# Survey on USB Attacks and USB Filters

Kamalika Poddar
University of California,Riverside
SID: 862002289
kpodd001@ucr.edu

## ABSTRACT

USBs has been an integral part in data transmission and due to its flexibility and user friendly application, lot of attackers are using it as a tool in attacking the host machine. This paper provides an insight knowledge of how easily malware through USB can be propagated,USB-Based attacks that can make any USB prone to malicious malware. It also talks about the precautions like USB-Guard, USB Filter, Curtain which can be implemented on a system to protect it from attacks.

## KEYWORDS

USB attacks, BadUSB, USBGuard, USB Filters,Curtain

**ACM Reference Format:**
Kamalika Poddar. 2018. Survey on USB Attacks and USB Filters. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

With the rapid development of society and technology, data transmission has become an integral part of our life and with the easy trasmission with USB providing extensive storing capabilities, we are becoming more prone to USB attacks. For example, lots of secret messages can be stolen with USB storage devices by some Trojan horses, viruses or other malicious codes. All above-mentioned security risks brought by USB storage device pose a serious threat to network security and can put some corporations or organs of government to a dangerous level. This paper is divided into two parts, Section 2 talks about USB Attacks and Section 3 talks about USB Filters, Section 4 summarizes all the seven papers.

## 2 USB ATTACKS

In this section we first talk about how easily an attacker can transmit viruses to one's system without any external precautions. Then light is led upon different USB-based attacks and the last part talks about BadUSB which is very powerful as it can infect a system without needing the owner's permission.


*Conference'17, July 2017, Washington, DC, USA*

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

### 2.1 Users Really Do Plug in USB Drives They Find

This paper tells us about the research in which 297 flash drives were dropped in the University campus and how people plugged in these drivers into their system without finding proper authentication. It was found that any device could be attacked in less than 6 minutes and the attack success rate was found to be 45-98 %. Researchers placed the malicious USB devices in different geographic locations, changed the drivers appearence to find out which one an individual is more susceptible to pick up. They also dropped the USBs at different time gaps (6-10am, 1-5pm) to find out which sector of individual are susceptible to fall prey for it. The driver contained all HTML files which allowed the researchers to get notified when the file was opened. Sometimes the files also contained some survey where they needed to answer few of the questions which could help the researcher detect the true intension of the people. Throughout the survey all the participants were given warnings on returning the USB to the respective vendors but none responded to it. At the end of the survey, the participants were asked what was the reason for them to take the USB, they responded that they wanted to return it to the user, some people told it was their curiosity that made them pick the USB. It was also found majority of the people did not take any precaution while openning the files but some told that they run anti-virus protected software , or thought their OS will protect them. The findings of the experiment was demontrated in the Red Hat conference and this paper led the foudation for social enginneering attacks and how pluggin of unknown devices in the host machine can inject malwares.

### 2.2 USB-based attacks

This paper was written by Nir Nissim, Ran Yahalom, Yuval Elovici and was published at ELsevier. In this paper the researchers from Ben-Gurion University of the Negev in Israel talked about 29 USB based attacks which could be categorized into four different categories. The different USB attacks under the following category is explained in the following subsections.

(1) Reprogrammable USB devices's internal microcontroller.
(2) Reprogramming USB device's firmware to execute malicious actions.
(3) Flaws in operating system of an unprogrammed USB
(4) USB based electrical attacks

#### 2.2.1 *Reprogrammable microcontroller USB attacks.*

- **Rubber Ducky**- It is a commercial keystroke injection platform released in 2010 which can inject a preloaded keystroke sequence when connected to a host.
- **PHUKD/URFUKED attack platforms**- It allows the attacker to select time when injected.

