

เผยวิธีการทำงานของเจ้าหนอนร้าย

```

xor     ecx, ecx
push   esi
pop    ecx
mov    [edx], ecx
mov    [edx+4], ecx
push   ebx
mov    [edx+8], ecx
push   ebp
mov    [edx+0C], ecx
xor     ecx, ecx
push   esi
    
```

```

loc_935295:
mov    ecx, 15A4E35h
mul    ecx, ecx
add    ecx, 1
add    ecx, 0
xor    [esi], dx
mov    edx, ecx
mul    edx, edx
add    ecx, 1
add    ecx, 0
shr    ecx, 1
xor    [esi+4], dx
mul    ecx, ecx
    
```

นายพล เฟอร์กูสัน นักวิจัยภัยคุกคามชั้นสูง จาก เทรนด์ ไมโคร อิงค์ เปิดเผยว่า จากเหตุการณ์หนอนอันตรายระบอบช่วงวันเมษาหน้าโง่ หรือ April Fool's Day จากการวิเคราะห์ขั้นตอนการตรวจสอบการติดต่อสื่อสารที่ใช้โปรโตคอลเพียร์ทูเพียร์ (P2P) ของหนอนร้าย WORM_DOWNAD.KK พบว่าการกระทำดังกล่าวได้รับการสนับสนุนจากแหล่งที่มาบางแห่ง และค้นพบการทำงานของโค้ดบางอย่างที่สำคัญซึ่งถูกนำมาใช้จากเอกสารที่มีอยู่ย้อนหลังไปเมื่อ พ.ศ.2540

หน่วยงานที่ดูแลด้านการรักษาความปลอดภัยระบบคอมพิวเตอร์ ประเทศฝรั่งเศส และความช่วยเหลือด้านการวิเคราะห์จากหน่วยงานระดับชาติ SRI International เกี่ยวกับวิธีการทำงานของ WORM_DOWNAD.KK p2p ให้ความสนใจไปที่โค้ดเฉพาะในพอร์ตชุดคำสั่งประจำ (โปรแกรมคอมพิวเตอร์ซึ่งถูกเรียกใช้บ่อย) เพื่อแพร่กระจายหนอนร้าย WORM_DOWNAD.KK p2p

รูปที่ 1 - พอร์ตชุดคำสั่งประจำเพื่อแพร่กระจายหนอนร้าย WORM_DOWNAD.KK p2p

โดยเฉพาะค่า "15A4E35h" - ซึ่งเป็น "hard-coded" ต้นกำเนิดสำหรับสม-หมายเลขชุดคำสั่งประจำ (โปรแกรมคอมพิวเตอร์ซึ่งถูกเรียกใช้บ่อย) เพื่อแพร่กระจายหนอนร้าย WORM_DOWNAD.KK นอกจากนี้ ยังพบอีกว่าสิ่งที่สำคัญที่สุด คือโค้ดนี้ไม่ใช่ของใหม่ แต่มันเคยเกิดขึ้นเมื่อปี พ.ศ.2540 ในชื่อ "raZZia" ที่พบในการแพร่กระจายชุดคำสั่งประจำ (โปรแกรมคอมพิวเตอร์ซึ่งถูกเรียกใช้บ่อย)

ความจริงโค้ดดังกล่าวเกือบจะแพร่กระจายเชื้อร้าย WORM_DOWNAD.KK เพื่อใช้เป็นตัวแพร่กระจายชุดคำสั่งประจำ (โปรแกรมคอมพิวเตอร์ซึ่งถูกเรียกใช้บ่อย) โดยกำหนดว่าพอร์ตไหนจะใช้เพื่อการติดต่อสื่อสารแบบเพียร์ทูเพียร์ (P2P) กับเครื่องคอมพิวเตอร์ในเครือข่าย อีกทั้งผู้สร้างหนอนร้าย WORM_DOWNAD.KK ยังเปลี่ยนความต้องการใหม่ แทนที่จะใช้มันเพื่อพุ่งเป้าไปที่ชุดคำสั่งประจำหลัก เปลี่ยนเป็นการสมตัวอย่างหาพอร์ตชุดคำสั่งเพื่อแพร่กระจายเชื้อร้ายต่อไป

```

:00404470 mov  eax, [00400320] ;!-- Put "key" in EAX
:00404475 mul  ecx, ecx, 015A4E35 ;!-- EAX*EAX * 15A4E35
:00404478 shr  eax ;!-- EAX/EAX + 1
:0040447C mov  [00400320], eax ;!-- Replace the "key" with the new value of EAX
:00404481 and  eax, 7FFFFFF0 ;!-- EAX&EAX && 7FFFFFF0
:00404485 shl  ecx, 10 ;!-- EAX*EAX >> 10
:00404489 xor  ecx, ecx

The above code consists of a loop that does through all the letters of the name we entered. With each letter some value is calculated, all these values are added up together (in EAX). Then this value is stored in EAX and the function returns. And that was when we were looking for, we wanted to know how EAX got its value!
    
```

รูปที่ 2 - โค้ด "1997 raZZia"

