Science, Technology, & Human Values

Stabilised Instability. Hacking Tournament As a Laboratory

Journal:	*Science, Technology, & Human Values*
Manuscript ID	ST&HV-2018-02-040
Manuscript Type:	Original Article
Keyword:	laboratory, hacking, computer security, epistemology, ANT, ethnography

Abstract:

Capture the flag is a legal tournament in computer security, in which teams of hackers compete in solving tasks and attack or defence of systems. Using the Actor-Network Theory framework, the paper presents the results of ethnographical studies of such tournaments in order to reveal the mechanisms of the construction of hacking knowledge. Each element of the tournament is examined with this approach, from task construction through different types of competition to a write-ups and potential links between tournaments, companies, and military agencies.
By using selected concepts from laboratory studies, the paper demonstrates the concept of stabilised instability. As a result, a new understanding of errors and black-boxes is proposed. Instability may be treated not only as a way of opening technoscientific black-boxes but also as a product of stabilisation. This concept also serves as a potential link between laboratory studies, classical studies of hackers, and contemporary social studies of technoscience of hacking.

Stabilised Instability. Hacking Tournament As a Laboratory

Capture the flag is a legal tournament in computer security, in which teams of hackers compete in solving tasks and attack or defence of systems. Using the Actor-Network Theory framework, the paper presents the results of ethnographical studies of such tournaments in order to reveal the mechanisms of the construction of hacking knowledge. Each element of the tournament is examined with this approach, from task construction through different types of competition to a write-ups and potential links between tournaments, companies, and military agencies.

By using selected concepts from laboratory studies, the paper demonstrates the concept of stabilised instability. As a result, a new understanding of errors and black-boxes is proposed. Instability may be treated not only as a way of opening technoscientific black-boxes but also as a product of stabilisation. This concept also serves as a potential link between laboratory studies, classical studies of hackers, and contemporary social studies of technoscience of hacking.

Computer security, hacking, actor-network theory, laboratory, technology

It is evening. Several twentysomethings sit around a conference table. The table is a mess, there are several laptops, bottles of alcohol, crates of energy drinks and wires. Some of them chat, some of them set up connections or check communication channels. They may be working on Linux kernels (Coleman 2013), they might be working on hardware, cryptography, hacktivism or pornography featuring My Little Ponies. Hacking as usual.

But there is something odd, even for the odd-rich environment like hackIng meeting. They had been hacking for a long time in silence, but why this cryptographer started to discuss exploits with the pony porn guy? They are conversing in soft voices because the hardware guy told them to “shut the fuck up, at least until this task.” Long hours and discussions about encryption are common in our basement, but what about requests for silence and tasks, whatever they might be? What about exploits and vulnerabilities in computer systems? Is testing them not somewhat shady, something better done in privacy rather than in a crowded space?

If we follow these hackers, we will notice that, forty-six hours after our evening observation, something has changed. The whole group operates as several silent problem-solvers. They rarely talk but almost always in groups of two or three hackers analysing the same screen. Almost every quarter, somebody reports something about the scoreboard situation and the “fucking” Poles, Americans, Chinese, Koreans, or Israelis. Someone shouts, “I have got the flag.” After forty-eight hours, the tension suddenly drops. People move their eyes away from the screens, begin to chat casually. Groups from other tables come for a drink, somebody receives a job offer by a rather well-known security company. Was there an undiscovered error, also known as zero-day, in one of the tasks?

Capture the flag (CTF) tournaments play an important part in computer security and hacking community. Is there anything peculiar in knowledge used and abused in such events? Is there a hacking equivalent of scientific laboratory? If so, what are similarities and differences between hacking and non-hacking collectives (Knorr Cetina 1999, 1981; Latour and Woolgar 1986; Lynch 1985; Vertesi 2015)?

Theoretical Framework

I will show how hacking tournaments construct new actors, such as computer exploits, vulnerabilities and critical exceptions. While question about construction of scientific facts has a long history in STS, answering it in case of computer security remains a controversy. Are hackers merely discovering existing exploits, or do they create and construct new actors? By studying hacking tournaments, I wish to extend our knowledge on laboratories to a new ground.

Social studies of science discussed knowledge construction in relation with modern technologies (Knorr Cetina 1999; Vertesi 2014; Latour 2005), but the construction and dissemination of computer exploits, leaks, and bugs still await examination and comparison with other types of technoscience laboratories. We know that genes, engines, scallops and institutions are constructed, but what about errors in computer infrastructure? What about exploits and security leaks? How to raise a world, when our laboratory has to work on digital, exceptional and secretive?

By studying construction of knowledge during computer security tournaments, this paper aims to revisit the concepts of the laboratory, black box, and stability in digital actor-networks. Such concepts were grounded in ethnographies of laboratories (Knorr Cetina 1981; Latour 1983, 1987; Knorr Cetina 1999), but only rarely they are adopted to digital, non-formal knowledge-making practices, such as computer security.

In order to do so, I use the notion of black box understood as the result of stabilisation practices of laboratories. (Latour 1999, p. 184). In this sense, the technological black boxes are subjects of interest for both science and technology studies (STS) and hackers. Both parties agree, that some elements in technoscientific systems might be operated, basing only on knowledge about input and output signals.

