국가정보원 "한미 합동 사이버보안 권고문" 경고

의료기관 대상으로 유포되는 北 마우이 랜섬웨어

요약

1. 2023년 2월 9일, 국가정보원은 “한미 합동 사이버 보안 권고문 ” 발표
   북한 정권 지원 랜섬웨어 위협 행위자들의 전술/기술과 침해지표 공개
2. 위 권고문에서 언급된 마우이(Maui) 랜섬웨어는
   공중 보건 및 의료 조직들을 대상으로 공격
3. 한국의 경우, 중소병원 일반 사용하는 오픈소스 메신저
   "X-PopUp"을 이용하여 유포했을 가능성 존재
4. 암호화 작업 수행 시 스레드 개수 조절 가능.
   스레드를 적게 사용하여 PC 영향 최소화.
   성능 및 속도 저하 현상이 없어 PC 사용자가 눈치채기 어려움.

대응 방안

1. 논리적 망분리를 적용하여 악성코드 PC 유입을 원천 차단한다.
2. AV(패턴기반탐지) + EDR(행위기반탐지) 솔루션을 최신 형상으로 유지한다.
3. PC 취약점을 주기적으로 점검, 보완한다.
4. 신뢰할 수 없는 메일의 첨부파일은 실행을 금지한다.
5. 비 업무 사이트 및 신뢰할 수 없는 웹사이트의 연결을 차단한다.
6. OS나 어플리케이션은 최신 형상을 유지한다.

목차

1. 개요

1.1 배경

2023년 2월 9일, 국가정보원은 “韓美 합동 사이버 보안 권고문 ”[01]을 발표했다. 미국의 국가안보국(NSA), 연방수사국(FBI), 사이버인프라보안청(CISA), 보건복지부(HHS), 그리고 대한민국의 국가정보원(NIS), 777사령부(DSA) 등 6개 기관이 참여했다. 이 권고문은 북한 정권이 지원하는 랜섬웨어 위협 행위자들의 TTP(Tactics, Techniques, and Procedures)와 침해지표들을 공개하여 랜섬웨어로부터 기관들을 보호하고, 주요 인프라 분야 담당 기관들에게 경각심을 고취시키는데 그 목적이 있다.

※ 독자제재 지정 대상

개인(4명)	박진혁
	조명래
	송림
	오충성
기관(7개)	조선엑스포합영회사
	Lazarus Group
	Bluenoroff
	Andariel
	기술정찰국
	110호 연구소
	지휘자동화대학(미림대학)

[그림 1] 정부가 발표한 북한 관련 제재 대상 (자료: 외교부)

01 NIS <국정원 '한미 합동 사이버 보안 권고문' 발표>

※ 독자제재 추가 지정 대상

개인(4명)	리성운
	김수일
	이석
	AMTCHENTSEV Vladlen(남아공)
기관(5개)	송원선박회사
	동흥선박무역회사
	대진무역총회사
	Transatlantic Partners Pte. Ltd(싱가포르)
	Velmur Management Pte. Ltd(싱가포르)

[그림 2] 정부가 발표한 북한 관련 추가 제재 대상 (자료: 외교부)

북한 관련 랜섬웨어 행위자들이 악의적으로 취득한 암호화폐는
북한의 체제 유지와 첩보 활동을 위해 사용된다는 평가를 받는다.
첩보 활동은 韓美 정부와 국방망 및 방산업체를 대상으로 한다.
정부는 권고문 발표와 동시에 북한의 불법 사이버 활동을 통한 외화벌이에 대응하기 위해
개인 4명과 기관 7개에 대한 제재를 발표했다.[02]

더불어 2023년 2월 20일, 북한의 거듭되는 미사일 도발에 대응하기 위한 북한 대상 금융제재를 발표했다.[03]
제재 대상이 되는 북한과 외환거래 또는 금융거래를 하기 위해서는
한국은행 총재 및 금융위원회의 사전 허가가 필요하며,
허가를 받지 않고 거래하는 경우 관련법에 따라 처벌받을 수 있다는 내용이다.