- **USBdriveby**- backdoors can be installed and DNS settings can be overridden on an unloacked OS.
- **Evilduino**- Uses Arduino controllers and can be leaded by sending keystrokes or mouse cursor movements to the host.
- **Unintended USB channel**- Hardware Trojans can take out data as well.
- **Turnipschool**- a hardware implant concaled within a USB cable.
- **RIT attack via USB mass storage**-Changing the content of files while USB is connected to host.
- **Wireless USB dongles**-a tool that could decrypt keystrokes from a Microsoft RF wireless keyboards.
- **Default Gateway Override**-

*2.2.2 Maliciously Reprogrammed USB peripheral firmware attack.*

- **Smartphone**- Malicous driver interacts with the Android USB gadget API to simulate keyboard and mouse connected to the phone.
- **DNS override**- USB flashdrive firmware was modified and used to emulate a USB-ethernet adapter, which is allowed to hijak the local traffic.
- **Keyboard Emulation**- Positioning the USB flashdrives can incur different keystrokes
- **Hidden Partition Patch**- USB can be reprogrammed to act like normal drive„ creating hidden partition which cannot be formatted.
- **Password Protected Bypass Patch**-bypassing password protected USB flashdrives.
- **Virtual MAhcine Break-Out**- This could break out virtual machine environments.
- **Boot Sector Virus**-computer can be infected before it boots.
- **iSeeYou**- in this the Apple internal iSight webcams can be reprogrammed where the attacker could capture video without LED indicator warning.

*2.2.3 Unprogrammed USB devices.*

- **CVE-2010-2568**- exploited by Stuxnet and Fanny malware
- **USB Backdoor into Air-Gapped Hosts**- USB hidden storages to store preset commands that map computers in air gapped networks and the info is saved back to the USB.
- **Data Hiding**- Storing the data inside a hidden file and making the icon transparent.
- **AutoRun Exploits**- Depends on nthe configuration of the computers.
- **RAM Dump attacks**- a memory dumper can be stored in a USB flashdrive and data can be extracted from the RAM by booting the USB.
- **Buffer Overflow**- Exploits OS buffer overflows when USB is inserted into the host.
- **Driver Update**- Complex attack relies on obtaining a VeriSign Class 3 Organizational Cartificate and submitting drivers to Microsoft which can be delivered and installed on user PCs.
- **Device Firmware Upgrade**- an update can be a tool to change the firware settings to malicious version.
- **USB Thief**- flash drive based data-stealing malware.
- **USB Port**- malware propagated through phone chargers.
- **USBee attack**- Makes USB connector data bus give out electromagnetic emissions which is used to extract data.

*2.2.4 Electrical Attacks.*

- **USB Killer**- USB device that triggers an electrical surcharge and permanently destroys it.

## 2.3 BadUSB- On accessories that turn evil

In this conference,Karsten Nohl demonstrated how USB devices can take on the ability to act a keyboard and type in malicious commands into the attached computer and another drive will be reprogrammed as a network card which will cause connected computers to connect to malicious sites. They also presented about how android phones can fall into the same prey when attached to the targeted computers. They also demonstrated that their techniques works well for Webcams, keyboards and other USB devices.

- how USB stick takes over Windows machine- Transforming an USB stick into a computer keyboard that opens a command window in the host system and itself enters commands that cause it to download and install malicoius software. There is no no protection apart from just clicking a 'OK'.
- Windows infects USB stick which then takes over Linux machine- When a USB stick transforms into a network card, then this causes other computers to connect to malicious sites and attacking the machine.
- USB thumb drive changes DNS settings in Windows- When a programmed USB stick is used to inject a payload into a Ubuntu installation file and this malicious file gets attached to the host computer.
- Android diverts data from Windows machine and tranforms into a malicious network card.

