Botball Hacking 101 (Part 2)
Jeremy Rand
Team SNARC (Sooners / Norman Advanced Robotics Coalition)
jeremy.rand@ou.edu

Botball Hacking 101 (Part 2)

14 Welcome to Part 2!

Glad you made your way here. And now, Part 2.

15 Two Ways to Go About Finding Useful Hacks

There are two good approaches for finding new stuff to hack. The first approach is if you have an idea of what you want to achieve, and you simply search for code or documentation which allows achieving this. My camera hacks for the CBC used this approach. This approach tends to yield faster results, but since you're limited by your imagination (which may not be very creative), it tends to also yield less innovative results.

The second approach is to simply pick a file or document you've never read, and see if there's anything interesting in it. My sound playing hack and multiple program storage hack for the XBC were developed this way. This often yields more innovative results, but sometimes wastes time (I recommend doing this when you're not on a deadline).

Good hackers typically utilize both approaches.

16 Don't be Fooled by the Illusion of Security

For the first month when I was hacking the CBC, it was fairly common for me to describe the CBC as "pretty well locked down." As it turned out, the CBC was anything but locked down; both Chumby Industries and KIPR had left huge amounts of hackable stuff on board. Usually, a device will appear "locked down," up until about 5 minutes before it's "completely hacked."

On the other hand, sometimes things really are locked down. The SRF04 sonar, which was used in Botball through 2008, really does seem to be impossible to hack in Botball, because its code is stored on a one-time programmable chip. However, such cases are rare, and attempts to classify devices into this category are typically premature.

17 Study Firmware Updates

When you have a device that doesn't seem hackable, check whether it accepts firmware updates. Accepting firmware updates essentially means that it has the ability to modify its inner workings. In other words, this is a vector for hacking.

The XBC, CBC, and AR.Drone all have firmware update processes which are easily hijackable to install modifications. Sometimes the firmware update process is locked down, as in the case of the iRobot Create, but it's still worth investigation.

Be warned that modified firmware updates are also a great way to brick a device. Diligent research on the recoverability of a bad firmware is a prerequisite for this attempt. Such research revealed that firmware updates were the best way to hack the XBC, while other methods such as custom Linux programs were preferable for hacking the CBC and AR.Drone.

18 Recognize "Black Box" Systems

"Black Box" systems are systems whose internal operations are opaque. Interactive C, the Create Open Interface, and the AR.Drone UDP protocol are examples of black box systems. As a general rule, if you're communicating with or programming a device in something other than C, C++, or Assembly, there is almost certainly a lower level of programming that is possible. Whether this lower level is easily accessible (e.g. the XBC), very difficult (e.g. the Create), or somewhere in between (e.g. the Drone) is not always predictable, but is worth investigating.

19 Bricking is a Risk, but Can Be Mitigated

Here's a statistic which many people don't realize. I have never permanently bricked any device in Botball via hacking, nor has anyone at La Jolla or Nease. This is mainly because we take some precautions while hacking. If something looks important, but can't be backed up, we don't touch it, no matter how tempting. (The cat command in Linux is a good way to analyze what something is supposed to do without changing it. If you don't know how important something is, you may want to try making a backup in a different location, and only make changes to the backup. That way, if you crash something, on reboot the original will still be untouched.) We learn as much as possible before changing things. In particular, anything that overwrites flash sectors or system files, or that affects the boot process, is something we avoid unless we are very, very confident that we know what we're doing. Yes, bricking is possible, but it can be avoided most of the time.

20 Don't Reboot in the Middle of a Hack

If you're in the process of changing a system file, don't reboot the machine unless you're sure it's in a bootable state. Use a remote terminal program such as SSH or Telnet to make changes and test them without a reboot. For example, I replaced the CBC GUI in a recent hack. Unfortunately, my new version had a bug, which made it not boot. If I had rebooted the CBC after installing it, I would have bricked the CBC. Instead, I ran the new version from my SSH session, and noticed that it didn't boot. I then tinkered over SSH until it did boot, before I rebooted the CBC (closing the SSH session in the process). Only reboot when you're certain that the boot process will go far enough to at least let you continue working.

21 Know What an Operation Is Expected to Do

I'm going to tell a little story about a nameless Botballer whom I'll call John Doe. One day, a CBC started acting up. Several experienced Botballers (including a hacker) were in the room, and diagnosed it as a hardware failure. One of the experienced Botballers called KIPR, and was told to send the CBC to KIPR for repair under the warranty. Before the CBC could be packaged up for transport to KIPR, John decided to try to be a hacker and repair it himself. Without telling anyone what he was doing, he went into the Chumby boot menu, and told it to reset to factory settings, thinking that "factory settings" meant "the way KIPR gave it to Botballers." Except that it didn't mean that. The "factory settings" were Chumby Industries' factory settings, not KIPR's. John wiped all of KIPR's modifications and turned the CBC into a Chumby.

Luckily for the team involved, KIPR still honored the warranty, because it was obvious that the hardware problem wasn't caused by John. But this hammers home a point. John hadn't noticed that there was another option for restoring from a USB drive. He had ignored this setting because he didn't have a USB drive. He hadn't read Hacking the CBC Botball Controller [2], which explained that the USB drive was internal. Nor had he read KIPR's instructions stating that restoration should be done from USB instead of factory settings. He had never heard of a Chumby, nor had he ever seen the Chumby "sexopus" logo, so he was totally unaware what was going on when the CBC turned into an animated alarm clock. And he didn't notice that he couldn't access the CBC GUI anymore (he figured that rebooting would make the Chumby clock go away, but didn't try), so he didn't realize that he had bricked the CBC.