사이버 보안 권고문에서 언급되는 북한의 랜섬웨어 중 마우이(Maui) 랜섬웨어는
공중 보건 및 의료 조직들을 대상으로 공격을 해왔다.
미 연방수사국은 2021년 5월부터 마우이 랜섬웨어의 공격에 대응했다고 전했으며,
국내 보건 및 의료 인프라도 예외가 아니었다. 공격자는 대한민국의 중소병원이 일반적으로 사용하는
오픈소스 메신저인 X-PopUp(빨간전화기로도 불림)을 사용하여
악성코드를 유포했을 가능성이 존재한다고 밝혔다.

02 외교부 <사이버 공간을 활용한 북한의 불법적 외화벌이 차단>

03 외교부 <우리 정부의 대북 독자제재 추가 지정>

[그림 3] X-PopUp(빨간전화기) 사용자 모임 페이지

X-PopUp은 P2P(Peer-to-Peer Network) 기반 메신저로 사용자 간의 채팅, 파일 전송 등을 지원한다.
문제는 파일 전송기능에서 발생한다. X-PopUp의 파일 전송은 수신자의 동의 없이 시스템에
파일을 업로드할 수 있다. 따라서 메신저가 설치되어 있고
통신이 가능한 PC는 모두 악성코드 유포 대상이 된다.

사용의 간편함과 프리웨어라는 장점이 있는 반면, 공격에 취약하다는 단점이 존재하기 때문에
해당 메신저를 사용하는 조직은 프로그램 교체가 불가피한 상황이다.
해당 프로그램의 업데이트는 중단된 상태로 사실상 보안 패치를 기대할 수 없어
백신 회사들은 X-PopUp을 악성 프로그램으로 탐지하고 있다.
단, 파일 전송을 하더라도 실행은 불가하기에 원격 실행이 가능한
추가적인 악성코드(e.g. RAT)가 설치되어야 할 것으로 보인다.

1.2 파일 정보

Name	maui.exe
Type	PE32
Behavior	File encryption
SHA-256	5b7ecf7e9d0715f1122baf4ce745c5fcd769dee48150616753fec4d6da16e99e
Description	Maui Ransomware

[표 1] Maui 랜섬웨어 파일 정보

2. 분석

[그림 4] 마우이 랜섬웨어 실행 순서도

① 명령줄 옵션 확인
- 마우이 랜섬웨어를 옵션과 함께 실행하였는지 확인한다.

② 공격자 공개키 획득
- 마우이 랜섬웨어에 숨겨놓은 공격자의 RSA 공개키를 획득한다.

③ RSA 키 페어 생성
- 런타임에 RSA 키 페어를 생성한다. 공개키는 AES 대칭키의 암호화에 쓰인다.
  생성한 키 페어는 각각 인코딩 또는 공격자의 공개키에 의해 암호화되어 피해자의 PC에 저장된다.

④ 암호화 대상 선정
- 입력된 최상위 경로부터 최하위까지 탐색하여 모든 파일 경로를 수집한다.
  단, 이미 감염된 파일은 제외된다.

⑤ 파일 암호화
- 파일마다 고유한 AES 대칭키를 생성하여 암호화한다.

2.1 명령줄 옵션 확인

```
maui.exe [-ptx] [PATH]
```

[표 2] 마우이 랜섬웨어 실행 형식

마우이(Maui) 랜섬웨어의 실행 형식은 위와 같다. 암호화 대상 최상위 폴더 경로를 입력하고,
필요에 따라 옵션을 사용하여 세부적인 설정을 하거나 또는 추가 기능을 이용할 수 있다.

옵션	설명
-p	로그 파일이 저장될 폴더 경로를 지정한다. (기본값 : 랜섬웨어가 위치한 폴더 경로)
-t	파일 암호화 작업을 수행할 스레드의 개수를 지정한다. 암호화 대상의 개수와 크기에 비례하여 조절할 수 있다. (기본값 : 1)
-x	자가 삭제를 하기 위한 배치 스크립트를 실행한다. (기본값 : false)

[표 3] 마우이 랜섬웨어 옵션 목록

사용 가능한 옵션은 세 가지이다.
특이한 점은 암호화 작업을 수행하는 스레드 개수를 조절할 수 있다는 것이다.
이는 활용 가능한 프로세서의 자원을 최대로 동원하여
파일 암호화를 빠르게 끝내는 랜섬웨어들과 대비되는 점이다.
스레드를 최대로 동원하면 작업을 빠르게 마칠 수 있지만,
가용성을 급격히 저하시키므로 피해자가 이상을 감지할 확률이 높다.
따라서 암호화 대상의 파일 개수가 적고 크기 또한 크지 않다면
스레드를 적게 사용하여 피해자가 눈치채지 못하도록 은밀하게 작업할 수 있다.