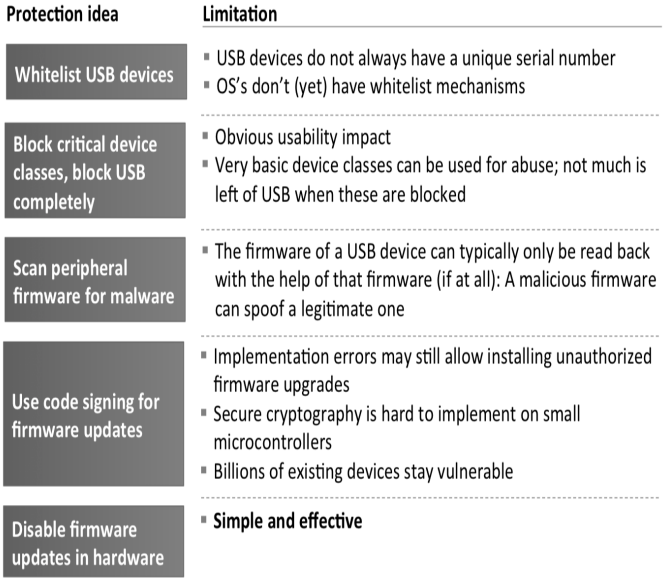


**Figure 1: Protection ideas presented at the Red Hat conference on BadUSB**

These types of attacks can be programmed on any type of USB device and its very difficult to detect a tampered device when attacked by a BadUSB. He also demonstrated that although there are few ways to get protected but people can try to limit their system connected to the network. They also presented different protection ideas and the limitaion on them which are shown in figure 1.

## 3 USB FILTERS

This section first talks about USBGuard which can protect a device by whitelisting, blacklisting and triggering inputs. Then we talk about design of Filter driver and its working functionality. Next part includes USBfilter proposed by Dave Tian which works on Linux machines and Curtain which works on Windows system in protecting from USB attacks.

### 3.1 Built-in protection against USB security attacks with USBGuard

This article tells us about how USB attacks can be prevented by using USBGuard by a Red Hat Enterprise Linux user. USBGuard is a software which has the capability to whitelist and blacklist device attributes in order to protect the system. In this a user can define what kind of USB devices are authorized and how a USB device can interact with the system. Three main cases for USBGuard are:

- USB device Whitelisting- in this the USB device can permit only particular devices to interface with it. It basically list out the devices with which the host will interact.
- USB device blacklisting-In this the device is programmed to block certain interfaces from communicating with the host.
- Triggering actions on USB device events- This tells us when a particular device is inserted or removed, a trigger is raised to make the user aware of the USB device.

USBGuard are not installed by default but can be installed from the Red Hat enterprises packages. After installing, the USBGuard settings can be controlled by usbguard-daemon.-conf file. When a USB device is inserted, the daemon scan through all the rules and it blocks, rejects or allows the device. by default, it blocks the rule and ask for permission from user. To start with USB Guard we can use CLI command and the generate policy rather than writing rules from scratch. This gives a very good software that can protect Red Hat Linux system.

### 3.2 Design and implementation of encryption filter driver for USB storage devices

In this paper the author talks about three kinds of solution which can protect from security hazards brought by USB storage devices.

- First is to disable the USBport to aim at preventing inner information betrayed, which shields USB port physically and modify registry. Example-It can make some input or output devices not working like mouse or keyboard.
- Encrypting secret information to aim crpytograph, which is inconvenience for user to encrpyt every file that needs to be protected.
- Encrypting the boot sector to aim to control the access policy of USB storage devices.

This paper presents a scheme to encrpyt data input or output based on WDM model. This scheme divides the system into-

- User layer module- Functions it completes are: Load and uninstall the filter driver; configure and mange the filter driver, set and exchange the key of the filter driver. It is implemented in the user level.
- Kernel module- It is the main part of the system whihc completes the encrption of all the data which is the input for USB storage devices and decrypts the data which is the output of the USB storage devices. It creates one object of the device and attaches it upon Usbstor.sys in the stack and then hooks all IRP about read and write.

