

UCLA
limn

Title

I am Not a Hacker

Permalink

<https://escholarship.org/uc/item/5xq0v7rb>

Journal

limn, 1(8)

Author

Bialski, Paula

Publication Date

2017-03-10


Copyright Information

This work is made available under the terms of a Creative Commons Attribution-ShareAlike License, available at <https://creativecommons.org/licenses/by-sa/3.0/>



61 11 2 12
HERBERT E. CLIFF
26 NOTTINGHAM RD.
SHORT HILLS N. J.
RSG 010 20004 135790
1/ 6/64 2448
RS 010 00004 246802
1/ 6/64 8774

TELEVISION



The term “hacker” is notoriously slippery. **Paula Bialski** dives into the practices and micropolitics of self-proclaimed non-hackers.

I am not a hacker

NOAH, A CORPORATE SOFTWARE DEVELOPER, AND I MET AT

a German language class when we were both living in Hamburg. As the years went by, our friendship flourished through our love for hummus and our mutual interest in tech culture. He graciously fielded my endless tech questions, and was the first techie I ever met who wasn't bored by my ignorance, but rather reveled in my queries about the logics and logistics of computing. When speaking to Noah, I came alive: my mind racing, picking apart the world of our smartphones that we (as regular users) wouldn't normally see when simply looking down at the object sitting in our hands.

Thanks to Noah's stories about where he worked and how he worked, I finally could picture the people behind that screen: their frustrations, the tests they were doing on us, the conversations they were having over one feature or the other. Each button, each little tiny object suddenly had a backstory, even a logic to it. My chat app, whose features once struck me as odd, even arbitrary—a particular swipe capability, specific colors, certain moments of flashing on and off, and other bizarre ways of behaving—finally made some sense. Who made my thumb able to swipe left and not right? When my phone collects my GPS data when I run, where does the data go, and what group of people are making the decision that my data will trigger another feature that allows me to listen to music at the speed of my running pace? Noah made me want to meet those people like him who designed the technologies saturating our daily lives, to talk to them and see what exactly they looked like, what food they ate for lunch, where they were born, and what music they listened to while coding. Through Noah, digital media technology became nonstatic, viscose, constantly shifting like a ball of clay that a group of previously mysterious and magical people were collectively pushing and pulling on, reshaping its size, purpose, and scope.

CORPORATE SOFTWARE DEVELOPERS ARE a rather enigmatic bunch of tech workers, at least when compared with the hacker, who has received far more academic and public scrutiny. Still, these technologists make all

sorts of design choices and decisions that shape the way our practices or certain forms of sociality unfold when using the digital media they create. They also hold “control over a valuable skill” (Ensmenger 2010: 231), meaning at times they—and only they—know what the heck is going on. For example, some developers are the only ones who know what beta version of an app feature is being run at what time on what specific group of users. To their bosses, their managers, their partners and mothers, and all other nondevelopers, the systems they build can even resemble the stuff of magic. After starting fieldwork at Noah's workplace—a large (1000-plus employee) software company I call BerlinTech—I also began to uncover myriad moments when they exerted their “control over a valuable skill.” I became attuned to these moments where developers used their skillful power over those who had less (e.g., their managers), or no skill at all (e.g., the user experience [UX] designers in their team). Their power and skill may seem nothing but hackish, especially in the eyes of nondevelopers who often only know about the world of technology from sensationalistic headlines and increasingly popular TV shows like *Mr. Robot*.

Yet this is where things get tricky: corporate software developers vehemently deny they are hackers. During our conversations, or overhearing their discussions online, in team meetings, or by the coffee machine, developers wanted nothing to do with the label “hacker,” and would shake their heads when asked if they “hacked.”

In the words of Sam, one of the coders I worked with, most corporate software developers associate the term “hacker” to two specific behaviors: “(1) Anarchist activism against oppressing institutions in order to regain in a way a certain flavor of human liberty; (2) direct work with security systems...either trying to break or to protect.” (Sam, BerlinTech developer, December 2016).

Sam explained that the word “hack” is merely used for “dirty coding, in order to conceptualize and create quick solutions” (field notes, December 2016), and other developers, during my conversations with them at work, linked the “hacker” with criminal activity.

