spec.txt

1 Introduction:

This document describes the details of Intel AMT with respect to the Solaris operating system. There were lots questions asked by our initial attempt from the PSARC 2007/601 review of this case. Based on the interested questions, we decided to break down AMT technology into the following numbered sections so that the project team can address each associated AMT technology specifically. Please note, this document's primary audience is Sun Microsystems' PSARC members. We assume no explicit no knowledge of Intel AMT. In addition, we would like to keep all the reviewers in perspective of what this project team will be adding to Solaris as the result of this project at the end of the phase I:

1) a Solaris AMT device driver (3500 ~4000 lines of code)
2) a Solaris user-land daemon (2000 ~2500 lines of code)
3) So, as you can see, this is not a large body of code by any stretch. Solaris is one of the OS consumers of Intel's AMT. Much of this AMT technology is already reviewed, published, and deployed. As Solaris is AMT's consumer, it is not the objective of the project team to improve Intel's AMT at phase I of the project. Much of this document is derived from public publications on the web and our Sun's relationship with Intel engineers, and, of course, we have edited with Solaris in mind.

Index to PSARC luminaries:
Garrett D'Amore = (gd)
James Carlson = (jc)
Bill Sommerfeld = (bs)
Nicolas Williams = (nw)
Paul Jakma = (pj)
Gary Winiger (gw)
Randy Fishel (rf)

2 Terms

AMT – Intel Active Management Technology
Host OS – The operating system running on the machine equipped with AMT chip
ME – Management Engine (The AMT firmware)
LMS – Local manageability Service (A daemon running on the Host OS accepting local http request and pass them to/from the ME)
MC – Management Console (A remote computer managing the AMT enabled host)
System Defense Filters - A set of filters applied by ME to incoming and outgoing network packets combined with actions to take when intrusions are detected.
Agent Presence: There's a watchdog in the ME firmware. An **Agent** is any process that is willing to report its **presence** to the watchdog in ME. The idea is that if an Agent doesn't tickle the ME periodically, then the ME can take some assertive action. (Like sending an alert out over the network to a Management Console.)

3 High level description

3.1 Architecture

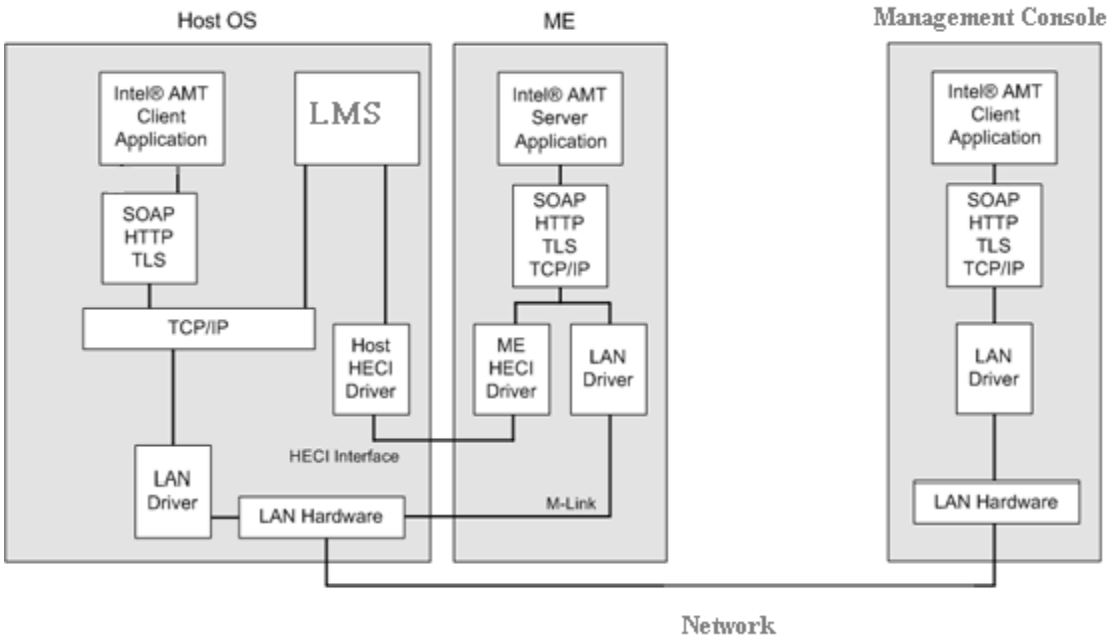

Figure 1 AMT architecture

Intel AMT's core hardware (micro-controller) is within the chipset's graphics and memory controller hub. The firmware (ME) is an embedded OS which implements various web services over XML/SOAP over HTTP(HTTPS).