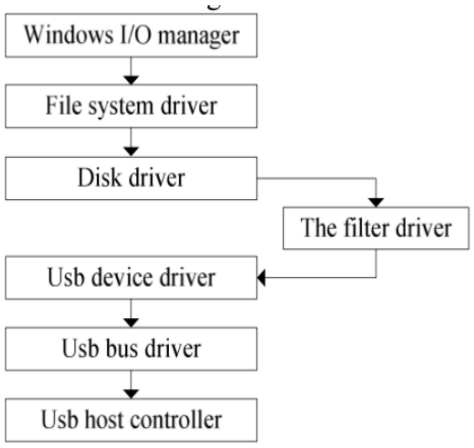


**Figure 2: location of filter driver**

In a windows system, first the program calls one of the application program interface named as createfile() which is defined in one of the dynamic link library named kernel.dll. Createfile() calls function Ntcreatefile() i.e, defined in the Ntdll.dll then the program get into kernel ode by call soft interruption INT 2E. Kernel can call I/O manager which creates IRP and transfers it to driver stack. First on eto get the IRP in the stack is partition driver, then disk driver and then USB mass storage device driver. The IRP packet is trasferred from the top of the stack to the bottom- application program transfer IRP to the file system filter driver, then the filter driver transfers to disk driver (disk.sys), then to UsbStor.sys. In Windows Driver Model(WDM), filter can be loaded before or after one level of the driver stack. During attaching process, filter driver produces one device object to insert in the stack of the USB devices. During the process of filter driver getting IRP, it can encrpyt or decrypt data to provide transparent encrypting of data. The key technology of the whole thing is IRP tracking. IRP is a struct with volatile size and two parts: header and location of the stacks. Hosts can read data from the USB storage device by "ReadFile" and write by "WriteFile". The IRP is send to the top of the stack each time it is called, and then to the next level. Cureent pointer of the stack can also be determined by calling"IoGEtCurrentIrpStackLocation()". The location of the filter driver is shown in figure 2. The process of encryption and decryption is shown in figure 3. Another paper

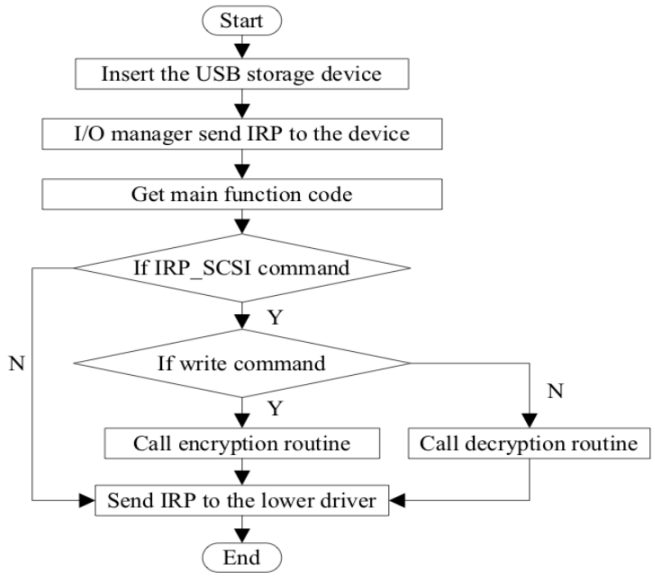


**Figure 3: The process of filter driver working**

"Reseach and Application of USB Filter Driver based on Windows Kernel" also focuses on the same topic as the paper discussed here.

### 3.3 Making USB Great Again with usbfilter

This paper was demonstrated at the Usenix Security'16, Austin Symposium. In this paper the author talk about USB Filters. USB filters is the kernel component working under the host controller, it consists of :

- USBFilter Modules- These develops, writes new features and plug-in kernel modules in USB filters.
- -Rule Database-Before transferring the USB packet it has to go through all the Rule databases, policies to check the rule.
- USBTables- It manages all the rules, policies and saves all active rules. It also verifies the correctness of the rules.