เช่นเดียวกันกับชุดคำสั่งด้านบน โดยเฉพาะค่าความสำคัญของ "15a4e35h" เป็น hard-coded ในหนอนร้าย WORM_DOWNAD.KK ซึ่งเป็นตัวแปรสำคัญในการแพร่พันธุ์ ทั้งการเลือก key-gen code และ WORM_DOWNAD.KK จะใช้หลักตรรกวิทยาที่ใกล้เคียงกันมากที่สุด โค้ด WORM_DOWNAD.KK จะมีข้อแตกต่างเพิ่มขึ้นมาเล็กน้อย แต่ว่าคุณสามารถที่จะดูได้จากตารางด้านล่าง

```

Both
WORM_DOWNAD.KK Only
Key-Gen Only
    
```

```

Sout = 0;
Sout = Susername[0];
Skey = Sp;
    
```

```

foreach(Skey@Susername){
    #key-gen iterates over each character in a username.
    for (Sj = 1;Sj <= S; Sj < 10){
        #where keygen iterates over characters in a username
        # WORM_DOWNAD.KK performs 10 iterations.
        Skey = Skey & 0xFFFFFFFF;
        # Ensure the incoming value is limited to 32-bit.
        # The keygen uses a stream of ascii one byte at a time
        # So the input is known to never be larger than 8-bit.
        Skey = (Skey * 0x5A4E35) + 1; #multiply key by <token>, add 1.
        Skey = Skey & 0xFFFFFFFF;
        Sout ^= Skey >> 10;
        #Mask out (xor) the first 16-bits and bit-32 and above
        #Result now 21-bits. These 2 steps leave us with the value of Skey
        #that is inhibited the bits above the first 16-bits.
        Sout ^= (((Skey >> 32) >> 0x5, 0xFFFFFFFF) & Skey >> 32); #Skey >> 32 removes the first 32-bits from Skey, this
        # eliminates the original IP address, each iteration keeps fewer
        #and fewer bits of the Skey value. After the second iteration all
        #but the first 16-bits are removed then XOR'd to the output value.
    }
    return Sout;
    
```

รูปที่ 3 - ภาพแสดงตัวอย่างว่าโค้ดนี้เคยถูกใช้โดยอาชญากรได้ดิน

รูปที่ 4 - ตัวอักษรจีนที่ไม่ชัดเจน

เป็นตัวอักษรจีนที่แปลความหมายได้ว่า "บริเวณทุ่งหญ้าที่หนาแน่น" หรือ "ต้นหญ้า, วัชพืชที่หนาแน่น" ซึ่งเป็นคำเหม็บเหม็บที่หมายถึงโฮสต์ที่ติดเชื้อร้ายเป็นจำนวนมาก ตัวอย่างเช่นเครือข่ายบรอดเน็ตขนาดใหญ่

ความจริงแล้ว 15A4E35h คือรูปแบบเลขทศนิยม 22695477 และ 15A4E35 ไม่ได้มีความหมายพิเศษสำหรับ raZZia - เพียงแต่พวกเขาเคยใช้โค้ดดังกล่าว และขณะนี้วิธีการดังกล่าวถูกนำกลับมาใช้อีกครั้งโดยผู้สร้างโค้ด Conficker

นักวิจัยภัยคุกคามสรุปข้อมูลได้ว่า กลุ่มคนบางพวกซึ่งอาจจะเป็นอาชญากรไซเบอร์ หรืออีกคำหนึ่งที่เรามักจะพบ คือ Russkrainians (อาชญากรไซเบอร์ชาวรัสเซียและยูเครน) เลือกใช้ลักษณะบางอย่างของวัฒนธรรมจีน เพื่อก่อให้เกิดผลกระทบ และเกิดความเสียหายแก่ชาวจีน ขณะนี้อาชญากรไซเบอร์ชาว Russkrainians มีแนวโน้มที่จะสร้างความปั่นป่วนเพิ่มขึ้นโดยใช้สินทรัพย์ของชาวจีน (เช่น ในการจดทะเบียนโดเมน .CN ฯลฯ) เพื่อหลอกลวงกลุ่มคนที่ไม่รู้เรื่องให้ตกเป็นเหยื่อของอาชญากรไซเบอร์ดังกล่าวด้วย

พวกเรายังคงวิเคราะห์โค้ดร้ายนี้ และคิดว่าน่าจะมีประโยชน์ในแง่ของการพยายามตรวจสอบหาข้อเท็จจริงให้แน่ชัด หรือถ้าหากเกิดเหตุการณ์ใดๆ ที่แสดงให้เห็นว่าโค้ดนี้ถูกนำกลับมาใช้โดยเหล่าอาชญากรไซเบอร์ใต้ดิน และถูกพบในสถานที่ที่ไม่ควรจะเกิด

เทรนด์ ไมโคร มีแนวทางการป้องกันข้อมูลเบื้องต้นสำหรับผู้ใช้งานออนไลน์ว่า ป้องกันโดยตรวจสอบชื่อเสียง และประวัติเว็บไซต์ (Web Reputation) สามารถวัดระดับความปลอดภัย และความน่าเชื่อถือของเว็บไซต์ก่อนเข้าชมได้ และต้องมั่นใจว่าอัปเดตหรือติดตั้งซอฟต์แวร์ป้องกันไวรัสใหม่ ทั้งควรหมั่นอัปเดต วินโดวส์ (Update Windows) ทุกครั้งที่มีการเตือนโอกาสที่จะมีปัญหากับเว็รมหรือไวรัสเข้ามาแทรกแซงการทำงานของเครื่องคอมพิวเตอร์ก็จะมีน้อยลง

มติชนรายวัน วันที่ 15 เมษายน พ.ศ. 2552 ปีที่ 32 ฉบับที่ 11358 หน้า 26

ที่มา :

http://www.matichon.co.th/matichon/view_news.php?newsid=01epe02150452§ionid=0147&day=2009-04-15