As seen in Figure 1, the firmware (ME) has its **own TCP/IP stack, and its own driver** for the NIC. Remote management applications can access Intel AMT securely, **even when the platform is turned off**, as long as the platform is connected to line power and to a network.

3.2 LAN Out-of-Band Communication

Intel AMT provides for remote communication of ME firmware with a central management console via SOAP, regardless of power state and OS condition, as shown in Figure 2. This mechanism allows the ME firmware to share a common LAN MAC, hostname, and IP address with the OS, helping to minimize the IT infrastructure cost to support functionality based on Intel AMT.

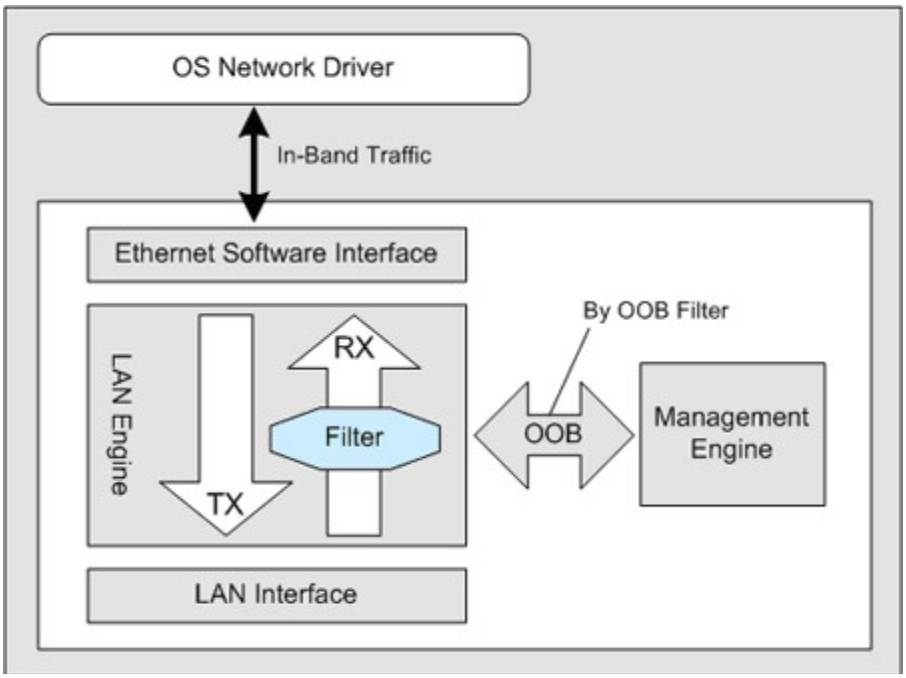

Figure 2. Intel AMT out-of-band communication

The out-of-band communications architecture supports the following filters:

- **ARP:** Forwards ARP packets containing a specific IP address to the host and/or the micro-controller
- **DHCP:** Forwards DHCP Offer and ACK packets to the host and/or the micro-controller
- **IP Port Filters (HTTP and Redirection):** Redirects incoming IP packets on a specific port to the micro-controller

3.3 What we propose in this case

This case proposes a HECI driver and a LMS daemon to be integrated in Solaris.

The Intel Management Engine Interface (Also known as HECI) driver is a software interface that is used to communicate to the Intel® AMT subsystem (Management Engine) to access Intel® AMT capabilities. Communication between the local host operating system (OS) and the Management Engine (ME) is accomplished by means of the Intel Management Engine Interface driver. The Intel Management Engine Interface is bi-directional, as either the host OS or Intel® AMT firmware can initiate transactions.

The Local Manageability Service (LMS) is a service that runs locally (in the user space) in the host operating system. LMS binds and reserves to TCP ports 16992/16993 on localhost with

INADDR_ANY and listens for HTTP(s) requests. Note, this is necessary to prevent host applications from accidentally trying to use the AMT ports. (If an application got unlucky, and bound to those ports, then it would fail to operate as AMT intercepts all traffic on those ports!) When an application sends SOAP/HTTP(s) message addressed to the localhost on 16992/16993, then LMS intercepts the request and routes the request to the Management Engine Interface via the Intel Management Engine Interface driver. In short, the LMS enables local management applications to send requests, and receive responses to the local Intel® AMT device.