Here I present an apparent paradox, common to ethnographers, through a tongue-and-cheek retelling of three stories from my field: what your respondents tell you often fails to match their actions and behavior. When asked if a software developer is a hacker, or identifies with being a hacker, they will quite often say “No.” Yet after months of observation, many of their actions resemble those we commonly attribute to the hacker. As we have seen in many other ethnographies of hacker communities, hacking is about experimentation, political gestures, and craftiness (Coleman 2013, 2014). Software developers are no exception. Sometimes the only difference lies in their verbal disavowal of this identity or, if pressed, they may admit to some limited resemblance: at most, they may frame their activities as a species of micro-hacking, intended for their own personal use, or only for their employer.

CONTROL OF A VALUABLE SKILL

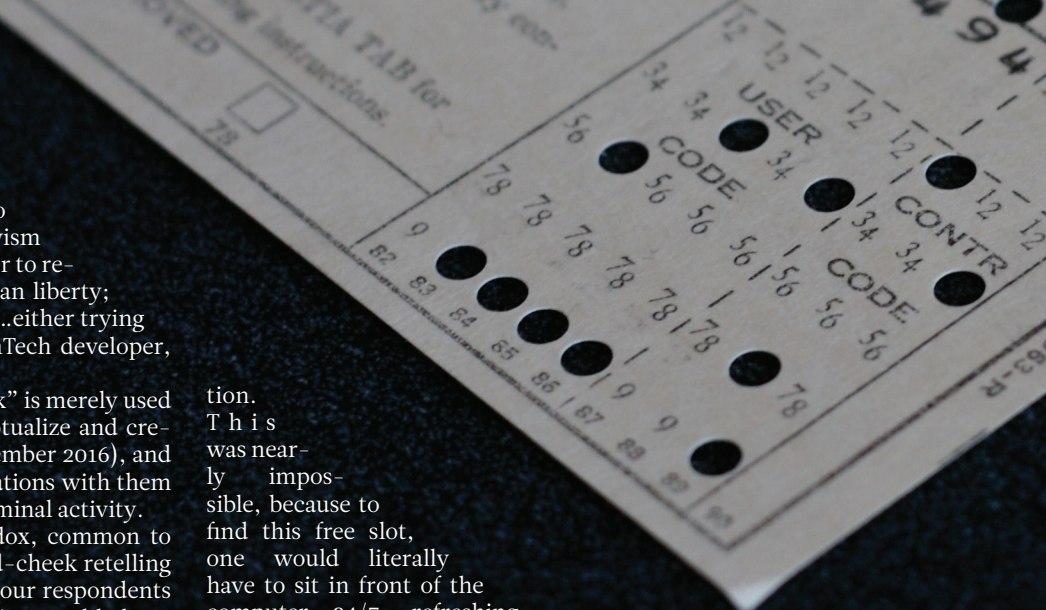
With this in mind, I rewind a few months back to a moment before I started my ethnography at BerlinTech. Noah and I were sitting together on the grass, enjoying the heat of July. Noah started mid-sentence as if he opened his head for me to see what was mulling through his brain: “I really doubted what you said last week. I doubted that programmers are really superstars, the new and powerful class of worker. I wasn’t so sure.” I nodded, listening, returning to the moment that Noah questioned my thesis that all programmers—not only the geniuses and those deemed “superhackers”—command more power not only for building technologies but maneuvering and outmaneuvering these systems. But then Noah offered a recent revelation: after years of having one foot in Tel Aviv and one foot in Berlin, Noah decided to really move to Berlin, register himself as a resident, gain a German driver’s license, and start taking intense German courses. To do so, he would have to register himself in the *Stadtamt* (city office), infamous for its very annoyingly snail-slow bureaucracy. The city of Berlin offers an online sign-up system for appointments, yet the downside is that the system has a three-month-long waiting list. The previous week, Noah explained, he had logged onto the Berlin city hall website, and managed to get an appointment for late August (which, being early July, was already pretty awesome). The only way to land an earlier spot is through a cancella-

tion.

