_1025F0
    struct_offsets
    cmt
      xor    [ecx], eax; CTO-xorloop
  xorloop_1_sub_102AB0
    struct_offsets
    cmt
      xor    cl, byte_159E20[edi]; CTO-
  _memchr
```

```
var_243= word ptr -243h
var_241= byte ptr -241h
var_240= dword ptr -240h
var_23C= dword ptr -23Ch
var_238= dword ptr -238h
var_234= dword ptr -234h
var_22E= dword ptr -22Eh

; FUNCTION CHUNK AT .text:00103DB0 SIZE 00000026 BYTES
; FUNCTION CHUNK AT .text:00143A20 SIZE 000000BD BYTES

; __unwind { // SEH_102AB0
push    0FFFFFFFFh
push    offset SEH_102AB0
mov     eax, large fs:0
push    eax
sub     esp, 474h
```

```
100.00% (750,1596) (477,6) 00001EB0 00102AB0: xorloc (Synchronized wi 100.00% (-50,-63) (2,0) -1.0
```

The main application class initializer

```
dynamic_initializer_for__g_ItemGenerationSystem_
mov dword_15D4A8, offset ??_7Cselfmake3App@@6B@
??_7Cselfmake3App@@6B@
dd offset sub_101030
sub_101030
call xorloop_1_sub_102AB0
xorloop_1_sub_102AB0
```

vftable

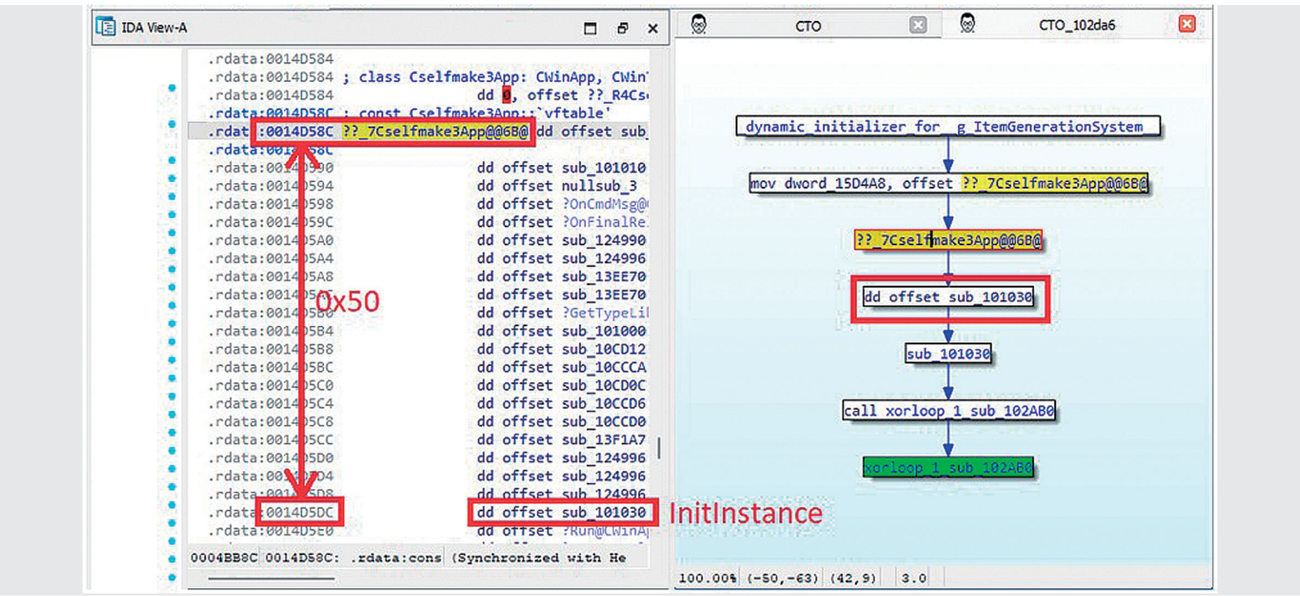
InitInstance method

Figure 5: Path Exploration

[*7] initterm (https://learn.microsoft.com/cpp/c-runtime-library/reference/initterm-initterm-e) is a function that initializes global objects within the CRT before executing the main function. When initterm is called within the CRT, it takes global variables as its first and second arguments, so these are relatively easy to find even if IDA does not recognize this function. initterm executes the function pointers between the addresses specified by its two arguments in sequence. Each function pointer is encapsulated in dynamic initializer code (https://learn.microsoft.com/cpp/c-runtime-library/crt-initialization). Within that code, the global object’s constructor is executed, and the class instance is stored in a global variable.

global variable. The function name displayed in the panel is automatically assigned by Lumina[*8], and clearly the part of the name following “for” is wrong. On the other hand, the path shown by CTO indicates there is access to a vftable with the class name Cselfmake3App in the constructor code. It can also be confirmed from the class inheritance hierarchy, which can be obtained from Class Informer[*9], that this class inherits the CWinApp class. These facts make it clear that Cselfmake3App is the main application class of this malware..

Next, Cselfmake3App’s vftable connects to a function called sub_101030. CTO extracts and caches access to global variables that exist within functions. In particular, if it finds the string “vftable” or “vtable” at the beginning of a variable name or at the end of the comment attached to its address, it treats the global variable as a vftable, parses the table, and follows certain rules to recognize the function pointer group as belonging to that vftable. Since IDA can recognize RTTI, a string containing “vftable” is added to the comment for this address. So, the vftable analysis is executed when CTO is run for the first time, and thus within CTO, sub_101030 is already recognized as part of this vftable. So, when access to a function belonging to the vftable occurs, CTO can connect this function pointer as a virtual method. Figure 6 shows the IDA screen when the Cselfmake3App vftable node (third from the top, “??_7Cselfmake3App@@6B@”) in CTO is clicked. IDA View-A shows a series of function pointers of the vftable. We can see that sub_101030 is located at an offset of 0x50 from the beginning of it. Incidentally, in 32-bit MFC main application classes, there is a virtual method called InitInstance at an offset of 0x50 of the vftable. Hence, sub_101030 is InitInstance.



```
.rdata:0014D584
.rdata:0014D584 ; class Cselfmake3App: CWinApp, CWin
.rdata:0014D584                            dd 0, offset ??_R4Cs
.rdata:0014D58C ; const Cselfmake3App::`vftable'
.rdata:0014D58C ??_7Cselfmake3App@@6B@ dd offset sub_
.rdata:0014D58C
.rdata:0014D590                            dd offset sub_101010
.rdata:0014D594                            dd offset nullsub_3
.rdata:0014D598                            dd offset ?OnCmdMsg@
.rdata:0014D59C                            dd offset ?OnFinalRe
.rdata:0014D5A0                            dd offset sub_124990
.rdata:0014D5A4                            dd offset sub_124996
.rdata:0014D5A8                            dd offset sub_13EE70
.rdata:0014D5AC                            dd offset sub_13EE70
.rdata:0014D5B0                            dd offset ?GetTypeLi
.rdata:0014D5B4                            dd offset sub_101000
.rdata:0014D5B8                            dd offset sub_10CD12
.rdata:0014D5BC                            dd offset sub_10CCCA
.rdata:0014D5C0                            dd offset sub_10CD0C
.rdata:0014D5C4                            dd offset sub_10CCD6
.rdata:0014D5C8                            dd offset sub_10CCD0
.rdata:0014D5CC                            dd offset sub_13F1A7
.rdata:0014D5D0                            dd offset sub_124996
.rdata:0014D5D4                            dd offset sub_124996
.rdata:0014D5D8                            dd offset sub_124996
.rdata:0014D5DC                            dd offset sub_101030
.rdata:0014D5E0                            dd offset ?Run@CWinA
0004BB8C 0014D58C: .rdata:cons (Synchronized with He
```

Figure 6: MFC Main Application Class vftable and InitInstance Function

[*8] As mentioned, Lumina uses names provided by ordinary users, so the accuracy of any given name depends on the skill of the user who created it. Thus, names are often unreliable and should be taken only as a reference. The example here also shows an incorrect name.

[*9] Class Informer is a third-party plugin for IDA Pro. It is a tool that can be used to analyze C++ RTTI (Runtime Type Information) and identify class names and the class inheritance hierarchy (https://sourceforge.net/projects/classinformer/). RTTI analysis itself has been possible since IDA 7.0, but I still use this plugin as it remains superior in some respects—e.g., hierarchy display, class search functionality. An improved version that can restore class information on PE32 binaries with the 64-bit version of IDA is also available on my GitHub repository (https://github.com/herosi/classinformer-ida8). I released this because IDA began phasing out 32-bit IDA starting with 8.0 and moving to the 64-bit version only, and the original Class Informer was unable to parse PE32 on the 64-bit version of IDA.