There is a growing interest in hacking in the STS community, because hackers [1] provide an opportunity to study relations between technology and politics (Coleman 2015, 2013), construction of users (Söderberg 2010), and dynamics of online capitalism (Söderberg and Delfanti 2015). Scholars argue that hacking may be understood as a practice of resistance or obfuscation (Nissenbaum 2004), as well as a practice subsumed into capitalist frameworks (Soderberg 2013). Other works focus on the new shapes of hacking, including hackathons (Irani 2015) and the evolution of hackerspace outside of the USA and Western Europe (Lindtner 2015; Zaród 2015). Recent studies concentrate on the broader meaning of hacking, which often makes them focus less on computer security than the broader spectrum of less formal computer engineering, namely the construction of Linux kernel, communication networks, 3D printing, or hardware construction. With some notable exceptions (Coleman 2015), the construction of knowledge in computer security no longer dominates the inquiry. Although a significant number of hacking tournaments are legal, they are difficult to study by social sciences. As a result of hermetic community and high complication of exchanged knowledge, studies of CTF are rare, mostly focusing on the social aspects of computer security (Cowan et al. 2003).

Discussions on computer security knowledge governed previous studies (Jordan and Taylor 1998; Taylor 1999; Turgeman-Goldschmidt 2005, 2008). Such works focused more on individual motivations and a narrower meaning of hacking understood as non-formal or illegal computer security practice. Gabriela Coleman and Alex Golub explored the connection between the broader and narrower understanding of hacking, which revealed political similarities between the security-hackers and open source software-hackers (Coleman and Golub 2008).

This paper builds on both these traditions to explore how individual practices in computer security constitute large-scale knowledge circulation mechanisms. Computer security and reverse engineering practices do not account for the whole hacking culture but remain an important point of reference for other types of hackers. Hacktivists and politically-minded hackers (Coleman 2017) still need the security of communication. Hackers interested in electronics or 3D printing follow computer security patterns in their own practices of reverse engineering (Söderberg 2013). Non-hacking organisations, such as military intelligence agencies and private companies, share this interest, as they sponsor security tournaments (Enserink 2008). Despite internal differences and external pressures, hackers retain significant control over some of the most important computer security events, such as Capture the Flag (CTF) hacking tournaments.

I will show that studying hackers might contribute not only to current STS-hackers-politics debates but also to the classical themes of STS, such as construction of knowledge, theory of black-boxes and relationship between stability and exception. As a result of studying computer security tournaments, I propose a concept of stabilised instability: a new type of actor, which may be an exception in technological system, but needs similar laboratory practices to be discovered, circulated, and recognized.

Data Gathering and Analysis

I gained access to CTF tournaments as part of my ethnographical research in selected hackerspaces in Poland, Sweden, Netherlands, and Denmark. In 2014, I followed one team to a hacking convention which I meticulously observed for four days and where I conducted more than ten semi-scripted interviews and improvised talks. For the following two years, I continued with the observations in the physical location of the hackerspace as well as through netnography of mailing lists and IRCs, from which I gathered above sixty notes from observations and 200 excerpts from digital channels. Furthermore, I explored CTF in twenty semi-structured interviews with CTF players, individual security specialists, and aspiring hackers. I also observed one game played remotely, from the primary site of my ethnographical project.

This groundwork proved very useful in December 2016, when I shadowed a CTF team during a tournament classified among the top three most important competitions in the annual league, which contribute to the annual score. During the tournament, my observations lasted four days of competition when I did twenty semi-structured interviews and improvised talks with the team, other participants, organisers, and authors of the challenges. During this process, I observed interactions between six to ten participants, all of whom agreed to give two or more short interviews at the task-solving phase.

The tournament in 2016 involved more than 200 teams who played remotely and locally. I decided to focus the observation on the top five contesting teams because they regularly scored high positions in the previous two years. I also deliberately focused on two tasks, which the participants selected as average and hard, by interviewing task authors on the various the aspects of task design. In 2017, I observed an attack-defence CTF and utilised the six-hour-long event for interviews with participants and observation of team interactions.

After transcription, I analysed the materials in MAXQDA qualitative analysis software, coding both in emic and etic approaches. The examples of emic (descriptive) coding are: constructing the task, evaluation of the task, coordination work. The examples of etic (literature) coding are: hybridisation, opening the black box. I have done both the initial ethnographical work and the coding with the use

of the Actor-Network Theory framework, from Bruno Latour's *Reassembling the Social* (2005), and special focus on controversies, scale-shifting, and stabilisation.

All interviewees gave individual informed consents and a group consent during the observations.

Results

While CTF may differ depending on the level and concepts of organisers, the following results stem from direct observation of two high-profile events, regarded by both organisers and contestants as the most successful and prestigious, and several smaller ones, which participants discussed after the events.

CTF is not the only type of competitive computer security exercise, but it is one of the most common for hacking conventions. Some CTFs like Def Con accept sponsoring from multiple organisations, but the majority of tournaments try to remain independent in terms of task creation. CTFs which this paper analyses are even less directly connected with non-hacking groups. All interviewed organisers and task authors presented themselves as independent contributors, operating outside of their professional duties.

Overview. Rhythm of the game

One of the interviewees explained the usual rhythm of the game with the following: "We sit, and we fuck on." When asked to elaborate, the participant said: "How do we fuck on depends on the actual squad and depends on particular task mixture. Some guys are experts in particular categories, so they gather and do tasks from specific categories. As CTF proceeds, there are more and more tasks, but everybody tries to stick to his category." Tasks and categories provide a basic structure for the game while, to maintain intensity, the organisers propose new tasks every twelve or twenty-four hours. The intensity of cognitive work in CTF resembles hackathons (Irani 2015), but the major challenge shifts from general design concepts to solving specific, constrained problems. Unlike hackathons, CTF distinguishes between wrong and right answers, even if it allows the use of a broader range of methods.

