A Unified Model for Computer Threat Protection (UMCTP)

Muhammad Murtaza, Muhammad Sharif, Mudassar Raza, Aman Ullah Khan

***Abstract*— This paper presents a unified way to unite major computer threat researchers, students, volunteers and amigos in antivirus industry under the umbrella of a Unified Model for Computer Threat Protection. The suggested model slices protection mechanism into two compatible parts to promote indigenous custom antivirus development. First part, rather static in nature, is unique custom antivirus software developed by individual organizations (users) that meets their specific needs. The other one, dynamic in nature, is the threat definition and research part contributed by global researchers' community in a standardized way to form a global public virus definition library. At the end a tentative sketch of such custom software for a strategic organization is also presented to model basic construct and essentials of an effective antivirus.**

***Index Terms*— Antivirus, Virus, Open Source, Model, Threat, Protection.**

I. Introduction

Most of the users ensure [1] data and information security through proprietary AV and anti-spyware software. But that is not the case with large organizations, especially strategic organizations. Priorities of large organizations are unique to their nature of work. Outsourcing a security system, itself, opposes spirit of security. Think of the term "hired security"; it seems funny when in cyber world your secrecy, confidentiality, privacy and even defense is at stake and cyber war jeopardizes international concerns. Monopoly or total dependency in field of security is least acceptable as compared to other domains of computer world. Concept of custom built software is very common in databases, word processors, spread sheets, web, ERP etc, but in field of security, users are more dependent on outsiders.

The main reason of current scenario is dynamic nature of threats that requires dynamicity in protection mechanism. So it is a continuous war between evil and good that means continuous involvement of an outsider. The only solution that we are left with is indigenization by empowering user in the field of security.

Manuscript received March 8, 2008. This work was supported in part by the Higher Education Commission of Pakistan under Grant No.59-42/1/HEC (R&D)/08/439

Muhammad Murtaza is with Department of Computer Sciences, COMSATS Institute of Information Technology Wah Cantt. Punjab-Pakistan (Email: leothegreatpk@yahoo.com)

Muhammad Sharif is with Department of Computer Sciences, COMSATS Institute of Information Technology Wah Cantt. Punjab-Pakistan, (Phone: +92-300-5188998, Email: muhammadsharifmalik@yahoo.com)

Mudassar Raza is with Department of Computer Science, COMSATS Institute of Information Technology, Wah Cantt. Punjab-Pakistan, (Email: mudassarkazmi@yahoo.com)

Aman Ullah Khan is with Department of Computer Science COMSATS Institute of Information Technology, Wah Cantt. Punjab-Pakistan (Email: auk_pk@yahoo.com)

Different threats are identified by different researchers at different occasions [5]. Threat definitions by one provider can not be used on other platforms. Using more than one AV is not feasible. So if you can benefit only one at a time then two are certainly better than one.

UMCTP points out need of a standard to be developed that enables users to develop their own custom security software.

The software can be "armed" with compatible virus definitions and threat detection techniques developed by numerous volunteer researchers and AV developers through Local Update Centers or a joint Central UMCTP Update Server forming a pool of virus signatures that anyone can benefit from (see figure 1. derived from [9]). UMCTP can be better understood with the analogy that you do not hire a guard for security but post your own watchman and buy weapons and detectors from open market to facilitate him. The concept provides solutions to the problems that open source AV model [2] is facing.

A. Potential Grounds

Development of indigenous AV software requires a long period spanning years of research and continuous support afterwards in form of virus definitions. The only way to a quantum leap in this field is "going open source". FreeAV/OpenAV model is in its early stages as compared to proprietary AV, but it has gone a long way if we look at likes of [2]. The vast presence of OS/FS based organizations, volunteers, researchers, universities, students and OpenAV and Free AV providers in the market [8] provides a cultivating ground for UMCTP.

Best practices regarding an AV design can be unified to develop a standard that can be followed by users to develop their custom AV Software that can extract updates from UMCTP Update servers. The present fore runners in proprietary AV business should also join the collective effort against computer threats to neutralize [5] kind of notions.

Some of them partially support the idea "We think [open-source antivirus products] are fine...we've always been big supporters of open-source anti virus," says Marcus at McAfee [18].

