dan farmer/ zen@trouble.org/June-4th-2014/v-1.0

Sold Down the River

Summary

A world-wide scan of the Intelligent Platform Management Interface (IPMI) protocol identified over 230,000 Baseboard Management Controllers (BMCs) exposed to the internet, of which upwards of 90% could be compromised by just a handful of basic configuration and protocol weaknesses[i]. The real exposure is even greater, as access to a BMC allows an attacker to compromise its host server as well as other BMCs within its management group, since they share common passwords.

For over a decade major server manufacturers have harmed their customers by shipping servers that are vulnerable by default, with a management protocol that is insecure by design, and with little to no documentation about how to make things better. These vendors have not only gone out of their way to make their offerings difficult to understand or audit but also neglected to supply any substantial defense tools or helpful security controls. As an industry they've told us that we should trust them explicitly and implicitly with the security of our servers. It hasn't worked.

Customer demand for IPMI-enabled servers continues to grow as computing resources migrate from corporate server rooms to large offsite or cloud-based datacenters. Large-scale rollouts and provisioning rely heavily IPMI for the management and deployment for the racks upon racks of servers. All of these assets are at risk due to weaknesses in the protocol itself and poor implementations on behalf of BMC manufacturers.

Many of these problems would have been easy to fix if the IPMI protocol had undergone a serious security review or if the developers of modern BMCs had spent a little more effort in hardening their products and giving their customers the tools to secure their servers. At this point, it is far too late to effect meaningful change. The sheer number of servers that include a vulnerable BMC will guarantee that IPMI vulnerabilities and insecure configurations will continue to be a problem for years to come.

The findings in this document are based on a recent data gathering collaboration between HD Moore of Rapid7 and myself during the month of May 2014. I'll continue to stand by the words in my previous treatise on IPMI and BMC security[ii], but that was more theoretical and sprawling, while this document is more of an hors d'oeuvre focused on the current state of IPMI exposure and what can be measured today with a modest amount of effort.

Background *(Feel free to skip this section if you've seen previous work and know what IPMI, BMC, OOB, etc. are)*

IPMI is an acronym for a protocol: the Intelligent Platform Management Interface. Initially developed by Intel, Dell, HP, and other large vendor manufacturers, it was designed to help manage what is sometimes known as Out-of-Band (OOB) or Lights-Out communication. It's an OS-agnostic and pervasive protocol that is implemented by a BMC (Baseboard Management Controller), which is usually implemented as an embedded Linux system on the motherboard of modern servers. Pure IPMI is usually implemented as a network service that runs on UDP port 623 and can either piggyback on the server's network port or may use a dedicated Ethernet port.

Vendors take IPMI as a base, add on a variety of services like mail, SNMP, and Web GUIs, and then rebrand the new package – Dell has iDRAC, Hewlett Packard iLO, IBM IMM2, etc. It's also used as the engine for higher-level protocols such as those put out by

the DMTF (WBEM, CIM, etc.) the OpenStack Foundation, and others. It's particular popular for large scale provisioning and rollouts, supercomputer node management, and remote troubleshooting, console access, and the like.

The parasitic BMC has near-complete control and oversight on of the server it rides upon, including its memory, networking, and storage media[iii], and cannot be truly turned off; instead it runs continuously unless the power cord is completely pulled – an owner may only temporarily disable outside interaction unless you take a hammer to the motherboard.

Last year I worked with HD to scan the Internet in order to find servers that would respond to a packet sent to their Out of Bound (OOB) management port[iv]. The results were a bit dire, and some were summarized in my paper about IPMI and BMC security[v]. HD also wrote up a fine piece discussing various aspects of how to test for issues, and along the way various vulnerabilities were discovered and a flurry of vendor and advisories came out[vi].

Numbers

I'm going to focus on a few serious configuration and protocol issues that are nearly universal BMCs living on the Internet. I ignored numerous bugs and security vulnerabilities, which not only abound but instead make the situation even worse. Guidelines on how to address or manage some of the problems may be found in the Fixit section of this paper.