로그 파일과 자가 삭제에 대한 부연 설명은 아래와 같다.

2.1.1 로그 파일

- 암호화 대상 최상위 폴더 경로
- 암호화 대상 최상위 폴더 크기
- 암호화에 실패한 파일 경로와 실패 사유
- 작업 완료 여부
- 작업 날짜와 시간

로그 파일은 랜섬웨어 실행 시 설정한 경로에 “maui.log”라는 이름을 가진 파일로 생성된다.
로그는 공격자의 디버깅을 돕기 위해 위와 같은 정보들을 포함한다.

오류 메시지	설명
Create file error	새 임시 파일(암호화된 내용이 저장될 파일) 생성 실패
Unknown file	확인되지 않음
Access denied	원본파일에 대한 핸들 획득 실패
Crypo error	AES 초기화 실패
Incorrect password	유효하지 않은 AES 대칭키
Memory insufficient	파일 버퍼 동적 할당 실패
Shred error	원본파일을 암호화 된 파일로 덮어쓰기 실패
User Cancelled	확인되지 않음
bad allocation	확인되지 않음

[표 4] 실행 오류 관련 메시지 목록

실패 사유 목록은 위와 같다. 메시지에 간혹 오타가 존재한다.

2.1.2 자가 삭제

```
*(_DWORD *)v0 = 'g\0d';                                 // L"dgod.bat\x00"
*((_DWORD *)v0 + 1) = 'd\0o';
*((_DWORD *)v0 + 2) = 'b\0.';
*((_DWORD *)v0 + 3) = 't\0a';
v0[8] = 0;
v2 = _wfopen(Buffer, L"wt");
if ( v2 )                                               // 배치 스크립트 생성
{
  GetModuleFileNameW(0, Filename, 0x200u);
  fputs("@ECHO OFF\n", v2);
  fputs(":REPEAT\n", v2);
  fputs("ping localhost -n 1\n", v2);
  fwprintf(v2, L"TASKKILL /IM \"%s\"\n", Filename);
  fwprintf(v2, L"del /f /q \"%s\"\n", Filename);
  fwprintf(v2, L"if exist \"%s\" goto REPEAT\n", Filename);
  fputs("(goto) 2>nul & del \"%~f0\"", v2);
  fclose(v2);
}
_swprintf(Command, L"start /b \"\" cmd /c \"%s\"", Buffer);
_wsystem(Command);                                      // 명령 프롬프트로 배치 스크립트 실행
```

[그림 5] 배치 스크립트 파일 생성

GetTempPathW API를 호출해 획득한 임시 폴더 경로에 “dgod.bat” 배치 스크립트 파일을 저장한다.
“dgod”은 마우이 랜섬웨어에서 종종 볼 수 있는 단어로 반신(demi-god)을 뜻한다.
스크립트는 마우이 랜섬웨어를 종료하고 이미지 파일 삭제 후 배치파일 또한 삭제한다.
해당 API가 임시 폴더를 찾는 순서는 MSDN[04] 에 명시되어 있다.