© Internet Initiative Japan Inc.

Once the MFC application has processed the main application class constructor within the CRT, as described above, it executes several methods such as InitInstance and Run within the WinMain function (specifically, AfxWinMain). In particular, according to the MFC application document, you must override the InitInstance function[*10], so in many cases, this is effectively the malware's main function. The malware we are looking at here also calls InitInstance (sub_101030), and it is easy to see that the routine (sub_102AB0) to decod the malware config is called from the function by using the CTO call tree graph.

Another feature of CTO's path exploration is the ability to create paths even for global variables (including strings) as long as you have a cross-reference[*11]. IDA also has a feature called Proximity View (or Browser), but it can only be used for functions. This is one advantage of using CTO.

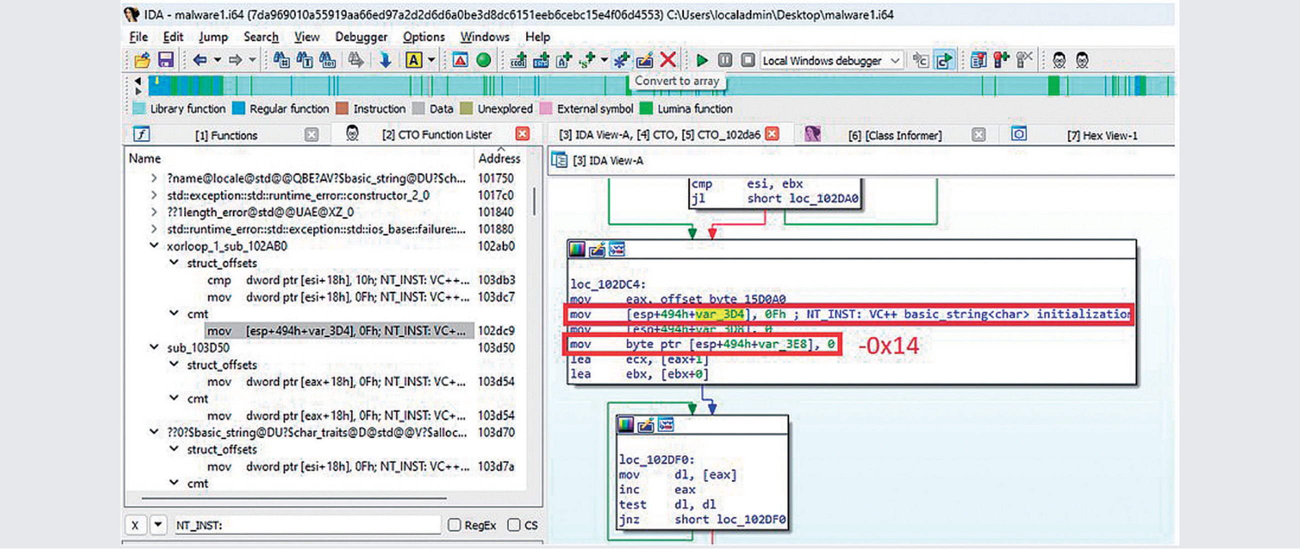
Note that in order to use the CTO Function Lister features as described here, you first need to run CTO.

2.4 Detecting std::string / std::wstring

A lot of malware written in C++ uses std::string and std::wstring for string manipulation. The constructors and some methods of these classes are expanded inline, which can make it hard to determine that these classes are being used at first glance. But because the code that initializes the class layout uses a distinctive initial value, they can be detected with a few simple pattern matching albeit with a few false positives.

These classes can be found by selecting "Built-in scripts", "Find notable instructions" from the CTO Function Lister drop-down menu introduced earlier. You can also select "Preset filters", "Notable instruction" from the drop-down menu to filter the results of this command.

As an example, we'll look at std::string as used in the code that parses the config data decoded by the malware. Figure 7 shows the initialization code for std::string detected by CTO. In the first red box in the figure, the stack variable is


IDA - malware1.i64 (7da969010a55919aa66ed97a2d2d6d6a0be3d8dc6151eeb6cebc15e4f06d4553) C:\Users\localadmin\Desktop\malware1.i64

```
Name                                                          Address
>  ?name@locale@std@@QBE?AV?$basic_string@DU?$ch...           101750
>  std::exception::std::runtime_error::constructor_2_0        1017c0
>  ??1length_error@std@@UAE@XZ_0                              101840
>  std::runtime_error::std::exception::std::ios_base::failure::...  101880
v  xorloop_1_sub_102AB0                                       102ab0
   v  struct_offsets
         cmp   dword ptr [esi+18h], 10h; NT_INST: VC++...     103db3
         mov   dword ptr [esi+18h], 0Fh; NT_INST: VC++...     103dc7
   v  cmt
         mov   [esp+494h+var_3D4], 0Fh; NT_INST: VC+...       102dc9
v  sub_103D50                                                 103d50
   v  struct_offsets
         mov   dword ptr [eax+18h], 0Fh; NT_INST: VC+...      103d54
   v  cmt
         mov   dword ptr [eax+18h], 0Fh; NT_INST: VC+...      103d54
v  ??0?$basic_string@DU?$char_traits@D@std@@V?$alloc...       103d70
   v  struct_offsets
         mov   dword ptr [esi+18h], 0Fh; NT_INST: VC++...     103d7a
   v  cmt
```

X ▾ NT_INST: □ RegEx □ CS

```
cmp     esi, ebx
jl      short loc_102DA0
```

```
loc_102DC4:
mov     eax, offset byte_15D0A0
mov     [esp+494h+var_3D4], 0Fh ; NT_INST: VC++ basic_string<char> initializatio
mov     [esp+494h+var_3D8], 0
mov     byte ptr [esp+494h+var_3E8], 0
lea     ecx, [eax+1]
lea     ebx, [ebx+0]
```

-0x14

```
loc_102DF0:
mov     dl, [eax]
inc     eax
test    dl, dl
jnz     short loc_102DF0
```

Figure 7: Detecting std::string

[*10] The following URL describes the methods that can be overridden and methods that must be overridden when deriving an application class from CWinApp. Only InitInstance is required (https://learn.microsoft.com/cpp/mfc/overridable-cwinapp-member-functions).

[*11] Cross-references, also known as xrefs, are one of IDA's key features. This feature lists code and data referring to a specific address. There are two types: xrefs from and xrefs to. IDA can display and use both, and they are thus collectively called cross-references. CTO also uses cross-references to create parent-child relationships.

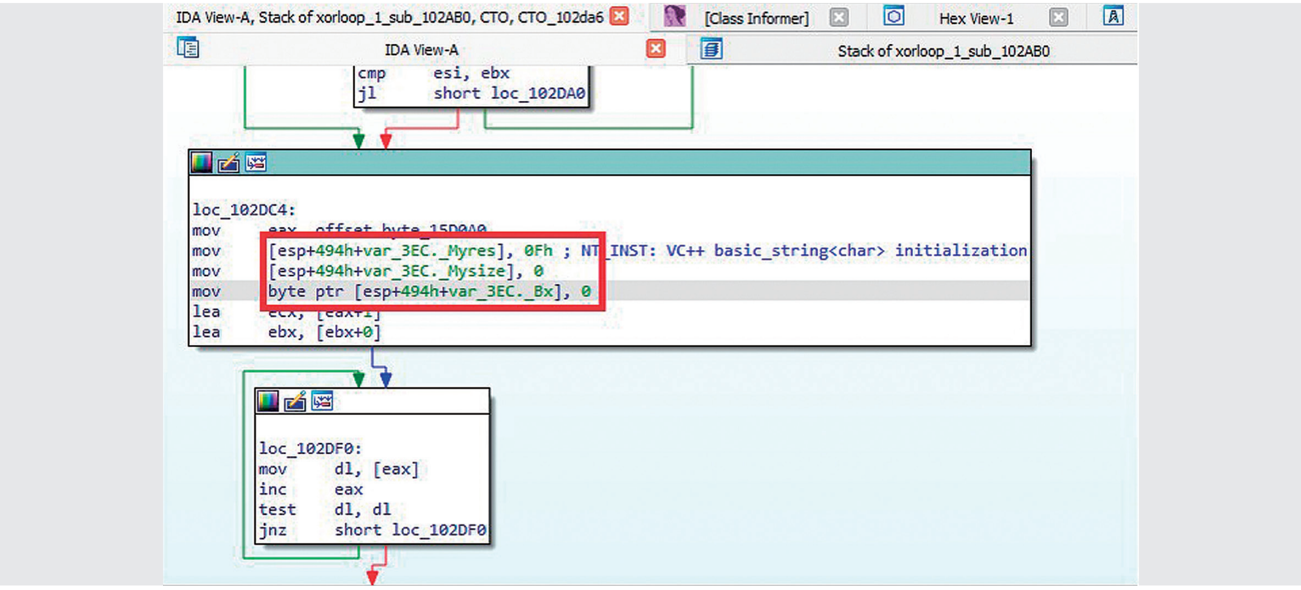
initialized with the immediate value 0xf. This is part of the initialization code for std::string that has been used in Visual Studio for many years. Two instructions below (second red box) is the code that initializes the beginning of the buffer (position -0x14 from the address initialized with 0xf above) with a 1-byte NULL character. When these instructions appear in a set like this, I consider this a use of std::string and apply that structure.