As mentioned, the Internet-wide[vii] sweep of UDP port 623 received over 230,000 responses. New UDP network scanning techniques and changes in network technology have accelerated such scans by many orders of magnitude since the days of the first UDP scanner two decades ago[viii]; today a complete internet search of a UDP port may be done in a couple of hours or less, and it won't be getting any slower. I also did some follow-up scans to verify and clarify some of the results.

While only a quarter of a million BMCs is only a tiny sliver of the total computing power in the world, it's still important indicator as a kind of canary in the coalmine. All the services that make the Internet so vital to our daily lives are powered by servers hiding behind the curtains of corporate firewalls; I would point to past surveys, studies, and personal observation that they too share the same sorts of problems found in this across-the-board survey. While management systems are often not directly assailable from the outside they're often left open once the outer thin hard candy shell of an organization is breached. Also OOB management networks are frequently not only kept separate from other internal networks but also left unmonitored for intrusions or spurious activity[ix], so security incidents can be especially troubling. Perhaps most importantly, however, is that when a run-of-the-mill server is compromised it exposes its own BMC to attack from its host, which could risk the sanctity of entire management network. And if the BMCs that I wasn't able to measure are anything like the ones I was, that's real trouble.

IPMI's BMC population is divided almost evenly into two parts – those running the older 1.5 protocol (1.5 is when networking was added, so no older versions should be on the 'Net) and those speaking the 2.0 version (which appeared in 2006 and is also backwards compatible with version 1.5.) Unless the servers are not answering truthfully to protocol requests, it was a bit surprising how popular 1.5 still is:

IPMI Version 1.5	# Found	% of total	IPMI Version 2.0	# Found	% of total
	109,726	**46.8**		124,413	**53.2**

BMCs running 1.5 only had a single simple problem, but it's a whopper - nearly all server management ports had the NULL authentication option set, meaning that all accounts could be logged into without authentication. Furthermore virtually all BMCs also had the NULL user enabled, by itself a problem but not a serious one, but working in tandem with the first it means that you can login to pretty much any older IPMI system without an account or a password. For example typing this command will print out a unique ID for a remote BMC:

```
$ ipmitool -A NONE -H 10.0.0.1 bmc guid
System GUID  : 0506d25f-c508-d711-9c9d-0004001e6c3c
Timestamp    : 09/02/1973 18:52:31
```

The privileges associated with the NULL account vary from vendor to vendor, but it seems to usually grant administrator access. No matter what, however, remote execution of commands on your server is bad.

About 90% of the 1.5 systems had the NULL security flaw:

IPMI Version 1.5 Major Problems	Number Found	Percentage of BMCs found vulnerable
NULL authentication	98,927	**90.1**
Null user enabled	109,050	99.4
Anonymous	1,806	**1.6**
Total percent vulnerable	98,949	**90.2**

The anonymous problem is almost synonymous with the NULL authentication & user problem and simply tosses a bit more fuel to the fire.

Before you start thinking a simple fix will kill off this problem and make 1.5 secure, keep in mind that besides optional password hashing the IPMI version 1.5 has no cryptographic protection at all between you and the BMC; the specification defines the RMCP (Remote Management Control Protocol) as a simple UDP datagram driven protocol. As a result it's vulnerable to a plethora of network security attacks such as password sniffing, network spoofing, connection hijacking[x], Man-in-the-Middle attacks, and more. Also when you set or change a user's password the actual password is sent over the network in clear text. You might think of the security of version 1.5 as something akin to using the old, reviled, unencrypted, and easily subverted `telnet` command for remote logins.

IPMI version 2.0 rolled out cryptographic protocols that were supposed to give additional security, but it also added a few serious security issues that cause yet more vulnerabilities. The numbers below aren't quite as precise as above, since the confirmation of some issues requires imprudently invasive strategies. The total number is therefore an estimate, which I'll discuss below in greater detail.