04 learn.microsoft > fileapi.h > gettemppathw

2.2 공격자 공개키 획득

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
000BEDE0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000BEDF0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000BEE00	30	81	9F	30	0D	06	09	2A	86	48	86	F7	0D	01	01	01	0.Ÿ0...*†H†÷....
000BEE10	05	00	03	81	8D	00	30	81	89	02	81	81	00	B9	08	93	.....0.‰....¹.“
000BEE20	47	B1	44	4E	7C	AA	26	27	6D	01	DD	0A	B8	2D	91	B0	G±DN|ª&'m.Ý.¸-‘°
000BEE30	E9	80	04	E2	2A	45	16	3C	55	5A	5D	FF	67	61	74	FD	é€.â*E.<UZ]ÿgatý
000BEE40	DC	86	97	8A	FA	81	BF	73	3C	E1	7D	38	D5	40	C6	15	Ü†—Šú.¿s<á}8Õ@Æ.
000BEE50	63	F3	A5	E1	EA	A1	CF	3C	AB	43	BB	EF	ED	7C	56	9E	có¥áê¡Ï<«C»ïí|Vž
000BEE60	99	92	3B	46	09	59	84	3E	9A	E5	A8	E4	5E	8C	A0	1F	™';F.Y„>šå¨ä^Œ .
000BEE70	5C	64	6E	68	20	9D	7D	74	8D	7E	59	E0	AC	4B	61	8F	\dnh .}t.~Yà¬Ka.
000BEE80	8C	7D	A1	41	E5	ED	54	27	15	12	E7	FD	B3	07	56	A6	Œ}¡AåíT'..çý³.V¦
000BEE90	90	31	D0	5A	81	FC	1A	80	2B	B6	BA	CC	95	02	03	01	.1ÐZ.ü.€+¶°Ì•...
000BEEA0	00	01	4B	42	55	50	01	00	00	00	A2	00	00	00			..KBUP....¢...

[그림 6] 바이너리 끝에 위치한 데이터의 구조

오프셋	크기	설명
-12	4	공격자 RSA 공개키 시그니처 (PUBK)
-8	4	키 호환 여부
-4	4	공격자 RSA 공개키 길이
-12-a	a	공격자 RSA 공개키

[표 5] 바이너리 끝에 위치한 데이터의 구조

공격자의 RSA 공개키는 마우이 랜섬웨어 바이너리의 끝에 위치해 있다.
마지막 12바이트는 공개키의 메타데이터이고, 그보다 앞에는 하드코딩 된 공개키가 존재한다.
메타데이터에 공개키의 길이가 포함되어 있는 것으로 미루어 봤을 때,
마우이 랜섬웨어 빌드 시 키의 크기는 가변적으로 조절 가능해 보인다.
이 공격자의 공개키는 이후 생성할 RSA 키 페어 중 비밀키를 암호화하는 데 쓰인다.

2.3 RSA 키 페어 생성

```
if ( !bits )
  bits = 1024;
rsa = RSA_generate_key(bits, 0x10001, 0, 0);
if ( !rsa )
  break;
```

[그림 7] 런타임 RSA 키 페어 생성

RSA-1024 키 페어를 생성한다. 여기서 생성한 공개키는
이후 파일 암호화에 쓰일 AES 대칭키를 암호화하는 데 쓰이며,
비밀키는 공격자의 공개키에 의해 암호화된다.
암호화된 비밀키는 마우이 랜섬웨어와 동일한 경로에 “maui.evd”라는 이름의 파일로 생성된다.
evd는 증거(evidence)의 준말이다.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	44	49	56	45	01	00	00	00	10	03	00	00	4B	5D	2A	51	DIVE........K]*Q
00000010	0A	72	7C	07	27	69	7C	1F	66	46	2A	49	96	95	D2	BC	.r|.'i|.fF*I-•Ò¼
00000020	AB	96	34	07	88	51	DB	D0	77	26	F9	34	A7	0D	D5	5F	«-4.ˆQÛÐw&ù4§.Õ_
00000030	C2	1A	55	6C	14	E3	E0	71	BF	06	9F	FC	03	CE	37	44	Â.Ul.ãàq¿.Ÿü.Î7D
00000040	66	E6	6E	CD	78	AF	E3	2E	AD	B7	88	BD	4A	F3	50	26	fænÍx¯ã..·ˆ½JóP&
00000050	92	78	2F	23	04	48	9D	1A	07	EC	E9	EA	C9	38	40	AB	’x/#.H...ìéêÉ8@«
00000060	F2	35	C3	D3	35	19	0D	AA	C8	5B	DC	95	6C	E6	B4	7C	ò5ÃÓ5..ªÈ[Ü•læ´|
00000070	62	93	A7	7A	C1	E6	25	A2	5A	31	D3	64	96	CE	DD	54	b“§zÁæ%¢Z1Ód-ÎÝT
00000080	84	07	86	81	A9	4A	30	7B	44	7B	88	0F	7A	F7	B9	1C	„.†.©J0{D{ˆ.z÷¹.
00000090	8D	7D	C0	0D	D4	57	DC	0C	5D	F8	7C	04	55	7A	8C	F9	.}À.ÔWÜ.]ø|.UzŒù