While volunteering, hackers may organise CTF outside of their professional duties, the participating teams may emerge from hackerspaces, academic groups, or companies. The observed teams stemmed from a particular hackerspace or worked in the form of hackerspace/academia collaboration.

One team consists of five to twenty members, sometimes with a designated captain or another person responsible for handling communication with the organisers. Not all members of the team play in each tournament, although the top-tier teams need to play at least every week or two in order to maintain their high ranking. The interviewed players explained that their team played about fifty tournaments per year, with each competition likely lasting for twenty-four to seventy-two hours.

Teams may play CTF remotely or directly during the convention. Some organisers limit the maximal number of players or the number of remote players. During the observed contest, the team played directly at the convention. Members of the hackerspace discussed CTF, helped with particular tasks, and provided caffeinated drinks and alcohol for the five regular members of the team. About the same number of members participated remotely.

We may divide CTF into two major types: jeopardy and attack-defence. Jeopardy focuses on solving tasks provided by the organisers. Attack-Defence is about maintaining the stability of a service while disrupting the services of other teams. The annual league consists of both. I directly observed two jeopardy games and one attack-defence CTF.

One of the jeopardies started on the evening of the first day of the convention and lasted non-stop for forty-eight hours. The teams received tasks in two cycles: at the beginning and after twenty-four hours. Participants could work on the first batch even after the release of the second one. Both parts consisted of approximately fifteen tasks with different difficulty levels and rewards. The first hours were not very tense. Despite the release of the tasks, some players finished watching conference talks while others tinkered with hardware. As one of the participants commented: “The first 5 hours don’t matter much. The team who scores first rarely finishes first.”

Jeopardy and attack-defence CTFs differ in intensity and duration. Jeopardies are longer, have sleep breaks, and give tasks in batches. Attack-defence CTFs are shorter and have all tasks available from the beginning. Different time organisation creates different intensity, further diversified by the types of tasks in both types of competition.

As jeopardies usually take longer, sleep and endurance become issues. The observed teams slept for about four hours a day and worked for the rest of the time. The teams rested mostly after finishing a task, in order to maximise concentration. The intensity of this CTF matched the individual hacking sessions observed in hackerspace and described in previous studies (Coleman 2010; Irani 2015; Soderberg 2012; Söderberg and Delfanti 2015).

Constructing a task

Constructing a task is both a public demonstration and laboratory experiment. Author of a task demonstrates concept of an error, by providing deliberately faulty system, to be hacked in particular way. It is an experiment, because participants, who often are security professionals, are not aware of details. They need to repeat author’s path to discovery, but basing only on their own experience. In a sense: it is an equivalent to blind-tests. From the ethnograhpies of laboratories, we know that we cannot understand an experiment if we focus only on final result or aftermath. Similar principle applies to CTF: we need to study construction of a task as well as a process of solving it or documenting the solution.

Most popular categories in the jeopardy require the participants to gain control over software provided by the organisers (pwn), find and decrypt a message (crypto), reverse-engineer devices like game consoles (reverse), or hack an online service constructed by the organisers (web).

To solve a task in the jeopardy CTF, one has to find a string of characters called ‘flag’ hidden in the memory of the provided system, the code of the provided software, data files, etc. Sometimes, the flag is encrypted, and one has to find a way to decrypt it. At other times, the flag is an answer to a specific query in an online service. After finding the flag, the organisers assess if it is the correct one and rewards the team. There is no penalty for an incorrect flag.

Usually, organisers prepare tasks for the particular event and require no hacking of publicly available systems. Paradoxically enough, the scoring system itself is a local or public service and, as such, might be considered a target. Similarly, the flag is usually a rather short string of ten to twenty characters, hence prone to brute-force attacks, which involve checking all the possible combinations by raw processing power. However, an attack on the scoring infrastructure or brute-forcing the flag is

considered penalty, usually punished by a warning and, then, disqualification. In other words, one must hack the system within a particular task but not the competition system or the computer-aided flag guessing.

The organisers usually begin the construction of tasks either from task category or from an interesting type of bug. For example, if there is a need for a task in cryptography, one may start with a documentation of a particular file format like 'pdf' or a list of known bugs in the format in order to find a mechanism to hide the flag. Sometimes the task author may leave true or misleading signs by placing traces of bugs like wrongly coded content. Then the hacker must examine how is that particular data organised and what are its usual transformations; e.g., during one of the competitions, the flag was hidden in the result of spectral analysis of an audio file. The file's content will sound like noise.

Another way to construct a task is to start with a particular type of a bug. In this case, one has to consider a system prone to that type of bug. As an interviewed task author explained, "You have this idea of a great, cool bug, and you need to design something around it, and I think that's a bit harder. And you need to make sure that you don't mess up by adding other bugs." The bug should not result in random effects but lead to a precise, repeatable effect. Some tasks may involve multiple stages, in which finding and exploiting one bug leads to another element of the system.

If we analyse task construction from the ANT perspective, we will see a purposely designed instable system: an Internet service; a piece of software or hardware that challenges players. Paradoxically, the apparent instability of the system requires stabilisation work by an author or task testers. The object must be faulty but in a repeatable and traceable way. To ensure such stabilised instability, task authors for larger tournaments do some form of peer review. One of them said, "I think that's really hard because you probably should have somebody else look at it because you missed something. We had two web challenges here that had independent bugs and were made easier and breakable by that. That's bad because you didn't test it enough."