In the BMCs supporting IPMI version 2.0 the NULL authentication default of 1.5 dropped by almost 50%; at least some vendors recognized it as a security problem:

IPMI Version 2.0 Major Problems	Number Found	Percentage of BMCs found vulnerable
NULL authentication	19,013	**15.3**
Null user enabled	55,414	44.5
Anonymous	4,773	**3.8**
Cipher Zero (See below for details)	80,712 72,641	Total found **~64.8** Est. Vulnerable **~60.2**
RAKP passwords (See below for details)	103,393 38,558 92,020	% Recovered **~83.1** IPs cracked **29.8** Upper Limit **~74**
ESTIMATED Total percent vulnerable	90,635 111403	Lower bound **72.8** Higher bound **92.5** (See below for details; higher % includes breaking passwords of 12 characters or less on known user names)

The first two lines are the same as their counterparts in the previous table. The multiple results for the Cipher zero and RAKP lines will require a bit of background and explanation, but using the bottom-right percentage in each as a result is not crazy, although it might be a bit pessimistic depending on your situation. The total adds up all the problems found for a given unique BMC and represents the number of systems with one or more problems.

IPMI defines 16 different cipher groups for protection. Unfortunately the very first one, the infamous cipher zero, is essentially the un-protocol, with no checks for authentication, integrity, or confidentiality – a step back from even the minimal security offered in 1.5. To authenticate you need a valid user name but **any** password may be used; it is simply ignored. The majority of servers have cipher zero enabled on their BMC by default, and HP, who is one of the largest if not largest vendor of BMCs, had apparently never allowed you to turn it off until just recently. Examples and troubleshooting tips advising the use of cipher zero as a solution to authentication issues abound on the Internet.

A Get Channel Cipher Suites Command[xi] sweep discovered that 63% of BMCs running IPMI 2 had cipher zero enabled, but since you do need a valid user name the number of vulnerable systems by a casual attacker is a bit less. Given that the RAKP and cipher scanning data suggest that about 10% of BMCs have their default accounts disabled, so taking 90% of the observed value seems a reasonable estimate; last year I tested over 35,000 systems and had nearly identical results. Finally another 2334 IPs had the NULL user set, so guessing the user was unnecessary. This brought the total to 60.2 percent (124,413 * 64.8 % * 90% + 2334) who were vulnerable to Cipher 0.

The account you use to login with cipher zero well vary in privilege level, but at least from what I've seen it generally it seems plausible that you'd be able to login with administrator access. As said before, however, no matter what, remote execution of commands on your BMC is serious.

A second and perhaps even more serious security problem introduced in version 2.0 was RAKP (aka the RMCP+ Authenticated Key-Exchange Protocol), which IPMI uses to negotiate a secure connection. RAKP allows an anonymous user the ability to remotely get the password hash from the BMC, and it **cannot** be prevented if the attacker knows the account name. This is an astonishingly bad design, because it allows an attacker to grab your password's hash and do offline password cracking with as many resources as desired to throw at the problem[xii]. Unfortunately this means that even if you're on the ball and up-to-date with your patching, have all known security problems fixed, and everything is working as planned, if an attacker can guess a valid account name they may get its password hash and crack your password without you knowing it[xiii].

To further investigate the RAKP problem I did a follow up scan on IPMI 2.0 BMCs with Rapid 7's Metasploit[xiv] and was able to gather password hashes from over 83% of the target BMCs[xv]. I then used the popular John The Ripper password-cracking engine[xvi] along with a modestly sized dictionary (4.7 million words) to test the strength of the passwords recovered. While running in the background inside VMware for less than a day Jack was able to guess passwords from about 30% of all BMCs running 2.0; The top 7 passwords guessed were all vendor default passwords, with "admin" being the most popular.