USBFilters co-relate the USB packets within the kernel with the I/O operations of search applications within the used space. To implement general USB drivers to trace back the process identifier of the originating application certain USB packets. The requests from the drivers to the host controller are sent by USB request blocks(URBs). The goals of the USB Filters can be *complete mediation*of all USB packets within the host machine, *tamperproof* Not able to formally verify the kernel instead focused on rule consistency. Before a rule is added, it was made sure that there is no conflict between the new rule and the existing rule in the kernel, *granularity*-rules to implement has right functionality and flexibility, *extensibility*-This allows to write new features and plug-in them into USB Filter. Using *udev* can erase all the saved rules. They also mentioned about Rule construction where they divided 21 rules into four types- *process, Device, Packet and LUM* (Linux USBFilter Module). LUM has a header file to write a kernel module. Once a plug is plugged in USBFilter, then one can write new rules with LUM. LUM is a way to look to USB packet and can have packets like SCSI commands, IP packets, HID packets. SCSI(Small Computer System Interface)-it physically connects and transfers data between computers and peripheral devices. Protocols used by USB. to read or write a file between USB storage. The SCSI commands were written in the order of first showing the direction of the packet(host to device/ device to host), then the packet should be large enough to carry out commands,making sure it is not empty and not a USB packet hacker, extracting the opcode of the SCSI command from the packet, we should check the opcode and return o or 1 as needed. Rule consistency is ensured as weak conflict and strong conflict rules are identified and rules are build accordingly. They tested the USB-Filters on different devices like keyboard, mouse, webcams. Here they set certain rules identifying the port number, device name which helps the host machine to accept request from the specified device and drop all the rest of the devices. Data exfiltration could be stopped as LUM is able to detect the SCSI command into the USB packet. They also tested on devices by blocking one of the feature in the device and running the test. They concluded that 20 Base rules were needed to make the filter work efficiently. There was a comparison between the USBFilter kernel and Stock Kernel and it was found that their throughputs and latency are same but after 1MB there is a drastic change between their latency. USBFilter was also tested on KVM -Kernel Virtual Machine(creating and installing KVM from USB flashdrive), running benchmark on Chrome using wifi adapter, Scanning viruses using USBFlasdrive using ClamAV. Some limitaions proposed by them was-

- If K is pressed in keyboard, it does not have any packet associated with it, so for K stroke, we cannot use process information to write a rule.
- Used USB drivers to get process identifiers but some USB devices which require specific devices, then we have to look into the particular process identifier.
- Host to device, master-slave protocol works for USBs 1.0 , 2.0 but new requests for USB 3.0 can be created so proper USBFilters is needed in the response path.
- More LUMs to write powerful rules
- USBFilters rules requires some knowledge of USB in general, Linux distribution key, to provide default rules which can help users to customize their rules for their working environment.

USBFilters was a major step in stoping BadUSB attacks but these technology of Dave Tian was based on Linux systems.

### 3.4 Curtain: Keep Your Hosts Away from USB Attacks

The GoodUSB and USBFilter introduced by Dave Tian can't protect the host from channel based USB attacks like MouseJack or Keyboard Sniffer. Based on USB authorization and working process of USB devices, this paper proposed "Curtain", a multi-layered USB defending system installed as a filter driver in Windows. This system doesn't needs to modify the upper and lower level software. Curtain can monitor, hook, modify the IRP flow and it can protect the host from HID attack, BadUSB attack and also from channel based attack. In this paper they proposed a new effective technologies for the USB defence on Windows and demonstrated validity in real-world scenarios and characterize performance. In Windows when a USB

device is connected to the host, the host controller driver starts to work. Depending on the descriptor content, the host controller will assgn different functional drivers for the device which will make up a stack called device driver stack. When transmitting data from device to host, the functional will recieve data by sending the IRP to the top of the stack and when transmitting from host to device IRP will send to the functional driver at the top of the stack, this will in turn return URBs. The architecture of Curtain is shown in Figure 4. In the User Layer, Device Identifier is used for identifying