The class layout of std::string is undocumented, and we have determined there to be several patterns depending on which version of Visual Studio is used. On the other hand, I found the malware we are looking at here was compiled using Visual Studio 2008. So, loading the appropriate structure for that version and applying it to the top of the std::string instance on the stack results in a nice, clean recognition of std::string as shown in Figure 8.

2.5 CTO / CTO Function Lister in Practice

At GCC 2023 Singapore[*12] in February 2023, my colleague and I delivered a training course on malware analysis using the IDA plugins discussed here. At the end, we had people randomly form teams of four to six and presented them with characteristic functions and code obtained from the malware sample discussed here in CTF format, and asked them to analyze the malware in some game-like exercises.

While the participants were students that had been specially selected from various countries, many of them had no experience with IDA or reverse engineering, so we had


IDA View-A, Stack of xorloop_1_sub_102AB0, CTO, CTO_102da6
[Class Informer]
Hex View-1
IDA View-A
Stack of xorloop_1_sub_102AB0

```
cmp     esi, ebx
jl      short loc_102DA0
```

```
loc_102DC4:
mov     eax, offset byte_15D0A0
mov     [esp+494h+var_3EC._Myres], 0Fh ; NT_INST: VC++ basic_string<char> initialization
mov     [esp+494h+var_3EC._Mysize], 0
mov     byte ptr [esp+494h+var_3EC._Bx], 0
lea     ecx, [eax+1]
lea     ebx, [ebx+0]
```

```
loc_102DF0:
mov     dl, [eax]
inc     eax
test    dl, dl
jnz     short loc_102DF0
```

Figure 8: Applying the Structure to std::string and Recognizing Member Variables

[*12] GCC (Global Cybersecurity Camp) (https://gcc.ac/) is a training program for students selected from eight, mainly Asian, countries (as of February 2023). GCC 2023 was the fifth such camp and was held in Singapore in February, and we presented there on the content in this chapter (https://gcc.ac/gcc_2023/lectures/#reverse-engineering-malware-written-in- c-with-ida-and-semi-automated-scripts). Hoping to contribute to the industry, IIJ has continued to provide training at this event since its inception.

to give them a brief lecture before starting the CTF exercise. Yet by using techniques like those presented here to save time, the best teams were able to finish most of the malware analysis in around an hour and a half. Over two-thirds of the teams got through most of the important parts in around three hours. This exercise was solely about reverse engineering, so we did not give the teams the executable file itself. They only received an IDA database with the file loaded in. What the malware does is simple, so it is easy to get an overall idea of what's happening under the hood once it is executed, but I deliberately made things harder for the participants because the ability to properly dissect malware by reading the code is also crucial. Even under these conditions, the students used the tools and techniques presented to flesh out their understanding, and it was exciting to see them develop their skills so quickly.

2.6 Final Thoughts

Aside from what I have described here, CTO and CTO Function Lister also implement a range of features that I needed based on past malware analysis. I plan to continue implementing new ideas, such as automation, going forward. I hope these tools prove useful in your malware analysis endeavors.

Hiroshi Suzuki
Malware & Forensic Analyst, Office of Emergency Response and Clearinghouse for Security Information, Advanced Security Division, IIJ
As a member of IIJ-SECT (IIJ's CSIRT), Mr. Suzuki is engaged in internal and customer incident response. He is primarily a malware analyst and forensic investigator. Drawing on the insight and knowledge from this work, he has spoken at international conferences including Black Hat (USA, Europe, Asia), Virus Bulletin, and FIRST TC, as well as at a range of domestic organizations including Japan's National Center of Incident Readiness and Strategy for Cybersecurity (NISC), the Ministry of Internal Affairs and Communications, the Ministry of Justice, IPA, and the National Institute of Advanced Industrial Science and Technology (AIST). He also delivers training courses for experts and students at domestic and international conferences and training programs, including Black Hat USA, FIRST (Annual, TC), Global Cybersecurity Camp, MWS, Japan's National Security Camp, and Cyber Colosseo. He was the first Japanese trainer to be selected for Black Hat USA, where he has given trainings on incident response using forensic investigation and malware analysis. He has dedicated over 17 years to these areas.

Authentication/Authorization with Cross-Device Flows

3.1 Introduction

The rapid proliferation and functional evolution of smartphones continues to change our lives in significant ways. We now use smartphones in every aspect of our daily lives. Authentication and authorization, which are crucial for ensuring that we can use Internet-based services safely, are no exception. In this article, I explain a smartphone-based authentication/authorization method called cross-device flows[*1], something that has been attracting attention in recent years.

A cross-device flow is an authentication/authorization method in which the device (e.g., a PC or smart TV) on which a service is used is separate from the device (e.g., a smartphone) that handles the service authentication/authorization. Say, for instance, that you want to stream video on your smart TV, but that entering your user ID and password into the TV's remote control is awkward, so you use your smartphone instead.

In this case, the cross-device flow solves the problem of using a service on a device with a limited input interface. Cross-device flows are needed in many other situations as well, with a wide range of use cases being proposed. You might, for instance, want to use a service via a device on which you want to avoid entering confidential information, such as a shared or public device. Or you might want to add multi-factor authentication to an existing authentication/authorization flow. Or perhaps you want to perform authentication/authorization on multiple devices using the same private key, but you want to avoid copying that private key.

A number of cross-device flow standards specifications exist, including some that are under development, each with different use cases. Below are some major ones, at which we will take a closer look.

- OAuth 2.0 Device Flow
- OpenID Connect CIBA Flow
- OID4VP's Cross Device Flow
- SIOP v2's Cross-Device Self-Issued OP
- CTAP v2.2's Hybrid transports

3.2 OAuth 2.0 Device Flow

OAuth 2.0 Device Authorization Grant (RFC8628)[*2] is an OAuth 2.0 authorization flow. It was standardized by the IETF in 2019. It is commonly called Device Flow. This cross-device flow was designed to allow other devices to be used to assist with applications running on devices with limited user input capabilities, such as smart TVs, digital photo frames, and printers. The case of using a video streaming app on a smart TV mentioned in the previous section is a prime example of this.

Device Flow is an authorization flow. The protocol is designed such that an authorization server issues access tokens that allow client applications to use a service (usually provided as an API). Since it is not an authentication flow, the Device Flow specification does not encompass functionality by which client applications can authenticate end users (functionality for identifying end users, such as the issuance of ID tokens). If you want to perform authentication as well, you need to combine it with something like OpenID Connect.

[*1] Cross-Device Flows: Security Best Current Practice (https://datatracker.ietf.org/doc/draft-ietf-oauth-cross-device-security/).

[*2] RFC 8628: OAuth 2.0 Device Authorization Grant (https://datatracker.ietf.org/doc/rfc8628/).

© Internet Initiative Japan Inc.

Figure 1 is an example of the Device Flow authorization flow. In OAuth 2.0, the application that uses the service is called the client, and the application that performs authorization (usually a web browser) is called the user agent.

1. The end user launches the client on the device.
2. The client sends an authorization request to the authorization server (a).
3. In response, the authorization server returns a device verification code (device code), an end user verification code (user code), and a verification URL for the end user to access.
4. The client displays on screen the user code and verification URL that it received. Verification URLs are usually displayed in the form of QR codes.
5. The end user scans the QR code with a smartphone or the like (b) to obtain the verification URL.
6. The user visits the verification URL via the user agent. The user is asked to authenticate and thus signs in.
7. After signing in, a user code is displayed on screen. (In some cases, the end user is required to enter the user code).
8. While the end user is working with the user agent, the client repeatedly sends access token requests to the authorization server. The requests include the device code as a parameter.
9. The end user confirms that the user code displayed by the client and the user code displayed by the user agent match, confirms any other notes displayed, and then approves (c).