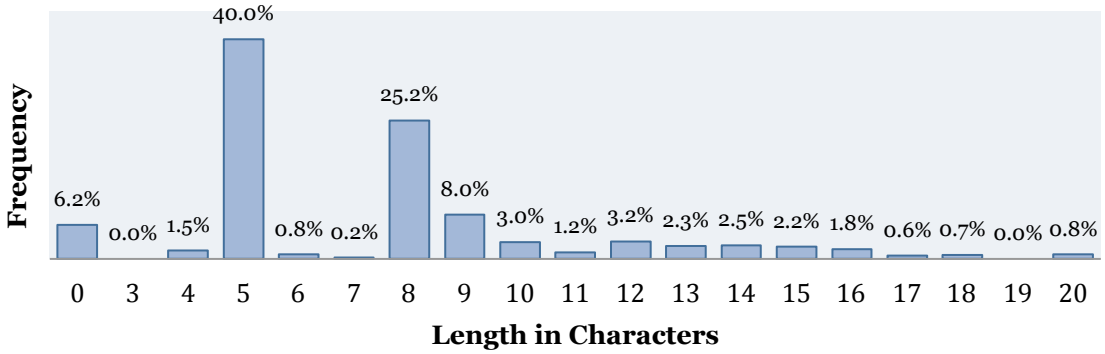
Of course numerous past studies have shown the effectiveness of what a serious attacker can do, and with orders of magnitudes faster speeds than I could muster on my consumer grade iMac. I'd say that even a well-chosen non-dictionary based password of a dozen characters or less is suspect.

Note that passwords are stored in plaintext on the BMC (or in recoverable form by the BMC, which in my experienced may be easily captured by examining the BMC's memory), so that the best password in the world won't do you much good if the BMC is compromised, or your de-provisioned server shows up on EBay without the BMC's media incinerated[xvii].

Last year it was revealed that SuperMicro BMCs were vulnerable to having their BMC password file snatched, which, as said in the previous paragraph, has the IPMI password in clear text[xviii]. Using nearly 24,000 passwords from this – accessed from less than 10% of the total number of BMCs found **over one year ago** – the total number of passwords recovered from the current data set increased by over 50%, going from over 38,000 to 56,700 and testifying to the longevity of passwords on BMCs.

Looking at the SuperMicro password data just under 11% of IPMI passwords were over 12 characters in length - if this is true across the board and it is indeed possible to crack a password of a dozen of characters or less, then the RAKP problem alone would close to 90% of all IPMI 2.0 BMCs to be compromised. Since I don't know the usernames for all of the target machines, however, the theoretical maximum cracking of 12 characters or less covered 74% of the BMCs (e.g. 90% of the 83.1% percent of BMCs with recovered hashes.)

Password Length Distribution(from SM data)



The Final Theoretical Number Of Vulnerable 2.0 BMCs

Obviously some BMCs will have multiple problems and others none, so you can't just add up all the percentages. Instead I made a sort of spreadsheet, added up all the individual vulnerabilities, and got the final result.

When calculating the final theoretical coverage of RAKP's effect on the total vulnerable BMCs you have to look at how many IP addresses weren't covered by any other vulnerability (e.g. NULL, anon, etc.) but we were able to recover a password hash. This number was 25,431. Using the same assumptions as above, we'd get (25,431/103,393) * 90% chance of guessing the right user name * 90% the password is 12 or less characters = an additional 19.7% otherwise not-exploited BMCs would be exploited by a determined password attacker.

Grand Conclusion

I suppose the results might not be surprising to some, but as an eternal optimist seeing something close to 90% of the BMCs out there was a bit depressing to me. I suppose we might all take solace in the fact that everything is pretty damn insecure, but things keep humming along, but I personally think it'd be nice to think that our vendors aren't knowingly driving knives in our backs or otherwise sabotage our possibly futile efforts to secure our systems as we sign checks to them.

One interesting result of the testing is that it revealed an unusual password[xix] that was shared among over 11,500 BMCs. Given the widespread nature of the distribution of the BMCs sharing the password (over one hundred and fifty one different class A CIDR blocks, or roughly 60% of the Internet's usable addressing space) and the homogenous nature of the vendor (SuperMicro coming up again) I suspect an undocumented default password on the NULL account, but I don't really know (and if anyone from SuperMicro is listening to this, drop me a line!) It's interesting to be able to spot this – presumably there are more to discover out there.

Another data point was a shared password between some 1300+ BMCs over what appeared to be sites covering a fair swath of Europe in a half-dozen networks or so. It could be coincidence, but it also might be that a global managed service provider uses a common password to manage geographically diverse systems, not a practice I'd personally advocate. Presumably this password was also used on BMCs inside the organizations (or, I suppose, perhaps someone is breaking into a whole lot of BMCs and

leaving a backdoor account?) It certainly would be a cruel blow to have your company compromised because another organization's password was nabbed, let alone your outsourced IT was lazy enough to share the same password across their customer base.