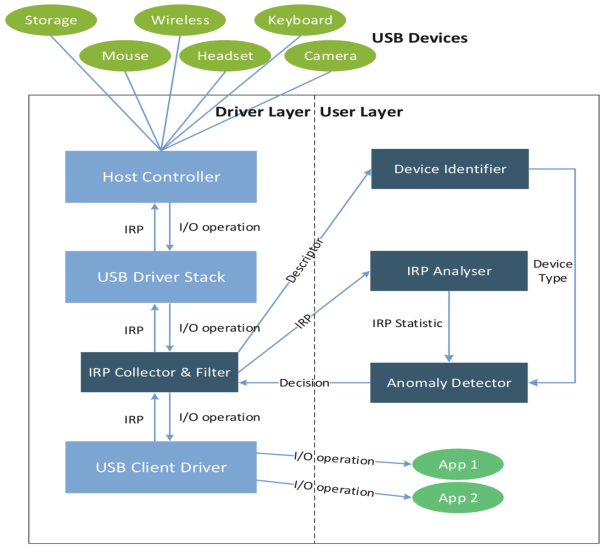


**Fig. 3.** Main architecture of Curtain System

**Figure 4: Architecture of Curtain System**

device type, IRP Analyser counts IRP features and sends statistic to anamoly detector to judge whether an USB is malicious. Then Anamoly detector sends decisions to the IRP Collector and Filter, to manage USB device's behaviour. They designed three algorithms to detect maliciuos behaviour.

- Unauthorized Access to Important Files Detection
- Abnormal Behaviours Detection
- Fake Device type Chain

This paper presents Curtain which catches all the IRP flows of each device and analyze their behavors to detect attacks.

## 4 CONCLUSION

This paper summarizes the findings of seven papers that tells us about USB attacks and how we can prevent them from occurring. USBs was introduced in 1994 and within few years it became mandatory for all the devices to have USB ports where external flashdrives can be easily connected and files can be transferred without much difficulty. This kind of easiness actually led to a huge attack of Stuxnet worm which could shut down the entire power plant unit in Iran. No one paid much attention to what a small USB device could do until this huge attack. After this researchers started understanding the architecture on how these attacks could have been possible. They started finding loop holes which are prone to these kind of attacks. This led them to discover 29 different types of USB attacks that are possible. Security researcher Karsten Nohl demonstrated in BlackHat conference that BadUSB attacks can make any device prone to malicious malware. In this he introduces BadUSB which can affect a sytem without the owner's permission. He also tells any individual can access data from the internet and Android phones can be easily attacked. He also talks about the limitations on USB which are not able to protect itself. To protect devices from these attacks researchers came up with USBGuard, USBFilter and Curtain. They demonstrated the design and implementation of each of the filters and how these filters can be used in different Windows and Linux machines to protect itself from any unknown attacks. Still more research needs to go on in this field as day by day the risk of USB attacks are increasing.

## A HEADINGS IN APPENDICES

### A.1 Introduction

### A.2 USB attacks

*A.2.1 Users Really Do Plug in USB Drives They Find.*

*A.2.2 USB-based attacks.*

*A.2.3 BadUSB- On accessories that turn evil.*

### A.3 USB FILTERS

*A.3.1 Built-in protection against USB security attacks with USB-Guard.*

*A.3.2 Design and implementation of encryption lter driver for USB storage devices.*

*A.3.3 Making USB Great Again with usb lter.*

*A.3.4 Curtain: Keep Your Hosts Away from USB Attacks.*

### A.4 Conclusions

### A.5 References

## ACKNOWLEDGMENTS

I would like to thank Dr. Nael Abu-Ghazaleh for giving the opportunity to explore USBFilters. I was not aware about USB attacks and how much naive we are that we don't give a second thought about plugging someone else USB in our system. With the help of this survey, I could explore different USB Filters that protects us from USB attacks. I would also like to thank all the researchers who are working day and night to make our life easy and protect our personal information from these kind of malicious attackers.

## REFERENCES

[1] Rafal Ablamowicz and Bertfried Fauser. 2007. CLIFFORD: a Maple 11 Package for Clifford Algebra Computations, version 11. (2007). Retrieved February 28, 2008 from http://math.tntech.edu/rafal/cliff11/index.html

[2] Patricia S. Abril and Robert Plant. 2007. The patent holder's dilemma: Buy, sell, or troll? *Commun. ACM* 50, 1 (Jan. 2007), 36–44. https://doi.org/10.1145/1188913.1188915

[3] American Mathematical Society 2015. *Using the amsthm Package.* American Mathematical Society. http://www.ctan.org/pkg/amsthm.