Stabilised instability matters also for the players. When asked, "What constitutes a good task?", several players answered it should be secure from blind guessing. It is acceptable to provide some baits or to suggest false routes to the solution, but the core bug needs to test knowledge about the peculiarities of given object. As one participant explained, "Least fun CTFs, in general, are those where I need to guess what I need to be doing and not by skills or so. I'm okay when I learn something, even if I don't succeed, I'm fine. But if I need to guess, I then read how it was done, and there was no skill part involved, only random trying or something, then I'm getting frustrated." When asked to compare CTF tasks with their professional computer security challenges, participants also said that CTFs are "the essence of hacking," due to the lack of randomness and concentration on technical aspects of a given object.

In addition to technical content, tasks often have stories or soundtracks connected with them, e.g. cryptography task might have introduction about secret operative lost in enemy territory that managed to send only one picture. Sometimes stories are obscene, taunt the player or focus on elements from anime or fantasy mythologies (e.g. My Little Pony universe). The observed players did not focus on the story content during any of the observed contests. Stories and sounds are regarded as distraction, an additional hurdle to make maintaining focus even more harder.

But players cannot disregard the stories or sound completely, as some tasks have hidden messages in images or songs. Within the observed groups, I noticed that such mechanisms were not popular, although accepted as a form of taunt or challenging the players to stay focused on the key issues, without losing a focus on context. It is the classical hybridity of hacking: one has to analyse technical

aspects of communication, but without ignoring its content. In order to master the medium, one has to switch between ignoring and analysing the message.

It is reflected in narrations about tasks. When asked to discuss a task, most participants told a story around an exploit or a system, disregarding the theme. Theme was brought to attention mostly when it was connected with a flag (e.g. song contained some elements of flag when analysed in some way), not when content was especially bizarre or offending.

Jeopardy CTF. Give me the laboratory, and I will hack the world

Jeopardy-type CTF treats tasks as closed challenges. As soon as the team finds the flag, it scores and moves to the next task. A notable exception occurs only when higher-value tasks have lower-value counterparts as introductions.

When asked to describe the initial steps in a highly-scored binary exploit task, a player said: "I run it on a virtual machine, look what is going on. I use debugger, load it to AIDA, look on the strings, then glance what the code is doing, more or less. When I know what the general idea is, what am I attacking, I switch to the high-level view. I noticed when virtual machine was active, so I looked at it using the debugger to find the point to hold into to download partial records of the virtual machine, without analysing its exact content. I figured out what it is doing without reading it from inside."

Task-solving usually starts with the analysis of the input-output of a given object in a virtual machine, sometimes with a dedicated software analysis tool like AIDA64. The virtual machine isolates the state of the system in order separate the workings of the object from noise. Outside of CTFs, programmers use virtual machines as sandboxes which enable them to look into a dangerous program without compromising the whole machine. It might be compared to the laboratory bench as described in laboratory studies (Knorr Cetina 1999; Latour and Woolgar 1986; Latour 1987, 1983). The virtual machine may function as a place for manipulation or a measurement tool. It is like having a laboratory bench with a battery of sensors that measure temperature and air contents around the sample.

Unlike in the usual laboratory experiments known from the works of Knorr Cetina's and Latour's, CTF participants deliberately avoid accessing the object too soon and focus instead on understanding its input-output characteristics. As we know from laboratory studies, to open a black-box may require considerable effort. In this scenario, the hacker opens it only partially and grasps some intermediary states. Rather than seeking complete access, the hacker activates a network of instruments around the object and repacks it into larger systems. Unlike the engineering projects (Vinck 2009; Bailey and Leonardi 2015), this system succeeds without stabilizing all of its actors. In this case, hackers stabilise all the inputs and outputs around the analysed object.

However, not all tasks permit such constructions. In some challenges, the hackers must gain access to several layers of a system. For example, in one high-value access control task (pwn), the team hacked the first layer of security by solving a modified version of a standard computing problem like the Hanoi towers. Then, they found that the system permits only a very limited array of operations. The team divided the work: some hackers concentrated on extending the array of operations while others programmed a tool to analyse the key points of the internal layer of the system. None of the hackers had the luxury of using the standard tools from outside of the analysed object because not even the input like keyboard operated as usual. In this case, the team had to build the mechanisms of stabilisation from scratch, within the system, and via very limited access routes. Metaphorically speaking, it was laparoscopic technoscience.

In both cases, the hackers constructed a stable perimeter of digital actors around the predicted vulnerabilities. The perimeters functioned as laboratories because they made each vulnerability repeatable and usable for the hacker. In the first case, the team set the perimeter outside the analysed object and use customised standard tools. In the second case, the team divided cognitive labour, created ad-hoc tools, and prepared the perimeter inside of the system.

The majority of the observed tasks were solved by one hacker who only sporadically communicated with the rest of the team to ask about details, brag, or discuss the initial strategy. Only the highly-scored tasks require a more refined division of cognitive labour. The teams employ several communication tools, most commonly a table with hackers assigned to each task, to avoid the unnecessary multiplication of workload. They also use chats and emails to communicate with other teams, organisers, and remote players.