What would have prevented this? John should have told the more experienced Botballers what he was going to try. John shouldn't have restored from factory settings without knowing what that was expected to do. John should have read the existing documentation from KIPR and from the hacking scene before trying to be a hacker. And he should have panicked and told someone when the CBC turned into a Chumby instead of turning it off and hoping no one would notice.

This is the difference between successfully hacking a CBC and bricking it.

22 Don't Aggressively Market Your Hacks to End Users

When I listed above some reasons for hacking, one reason I intentionally didn't list was having your work used by loads of other teams. There's a simple reason why. The average hack, even if perfectly user-friendly and safe for the team who developed it, is not user-friendly and safe for many other teams. Think of how long it took you to show a teammate how to use something complex which you developed. Think of what you forgot to explain the first time. Think of what might happen if someone completely unfamiliar tried to use it. If you're in high school, think of a 6th grader trying to use your work. If the work in question is a hack, any of these points could cause serious problems for the team using it (ranging from a brick, to a crash, to

strange error messages, to a lost tournament round).

Hacks which I've developed, and which I thought were stable, have caused crashes and strange behavior for my teammates. I was present to deal with them as they came up, and as a result we never lost tournament rounds due to my hacks. If I had given the same hacks to other teams, with the best of intentions, those teams would have been in trouble on tournament day. Some teams can handle working with buggy third-party code and hacks, and are capable of figuring out how to deal with problems as they come up. Many teams are not. There's a reason why several of my hacking talks have begun with a video of a microwave exploding, and that reason isn't purely comedic.

Present your hacks at GCER. Post them on the Botball Community [14]. Help other teams who are interested. But don't aggressively market your hacks to teams who aren't prepared to deal with the consequences. Doing so is highly unethical. Most Botball hackers (I won't name any rare exceptions) are ethical, and are more interested in the best interests of other teams than in making a name for themselves by having their hacks used as widely as possible. Be like most Botball hackers; don't be unethical.

23 Espionage Guidelines Apply

Joe McCormick published an excellent paper on espionage in Botball [15]. All of these guidelines apply to hacking. Cyber-stalking of KIPR, KIPR's contractors (e.g. Charmed Labs and Chumby Industries), and other hackers is a great way to obtain knowledge. (I use the term "cyber-stalking" to refer to clever searching for usernames, etc., as Joe's paper describes. Obviously, violation of privacy or computer intrusion should not be used unless you enjoy the prospect of the police showing up at your door.) Press releases and papers published by KIPR, KIPR employees, or KIPR contractors are often full of useful information not found in Botball documentation [16].

24 Teachers Should Encourage Hacking

Hacking is a thought process based on applying the scientific method to computers. It's how students learn about the computer hardware and software around them. It often directly correlates with increased tournament performance and better career options. Yet, some Botball teachers ban their students from hacking, either because of the bricking risk, or because the teachers don't understand hacking themselves. This is counter-productive to the educational nature of Botball. Teachers should encourage their students to learn as much as possible in Botball, and hacking is one method of doing so. If there is a serious budgetary problem regarding paying for bricked equipment, consider letting students hack equipment from the previous year which is not being used in current tournament play.

25 Attend or Watch Conferences

GCER is more than a tournament; it's a full conference. As such, attending GCER is a tremendous help for hackers who want to know what the other hackers have been up to. The Norman/Nease hacking alliance was formed because Matthew Thompson saw my XBC hacking GCER talk the previous year.

But GCER isn't the only conference to keep an eye on. Non-robotics-related hacking conferences such as the Chaos Communication Congress [17] are an excellent way to learn new things. One of the hacking methods I used with the AR.Drone was directly inspired by a talk I had seen from the CCC. You don't have to understand everything in the talks to gain from them; many of the CCC talks are way over my head. And the talks don't have to be robotics-related; the AR.Drone hack I mentioned was inspired by a talk about password security (a subject which had nothing to do with my AR.Drone work). The Chaos Communication Congress is streamed live (and archived) every year in late December, and I highly recommend watching their talks.

26 Don't Violate Copyrights

Posting modified firmware images can violate copyright rules regarding derivative works and distribution. If the owner of the original firmware doesn't want modified images floating around, just post the instructions to modify an original to the state of your modified image. This way, you're not distributing derivative works, and you won't find yourself receiving copyright takedown notices.

27 Conclusion

Hopefully this paper has given some useful tips for would-be Botball hackers. I can be found on the Botball Community [14], and I'm always interested in hearing about new hacking projects. I'm looking forward to seeing *Hacking the CBCv3*. Happy Hacking!

28 References

[2] Jeremy Rand, Matt Thompson, Braden McDorman. Hacking the CBC Botball Controller: Because It Wouldn't Be a Botball Controller if It Couldn't Be Hacked (Parts 1 and 2). Proceedings of the 2009 Global Conference on Educational Robotics. July 2009.
[14] Botball Youth Advisory Council. Botball Community. http://community.botball.org , June 2012.
[15] Joe McCormick. Espionage in Botball: Motivation and Methodology. Proceedings of the 2009 Global Conference on Educational Robotics, July 2009.
[16] Richard LeGrand, Kyle Machulis, David P. Miller, Randy Sargent, Anne Wright. The XBC: a Modern Low-Cost Mobile Robot Controller. http://dpm.kipr.org/papers/xbc-iros05.pdf , June 2005.

[17] Wikipedia Contributors. Chaos Communication Congress. Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/Chaos_Communication_Congress . January 2012.