This was nearly impossible, because to find this free slot, one would literally have to sit in front of the computer 24/7, refreshing the page and checking for cancellations. Frustrated, Noah coded a script that automatically scanned for cancellations in the city hall’s appointment schedule. “So, Paula, in this moment I realized that we are sort of a new upper class. Not in a big way, but in all the little tiny ways like being able to create a script that helps you make a driver’s license appointment, or making a script to help with booking a cheap train ticket. There are all sorts of little everyday examples that help us get in the backdoor of all sorts of systems that run our lives.” This skill is obviously shared by hackers (for example, in their “craftiness” to break through various systems), yet the difference among software developers like Noah is that they do not use the common vocabulary to frame this activity like hackers do.

In my months at BerlinTech, I encountered numerous other examples of developers who built apps for their own personal enjoyment and needs (without ever “releasing” the app for general public use) or, in the case of Noah, building a program that would help circumvent the limitations of a given system like a city infrastructure or customize the infrastructure they were being paid to build without the knowledge of their employers or managers. Although I don’t want to paint every developer as always able to enter the backdoors of every system, the developers I encountered explained these practices with such ease, as if building or breaching was part of their being in the world. It struck me as something similar to how the well traveled explain navigating through an airport, or how a marathon runner effortlessly describes the 15-kilometer run they just completed. This effortlessness or ease of building a new digital tool, customizing an existent piece of infrastructure, or breaching a seemingly closed system was part of the life of being a developer in an increasingly digital society.

If this “control over a valuable skill” can help exert a certain power over an infrastructure, or an organizational system, then why this perceived paradox? Why are software developers not “hackers”?





1 “I’M NOT A HACKER BECAUSE WHAT I BUILD IS DONE FOR FUN”

“Creating apps [in one’s] free time is like having a Lego in your computer,” remarked Sam during an online conversation. Software development, he insisted, is about building, about creativity, about craftiness, and about playfulness. Whereas hackers break or build as a means to get to a specific ends (we “break” *something* to get *into* “somewhere”), software developers claim that they create just to make something, without a specific desired outcome of “breaking down” or “breaking in.” The irony of this approach is that both interacting with a computer in an exploratory way, as well as building something to produce exactly what is needed, are both definitions of hacks, but not seen as such in the eyes of these developers (see <http://catb.org/jargon/html/H/hack.html>).

Sam said that indeed, they do “gain power” with the competences to manipulate a certain technical system, but they do so:

...to regain a space for creativity. In the end, it’s our space, we do it in our free time, and by doing that we are not affecting the processes, or the institutions, just our status with ourselves” (Sam, BerlinTech, December 2016).

According to Sam, personal pet projects don’t mix well with the heady and serious world of protest and politics. Protest must be seen and heard, and what he builds for “fun” or “to be creative” is often invisible, for his eyes only: “Creating an app at home during my free time is as political as silently protesting on a Sunday afternoon on the streets.” In silence, in hiding, Sam can create for fun. Hackers, on the other hand, are not silent. Hackers create for something, to achieve a certain end.

2 “I’M NOT A HACKER BECAUSE MY POLITICS ARE TOO ‘NANO-SCALE’ TO BE NOTICED”

Sam and a few of his colleagues made their own feature for the app they were building for their company. Their product managers didn’t commission this feature, nor was their company ever going to use it. Yet when they finished what they were doing for work, they would (sometimes during office hours) breach their work system, subvert their bosses, and go behind their backs and built it anyway. They’d do so in their free time, when their bosses weren’t looking. While this gesture has everything resembling a hack, breach, or skillful subversive act done to regain a certain sense of power, Sam and his friends did not perceive it that way: “It’s really nano-political,” Sam explained.

We try to transcend either the process or the product. But that rarely happens.... [T]he culture is very heavily impacted by the hierarchical organisational structure.... [S]ometimes you try to bring those ideas to the normal working environment, which often acts as a bouncing wall [at least in this company, at the moment].... You throw ideas, but often those ideas bounce back to you, with no way to impact the structures or the culture.

These “nano-political” moments of breaching are attempts to regain power (or is *agency* the better word?), but they are humble, and done in a smaller scope. Sam explained that these acts couldn’t be seen as true hacking because they “barely impact their work processes,” let alone a larger scale. Hackers, on the other hand, act big. Hackers hack on a large scale.