Jeopardy CTF closely resembles popular image of a hacker. Solitary participants, each struggling with their tasks. At first glance: not much to study for a ethnographer. But if we follow digital communication channels, if we interview task authors and follow fates of stabilised instabilities, we will see a plenty of social (Latour 2005). Participants recognize their expertise and match with expertise of task authors, but mediation between them occur by constructing and solving tasks, not by oral or written messages.

Attack-Defence CTF. Give me reference, and I will raise myself

Social nature of hacking is even more visible in second type of CTFs, as colleagues from one team become defenders and attackers of a given system. System is no longer a one-time riddle, it becomes an infrastructure, which needs to be maintained taken care of. Stabilized instabilities become more dynamic, as they are constructed both by attackers and defenders from the same team.

Inside of each team there is a division between defenders, who are responsible for maintaining the service and attackers, who try to get the flag from the enemy infrastructure. Non-humans are similarly divided, as the participants cannot use other sources of computing power than their servers and personal computers. It is an important restriction because one of the viable options is a brute-force attack; e.g., an attempt to crash the system by excessive requests. During the observed CTF, the participants constantly balanced the load between scripts used to monitor services, loggers, and offensive programs. In the observed six-person team, one was to delete useless programmes or scripts once per hour.

The objects in attack-defence CTF are more complicated and more closely resemble real-world systems because so they may be attacked in numerous ways. The observed CTF teams constantly updated the possible attack routes; once one was secured, they switched to another. Constant rearranging was also the standard practice in defence, as some players worked on immediate crises while other developed a larger patch to secure several routes at once. As the game is in constant flux, hackers develop several semi-stable points of reference. Each team has to defend and attack at the same time. You cannot ignore either because points are given for both. But you cannot ignore either for more important reason: as each activity provides a reference for another. Your opponents constantly modify their systems and try to develop new entry points. When you have to constantly rearrange your theories about own system and systems operated by other teams, you have to cooperate with your teammates.

Cooperation within a team is also a mechanism of stabilisation of an instability. If you are a defender, you still try to implement experiences of the attacking part of a team, you try to use their

experiences into patches, fixes and protection measures. Similar applies to attackers, over time enemy systems are patched and you need to discuss patches with defending colleagues: not only to break them, but to enhance your own defences.

As the tournament goes, systems become more and more different. As one of the players said: "I've got it. This task is easier at the beginning of CTF when different teams have similar services. As CTF develops, competitors' systems begin to differ in multiple ways. Other teams patch up the system, restart after failed patches, or kill some processes in order to save processing power."

As the system changes rapidly, even the recognition of normal/irregular states becomes a daunting task. After a few hours of CTF, one participant described the defended system as a "nightmare" with unfinished processes, leaking from several routes, and operating at the limits of capacity. Both attackers and defenders use logs from their systems to position the current state against historical data and to recognise between leaks, ineffective resource allocations, and crucial elements of the service. Some players deliberately do not change the system during the first minutes in order to get 'the baseline:' the regular state of the system; that is, they lose points at the beginning to gain advantage in the future.

Another source of reference is the scoreboard which signals the stability of services and currently effective hacks. The team does not know which enemy is which hacking team, but they can see whether they are losing points due to an exploited leak or gaining them due to a stable system. During the observed CTF, the team looked at the scoreboard every fifteen minutes.

Speaking in ANT terms: the attack-defence CTF is a struggle over black-boxing of technological system (Latour 1999, p. 184). The defenders try to make the service more reliable. To score points, the following chain of actors needs to work without external actors: simulated user – defended system – system response – scoring mechanism. The attackers may try to interrupt the system at any of such passages by providing fake users, breaking into the system, changing its response, or disrupting the communication with the scoring mechanism. The situation is volatile, as each solution only rarely works safely for longer than forty minutes. Defenders work with multiple actors to create references. They construct the normal state of the system with the use of historical logs. They predict the current safety of the system by referring to the scoreboard. They construct potential future threats by using data from the attackers from their team. In this case, black boxing is not about closing the system from the world; it is about constructing and measuring the system by using additional stable actors.

The defenders aim for reliability and security of the system. However, it will be a mistake to understand the attackers' role only as causing random havoc in the proposed chain of actors. It may be a viable strategy for beginners, but the scoring mechanism favours the interception of data over destruction. Gaining access to a flag and swapping it for a fake one is the optimal scenario. Thus, the attackers try to construct a similarly stable chain of references with only minor detours.

Exploit in a computer system resembles the same paradox as bacteria in society. In order to reduce its presence, one has to build it over and over again in laboratories, constructing it anew in order to find other weak spots. Attack-Defence CTF show that in order to have stable, reliable computer systems, one also needs to have stabilised procedures of creating instabilities. But what happens when experiments end (Galison 1987)? How instabilities exist outside of particular systems?

Writing up

As the challenge ends, the teams usually meet at a party or on online chat channels. Some congratulate their rivals, some brag or chat with acquaintances from other teams. The intensity of the game is slowly dissipating, and the non-hacking matters gradually emerge. Some exchange public keys to encrypt future communication while others discuss the ongoing CTFs or possible professional collaborations.

After the party, write-ups began to emerge. Players record their solutions, compare different approaches and publish them on repositories, blogs or in papers. Questions arise: Were the obvious routes really apparent to the majority of teams? Did the system fail in a standardised, predictable manner? Was my approach doomed from the start, or did I just drop too early?