10. The authorization server issues an access token and returns it to the client in response to the access token request (d).

A major difference between Device Flow and other OAuth 2.0 authorization flows is how the front channel is implemented. The term front channel refers to the link between the client and the user agent. Authorization Code Flow as defined in the OAuth 2.0 Authorization Framework (RFC6749)[*3] is the most commonly used OAuth 2.0 authorization flow and works by redirecting the front channel (using HTTP redirects or redirects that use inter-application linking mechanisms such as deep links (Universal Links on iOS and App Links on Android)). But with Device Flow, redirects cannot be used because the client and user agent run on different devices, so instead, the end user acts as an intermediary by scanning a QR code or reading off and manually entering a code.

Creating a Device Flow front channel is simple and does not require specialized hardware, so it is easy to implement, yet it offers less-than-robust security in some respects. It may be susceptible to access token theft via social engineering or man-in-the-middle attacks, and users could be redirected to malicious sites. So it's generally thought that Device Flow should be avoided for clients that access sensitive or important data.

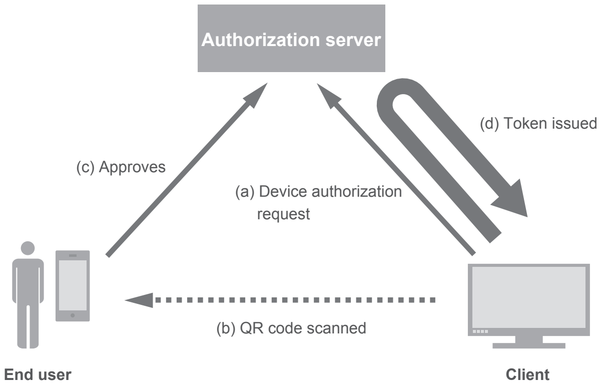


Figure 1: Example of Device Flow-based Authorization Flow

[*3] RFC 6749 - The OAuth 2.0 Authorization Framework (https://datatracker.ietf.org/doc/rfc6749/).

3.3 OpenID Connect CIBA Flow

OpenID Connect Client-Initiated Backchannel Authentication Flow[*4] is an OpenID Connect authentication/authorization flow, abbreviated as CIBA. It was standardized by the OpenID Foundation in 2021. Like Device Flow, CIBA is a cross-device flow that allows the client that uses a service to be on a different device from the one that handles authorization. It is conceptually very different, however. With Device Flow, a single end user operates both devices in most cases, but CIBA was designed with cases in which each device is operated by a different user in mind. This opens up the following sort of use cases for CIBA.

- When a call center rep needs to obtain information from a customer over the phone. In this case, the customer gives their member number to the rep, who then searches for it in a customer management system. A notification is then sent to the customer's smartphone, with a prompt for permission to disclose personal information. The customer management system displays the customer's information to the rep only after the customer provides permission. This mechanism can prevent information breaches caused by staff viewing customer information without permission.
- When approving credit card payments at a store. In this case, when a customer tries to pay via credit card at the cash register, a notification appears on the customer's smartphone with a message confirming the payment details. The payment is completed once the customer approves the message. This offers a more reliable way of identifying people and obtaining consent than asking for a signature or PIN.

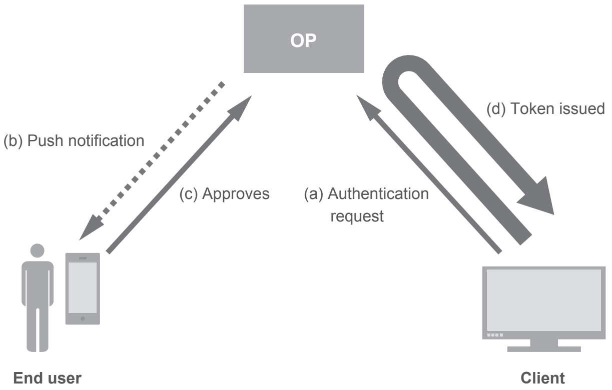


Figure 2: Example of CIBA Authentication/Authorization Flow

Let's take a closer look at CIBA to see how authentication and authorization are implemented (Figure 2). First, some terminology. In CIBA, the device that runs the client is called the consumption device, and the device on which the end user performs authentication is called the authentication device. The authentication device is typically a smartphone. CIBA does not define a term for the application that performs the permissioning operations on the authentication device, but for convenience, I will refer to it as the authentication application. CIBA is an OpenID Connect authentication/authorization flow, so it issues an ID token together with an access token. The server that issues these is called the OpenID Provider (OP).

1. The client sends an authentication request to the OP (a). The request contains a parameter identifying the end user.
2. The OP returns an authentication request ID in response to the authentication request.
3. The OP searches the end user database for an authentication device associated with the end user and then sends a message requesting consent to that authentication device (b). Push notifications (Apple Push Notification Service or Firebase Cloud Messaging) are often used here.
4. The authentication device that receives the consent request starts the authentication application and displays the message on screen.
5. The end user chooses to either consent or decline, and this response is sent to the OP (c).
6. If the end user consents, the OP will issue an access token and an ID token.
7. The client polls the token endpoint and obtains a token (d). The request here includes the authentication request ID as a parameter. If the client is able to expose a notifications endpoint, there is also the option of receiving notifications when a token is issued without any polling.

[*4] OpenID Connect Client-Initiated Backchannel Authentication Flow - Core 1.0 (https://openid.net/specs/openid-client-initiated-backchannel-authentication-core-1_0.html).

The CIBA specification does not define a protocol for communications between the OP and the authentication device. Both the communications method and message specifications are left up to the implementer.

CIBA differs considerably from Device Flow and the other cross-device flows discussed below in that it does not use the front channel; everything is completed via the back channel. The back channel is where interactions between the client and the OP and between the authentication application and the OP occur. Because there is no direct interaction between the client and the authentication application, it supports use cases in which the consumption device and authentication device are separated geographically, as in the call center example above.

Another major feature of CIBA is that it is a client-initiated authentication/authorization flow. With other OAuth/OpenID Connect flows, when a client attempts to access end user resources, authentication/authorization is carried out a single time and the client then holds the token for a lengthy period of time. CIBA makes it possible to issue short-term tokens for each client request, enabling more flexible resource protection.

CIBA is thus quite valuable in that it supports use cases that can be difficult to handle with other authentication/authorization flows. It is attracting attention from the financial industry in particular, and it has also been incorporated into FAPI (an OAuth/OpenID Connect profile for areas that require strong security, such as finance)[*5], which the OpenID Foundation is working to popularize[*6].

3.4 OID4VP's Cross Device Flow

This section describes OpenID for Verifiable Presentations[*7] (abbreviated OID4VP), currently being developed by the OpenID Foundation. Before diving into OID4VP, I will briefly explain verifiable credentials, which are used in OID4VP.

Verifiable credentials (VCs) are a verifiable form of digital credentials. They include, for example, digitized versions of passports, graduation certificates, and employee ID cards[*8]. The issuer digitally signs the credential, and they can be verified by third parties. Multiple VC standards exist, including ISO/IEC 18013-5 Mobile driving license (mDL)[*9] and W3C Verifiable Credentials[*10], which provides a general-purpose data format.

VCs are typically stored in an application called the credential holder's wallet. As mDL and W3C Verifiable Credentials are only VC data specifications, however, they do not define a protocol for obtaining credentials from an issuer and storing them in a wallet, nor a protocol for presenting credentials from a wallet to a verifier. The design of these protocols is up to the implementer. One example is SMART Health Cards (SHC)[*11], a specification for handling VCs (W3C Verifiable Credentials format) for medical information (incidentally, the Covid-19 vaccination certificates provided by Japan's Digital Agency are based on SHC[*12]). The OpenID Foundation is working to standardize these protocols in an effort to promote the adoption of VCs. This is in the form of OpenID for Verifiable Credential Issuance (abbreviated OID4VCI)[*13], a protocol for issuing VCs, and OID4VP, a protocol for presenting VCs. Both OID4VCI and OID4VP are independent of the VC data

[*5] FAPI 2.0 Security Profile (https://openid.bitbucket.io/fapi/fapi-2_0-security-profile.html).

[*6] FAPI: Client Initiated Backchannel Authentication Profile (https://bitbucket.org/openid/fapi/src/master/Financial_API_WD_CIBA.md).

[*7] OpenID for Verifiable Presentations (https://openid.net/specs/openid-4-verifiable-presentations-1_0.html).

[*8] Verifiable Credentials Use Cases (https://www.w3.org/TR/vc-use-cases/).