B. Advantages

UMCTP inherits all OSS/FS advantages and adds a few of its own that are fairly strongest advocates of this model.

The success of an AV, largely, lies in the virus signature database and heuristic techniques. Heuristics, being comparatively static in nature, need less frequent updating as compared to virus signatures that are more dynamic in nature. The approach adopted in UMCTP, having such a large infrastructure [2] [8],

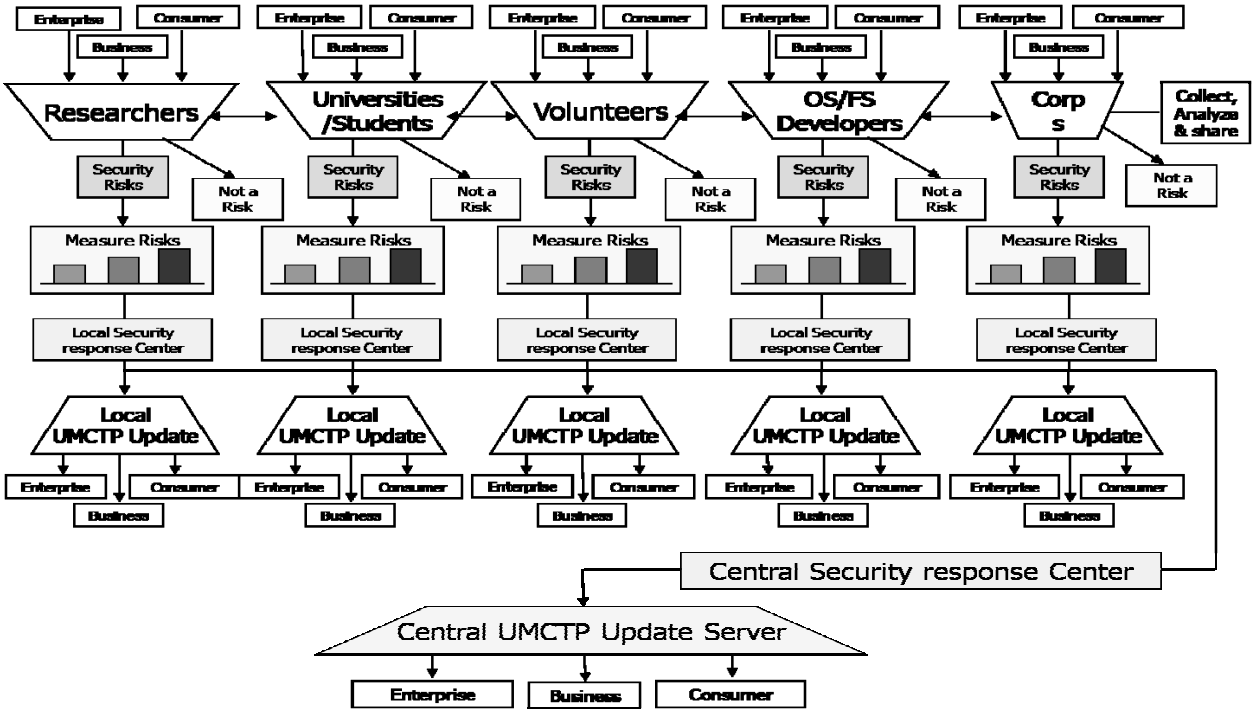

Fig 1: UMCTP Approach to Virus Signatures

Open Source orientation and indigenization provides a holistic solution and a collective approach to threat discovery problems.

The UMCTP approach to threat identification and risk calculation certainly spans worldwide resources.

UMCTP provides the user liberty to choose definition provider rather than to be bound to one vendor. User can opt for any local update provider or Central UMCTP update Server or whoever he trusts.

One of the biggest advantages of open source is its customization and individuality that makes it more threat resistant. The same feature, augmented in UMCTP, proves more beneficial as we know expert attackers target popular AV while writing threats. Most of viruses are programmed to avoid detection by popular AV software.

There are many arguments in favor of OS/FS that revolve around preserving, protecting and promoting rights of the users [6].