Write-ups focus on the critical details of individual tasks, only rarely considering the whole CTF. They usually concentrate on the key operations in jeopardy summaries while considering the whole logic of the given system in attack-defence. Jeopardies question vulnerability mechanism and exploitation possibilities, whereas defence-attacks motivate a reflection on vulnerability in the context of whole systems. In both types of write-ups, readers value abstraction. One task author described the role of write-ups when assessing the writer: "If he[1] cannot explain the problem well, that's a sign that he doesn't really understand the problem. He may have solved the task, but he didn't really understand it fully to a degree that you can verbalise in a very good way how the task was built or how the task worked, or was supposed to work, and where the bug was. A very good player will be able to abstract how the task worked and explain it on a very high level what it should be doing, where is the bug, why it happens, how you can prevent it, and so on."

Depending on the level of generalisation, write-ups vary from tales from the field (Orr 1996) to inscriptions (Latour 1990). The former focus more on particular vulnerability and solution, which makes them closer to case studies or personal accounts. The latter concentrate on the general mechanism and the role of particular vulnerability in a wider class of systems. Description of particulars is more common, but generalization is recognized as a sign of exceptional mastery. In result: write-ups not only record solutions but also categorize them and construct meta-theories of exploits.

Discussion

By adopting the STS approach to the construction of technology, we furthermore address the discussion about the moral and ontological dimensions of computer exploits (Taylor 1999, p. 75-90). To paraphrase Latour: computer exploits are real because they are constructed in CTFs. The specificity of the technoscience of hacking lies in treating exceptions and errors as stabilised, usable products rather than controversies or failures to be solved. As the laboratory life of hackers expands to new domains (Hunsinger and Schrock 2016), hacking laboratories will likely construct stabilised instabilities in areas as biohacking or 3D printing.

If CTF is successful, then we may notice the following process. A task author constructs a faulty system. The whole system is prone to instability but has to be instable in a reliable, repeatable way. Stabilised instability enables sharing of the exploit among the hacking community. The task author constructs the error along with the participant. Unlike black-boxes in formal technoscience, the objects in CTFs are expected to be misused (Söderberg 2010). The misuser is not a bug in CTF; it is a deliberate effect of action between the task author, the task, and the player. The construction of the participant is symmetrical to the construction of the object in the task; he/she is expected to misuse,

hack, and utilise the non-normal state of the error. Both the participant and the object of CTF are expected to be stabilised exceptions, bugs and exploits in computer systems.

An open task may be understood in terms of ANT as controversial because it is unstable, prone to manipulation, and awaiting closure. The task may also be understood as a trial of force for the hacker: solve the task and you will enter the whole collective of CTF. Your professional life may develop as winners of the major CTFs receive recognition on the HackerNews, a website with everyday information about hackers' feats. If the participant succeeds, then the particular bug that he used in the task becomes a point of reference. This point of reference was passed from the task author to the participant through that task. It may be described and disseminated to further actors: other hackers, hacktivists, or automated learning systems used by the military agencies.

We saw the examples of establishing different stabilisation mechanisms: from weaving a bug into a measurement network to orientating the defended system by references to the past and present. We saw how successfully CTF circulates knowledge from the author through the participant to inscriptions and tales. We now see that the whole CTF tournament acts as a laboratory, understood in the STS terms. CTF constructs new actors; by providing a trial of force, it constructs both hackers and exploits. CTF facilitates knowledge circulation with write-ups and new loops of circulating references. Finally, CTF stabilises some computer exploits as known exceptions and stable instabilities.

Unlike solitary hacking, CTFs are deliberately closed in terms of time and space, as well as other experiments they have to end (Galison 1987). Designed tasks allow to reduce blind-guessing, which would be otherwise a regular part of security work. This allows to concentrate on intricacies of tools, quirks of the systems and overlapping of specializations. Another difference between CTF and "normal" hacking-security practice (Turgeman-Goldschmidt 2005; Taylor 1999), is the opportunity to specialize in different aspects of security cycle (recon, intercepting communication, cracking cyphers, reverse engineering of existing systems, hardware). Hacking groups were noted in history, but CTF provides compromise between everyday collaboration in a company or research group and solitary practice. I noted that some teams recruit some extra specialists for important CTFs and use others to test potentially aspiring candidates. But CTF teams are not set in stone, as many of the teams told me that they wish to go beyond usual faces, known from work.

One may argue that the objects used in CTF do not resemble 'real-life' systems, they are intentionally exploitable, incomplete and only rarely updated. Bank services, as used in many web-based CTFs, are clumsy, without many basic tools. Bu, as we know from laboratory studies (e.g. Knorr Cetina 1999, p. 26 and 71-75), laboratories often work with assemblages, which are simplified, reduced in space or in time scale. Laboratory situation or trial of force (Latour 1987) does not have to resemble external world. In case of CTFs: using bizarre configurations, incomplete or substandard systems allows to test limits of particular security approaches and range of application for particular exceptions.

Systems, programs, and hardware parts are not products of a CTF laboratory. The task authors, write-ups, and participants only use them to construct the exploit, error, or possible attack route. This also marks the difference between CTFs and laboratories in other technosciences. The latter constructs stable objects while the former establish exceptions. 'Regular' technoscientists construct stabilised black-boxes. Hackers create instabilities in stable objects. As we know from laboratory studies, a lot of artificial conditions is required to show a state of nature. The situation is symmetrical in the case of 'hacking laboratories:' a lot of artificial conditions is required to construct 'a real' computer vulnerability. The flag itself acts not only as a quantifiable proof of access but also as a framing device, helping to focus hacker's attention on the process, not on the content.