[*9] ISO/IEC 18013-5:2021 —Personal identification —ISO-compliant driving licence —Part 5: Mobile driving licence (mDL) application (https://www.iso.org/standard/69084.html).

[*10] Verifiable Credentials Data Model v1.1 (https://www.w3.org/TR/vc-data-model/).

[*11] SMART Health Cards (https://smarthealth.cards/en/).

[*12] Digital Agency, "FAQ: Contents of vaccination certificates" (https://www.digital.go.jp/policies/vaccinecert/faq_06/, in Japanese).

[*13] OpenID for Verifiable Credential Issuance (https://openid.bitbucket.io/connect/openid-4-verifiable-credential-issuance-1_0.html).

specification and can be used with mDL, W3C Verifiable Credentials, or other formats.

So, we now turn to the main focus of this section, OID4VP. What does it mean to present a VC? Imagine a situation in which you are asked to provide age verification to purchase alcohol. With a physical ID, you have to present it face-to-face to a clerk at a brick-and-mortar store. VCs, on the other hand, are electronic data, so the interaction does not have to be face-to-face. You can use them for online shopping.

1. You put liquor in your cart on the liquor store website and click the purchase button. The site asks you to present a VC proving you are at least 20 years old.
2. When you click the submit button, your wallet is launched via a deep link, and a message asking if your VC can be presented to the liquor store is displayed.
3. If you consent, you are redirected back to the liquor store website, and your VC is passed to the liquor store. At this point, it is possible to use a mechanism called selective disclosure to ensure that the store only sees what it needs—your date of birth—and none of the other information in your VC.
4. The liquor store website verifies your VC, checks your age, and allows you to make the purchase if you are at least 20 years old.

The above is known as a same-device flow. This is when the software running OID4VP and the wallet are on the same device, that is, when inter-application redirects are possible. OID4VP also accommodates cross-device flows. In the previous example, this corresponds to the use of a VC stored in a smartphone wallet when shopping online on a PC. Instead of redirects, cross-device flows use QR codes to connect the two devices.

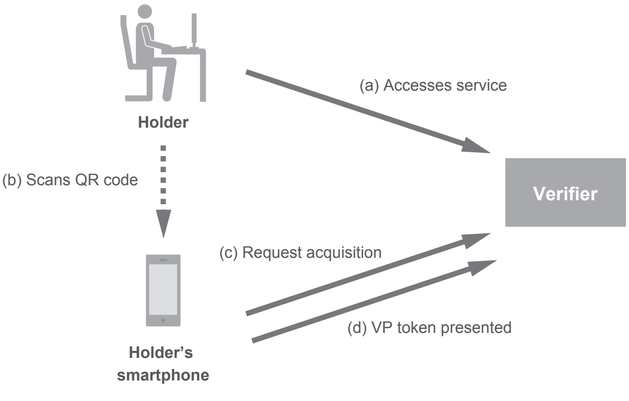


Figure 3: Cross-Device Flow Authentication Example

Let's take a closer look at OID4VP's cross-device flow (Figure 3). In OID4VP, the end user who has the VC is called the holder, the person to whom the VC is presented is called the verifier, and the data format used to present the VC to the verifier is called the VP token. A VP token can contain multiple VCs. The verifier's application needs a server that will receive HTTPS requests.

1. The holder accesses the verifier's services via a PC (a).
2. The verifier application converts the request acquisition URI into a QR code, which is displayed on screen.
3. The holder scans the QR code with a smartphone wallet (b).
4. The wallet accesses the verification server's request acquisition URI (c).
5. The verification server returns the details of the request in response. The request contains a detailed description of the requirements of the VC that will be presented.
6. In accord with the request received, the wallet displays a message asking the holder for consent regarding the content of the VC that will be presented.
7. The holder reviews the content and consents to the VC being presented.
8. The wallet sends the VP token to the verification server (d).
9. Once the verifier verifies the VC, the holder can continue to use the verifier's services via the PC.

With OID4VP's cross-device flow, all communication between the verifier and the wallet after the URI is initially acquired using a QR code is assumed to take place over the Internet. OpenID for Verifiable Presentations over BLE[*14] is an extension of this currently being developed to facilitate the use of OID4VP in environments where the

[*14] OpenID for Verifiable Presentations over BLE (https://openid.bitbucket.io/connect/openid-4-verifiable-presentations-over-ble-1_0.html).

Internet is unavailable. Possible use cases for this include patrons presenting e-tickets in VC form wirelessly over BLE (Bluetooth Low Energy) at venues where smartphones are unable to establish a stable Internet connection, such as large concert venues or below-ground entertainment venues.

The potential use cases for VCs span all kinds of everyday scenarios. Once the OID4VP standardization process is complete and it truly starts to become widespread, we will no doubt encounter this cross-device flow in many aspects of our daily lives.

3.5 SIOPv2's Cross-Device Self-Issued OP

Self-Issued OpenID Provider v2[*15] (abbreviated SIOPv2) is a specification being developed by the OpenID Foundation. It extends OpenID Connect to allow end users to issue ID tokens themselves. The previous specification (SIOP sans v2) was part of the OpenID Connect Core 1.0[*16] specification, whereas SIOPv2 is now being standardized as an independent specification.

With OpenID Connect, an OpenID Provider (OP) issues an ID token that proves the end user's identity, and this is presented to any third party (the Relying Party (RP)) who wants to authenticate the end user. Social login (logging in via an account with Google, Apple, etc.) is a typical example of how this is used with web services. In these cases, Google or Apple or the like is the OP, and the web service is the RP. With SIOP, the end user acts as the OP and issues their own ID token.

The advantage of SIOP is that it allows end users to manage their own IDs, away from the mega platforms' centralized identity management. With social login, the OP is able to collect information on which RP was used. And if a user's OP account is suspended, this will also render the RP's service unavailable to that user. The idea of SSI (Self-Sovereign Identity) is beginning to gain traction as a means of overcoming these undesirable aspects of centralized identity management. The SIOP specification is designed to make OpenID Connect work with SSI.

The SIOPv2 protocol defines two flows. One is the conventional Same-Device Self-Issued OP, in which the RP client application and the OP run on the same device. Redirects are used to link the RP and OP. The other is Cross-Device Self-Issued OP, which was newly added in SIOPv2. Here, the OP runs on a different device (usually a smartphone). Let's take a look at the Cross-Device Self-Issued OP flow (Figure 4).

1. The end user accesses the RP (a).
2. The RP displays the self-issued request URI on screen, usually as a QR code.
3. The end user scans the QR code with a smartphone (b). The self-issued request URI is a deep link that launches the OP.
4. The OP is launched via the deep link. A message requesting permission to issue an ID token is displayed on screen.
5. Once the end user approves, the OP sends the issued ID token to the RP's backend server (c).

In addition to Cross-Device Self-Issued OP, SIOPv2 is expected to have the following enhancements over the previous specification.

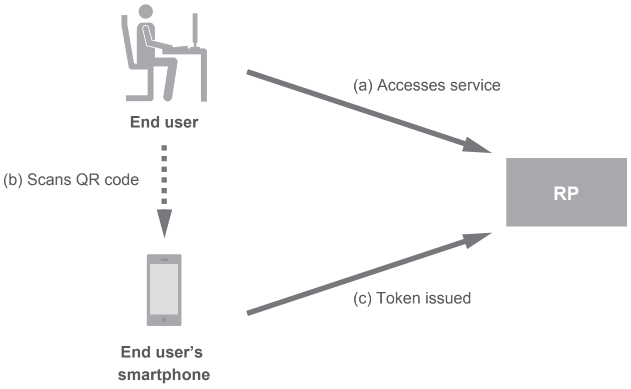


Figure 4: Example of Cross-Device Self-Issued OP Authentication

[*15] Self-Issued OpenID Provider v2 - draft 12 (https://openid.bitbucket.io/connect/openid-connect-self-issued-v2-1_0.html).

[*16] Final: OpenID Connect Core 1.0 incorporating errata set 1 (https://openid.net/specs/openid-connect-core-1_0.html).

- The end user's public key fingerprint is what has so far been used as the end user identifier included in the ID token. In addition to this, v2 will also allow the use of DIDs (Decentralized Identifiers). This will allow the use of an external verifiable data registry.
- When combined with OID4VP, it will allow VCs to be presented together with ID tokens. By verifying the VC, the RP will be able to associate the ID token with a VC issued by a trusted issuer. Since VC verification is completed on the RP (i.e., verifier) side, no information is collected by the VC issuer.