I've put a small section below on how to address some of the problems found as well as notes on the complex IPMI security model and why the results may well be worse than I found here. This document represents the situation to the best of my ability to sketch it out; if I get corrections or additional information I'll try to keep it updated.

I also keep a FAQ, resources, a bibliography, and what I consider to be best practices along with a longer paper and other miscellanea on OOB security at:

http://fish2.com/ipmi

If you have any additional links or material to add drop me a line. I will try to answer any questions that I can.

It seems to me that with both IPMI and the BMC in particular there seems a wealth of additional security material to look at, so prospective students of the craft take note. I have to find a job before I do more work on the topic; I've taken off too much time already on this stuff.

While any mistakes here are my own, I'd especially like to thank Albert Chu of the Lawrence Livermore National Laboratory and the FreeIPMI maintainer, HD Moore of Rapid 7, and Jarrod Johnson from IBM (who has told me that IBM has added or removed several insecurities on their own IPMI/BMC offerings, although I haven't seen it first hand) for their invaluable comments and feedback on this document. And without HD's data it wouldn't have happened at all, so dude, thanks again! Finally, much love thanks to Dona, my long-suffering wife, who would probably pay a lot of money not to hear the acronym IPMI again.

In any case, best of luck from Seattle, with IPMI & BMCs and all your fine endeavors -

Dan Farmer
zen@trouble.org
6/4/2014

Fixit: How To Know If You Are Vulnerable, Addressing Problems, Etc.

By far the best defense is to keep people as far from possible from your managed networks and systems as possible, but I realize this isn't really practical in some (if not many) cases. Remember that a compromised server can have access and compromise its BMC (and vice-versa)! Never, ever, ever put a managed network port on the Internet unless you really, really, really have to.

Also, if you are using managed services, hired guns for administration, or some other outsourced folks or automation that touches managed networks, you might consider a policy – or even putting it into a contract – that they are forbidden them to use the same passwords or access to your systems and networks that they use for other customers.

The best single technical thing to do, especially for BMCs supporting 2.0, is to disable or remove the default vendor user names and pick something that an attacker wouldn't immediately know; this disables several of the problems I outlined here (the NULL account may not be removed, at least according to the specification.) Unfortunately doing this en masse can be quite a challenge, but don't shoot the messenger, I didn't

design the specification. Also unfortunately – if word gets out of your account names, then attackers may try the same tactics I outlined above, so it's almost as bad as a password leak.

If at all possible try to rotate your IPMI passwords on not-too-infrequent basis; while this varies wildly on a site, organization, or technology basis, I'd say a password lifespan of a year is stretching your luck.

Another relatively easy defense is to simply increase your password lengths. Since IPMI passwords are so long-lived (with a lifespan often measured in years) and shared among so many other servers, attackers have a fair bit of incentive to spend a large amount of CPU and GPU time cracking a single password. I'd advise using either passphrases of 16 characters or more (IPMI 2.0 supports up to 20 characters) or strong two-factor authentication if possible (this often isn't possible in a production server environment because of the issues with automation, among other things.) Although implementations are non-standard (because it's not in the specification) and network authentication has its own issues, it may well be worth considering for this point alone.

Keep in mind that anyone who has administrative privileges on a BMC's server has administrative control over it and may disable or enable IPMI (along with the vendor added-on services such as the web GUI), add or remove accounts, change the IP address, etc., etc. – all without any authentication to the BMC. It cannot, however, be used to read passwords on the BMC, make a backup of its storage media, or any other low level stuff I so direly warn about at times. One needs to login to the BMC's operating system (some vendors, such as Sun/Oracle, SuperMicro, etc. allow this at least with some models) compromise the security of the BMC for that (which is at times not very difficult!)

Testing note

No servers were harmed while running these tests. Except where explicitly noted all of these may be tested for anonymously and non-invasively with a fairly high degree of accuracy with perhaps the exception of Cipher 0, which may be surmised by other data, but only validated by executing a non-anonymous command on the target BMC.