[그림 8] 런타임 비밀키 파일 형식

오프셋	크기	설명
0	4	런타임 RSA 비밀키 시그니처 (EVID)
4	4	키 호환 여부
8	4	암호화된 런타임 RSA 비밀키 길이
12	a	암호화된 런타임 RSA 비밀키

[표 6] 런타임 비밀키 파일 형식

maui.evd 파일의 형식은 위와 같다.

```
hFile = CreateFileA("\\\\.\\PhysicalDrive0", 0x80u, 7u, 0, 3u, 0x20000000u, 0);
if ( DeviceIoControl(hFile, IOCTL_STORAGE_QUERY_PROPERTY, in, 0xCu, &out, 0x400u,
{
```

[그림 9] XOR 키 생성을 위한 디스크 정보 쿼리

```
for ( i = 0; i < v17; ++i )
{
  *((_BYTE *)v3 + i) ^= v5->xorkey[v18];     // XOR 인코딩
  if ( (unsigned int)++v18 >= 0x10 )
    v18 = 0;
}
fwrite(v3, 1u, v17, hFile);
fclose(hFile);
```

[그림 10] 런타임 공개키 XOR 인코딩

공개키는 랜섬웨어와 동일한 경로에 “maui.key”라는 이름의 파일로 생성된다.
런타임 공개키는 XOR 키로 인코딩하여 저장한다.
XOR 키는 0번 디스크의 제품 ID와 시리얼 번호를 조합해 16바이트의 크기로 생성된다.
해당 키는 두 가지 용도로 쓰인다. 런타임 RSA 공개키를 인코딩하는 데 쓰이며
동시에 AES CBC 블록모드의 IV(Initialize Vector)로 사용된다.
디코딩한 공개키는 첫 4바이트로 0x44474F44(DGOD) 시그니처를 가진다.

2.4 암호화 대상 선정

일반적인 랜섬웨어는 EXE나 DLL과 같이 파일의 크기가 커서 암호화 속도를 저하시키거나,
시스템에 악영향을 줄 수 있는 주요 파일은 암호화 대상에서 제외시킨다.
그러나 마우이 랜섬웨어는 이미 감염된 파일을 제외한 모든 파일을 암호화시킨다.
감염 여부는 파일의 첫 4바이트 시그니처를 보고 구별한다.
감염 파일의 형식은 “파일 암호화” 장에서 상세히 설명한다.

2.5 파일 암호화

```
*(_DWORD *)key = 'dogd';
RAND_bytes((char *)key + 4, 28);
*((_BYTE *)key + 32) = 0;
return 32;
```

[그림 11] 무작위 AES 대칭키 생성

파일 암호화에 쓰일 AES-256 대칭키를 생성한다.
32바이트 중 첫 4바이트는 항상 0x646F6764(dgod)로 고정되며
그 뒤의 28바이트가 무작위로 정해진다.
대칭키는 매 파일마다 새로 생성되므로 모든 파일은 고유한 대칭키에 의해 암호화된다.

```
if ( !GetTempFileNameW(pwszTempDirPath, L"dgod", 0, pwszTempFilePath) )
{
```

이름	수정한 날짜	유형	크기
dgoAEF5.tmp	2023-02-28 오후 12:36	TMP 파일	11KB
dgoAF63.tmp	2023-02-28 오후 12:36	TMP 파일	727KB
dgoAF64.tmp	2023-02-28 오후 12:36	TMP 파일	11KB
dgoB0CD.tmp	2023-02-28 오후 12:36	TMP 파일	11KB
dgoB2C7.tmp	2023-02-28 오후 12:36	TMP 파일	10KB
dgoB2F7.tmp	2023-02-28 오후 12:36	TMP 파일	1,070KB