3.6 CTAP v2.2's Hybrid Transports

FIDO2[*17] is an authentication technology for passwordless sign-in to web services put forward by the FIDO Alliance. FIDO2 consists of W3C Web Authentication (WebAuthn)[*18] and corresponding Client to Authenticator Protocols (CTAP)[*19]. The WebAuthn specification is standardized by the W3C in collaboration with the FIDO Alliance. It is designed to facilitate web service sign-ins using biometric authentication entities, called authenticators, and authentication via security keys and the like. The CTAP specification is standardized by the FIDO Alliance. It is designed to allow the use of external authenticators that are not built in but instead connected to a device via USB or NFC.

CTAP v2.2, currently in the drafting phase, proposes a protocol called hybrid transports for using smartphones as external authenticators. In short, this would allow the use of a smartphone for authentication when signing in to a web service on a PC or the like. A number of operators already offer similar solutions, but they are all proprietary implementations. The FIDO Alliance is endeavoring to standardize the protocol. Authentication using hybrid transports will apparently be called FIDO Cross-Device Authentication flow (CDA)[*20]. Incidentally, there is also the somewhat similar sounding Multi-Device FIDO Credentials[*21]. This provides a mechanism for synchronizing credentials (authentication credentials) across an end user's own devices, and is also known as Passkeys[*22]. CDA and Passkeys are separate specifications, and hybrid transports can be used between PCs and smartphones even when credentials are not synchronized via Passkeys.

Figure 5 shows an example sign-in procedure using hybrid transports.

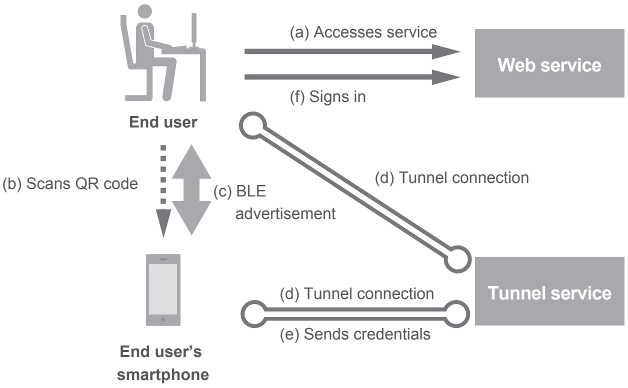


Figure 5: Example of Sign-in Procedure using Hybrid Transports

[*17] User Authentication Specifications Overview - FIDO Alliance (https://fidoalliance.org/specifications/).

[*18] Web Authentication: An API for accessing Public Key Credentials - Level 3 (https://www.w3.org/TR/webauthn-3/).

[*19] Client to Authenticator Protocol (CTAP) (https://fidoalliance.org/specs/fido-v2.2-rd-20230321/fido-client-to-authenticator-protocol-v2.2- rd-20230321.html).

[*20] Terms - passkeys.dev (https://passkeys.dev/docs/reference/terms/#cross-device-authentication-cda).

[*21] White Paper: Multi-Device FIDO Credentials - FIDO Alliance (https://fidoalliance.org/white-paper-multi-device-fido-credentials/).

[*22] Passkeys (Passkey Authentication) (https://fidoalliance.org/passkeys/).

1. Using a PC, the user opens the sign-in screen on a FIDO2-enabled website (a).
2. A dialog box for selecting the authenticator is displayed. The user selects “Smartphone”.
3. A QR code appears on screen.
4. The user scans the QR code using their smartphone (b).
5. The authentication application on the smartphone starts up.
6. To reduce the risk of phishing, at this point BLE advertisement is used to confirm that the PC and smartphone are in close proximity to each other (c).
7. The end user provides fingerprint or other authentication.
8. WebSocket is used to establish a reliable, secure communication link between the authentication application on the smartphone and the web browser on the PC (d). The tunnel specification is up to the implementer.
9. The authenticator application provides the credentials to the web browser through the tunnel (e).
10. The web browser uses the credentials to perform a WebAuthn sign-in (f).

Once the tunnel link is established, the QR code scanning step is skipped in subsequent authentications.

FIDO2 is specially designed to replace website sign-in procedures, so it can be used in combination with OAuth/OpenID Connect. Hence, it is expected that cross-device flows based on hybrid transports could be adopted for most of the areas covered by OAuth/OpenID Connect. While it is still in the drafting phase, the specification does have great potential when it goes into practical use.

3.7 Conclusion

In this chapter, I have introduced some cross-device flow specifications, both standardized ones and some still being drafted. Each has its own characteristics and target use cases. Yet they all use the features and functionality of smartphones (high penetration rate, always-on mobile, advanced biometric authentication, QR code support, push notification support, etc.) with the aim of providing safer, easier-to-use authentication and authorization flows. As cross-device flows become more prevalent, we can expect the security of online services and transactions to improve, providing an even better experience for users.

Kenzo Yotsuya

Research Laboratory, Internet Initiative Japan Inc.

Mr. Yotsuya is engaged in research and development on technologies related to next-generation authentication and authorization.

30 Years of IIJ and DNS

4.1 Introduction

IIJ was the first commercial service in Japan to provide Internet access, something previously limited to academic institutions. That was in November 1993. This year marks 30 years since. In this chapter, we look back over the past 30 years with an eye on DNS..

4.2 1990s: Working with Connectivity Services

No DNS

IIJ was established in December 1992 as Internet Initiative Planning Inc., subsequently taking on its current name the following May. In July 1993, it launched the UUCP Service, and then in November, it launched its Internet Connectivity Service, Japan's first commercial Internet connectivity service.

DNS is a service that uses host names to look up IP addresses and is essential for using the Internet. It is therefore common for a caching DNS server to be bundled in when you subscribe to a connectivity service. That's par for the course these days, so you might think that IIJ's history with DNS started back when it launched its first service, but that was not the case. We only provided connectivity services via dedicated lines. A caching DNS server was not included.

Although ours was Japan's first service, most of our users were familiar with the Internet as an academic network. The Internet is a decentralized or distributed system, as opposed to a centralized one. The participants are equals, and they either arrange what they need themselves, or help each other out in the spirit of mutual aid when things are lacking. This seems to have been the common understanding among IIJ and its users. IIJ's position was that it provides everything to get you connected to the Internet, and that you are on your own from there in terms of arranging DNS, email, and online news. And the users seem to have regarded this as normal.

The first DNS servers

Caching DNS servers were first made available to users in May 1994 with the Dialup IP Service. This did not provide a permanent connection over a dedicated line. Instead, users connected via a phone line only when they needed to, making the service accessible to small businesses and individuals as well. Tele-Hodai, NTT's nighttime flat-rate phone service, had not yet been launched (it appeared in 1995), so the service was used in much the same manner as people used the traditional dialup online services that were popular at the time—that is, they would connect, obtain the information they needed, and then immediately disconnect. Unlike with dedicated line connections, this mode of use does not really fit with the idea of users arranging the necessary servers, and so IIJ ended up providing caching DNS in this case.

Even after IIJ started providing caching DNS servers, its position that users should arrange their own authoritative DNS server did not change. Even so, per the textbook description of DNS, you need two servers, the primary and the secondary, and it was not easy back then for users to furnish both, so there were cases in which IIJ looked after the secondary. And this seems to have been done as part of the mutual aid that Internet users provided to each other as equals, rather than for business motives. If you can believe it, the secondary DNS zone was stored on a Dialup IP Service caching DNS server. While this is unthinkable by today's standards, things were small in scale at the time, and so it was deemed reasonable to share resources.

JP domains were an important aspect when it came to secondary DNS being operated in the spirit of mutual aid. A document giving background to IIJ's involvement is even still available via JPNIC[*1].

In 1994, IIJ took on the role of JP domain secondary, and it has been doing this ever since. Its efforts since then have gone beyond mere assistance among peers. To help with the stable operation of JP domains, it has actively incorporated advanced functionality, which has included providing early support for IPv6 in 2001 and establishing an overseas presence and providing support for anycast.

Rise of the Internet

The mid-1990s was when the Internet started to become more accessible to individual consumers. INTERNET

[*1] JPNIC, "Report on DNS Management Group's activities (including some background on agenda items)" (https://www.nic.ad.jp/ja/materials/committee/1994/0510/shiryou-2-4-1.html, in Japanese).

magazine was first published by Impress in September 1994, and Windows 95 with TCP/IP as standard was released in Japan in November 1995. Many other ISPs beyond IIJ began to pop up around this time.

IIJ4U was launched in December 1996 to serve demand for personal Internet connectivity. The equipment was designed specifically for large-scale consumer services, and this was the first time that IIJ put two caching DNS servers on different network segments to ensure availability, something that is commonplace nowadays.