Transparency of processes has a unique value in case of security.

Despite arguments, the presence of many open source projects [8] itself proves its future scope as it opens doors to research and learning.

Security becomes a fallacy without indigenization as trust cannot be purchased. Now or later, one has to be independent in this field when secrecy will not limit itself to merely passwords, card numbers and personal information but it shall start affecting national interests. UMCTP is the best road to indigenization that can be expanded by contributions and can be personalized at the same time. It is never too late taking the first step that is inevitable.

II. DESIGN METHODOLOGY

Effective AVs must utilize the host side (self) and virus side (non-self) information [3] as well as dynamics and propagating model of the viruses [15] to cover the threatening areas to the system. We have to focus on Access Points (AP) that provide an interface between inside and outside of a system and govern the behavior how systems exchange malware code or secret information. This three-way approach provides solid defense against threats as each one covers different aspects of threat detection and has its own pros and cons [12].

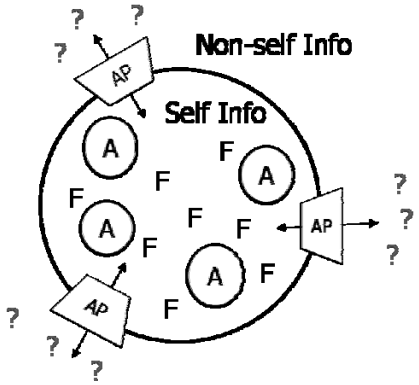

Fig 2: Virus Detection Philosophy

Moreover, the self information is divided into legal files (F) and activities (A).

Files portion constitutes OS system files and user data. It is a difficult task to identify which files are legal and which are not, but there are many methods that can detect corruption into files e.g. checksum method [3][10][14].

The illegal activities are identified through memory resident part of AV through both signatures and heuristics that enables the detection mechanism to recognize viral activities that are alien to the system or hide themselves behind legal activities e.g. Windows™ provides many services like the Win logon Notification Package, cookies, shell commands, registry editing, Auto run etc, that are usually exploited by viruses.

III. PROTECTION MECHANISM

A prototype system mechanism is designed to implement UMCTP using modular approach. The main emphasis is on threat detection and prevention schemes. The system is assumed to run on a LAN isolated from WAN that can be equipped with any proprietary AV or some WAN version of UMCTP AV; while on LAN, all clients are equipped with strictly UMCTP AV that can be updated through local server using active networks technique [16] and in other cases through online live updates.

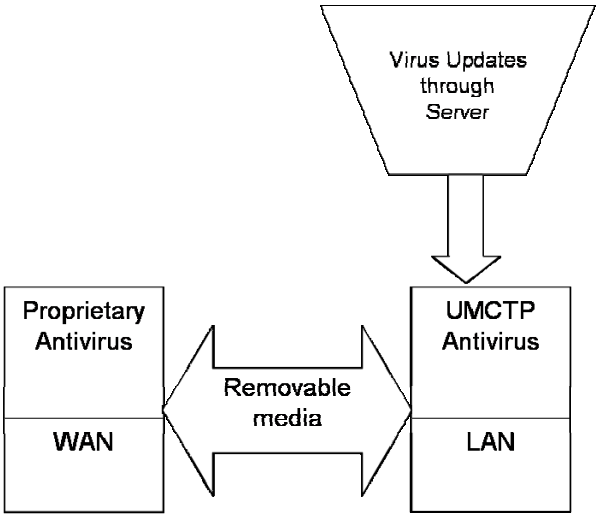

Fig 3: System Layout

Following methods can be used in combination for detection:

1) Scanner method [3]-for known threats (empowered by UMCTP updates)
2) Check-sum [4]/CRC method-for self information
3) Heuristics-for new threats (Xraying, Emulation, DCAM [17])

An improved modular version of basic design presented in [4] is devised for scan engine (see figure 4) distributed into kernel, Virtual File System Interface (VFSI), macro, ZIP, spyware plugins [4], Media Access Control Module (MACM), System Transparency Modules (STM) and Self Defense Shield (SDS).