[그림 12] 임시 파일 생성

접두어가 “dgod”인 임시 파일 생성을 시도한다.
그러나 GetTempFileNameW API로 임시 파일명을 생성할 때
접두어는 최대 세 글자로 제한되어 있기에 실제로는 “dgo”까지 적용된다.
임시 파일에는 원본 파일을 AES 알고리즘으로 암호화한 데이터가 저장된다.

```
  buf = malloc(0x1000u);
  if ( buf )
  {
    memset(buf, 0, 0x1000u);
    if ( hidword_size >= 0 && (hidword_size > 0 || lodword_size) )
    {
      do
      {
        v7 = lodword_size - lodword_nwritten;
        if ( (__int64)(__PAIR64__(hidword_size, lodword_size) - __PAIR64__(hidword_nwri
          v7 = 4096;
        v8 = fwrite(buf, 1u, v7, hFile);      // 원본파일을 4096 바이트씩 0으로 덮어씀
        if ( v8 <= 0 )
          break;
        v9 = v8 + __PAIR64__(hidword_nwritten, lodword_nwritten);
        hidword_nwritten = HIDWORD(v9);
        lodword_nwritten = v9;
      }
      while ( v9 < __SPAIR64__(hidword_size, lodword_size) );
    }
    free(buf);
  }
  fclose(hFile);
}
_wunlink(lpFileName);                         // 원본파일 삭제
```

[그림 13] 원본파일 덮어쓰기

암호화를 마치면 복구가 어렵도록 원본파일을 0으로 덮어쓴다.
모두 덮어쓰고 난 후에는 원본파일을 삭제한다.

```
if ( _wrename(pwszTempFilePath, pwszOriginalFilePath) != -1 )
  break;
Sleep(1u);
```

[그림 14] 임시파일 이름 수정

파일 암호화가 완료되면 임시 파일을 원본파일과 동일한 경로에 저장한다.
파일 확장자는 별도로 추가되지 않는다. 따라서 피해자는 피해 사실을 즉시 인지하기 어렵다.


```
hFile = CreateFileW(pwszOriginalFilePath, 0x100u, 2u, 0, OPEN_EXISTING, 0x80u, 0);
if ( hFile != (HANDLE)-1 )
{
  SetFileTime(
    hFile,
    &FileInformation.ftCreationTime,
    &FileInformation.ftLastAccessTime,
    &FileInformation.ftLastWriteTime);
  CloseHandle(hFile);
}
```