[4] Sten Andler. 1979. Predicate Path expressions. In *Proceedings of the 6th. ACM SIGACT-SIGPLAN symposium on Principles of Programming Languages (POPL '79).* ACM Press, New York, NY, 226–236. https://doi.org/10.1145/567752.567774

[5] David A. Anisi. 2003. *Optimal Motion Control of a Ground Vehicle.* Master's thesis. Royal Institute of Technology (KTH), Stockholm, Sweden.

[6] Mic Bowman, Saumya K. Debray, and Larry L. Peterson. 1993. Reasoning About Naming Systems. *ACM Trans. Program. Lang. Syst.* 15, 5 (November 1993), 795–825. https://doi.org/10.1145/161468.161471

[7] Johannes Braams. 1991. Babel, a Multilingual Style-Option System for Use with LaTeX's Standard Document Styles. *TUGboat* 12, 2 (June 1991), 291–301.

[8] Malcolm Clark. 1991. Post Congress Tristesse. In *TeX90 Conference Proceedings.* TeX Users Group, 84–89.

[9] Kenneth L. Clarkson. 1985. *Algorithms for Closest-Point Problems (Computational Geometry).* Ph.D. Dissertation. Stanford University, Palo Alto, CA. UMI Order Number: AAT 8506171.

[10] Jacques Cohen (Ed.). 1996. Special issue: Digital Libraries. *Commun. ACM* 39, 11 (Nov. 1996).

[11] Sarah Cohen, Werner Nutt, and Yehoshua Sagic. 2007. Deciding equivalances among conjunctive aggregate queries. *J. ACM* 54, 2, Article 5 (April 2007), 50 pages. https://doi.org/10.1145/1219092.1219093

[12] Bruce P. Douglass, David Harel, and Mark B. Trakhtenbrot. 1998. Statecarts in use: structured analysis and object-orientation. In *Lectures on Embedded Systems*, Grzegorz Rozenberg and Frits W. Vaandrager (Eds.). Lecture Notes in Computer Science, Vol. 1494. Springer-Verlag, London, 368–394. https://doi.org/10.1007/3-540-65193-4_29

[13] Ian Editor (Ed.). 2007. *The title of book one* (1st. ed.). The name of the series one, Vol. 9. University of Chicago Press, Chicago. https://doi.org/10.1007/3-540-09237-4

[14] Ian Editor (Ed.). 2008. *The title of book two* (2nd. ed.). University of Chicago Press, Chicago, Chapter 100. https://doi.org/10.1007/3-540-09237-4

[15] Simon Fear. 2005. *Publication quality tables in LaTeX.* http://www.ctan.org/pkg/booktabs.

[16] Matthew Van Gundy, Davide Balzarotti, and Giovanni Vigna. 2007. Catch me, if you can: Evading network signatures with web-based polymorphic worms. In *Proceedings of the first USENIX workshop on Offensive Technologies (WOOT '07).* USENIX Association, Berkley, CA, Article 7, 9 pages.

[17] David Harel. 1978. *LOGICS of Programs: AXIOMATICS and DESCRIPTIVE POWER.* MIT Research Lab Technical Report TR-200. Massachusetts Institute of Technology, Cambridge, MA.

[18] David Harel. 1979. *First-Order Dynamic Logic.* Lecture Notes in Computer Science, Vol. 68. Springer-Verlag, New York, NY. https://doi.org/10.1007/3-540-09237-4

[19] Maurice Herlihy. 1993. A Methodology for Implementing Highly Concurrent Data Objects. *ACM Trans. Program. Lang. Syst.* 15, 5 (November 1993), 745–770. https://doi.org/10.1145/161468.161469

[20] Lars Hörmander. 1985. *The analysis of linear partial differential operators. III.* Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. viii+525 pages. Pseudodifferential operators.