A. Media Access Control Module

Spreading mechanism is one of two essential parts of viruses [14][15]. First possible encounter with a threat by a system is during media access. Hence, defending the system at access point and preventing it entering the system should be the first priority. Media can be a LAN/WAN interface, removable media, storage media etc. As our system is supposed to be isolated from WAN, the MACM mentioned here focuses on removable media that is the only access point between self and non-self of the computer.

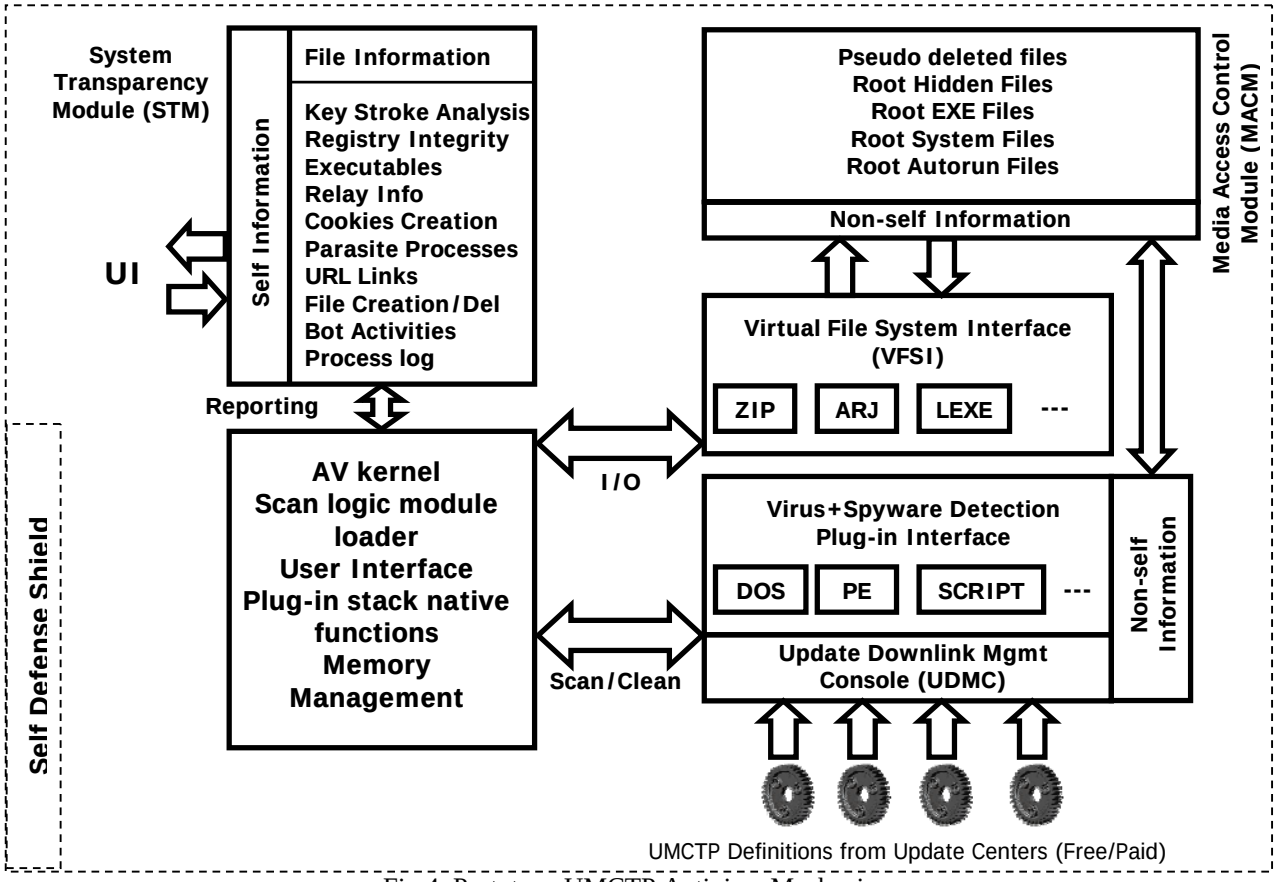

Fig 4: Prototype UMCTP Antivirus Mechanism

For other systems WAN access can be secured with a firewall module. At the entry point MACM is responsible for a safe access to media rather than an Open, Close or EXE scan method [11]. It is armed with non-self information to analyze information I/O for virus signatures and also benefits from virus spreading dynamics at initial access level. It detects hidden, system, .EXE and Autorun files found on media (esp. on root) and reports to System Transparency Module before letting them execute.