Around that time, IIJ also started providing caching DNS servers as part of its business connectivity services, but the basic approach of the customer being responsible for building and operating the servers themselves remained unchanged, so the idea was that IIJ would provide the servers only when the customer was absolutely unable to. It was in November 1997 when the IIJ Economy service made them standard.

The term SOHO has perhaps faded out of the popular vernacular somewhat by now. Short for small office / home office, it refers to the concept of using a home or a small office as your workplace. IIJ Economy was a low-cost leased line service for the SOHO market, and unlike with its traditional business connectivity services, IIJ did not expect users to set up and operate their own servers. That is, with this service, IIJ would arrange the caching DNS servers.

That is how the foundations of caching DNS services were established. IIJ subsequently released all sorts of connectivity services, including ADSL, optical fiber, VPNs, and mobile, but even with these changes in line types, the basics of caching DNS remained unchanged, albeit with enhancements to facilities and equipment.

4.3 2000s: Launch of DNS-only Services

The year 1999 saw a huge run-up in US stock market prices centered on Internet-based businesses, and even companies with nothing to do with the Internet saw their stock prices double simply after renaming to include .com in their names[*2]. The extraordinary market highs ended with the crash of 2001, and the episode is now remembered as the dot-com bubble.

With Japan remaining plagued by the Heisei Recession and the Employment Ice Age following the collapse of its bubble economy in 1991, it did not see a dot-com bubble like the US. Yet many of the companies experiencing significant growth during this era had a connection to IT, one such example being SoftBank, which saw its market cap expand to the point that it was second only to Toyota Motor.

While the stock price bubble was only temporary, the practice of companies owning their own domains was here to stay. This became commonplace at businesses across the board, not just major corporations and Internet-related names. Even if you buy a domain, you still can't use it unless you register it with an authoritative DNS. IIJ's stance up to this point had been that customers should handle their authoritative DNS themselves if they needed it, but that required quite a bit of expertise. The era of the Internet only being for those few people who "got it" was already over by this point, and there was rising demand for registering information on the authoritative DNS system even among people with no expertise in server operations.

In response, IIJ launched DNS-only services in March 2000: the DNS Outsourcing Service, which enabled overall authoritative DNS operations and the editing of zone information via the web; the DNS Secondary Service, through which IIJ handled secondary servers only; and the Domain Management Service for managing and maintaining domain registration.

Up to this point, JP domains were classified based on attributes such as co.jp (companies) and ac.jp (academic institutions). A domain is something that represents an organization, so you could only register one domain per organization. Then in 2001 came the release of a general-purpose JP domain system that did not tie domains to specific organizations or place limits on the number of domains registered. This prompted an increase in the number of domains being registered not

[*2] Burton G. Malkiel, Chapter 4, A Random Walk Down Wall Street.

just for individual organizations but also for specific products and brands. And so the number of domains registered and the use of the DNS Outsourcing Service / Secondary Service was growing every year.

It was no longer uncommon by this point for individuals with no organizational affiliation to also have their own domains. So in March 2002, we launched the IIJmio Personal Domain Service, making it possible for people to use email, web, and DNS hosting via their own domain at a low price although without all the bells and whistles, and then in March 2003, we launched the DNS hosting-only IIJmio Simple DNS Service.

The three services previously mentioned—DNS Outsourcing Service, DNS Secondary Service, Domain Management Service—all went on to become long-lived services running for over 20 years, with features and capacity additions being made along the way. Of particular significance here were DNSSEC support and the Site Failover Option, an optional add-on service.

DNS is one of the Internet's essential component technologies, but the protocol was first designed back in the 1980s, so it also has shortcomings that are the result of certain issues either not being envisioned or not being seen as a problem. One such shortcoming is that it is difficult to detect when response packets are forged as part of cache poisoning or man-in-the-middle attacks. These require a lengthy discussion, so I will skip the details, but it was DNSSEC[*3] that made it possible to sign DNS information so that response recipients can confirm the authenticity of the information by validating signatures.

DNSSEC support commenced on the DNS root servers in July 2010, and the JP zone was DNSSEC signed in December 2010. IIJ's Domain Management Service and DNS Outsourcing Service added DNSSEC support in January 2011. DNSSEC necessitates some complicated work not previously involved in DNS operations, including generating signature keys and signing zones. To make DNSSEC available without the hassle, we set this up so that it would be done automatically on the IIJ server side.

As information stored in the DNS system was static, any change to the response required a manual rewrite of the information. The Site Failover Option released for IIJ's DNS Outsourcing Service in March 2015 improved webserver availability by making it possible to monitor webservers externally, quickly remove any server encountering some sort of failure from the DNS response and switch to a standby server, and automatically return it to the DNS information once the webserver had recovered.

4.4 2010s: Wrestling with Attacks

The rise of DDoS attacks

The 2010s saw DDoS (distributed denial of service) attacks by botnets increase in scale, and we had to scramble to implement countermeasures.

DDoS attacks saturate server processing capacity by flooding the server with simultaneous requests from a large number of devices, resulting in a loss of availability. In the 2000s, computer viruses evolved rapidly into what are called worms, which, upon infecting a device, are able to spread themselves broadly to other systems, and they subsequently developed the ability to coordinate to form botnets. DDoS attacks in which the many bots comprising a botnet act at the behest of an attacker sending commands became a frequent occurrence all over the world.

Internet traffic, meanwhile, continued to grow rapidly, with the use of CDNs (content delivery networks) designed to efficiently deliver web content also spreading, and small, garden-variety DDoS attacks were no longer able to bring down services running on servers designed to handle huge amounts of traffic.

[*3] For example, see "What is DNSSEC?" (https://jprs.jp/dnssec/doc/dnssec.pdf, in Japanese).

Now, with DNS, only a few hundred bytes at most are exchanged at any one time, and because of the efficient caching mechanism too, the load on CPU, memory, and network bandwidth resources is miniscule relative to the loads imposed by web and email protocols and the like. So while the performance and bandwidth of webservers have continued to rise, it has long been the case that DNS servers are allocated only a minimum of resources.

In many cases, the attacker's goal is to render a website unavailable. The target of an attack need not be the webserver itself if this goal can be achieved by other means. To access a website, you first need to know the site's IP address, so the goal can be also achieved by interrupting the mechanism for obtaining that IP address—i.e., the authoritative DNS server. Rather than targeting webservers protected by CDNs and thus able to withstand the onslaught of large amounts of attack resources, it is more efficient for attackers to target authoritative DNS servers, which are easily saturated by modest loads.

The October 2016 DDoS attacks on Dyn (later acquired by Oracle) are a prominent example of this[*4]. Dyn was a truly major provider of authoritative DNS services, with prominent global web services such as Twitter and Spotify being hosted by Dyn. Dyn's servers were the subject of DDoS attacks emanating from hundreds of thousands of devices over a period of six hours, rendering them unable to return a response, which made many of the domains using Dyn unreachable.

In 2012, four years before the attacks on Dyn, IIJ also suffered a large-scale DDoS attack against an authoritative DNS server. The attack was targeted at the domain of a customer whose web and DNS systems were hosted by IIJ. The attackers initially attacked the webserver but subsequently realized that the server was performant enough that they would not be able to bring it down. So they refocused their sights on authoritative DNS. IIJ's authoritative DNS servers had what was an abundance of resources for the time, but this was utterly insufficient to withstand a DDoS attack designed to marshal enough resources to take down a broadband webserver, and thus the target server struggled to respond to requests.

This incident prompted a major shift in DNS server design philosophy at IIJ. The DNS server network configuration changed significantly from what it had been before. Multiple defenses were implemented, which included ensuring sufficient bandwidth to withstand saturation attacks, creating a mechanism for using anycast to localize the impact even if bandwidth was saturated, and isolating the DNS servers on a dedicated network so that the impact of attacks would not ripple into other services. The equipment configuration changes incorporating these measures were rolled out progressively on both the authoritative DNS servers used by the DNS Outsourcing Service and the like and the caching DNS servers used in our connectivity services.

■ Open resolver challenges

IIJ hasn't been exclusively on the receiving end of DDoS attacks. IIJ's DNS servers have, unfortunately, also been used as a springboard for DDoS attacks.

DNS primarily uses UDP as its lower-layer protocol, and UDP makes it easier for clients to spoof IP addresses than TCP. The DNS response packet size can also be tens to hundreds of times larger than for queries. A malicious attacker can take advantage of this by sending queries with the source IP address spoofed to be that of the target to a DNS server that will act as a stepping stone, such that it returns a response to that IP address with an amplified packet size, which can saturate the target network's bandwidth. Such attacks are called DNS amplification attacks (DNS amp) or DNS reflection attacks (Figure 1).