[21] Lars Hörmander. 1985. *The analysis of linear partial differential operators. IV.* Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Vol. 275. Springer-Verlag, Berlin, Germany. vii+352 pages. Fourier integral operators.

[22] IEEE 2004. IEEE TCSC Executive Committee. In *Proceedings of the IEEE International Conference on Web Services (ICWS '04).* IEEE Computer Society, Washington, DC, USA, 21–22. https://doi.org/10.1109/ICWS.2004.64

[23] Markus Kirschmer and John Voight. 2010. Algorithmic Enumeration of Ideal Classes for Quaternion Orders. *SIAM J. Comput.* 39, 5 (Jan. 2010), 1714–1747. https://doi.org/10.1137/080734467

[24] Donald E. Knuth. 1997. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms (3rd. ed.).* Addison Wesley Longman Publishing Co., Inc.

[25] David Kosiur. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY.

[26] Leslie Lamport. 1986. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, MA.

[27] Newton Lee. 2005. Interview with Bill Kinder: January 13, 2005. Video. *Comput. Entertain.* 3, 1, Article 4 (Jan.-March 2005). https://doi.org/10.1145/1057270.1057278

[28] Dave Novak. 2003. Solder man. Video. In *ACM SIGGRAPH 2003 Video Review on Animation theater Program: Part I - Vol. 145 (July 27–27, 2003).* ACM Press, New York, NY, 4. https://doi.org/99.9999/woot07-S422

[29] Barack Obama. 2008. A more perfect union. Video. (5 March 2008). Retrieved March 21, 2008 from http://video.google.com/videoplay?docid=6528042696351994555

[30] Poker-Edge.Com. 2006. Stats and Analysis. (March 2006). Retrieved June 7, 2006 from http://www.poker-edge.com/stats.php

[31] Bernard Rous. 2008. The Enabling of Digital Libraries. *Digital Libraries* 12, 3, Article 5 (July 2008). To appear.

[32] Mehdi Saeedi, Morteza Saheb Zamani, and Mehdi Sedighi. 2010. A library-based synthesis methodology for reversible logic. *Microelectron. J.* 41, 4 (April 2010), 185–194.

[33] Mehdi Saeedi, Morteza Saheb Zamani, Mehdi Sedighi, and Zahra Sasanian. 2010. Synthesis of Reversible Circuit Using Cycle-Based Approach. *J. Emerg. Technol. Comput. Syst.* 6, 4 (Dec. 2010).

[34] S.L. Salas and Einar Hille. 1978. *Calculus: One and Several Variable.* John Wiley and Sons, New York.

[35] Joseph Scientist. 2009. The fountain of youth. (Aug. 2009). Patent No. 12345, Filed July 1st., 2008, Issued Aug. 9th., 2009.

[36] Stan W. Smith. 2010. An experiment in bibliographic mark-up: Parsing metadata for XML export. In *Proceedings of the 3rd. annual workshop on Librarians and Computers (LAC '10)*, Reginald N. Smythe and Alexander Noble (Eds.), Vol. 3. Paparazzi Press, Milan Italy, 422–431. https://doi.org/99.9999/woot07-S422

[37] Asad Z. Spector. 1990. Achieving application requirements. In *Distributed Systems* (2nd. ed.), Sape Mullender (Ed.). ACM Press, New York, NY, 19–33. https://doi.org/10.1145/90417.90738

[38] Harry Thornburg. 2001. Introduction to Bayesian Statistics. (March 2001). Retrieved March 2, 2005 from http://ccrma.stanford.edu/~jos/bayes/bayes.html

[39] TUG 2017. Institutional members of the TeX Users Group. (2017). Retrieved May 27, 2017 from http://wwtug.org/instmem.html

[40] Boris Veytsman. [n. d.]. acmart—Class for typesetting publications of ACM. ([n. d.]). Retrieved May 27, 2017 from http://www.ctan.org/pkg/acmart