3 “I’M NOT A HACKER, I’M JUST EXPERIMENTING”

Software developers would sometimes hack, breach, and break systems “by mistake” or during the process of learning. The software developers at BerlinTech, much like many tech companies, were encouraged by their managers to experiment. As one manager explained, it is their “responsibility” to allow their software developers to experiment during hack-a-thons, team-coding sessions, and “research weeks.” During these sessions and others, “you experiment, you learn, and based on this, you bring ideas to your environments,” explained Sam. Yet Sam reiterated that his technical “experimentation” is *not* hacking. These experiments might involve breaking certain systems or breaching territory that is not generally intended for that specific use (mainly a system within one’s own company). What differentiates this from “true hacking” is that a company like BerlinTech can capitalize on whatever arises out of this experimentation, “bringing your ideas back into your environment.” Developers perceive that what they are doing is just experimenting to directly benefit the output of their company. Real hackers just hack, without a third party intending to capitalize on what they are doing.

SOME CONCLUSIONS

As ethnographies of corporations (e.g., Wittel 1997) or the working class (Bachmann 2014; Kracauer [1930]1998) have shown, the seemingly mundane, everyday practices of work—whether those of a software developer or clerical manager—are also about power, construction, destruction, dreams, fears, and foes. The ability for a developer to secretly create a feature, or cleverly bypass/route around government bureaucracy, can be political without being an “epiphenomenon, or a manifestation or instrument of grander movements” that affect a larger society or group (Burns 1961:264). The developer’s persistent efforts to improve their chances, or use of their skills in protesting the way in which an institution functions, can be seen, as sociologist Tom Burns explained more than 50 years ago, as “micropolitical” (Burns 1961): where physical and human resources present in institutions, corporations, or organizations “accumulate and then widen and alter the possibilities of political action” (Burns 1961:281). If the managing director or product owner is the central source of visible power, the software developer’s invisible power lies in acting in hidden ways, behind their field of knowledge. This is how they gain a sense of agency and power, which can become political. Today, with the protest, scandal, and criminality circling around the term “hacker,” developers perhaps have attempted to distance themselves from the term

when speaking to me (their ethnographer), much like the Certified Ethical Hacker attempts to destigmatize their own practice by disassociating themselves with the “political hacker,” as Rebecca Slayton describes in this issue.

Not all corporate software developers are powerful technological agents. Many also work with constraints and even through states of ignorance (because developers often don’t know what’s going on within the software system they are working on). Moreover, more than ever, automation and artificial intelligence makes their “power” in some ways obsolete. Still, software developers do enact agency and power with their skills and capacities; their actions, however small they may seem to be, are neither mundane nor inconsequential. Understanding the technologist’s “agency” and the micropolitics of software development can help us understand the various shapes and forms “hacking” takes on, as well as the weapons, skill, and control that is intrinsic to the culture of software development. ■

PAULA BIALSKI is a postdoctoral researcher at Leuphana University’s Digital Cultures Research Lab (DCRL), where she bides her time pestering students in the Digital Media BA program and conducting an organizational ethnography of a corporate tech company in Berlin.

BIBLIOGRAPHY

- Bachmann, G. (2014). *Kollegialität: Eine Ethnografie der Belegschaftskultur im Kaufhaus*. (Collegiality: An ethnography of employee culture in a department store). Frankfurt, Germany: Campus Verlag.
- Burns, T. (1961). “Micropolitics: Mechanisms of Institutional Change.” *Administrative Science Quarterly*. Vol. 6. No. 3. 257–281.
- Coleman, E. Gabriella. (2013). *Coding Freedom: The Ethics and Aesthetics of Hacking*. Princeton, NJ: Princeton University Press.
- . (2014). *Hacker, Hoaxer, Whistleblower, Spy: The Many Faces of Anonymous*. London, UK: Verso Books.
- Ensmenger, N. L. (2010). *The Computer Boys Take Over: Computers, Programmers, and the Politics of Technical Expertise*. Cambridge, MA: MIT Press.
- Kracauer, S. (1998). *The Salaried Masses: Duty and Distraction in Weimar Germany*. Originally published 1930. London, UK: Verso Books.
- Wittel, A. (1997). *Belegschaftskultur im Schatten der Firmenideologie. Eine ethnographische Studie*. (trans: Employee culture under the shadows of company ideology: An ethnographic study) Berlin: Stigma.