[그림 15] 파일 타임스탬프 변경

저장이 완료되면 해당 파일의 생성 시간, 마지막 접근 시간, 마지막 수정 시간 등
타임스탬프를 원본 파일과 동일하게 변경한다.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	54	50	52	43	01	00	00	00	57	00	69	00	6E	00	4D	00	TPRC....W.i.n.M.
00000010	65	00	72	00	67	00	65	00	5C	00	44	00	6F	00	63	00	e.r.g.e.\.D.o.c.
00000020	73	00	5C	00	52	00	65	00	61	00	64	00	4D	00	65	00	s.\.R.e.a.d.M.e.
00000030	2E	00	74	00	78	00	74	00	00	00	A3	08	00	00	00	00	..t.x.t...£.....
00000040	00	00	58	C7	27	A3	77	F2	24	85	BC	7B	BB	42	D3	27	..XÇ'£wò$…¼{»BÓ'
00000050	3E	E9	C5	3E	CD	10	7F	25	87	8F	5B	B6	F8	DC	31	A2	>éÅ>Í..%‡.[¶øÜ1¢
00000060	75	8D	ED	A3	76	BA	C3	14	72	AD	E9	8D	A9	98	0A	01	u.í£vºÃ.r.é.©˜..
00000070	CC	B1	78	FB	E8	AC	6D	9C	2B	E2	82	B8	C6	1F	6C	4B	Ì±xûè¬mœ+â‚¸Æ.lK
00000080	22	29	59	A6	3B	C6	D5	09	90	27	A5	E0	44	81	B8	EF	")Y¦;ÆÕ..'¥àD.¸ï
00000090	6F	DA	65	A4	85	AD	49	31	9A	00	AA	23	C9	AD	0E	87	oÚe¤…I1š.ª#É..‡
000000A0	37	9B	21	B3	62	97	1C	DA	B9	D6	24	2A	25	AC	6A	6F	7›!³b—.Ú¹Ö$*%¬jo
000000B0	65	79	AF	17	3A	9C	1D	1E	21	AD	7C	B6	DA	A2	E6	BA	ey¯.:œ..!.¦¶Ú¢æº
000000C0	93	53	AD	28	FE	EE	D0	68	3C	A2	99	01	9C	EA	52	CD	"S.(þîÐh<¢™.œêRÍ
000000D0	90	78	58	7D	71	CA	01	2D	06	5B	F6	BC	03	4F	5F	35	.xX}qÊ.-.[ö¼.O_5
000000E0	AE	CD	7B	27	17	07	5A	5A	FC	98	76	41	C5	EA	EC	57	®Í{'..ZZü˜vAÅêìW
000000F0	E6	3E	92	34	C4	30	9D	1A	07	0C	27	2F	E6	BE	27	2A	æ>'4Ä0....'/æ¾'*
00000100	CA	D2	5B	D1	C3	7C	FE	40	30	3F	D3	C7	7D	D8	98	6D	ÊÒ[ÑÃ|þ@0?ÓÇ}Ø˜m
00000110	84	8D	E8	06	45	86	57	80	06	AA	C1	A7	C3	A5	69	2F	„.è.E†W€.ªÁ§Ã¥i/
00000120	BB	2B	FA	13	B7	38	55	80	B9	83	94	14	5B	09	29	5A	»+ú.·8U€¹ƒ”.[.)Z
00000130	59	8D	5B	65	20	58	79	D9	5E	B2	D5	B5	C0	0E	A0	69	Y.[e XyÙ^²Õµ À. i
00000140	C4	80	B5	6C	10	B2	59	C6	C9	3F	F4	E9	0B	AC	77	B5	Ä€µl.²YÆÉ?ôé.¬wµ
00000150	9F	D3	14	F3	90	AF	BA	DA	8C	A6	C9	B1	57	6F	70	C8	ŸÓ.ó.¯ºÚŒ¦É±WopÈ
00000160	F8	45	2C	CD	2D	04	29	8D	02	BC	3E	B9	51	10	B6	18	øE,Í-.)..¼>¹Q.¶.
00000170	07	88	6F	0C	4D	CC	EC	3E	28	72	8C	41	13	C2	03	E5	.ˆo.MÌì>(rŒA.Â.å
00000180	E8	E3	6B	96	53	E0	7D	BC	06	F1	76	F3	46	AA	DD	C6	èãk–Sà}¼.ñvóFªÝÆ