Files that deleted from media physically reside on it and are not permanently deleted. These pseudo-deleted files may be a source of information leakage when accessed on a WAN connected computer through removable media. MACM ensures permanent deletion of such files on media and reports to STM. APIs can be written to "shred" such files by writing binary zeros from file starting address to EOF instead of "pseudo-deleting". Similarly the free space on removable media can be padded.

B. System Transparency Module

Around fifteen viruses are discovered daily. It is obvious that development of virus signatures is always one step behind development of viruses. That means there always exists a "first time" you encounter a threat. So it is necessary to empower user to monitor and interfere in system processes as automated detection is not always effective.

STM works on self-information and keeps log of registry modification, module additions and processes added to the system time to time and conditions their addition to user consent. Any process being added or any activity related to registry, system files, auto run files or executables is reported to user to make sure if it is in response to user actions or it is unintentional automated activity that may be specific to a threat. Rolling back facility for activities logged can be useful for advanced users. A similar approach is being used in Kaspersky™ and Symantec Internet Security™.

C. Update Downlink Management Console

This module implements user policies to select sources to be contacted to download virus definitions. In our scenario this source may be a location on network server and in a WAN connected system there may be one or more web locations.

D. Self Defense Shield

Expertly written viruses not only try to hide themselves from AVs but also attack them to disable detection mechanism. Some are so intelligent that prevent AV installation or even recognize windows and names specific to AVs. There emerges need for AV self protection scheme that prevents any changes into AV "self". AV's virus definition interface should be protected in software part to prevent misuse as a backdoor.

IV. VIRUS DEFINITIONS/DATABASE

Although most of the AV companies have shifted towards more advanced methods [13] it is always useful to detect viral infections through signature scanning method. Other methods like Heuristics, X-raying and Emulation can be implemented in AV Software portion. The virus definitions address virus specific code (signatures), files it creates, registry tempering, encryption behavior, links creation to URLs and custom repair routines.

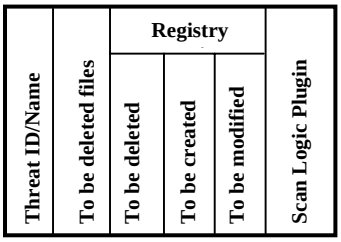

Fig 5: Virus Definition Construct

Viruses normally create files on root or system areas and they are usually .EXE, autorun.inf, system and/or hidden files.

Registry tempering includes deleting, creating, modifying registry entries.

Virus definitions mention which files are to be deleted, what registry entries are to be deleted/modified/created to disinfect the system. The decryption of encrypted files is the most difficult job and there is no guaranty that an infected file can be restored fully. For this purpose some updatable logic part of the AV can be incorporated into database part. Patches and advices can be useful in exceptional cases.

V. PROBLEMS AND FUTURE WORK

The suggested system focuses only on design for detection part of the AV system, implementation issues are not discussed in detail. The sample AV design given above does not incorporate firewall and network modules.

Formation of a central update server requires certain standards for naming and defining viruses and prevention of duplication in definitions.

Since UMCTP updates do not modify software part of AV; periodic improvements in scan logic algorithms remain a responsibility of the user to ensure *privacy*. Updates are passively used for extraction of definitions. If logic is to be updated, a separate and up to date part of scan logic must be identified that can be improved through periodic updates.

VI. CONCLUSION

The model presented opens vast opportunities to empower users managing, controlling and configuring security aspects as per their unique needs with minimal foreign dependence.

The model outlines a practical framework for fighting threat issues with a unified collaborative approach that can result in better and speedy protection solution.

The given framework can be benefited for developing a standard for OSS based AV and virus definition format to promote custom AV software that can be updated from selected sources. A Central Update Server can serve as a public global library reducing duplication of work, increasing integrity, solving virus naming problems and promoting open source AV development.