Technical Addendum: Notes on the IPMI Protocol Security Model

This gets even farther into the security aspects of IPMI, and should probably be best avoided but all but the fanatical or especially curious; it also discusses why the problems found are perhaps a lower-bound rather than upper one on the overall problem.

While I believe the numbers in this paper are correct to the best of my ability to validate, IPMI authentication, cryptography, and authorization mechanisms are among the most complex and unusual I've ever seen, and make the verification and testing of it (let alone understanding it!) very difficult.

There seems to be very little documentation, books, or articles written on large-scale IPMI management outside academic circles, which essentially leaves us with IPMI's 650 or-so-page specification essentially as the blueprint to manage and secure the BMC (along with its 484 page older sister that covers version 1.5), which is both a shame and a worry[xx].

It seems rather intuitive that complex, arcane, or ill-understood systems essentially guarantee problems that are essentially impossible to discover without the aid of specialized software or management tools, but the security and management tools I've seen only scratch the surface as well.

In any case I'm almost certain that I really don't understand all the implications of the specification, but here's a brief summary of how I think it works.

User IDs are numerically based and have names (e.g. "ADMIN", "root", etc.) associated with them, with user ID 1/one permanently associated with the NULL user name. Duplicate names are allowed, and while some commands use IDs and others usernames, if there are ambiguities the system will grant or deny access or authorization based on the lowest matching numeric ID username that is matched by the username.

Authentication is done via a password of up to 16 or 20 characters, but may be bypassed or controlled on a per user and channel basis. Most vendors have also added support for LDAP, Active Directory, or Radius network authentication. Users may also be disabled regardless of their authentication settings.

IPMI also allows multiple channels of communication that may be used in different ways over different interfaces or transport protocols, such as the LAN, internal buses, serial lines, VLANs, etc. (version 1.5 only had 9 channels, while version 2.0 has 14.) Each channel is completely independent of the others and may operate in the same or different mediums.

In my testing I only examined the default channels, which are actually a sliver of the overall potential of what IPMI can do, so there may well be additional undetected problems, both similar and unknown, lurking out there.

While LAN and serial channels share many characteristics with the basic channel settings, serial users have additional options and limitations with respect to access, authentication, and session management.

Channels have an access mode associated with them, granting access based on the state of the server. These modes are configurable and include pre-boot only, always available, shared, or disabled.

There are also 5 privilege levels that are associated with commands and users: callback, operator, user, administrator, and an OEM/vendor chosen one. Each user may be granted a maximum privilege level, and all commands have a minimum privilege level that must be met in order to be executed.

There are also a set of commands to manage and limit access of other commands (I count over 160 commands in the specification, plus OEMs and vendors are free to add to the set) in the terribly and misleadingly named firmware firewall, which allows individual commands to be limited on a per channel and per user basis. Commands may also be bridged or routed to other interfaces and media.

IPMI calls the data in its protocol payloads, which in version 2 were greatly expanded; they may be used to transmit both IPMI and non-IPMI commands and data. Payloads may use their own set of port numbers, and transports. Non-IPMI data is perhaps most commonly used for Serial Over LAN (SOL), but vendors may add just about anything here.

Channels also have support for different algorithms for authentication as well as data confidentiality and integrity; this also is set on individual channels and may be set for individual sessions, command or payloads.

In addition the vendors have the capacity to expand the protocol to do whatever they want. Part of the problem with analyzing IPMI security is that no one outside the various vendors knows what its actual capabilities are.

In any case what this all creates is a rather sizeable multidimensional matrix of possibilities. When I first saw all this I initially thought that no one would use all these options, but unfortunately some in fact do, and I've seen different settings, configurations, and restrictions for users, privileges, and commands on discrete channels.

This sets up an unfortunate situation where you might **think** you have disabled some undesirable setting (say, cipher zero to disallow unauthenticated access), but you might not be looking at all the users or channels. Or perhaps you only disabled it for the wrong privilege level on the right channel. Or… pick your confusion.

I know of no software that manages or reports on all this, but to my eyes it vastly too complex to understand any reasonably sized implementation that span multiple servers. Detecting someone setting up unauthorized or backdoor access by simply using stock IPMI commands would be a sizeable challenge.