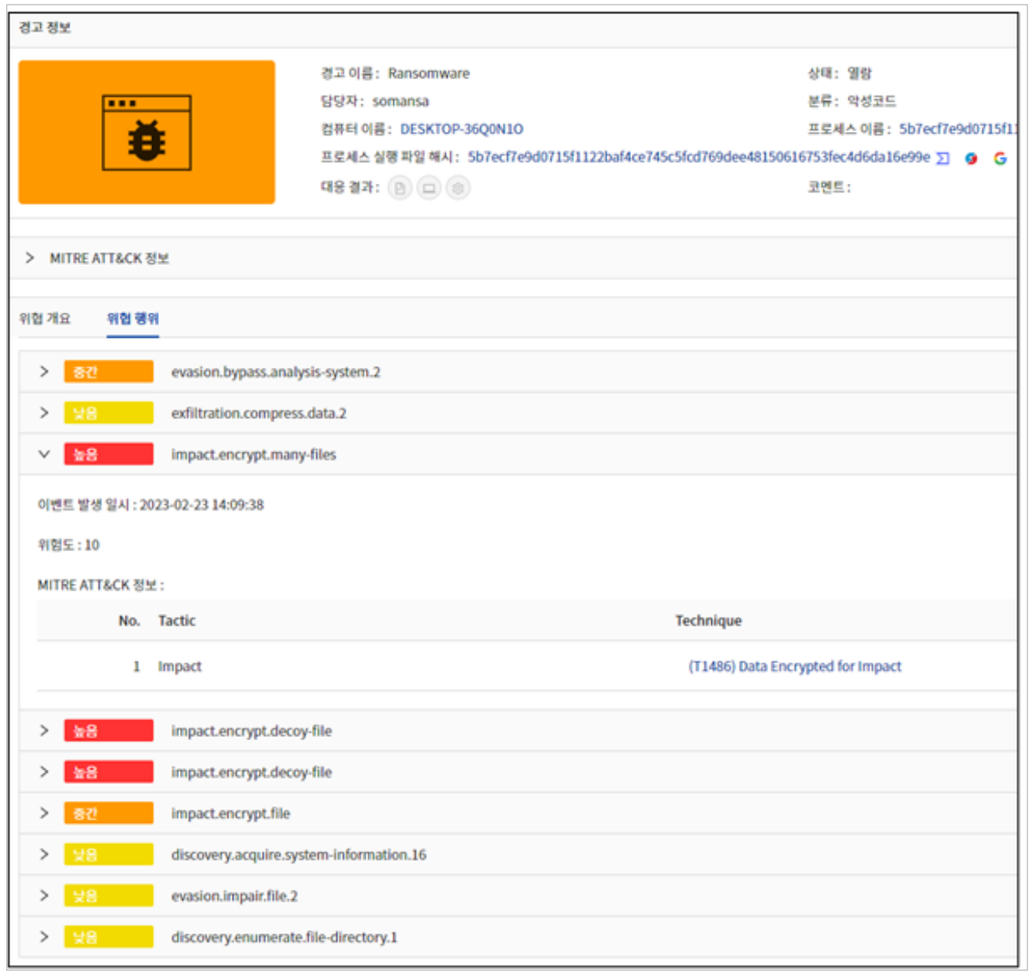
[그림 16] 감염파일 형식

오프셋	크기	설명
0	4	감염 파일 시그니처 (CRPT)
4	4	키 호환 여부
8	a	널 문자로 끝나는 원본파일 경로
8+a	8	원본파일 크기
16+a	128	암호화된 AES 대칭키
144+a	-	암호화된 파일 데이터

[표 7] 감염파일 형식

마우이 랜섬웨어는 파일의 첫 4바이트 시그니처를 확인하여 기존 감염 여부를 확인한다.
또한 매 파일마다 암호화에 사용한 AES 대칭키가 상이하므로
파일에 대칭키를 런타임 RSA 공개키로 암호화시켜 저장하였다.
감염된 파일의 형식은 위와 같다.

3. Privacy-i EDR 탐지 정보



[그림 17] Privacy-i EDR 탐지 정보

Privacy-i EDR은 마우이 랜섬웨어의 파일 암호화 행위를 탐지하여 랜섬웨어로 분류하고 있다.

4. 대응

1. 논리적 망분리를 적용하여 악성코드 PC 유입을 원천 차단한다
2. AV(패턴기반탐지) + EDR(행위기반탐지) 솔루션을 최신 형상으로 유지한다.
3. PC 취약점을 주기적으로 점검, 보완한다.
4. 신뢰할 수 없는 메일의 첨부파일은 실행을 금지한다.
5. 비 업무 사이트 및 신뢰할 수 없는 웹사이트의 연결을 차단한다.
6. OS나 어플리케이션은 최신 형상을 유지한다.

5. 참고자료

5.1 국내 피해 현황

피해사례 있을 것으로 추측되나 알려지지 않음.

5.2 국제 피해 현황

미국의 공중 보건 및 의료 분야가 공격 대상이었다는 점 이외에 확인되지 않음.

5.3 참고 자료

https://www.cisa.gov/news-events/cybersecurity-advisories/aa23-040a
(미국 사이버보안 및 인프라 보안국)

궁금하신 점이나 문의사항은 malware@somansa.com 으로 문의주십시오