One example was observed by me during one of the top-tier CTFs. Organizers provided slightly wicked configuration of a web service, uncommon but theoretically imaginable. Web service was used to provide back-end for some online shopping or online banking [2]. There was a deliberate vulnerability planted in this configuration by the task author, but one of the hackers found an alternative solution. When he consulted with organizers and documentation, it was revealed that it was zero-day. Zero-days are equivalents for new mathematical theorems or other discoveries, because this term applies to previously unknown vulnerability, or at least vulnerability unknown in civilian computer security technoscience. Team scored some points, because this zero-day enabled them to get the flag, but immediately after the tournament hackers started to work on testing this zero-day and publishing a report on it.

Discovery of zero-days happen extremely rarely, but one may say similar things about mathematical theorems or physical measurements. Most of the time, scientists confirm and slowly extend range of their thought collective (Fleck 1979). Similarly, majority of CTFs is about repeating the old exploits in slightly changed configuration. This dynamic is another argument for treating CTFs as laboratories in the STS sense.

By constructing, testing and repeating constant exceptions, hackers enable the continual rearrangement of networks in computer technoscience, thus creating new openings for the next generations of devices, products, and services. In the classical Schumpeterian model, new exploits and bugs create new digital frontiers for the companies. Similarly to other products of hackers, CTFs are included into capitalistic networks. The companies use CTFs to recruit or train new security specialists. Military agencies use CTFs to test their own actors. DARPA used of the most prestigious CTF, Def Con, to compare its automated hacking system with human teams. Human teams won this time, but the circulating reference perpetuated in this tournament was used to improve the military hacking machine. The observation of such hybrid CTFs may be used as a probe into the state of play between independent hackers and other actors.

The concept of stabilised instability provides non-human symmetry to the human figure of misuser (Söderberg 2010). Incomplete, constantly rearranged users are irreversibly linked with incomplete, constantly rearranged systems. CTF produces and facilitates such incompleteness and rearrangement which are subsumed into political and economic networks of computer security. In this sense, CTFs and hackers are elements of capitalism due to not only their contribution to the open-source programming but also acting as a mechanism for exceptions and re-arrangements.

Conclusions

Capture the Flag tournaments function as laboratories defined in terms of classical ethnographies. CTFs open technological black-boxes, rearrange actor-networks, and reshape force relations. Unlike academic thought collectives, CTFs and other hacking laboratories construct exceptions, vulnerabilities, and critical cases while emphasising further generalisations to a lesser extent.

We may understand the exceptions as controversies in existing black-boxes or products of the stabilisation work done by hackers. It leads us to the possible rethinking of the symmetry and stability in the ANT and studies of science: instability may result from separate stabilisations. Possibly, this approach to stabilised exceptions is not unique for hackers but may also be found in formal sciences like physics or mathematics. Perhaps, the difference between the role of exception in physics and hacking refers to Coleman's and Golub's proposal for understanding hacking as a liberal practice, liberal in sense that one exploit does not falsify the whole system, but opens possibilities for

its rearrangement. In this sense, hacking is a way of managing contradictories not only in politics but also in results of laboratory operations.

Perhaps the symmetries used in STS and the ANT since the Strong Programme should be a call to treat exceptions and stabilities with the same theoretical tools. Perhaps we should reconsider controversy not only as something to be exploited or solved as if it was a temporary inconvenience but also as something precious, which requires constant attention and care both from hackers and STS. After all, scientific facts are similar to exceptions generated by hackers: they are real because they are constructed.

Funding:

Research was funded by Polish National Science Centre (Narodowe Centrum Nauki), Preludium research grant 2014/13/N/HS6/04113

Notes:

[1] Almost every CTF player and task author interviewed, observed, or spotted in this study described himself as male. Some women write summaries after CTFs or try to solve tasks on their own outside of CTF events or as a part of a wider computer security community. Possibly, a CTF operates as a clubhouse mechanism, as described in previous works (Margolis and Fisher 2002), but it requires further examination in separate studies. I do not have data whether the teams from feminist hackerspaces (e.g. Fox, Ulgado, and Rosner 2015) participate in the CTF league, do they create separate networks or do not exist at all.

[2] I am being deliberately vague in order to protect anonymity of participants, who are still researching limits of this exploit.

References

Bailey, Diane E., and Paul M. Leonardi. 2015. *Technology Choices: Why occupations differ in their embrace of new technology.* Cambridge, MA: MIT Press.

Coleman, E. Gabriella. 2010. “The Hacker Conference: A Ritual Condensation and Celebration of a Lifeworld.” *Anthropological Quarterly* 83 (1): 47–72.

———. 2013. *Coding freedom: The ethics and aesthetics of hacking.* Princeton, NJ: Princeton University Press.

———. 2015. *Hacker, Hoaxer, Whistleblower, Spy: The many faces of Anonymous.* London, Brooklyn, NY: Verso.

———. 2017. “From Internet Farming to Weapons of the Geek.” *Current Anthropology* 58 (S15): S91-S102. doi:10.1086/688697.

Coleman, E. Gabriella, and Alex Golub. 2008. “Hacker practice: Moral genres and the cultural articulation of liberalism.” *Anthropological Theory* 8 (3): 255–77. doi:10.1177/1463499608093814.

Cowan, Crispin, Arnold Arnold, Steve Beattie, Chris Wright, and John Viega. 2003. “Defcon Capture the Flag: defending vulnerable code from intense attack.” In *Information Survivability Conference and Exposition*, edited by DARPA.