I think to understand a BMC basic security one would need to (at least):

- Enumerate through all the channels to examine all the various privilege levels assigned to commands and payloads
- Look at the cipher support for each channel and traffic
- Enumerate all the privilege constraints for each channel
- Enumerate all users, payloads, and commands on all channels and map their capabilities as granted and constrained by the various constraints and rights granted by the firmware firewall and other commands

Perhaps an N-dimensional spreadsheet would help?

Mind you, doing all of this on a BMC might well crash or wedge it into a sullen silence, as they are very easy to DoS into submission even unwittingly (I've completely broken BMCs from my testing both Dell and HP servers.)

But this seems simply ridiculous – am I smoking crack here or were the specification authors? There may be an easier way, but I don't know of it nor do the very knowledgeable people I've asked. Please let me know if you have any insight into the problem.

---

[i] I'll be the first to admit that concrete and provable numbers are hard to come by, especially when claiming things about the entire Internet! But for more on the numbers see the results section.
[ii] Summary of IPMI/BMC security: it sucks. It's atrocious. It's … well, you get the idea. I've a small site with bits and pieces of references and material along with my paper "IPMI: freight train to hell" on general IPMI and BMC security.
[iii] IPMI is designed to be able to restart stuck systems and boot from arbitrary media from remote sources, making it easy to browse or modify the server's drives; memory may be captured in real time with some inside knowledge or captured with something like this nifty hack that disables ECC memory clearing.
[iv] The packet was an IPMI Get Channel Authentication Capabilities packet sent to UDP port 623; the IPMI specification says that it must be answered if received.
[v] My own work and a set of references to other efforts may be found at http://fish2.com/ipmi.
[vi] In my opinion the advisories were fairly uneven and not terribly accurate, but no one asked me for my opinion despite being cited and bringing the problems to light in the first place.
[vii] HD will take people off his scanning list upon demand, and a few networks have taken him up on it, but the opt-out number is substantially less than 1% of the total address space.
[viii] SATAN came out in '95 and had the first UDP scanner I'm aware of.

[ix] For cost reasons the management backplane often runs a variety of services and systems beyond simply the BMC, but, but backed by my admittedly anecdotal evidence, often shuts out security and monitoring teams again due to cost, but also because they don't want their mission critical machines mucked with. In general as few folks and tools are allowed back there as possible.
[x] IPMI creates sessions tunneled over the UDP protocol, which is a connectionless protocol.
[xi] This is defined in section 22.15 of the IPMI 2.0 specification, and a BMC may be queried anonymously for its supported cipher suites; I wrote a small tool to query remote systems.
[xii] Password cracking isn't done by comparing password guesses to the password in question, but instead by comparing hashes against hashes to see if they match. Wikipedia has a nifty little article on the topic.
[xiii] http://fish2.com/ipmi/remote-pw-cracking.html has more details on how this works, as well as a program that may be used to demonstrate the remote hash-grabbing feature.
[xiv] HD Moore wrote a module for this and a few other IPMI checks last year and are freely available.
[xv] There was about a 30-45 day lag between the two scans, so one might expect some percent to move, be turned off, etc. The others missed were presumably due to savvy administrators disabling the default accounts.
[xvi] Both the very fast and popular John and HashCat now have native support for the IPMI hashing algorithm.
[xvii] There's been lots of activity done on recovery of data on flash (what all BMC's I've seen use) and other media due to its applications in forensics and data recovery; all results indicate a high probability of success. I've personally performed a few ad hoc tests on deleted data and passwords on BMC storage media and it proved an extremely simple task.
[xviii] HD Moore of Rapid 7 wrote about this in great detail last year, although oddly I don't see an CERT advisory or anything about it.
[xix] I'm not saying what the password is to protect the guilty, but it was 8 characters that seemed likely to be unique, and searching for it on Google revealed no useful results.
[xx] I've personally found the Free IPMI and IPMI source code invaluable at spots where I simply couldn't decipher the specification's meaning or intent, but I wouldn't want to learn how to use, manage, or secure IPMI from their code!