Standardization in virus updates field can unite global virus research strength on a single platform that can initiate a vigorous movement against cyber crimes.

The model provides solution to problems like availability of frequent update, customization, monopoly,

trust, cost, transparency and indigenization that are fairly more critical in the field of security than any other field.

REFERENCES

[1] Symantec's Internet Security Threat Report, Vol XI, March 19, 2007.http://www.symantec.com/about/news/release/article.jsp?prid=20070319_01

[2] www.clamwin.com, www.openantivirus.org, www.securityfocus .com, www.samag.com, Virus Test Center at the University of Hamburg, XoftSpySE, NOD32, eTrust EZ Antivirus, BufferZone Security Pro. www.Kaspersky.com, AVIRA Antivirus for Linux Server 1.1.5 (avira.com), MailScan for Linux (mwti.net), eTrust EZAntivirus Computer Associates http://www.my-etrust.com/microsoft "www.my-etrust.com/ microsoft, Avast Home Edition ALWIL Software www.avast.com, OpenAntivirus Project (http://sourceforge.net/project/), Open Source Initiative.

[3] Okamoto, T.; Ishida, Y, "A Distributed Approach to Computer Virus Detection and Neutralization by Autonomous and Heterogeneous Agents",. 1999, pp. 328 – 331.

[4] Costin Riau, "Developing Open Source AntiVirus Engines", online.securityfocus.com. 2002-12-16.

[5] Larry Seltzer, "Open Source Not Ready for Anti-Virus", eweek.com. August 9, 2004.

[6] Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

[7] Alex Woodie, "Is Antivirus Ready for Open Source?", Editor of IT Jungle, September 19, 2006.

[8] Apache - HTTP web server, Tomcat web server - Java web/servlet-container, Blender - 3D graphics application, Drupal - content management system, Eclipse - an IDE, GNU Compiler Collection - Programming language compiler for C, C++, Java and other languages, Linux kernel - operating system kernel based on Unix, Mediawiki - wiki server software, the software that runs Wikipedia, Mozilla Firefox - web browser, Mozilla Thunderbird - e-mail client, MySQL – database, OpenOffice.org - office suite, OpenSolaris - Unix Operating System from Sun Microsystems, PostgreSQL – database, phpBB - open source bulletin board system, WordPress - open source blogging platform.

[9] Symantec's Antispyware Approach. http://www.symantec.com /business/security_response/antispyware_approach.jsp

[10] Fred Cohen, "Current Best Practices against Computer Viruses". 1991, pp. 261-270.

[11] Yevgeniy Miretskiy, Abhijith Das, Charles P. Wright, and Erez Zadok, "Avfs: An On-Access Anti-Virus File System", Stony Brook University. 2004. http://www.fsl.cs.sunysb.edu/docs/avfs-security04/index.html

[12] Igor Muttik, "Stripping Down an AV engine", Network Associates Inc (McAfee Division) Alton House, Gatehouse Way, Aylesbury, Bucks, HP4 8YD, UK. Virus Bulletin Conference, September 2000.

[13] Munir Kotadia, "Why popular antivirus apps 'do not work", ZDNet Australia. 2006.

[14] S. J. Phillippo, "Practical Virus Detection and Prevention", PC Security Ltd. Marlow, BUCKS. 1990, pp. 1-4.

[15] Xi Zhang, Debanjan Saha, Hsiao-Hwa Chen, "Analysis of Virus and Antivirus Spreading Dynamics". Dept. of Electr. Eng., Texas A&M Univ., College Station, TX, USA. 2005.

[16] Akbar, Buhari, Sahalu, Saleem, "Automatic Signature files Update in Antivirus Software Using Active Packets". King Fahad University of Petroleum and Minerals, Dhahran, Saudi Arabia. 2001, pp. 457-460.

[17] A. Sulaiman, K. Ramamoorthy, S. Mukkamala, A. H Sung, "Disassembled Code Analyser for Malware (DCAM)". Department of Computer Science New Mexico Tech. 2005, pp. 398 – 403.

[18] "Malware Now a Group Effort", Source: IDG News Service (07/17/06) McMillan, Robert. http://pcworld.about.com/news/ Jul172006id126438.htm