Enserink, Martin. 2008. “Academic Hackers in Court.” *Science* 321 (5886): 189. doi:10.1126/science.321.5886.189b.

Fleck, Ludwik. 1979. *Genesis and Development of a Scientific Fact.* Chicago, IL: Chicago University Press.

Fox, Sarah, Rachel Rose Ulgado, and Daniela Rosner. 2015. “Hacking Culture, Not Devices.” In *18th ACM Conference Proceedings*, edited by Dan Cosley, Andrea Forte, Luigina Ciolfi, and David McDonald, 56–68.

Galison, Peter. 1987. *How experiments end.* Chicago, IL: University of Chicago Press.

Hunsinger, Jeremy, and Andrew Schrock. 2016. “The democratization of hacking and making.” *New Media & Society* 18 (4): 535–38. doi:10.1177/1461444816629466.

Irani, Lilly. 2015. “Hackathons and the Making of Entrepreneurial Citizenship.” *Science, Technology & Human Values* 40 (5): 799–824. doi:10.1177/0162243915578486.

Jordan, Tim, and Paul Taylor. 1998. “A sociology of hackers.” *Sociological Review* 46 (4): 757–80.

Knorr Cetina, Karin. 1981. *Manufacture of Knowledge: An Essay on the Constructivist and Contextual Nature of Science.* Oxford, New York, NY, Toronto, Sydney: Pergamon.

———. 1999. *Epistemic Cultures: How the sciences make knowledge.* Cambridge, MA: Harvard University Press.

Latour, Bruno. 1983. “Give Me a Laboratory and I will Raise the World.” In *Science observed: Perspectives on the social study of science*, edited by Karin Knorr Cetina and Michael Mulkay, 141–71. London, Beverly Hills: SAGE Publications.

———. 1987. *Science in Action: How to Follow Scientists and Engineers Through Society.* Cambridge, MA: Harvard University Press.

———. 1990. “Visualisation and Cognition: Thinking with Eyes and Hands.” In *Representation in Scientific Activity*, edited by Michael Lynch and Steve Woolgar, 19–68. Cambridge, MA: MIT Press.

———. 1999. *Pandora's Hope: Essays on the Reality of Science Studies.* Cambridge, MA: Harvard University Press.

———. 2005. *Reassembling the Social: An introduction to Actor-Network-Theory.* Oxford, New York, NY: Oxford University Press.

Latour, Bruno, and Steve Woolgar. 1986. *Laboratory Life: The Construction of Scientific Facts.* Princeton, NJ: Princeton University Press.

Lindtner, Silvia. 2015. “Hacking with Chinese Characteristics: The Promises of the Maker Movement against China's Manufacturing Culture.” *Science, Technology & Human Values* 40 (5): 854–79. doi:10.1177/0162243915590861.

Lynch, Michael. 1985. *Art and artifact in laboratory science: A study of shop work and shop talk in a research laboratory.* Studies in ethnomethodology. London, Boston, MA: Routledge & Kegan Paul.

Margolis, Jane, and Allan Fisher. 2002. *Unlocking the Clubhouse: Women in computing.* Cambridge, MA: MIT Press.

Nissenbaum, Helen. 2004. “Hackers and the contested ontology of cyberspace.” *New Media & Society* 6 (2): 195–217.

Orr, Julian E. 1996. *Talking about machines: An ethnography of a modern job.* Collection on technology and work. Ithaca, NY: ILR Press.

Soderberg, Johan. 2012. *Hacking Capitalism.* Routledge research in information technology and society. London: Taylor & Francis Ltd.

———. 2013. "Determining social change: The role of technological determinism in the collective action framing of hackers." *New Media & Society* 15 (8): 1277–93. doi:10.1177/1461444812470093.

Söderberg, Johan. 2010. "Misuser Inventions and the Invention of the Misuser: Hackers, Crackers and Filesharers." *Science as Culture* (19): 151–79.

———. 2013. "Automating amateurs in the 3D printing community: Connecting the dots between 'deskilling' and 'user-friendliness'." *Work Organisation, Labour and Globalisation* 7 (1): 124–39. Accessed January 12, 2015.

Söderberg, Johan, and Alessandro Delfanti. 2015. "Hacking Hacked! The Life Cycles of Digital Innovation." *Science, Technology & Human Values* 40 (5): 793–98. doi:10.1177/0162243915595091.

Taylor, Paul. 1999. *Hackers: Crime and the digital sublime.* New York: Routledge.

Turgeman-Goldschmidt, Orly. 2005. "Hackers' Accounts: Hacking as a Social Entertainment." *Social Science Computer Review* 23 (1): 8–23. doi:10.1177/0894439304271529.

———. 2008. "Meanings that Hackers Assign to their Being a Hacker." *International Journal of Cyber Criminology* 2 (2): 382–96.

Vertesi, Janet. 2014. "Seamful Spaces: Heterogeneous Infrastructures in Interaction." *Science, Technology & Human Values* 39 (2): 264–84.

———. 2015. *Seeing like a Rover: How robots, teams, and images craft knowledge of Mars.* Chicago, London: The University of Chicago Press.

Vinck, Dominique. 2009. *Everyday engineering: An ethnography of design and innovation.* Inside technology. Cambridge, MA, London: MIT.

Zaród, Marcin. 2015. "Hacking collective as a trading zone. Notes from the ethnography of hackerspaces in Poland." *Kultura i Edukacja* 4 (110): 134–52.