[*4] Wikipedia, "DDoS attacks on Dyn" (https://en.wikipedia.org/wiki/DDoS_attacks_on_Dyn).

DNS amp attacks work because DNS servers return a response to spoofed IP addresses, so these attacks could be prevented by not responding to queries from spoofed IP addresses. But because of the way UDP works, it is difficult to detect spoofing.

Caching DNS servers are usually set up for use by a single organization, so only making it available to users within that organization is generally fine. Not responding to external queries would mean that large responses are not sent outside of the organization, even if source IP addresses are spoofed, thus making it impossible to use the organization's server as a DNS amp stepping stone. Yet for a long time in the wake of the Internet's spirit-of-mutual-aid era, it remained uncommon to impose such restrictions on caching DNS servers. IIJ's caching DNS servers were no exception; they too were open resolvers with no access restrictions.

DDoS attacks were prevalent during this period, and malicious attackers began turning their gaze to these open resolvers and using them as stepping stones. We knew that restricting access would prevent our servers from being used as stepping stones, but doing so would also make them unavailable to users using IIJ's servers for their intended purpose and not nefarious ones. After struggling with this dilemma at length, in December 2013 we finally changed our settings to prevent access to IIJ's caching DNS servers from outside of the IIJ network.

Open resolver countermeasures did not end here. While we had dealt with IIJ's DNS servers, there were many cases in which caching DNS servers installed by users and the DNS functions of users' routers were acting as open resolvers and thus used as attack stepping stones, so we had to contact these users and ask them to take appropriate steps. Thanks not only to IIJ's efforts in this regard but also to diligent efforts around the world, the incidence of DDoS attacks leveraging DNS began to die down.

4.5 2020s: Further Developments

■ Encrypted DNS

DNS is public information. So initially, the emphasis was on the information not being tampered with (integrity) rather than on it not being eavesdropped (confidentiality). DNSSEC, released in 2010, is also a mechanism for ensuring integrity. But following the Snowden Incident[*5] in 2013, revealing that the US NSA was collecting large amounts of personal information, the Internet Engineering Task Force (IETF, a volunteer organization tasked with developing Internet standards) declared that "pervasive monitoring is an attack"[*6] and called for future Internet protocols to be equipped with mechanisms for mitigating widespread surveillance. The Snowden incident revealed

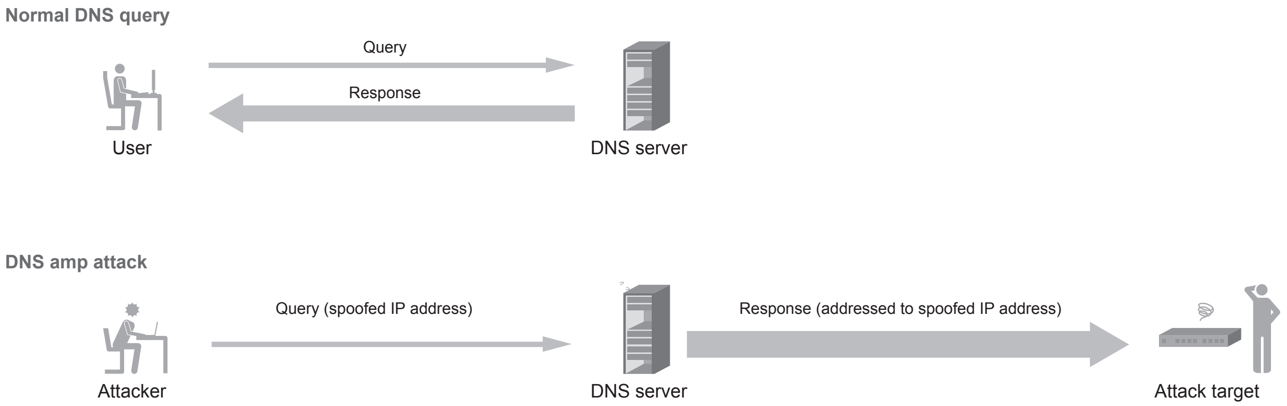


Figure 1: DNS amp

[*5] Wikipedia, "Edward Snowden" (https://en.wikipedia.org/wiki/Edward_Snowden).

[*6] RFC 7258: Pervasive Monitoring is an Attack.

© Internet Initiative Japan Inc.

that DNS was also a target of surveillance, and so it was concluded that, despite DNS being public, the information being sought was a matter of personal privacy and that confidentiality would be crucial to DNS going forward.

This led to the DNS over TLS (DoT) and DNS over HTTPS (DoH) standards. Traditional DNS puts DNS messages directly on top of UDP or TCP, but DoT and DoH put DNS messages on top of (respectively) TLS and HTTPS layers to prevent eavesdropping by third parties.

From 2018 to 2019, public DNS services such as Google Public DNS and Cloudflare 1.1.1.1 adopted DoT and DoH one after another, with client-side support via web browsers and OSs continuing to roll out.

Some issues still remain at present. DoT and DoH only encrypt communications between clients and caching DNS servers, not between caching DNS servers and authoritative DNS servers, and mechanisms for automating DoT/DoH servers are not yet widespread. That said, these protocols are expected to play an important role in the future.

This led to the May 2019 launch of the IIJ Public DNS Service, an experimental service for the purpose of verifying the technology. DoT and DoH do not use UDP and thus do not carry the risk of being used in DNS amp attacks, so we made servers running these protocols available as open resolvers to non-IIJ users. Support for DoT and DoH has since gradually been expanded to caching DNS services used in connectivity services.

■ New authoritative DNS service

Since the dawn of the Internet, people have been saying that multiple authoritative DNS servers should be set up to improve availability. Although the number of DDoS attacks using DNS as a stepping stone fell away, DDoS attacks themselves have actually been on the rise since. As such, an idea that is gaining traction in recent years, particularly for large sites, is that of distributing zones among multiple DNS operators so that name resolution can continue even if any particular operator experiences a fault that renders its servers unresponsive.

Although we had continued to enhance the features of the DNS Outsourcing Service and DNS Secondary Service launched in 2000, the focus was on the basic function of receiving and responding to queries. They did not have functionality for coordinating among multiple operators. For this reason, we set about overhauling the services, which also included bolstering other administrative functionality, and these efforts culminated in the release of the IIJ DNS Platform Service in November 2019. The service allows admins to freely configure their systems in ways that were not possible with the previous services. They can, for instance, use IIJ as the primary server and another operator as the secondary, or DNSSEC sign zones on the IIJ server transferred from the user's primary server. We also launched the IIJ DNS Traffic Management Service in March 2022 as successor to the Site Failover Option.

4.6 Conclusion

We have taken a whirlwind tour back through IIJ's 30-year history with a focus on DNS. Many other pertinent anecdotes could have filled these pages, but space limitations necessitated their omission.

The DNS protocol has been around since the old days, but is not stagnant and is constantly evolving. Now, as ever, it remains a cornerstone of the Internet's foundations. Looking ahead, IIJ plans to continue actively incorporating advanced features while providing robust, flexible DNS services.

Takanori Yamaguchi
Application Service Department, Network Division, IIJ. Mr. Yamaguchi works on the development of DNS services and the like.

August 2023 Vol.59

Internet Initiative Japan

About Internet Initiative Japan Inc. (IIJ)

IIJ was established in 1992, mainly by a group of engineers who had been involved in research and development activities related to the Internet, under the concept of promoting the widespread use of the Internet in Japan.
IIJ currently operates one of the largest Internet backbones in Japan, manages Internet infrastructures, and provides comprehensive high-quality system environments (including Internet access, systems integration, and outsourcing services, etc.) to high-end business users including the government and other public offices and financial institutions.
In addition, IIJ actively shares knowledge accumulated through service development and Internet backbone operation, and is making efforts to expand the Internet used as a social infrastructure.

The copyright of this document remains in Internet Initiative Japan Inc. ("IIJ") and the document is protected under the Copyright Law of Japan and treaty provisions. You are prohibited to reproduce, modify, or make the public transmission of or otherwise whole or a part of this document without IIJ's prior written permission. Although the content of this document is paid careful attention to, IIJ does not warrant the accuracy and usefulness of the information in this document.

©Internet Initiative Japan Inc. All rights reserved.
IIJ-MKTG020-0057

Internet Initiative Japan Inc.

Address: Iidabashi Grand Bloom, 2-10-2 Fujimi, Chiyoda-ku,
Tokyo 102-0071, Japan
Email: info@iij.ad.jp URL: https://www.iij.ad.jp/en/