The LMS is actually a proxy, which transfer TCP requests (open connection, close connection & TCP packets), between host management applications and the Local Management Engine in the ME, via the Intel Management Engine Interface host driver

Both the daemon (LMS) and the driver (HECI) are ported from Intel's code for Linux, under BSD/GPL dual license.

3.4 Local vs. remote interfaces accessibility.
The ME can be accessed in the same manner by both remote and local applications but the functions accessible by remote and local applications are not exactly the same:

Realm	Controls access to:	Network	Local
Administration	All of the Intel® AMT interfaces	X	
General Info	General Information Interface	X	X
Hardware Asset	Hardware Asset Interface	X	
Remote Control	Remote Control Interface	X	
Event Manager	Event Manager Interface	X	
Redirection	Redirection interfaces (SOL/IDER)	X	
Storage (3PDS)	ISV Storage Interface	X	X
Storage Admin	Storage Admin Interface	X	
Local Agent Presence	Local Agent Presence Interface		X
Remote Agent Presence	Remote Agent Presence Interface	X	
Circuit Breaker	Circuit Breaker Interface	X	

As seen above, most interfaces are only accessible from remote. The local interface accessible through HECI and LMS are those like 3rd Party Data Storage and Local Agent Presence (watchdog). **AMT does not provide administration interface on the Host OS.**

4 Administration / Configuration

Intel AMT devices arrive in un-configured state (also called “Factory Mode”), not available for use by management applications. **So it is off-by-default.** Setup and configuration is also referred to as “provisioning”. Intel AMT supports two provisioning models: Enterprise Mode and Small Business Mode. Either mode is done from outside Solaris. **Administration (Configuration) cannot be done from within the Host OS.**

Small Business Mode is set up from within the AMT Configuration screens from within the MEBX (BIOS Extensions.)

Enterprise Mode involves **remote** Management Console with secure communication using Transport-Level Security (TLS), and PID/PPS keys. Provisioning in Enterprise mode is always done via some Setup and Configuration application, such as the Intel SCS, the Intel AMT Commander, or by another vendor’s application that might have a provisioning server integrated in to it. Note that provisioning is not possible from a wireless interface. Part of the setup process involves adding credential data generated by Intel SCS to the BIOS of the client PC, which can be accomplished using either 'one-touch' or a 'zero-touch' configuration mode. One-touch configuration requires physical contact with the PC, whereas remote configuration does not.

One-touch mode requires an administrator to enter credential data to the client PC, either by saving data to a USB key and then using that key to boot the PC (note that the AMT Client must still have it’s default password for the ME) or by manually keying the information into the system's BIOS screens. In zero-touch mode, the PC's BIOS is populated with an encryption key at the time of manufacture that enables the PC to establish an unattended, secure connection with the Intel SCS server in order to obtain the credential data.

The choice between these two modes provides enhanced flexibility. Remote Configuration provides convenience that may help to cut deployment costs, especially in remote-site scenarios. Because one-touch configuration does not require the use of an encryption key that is known by a third party (the PC manufacturer), it may provide a somewhat higher level of security.

Small Business Mode which does not support TLS-based communication, is used when sufficient infrastructure is not available to support Enterprise Mode setup (which is recommended). In this mode, setup is carried out using browser access to a web server within the Intel AMT device.

5 Security

An IT administrator can configure Intel AMT for 1) certificate-based authentication, or 2) password-based authentication. Both the Intel AMT platform and the Setup and Configuration (S&C) Server start with two pieces of shared information – a platform ID and a pre-shared key (PSK). The first communication between Intel AMT and the setup and configuration server is an un-encrypted “hello” message from Intel AMT to the server that contains the platform identifier. The S&C Server then performs the setup and configuration process using the PSK and the TLS-PSK protocol for authentication and encryption of the configuration traffic. The S&C Server downloads Certificates to the Intel AMT platform, which stores them in non-volatile memory. The certificates trace to an enterprise certificate authority and are used by Intel AMT to authenticate to Management Console applications. If Intel AMT is configured for mutual authentication, the S&C Server must provide a client certificate for each application that will communicate with Intel AMT. The S&C server also establishes an Access Control List, enables certain Intel AMT features, and configures device settings. At the end of the setup and configuration process, the keys generated and used during the process are deleted. All

subsequent communications use the certificates and Transport Layer Security (TLS) for authentication, confidentiality (encryption), and integrity (mutual authentication). Intel AMT performs authorization using the Access Control List, as described in the following section. HTTP Digest authentication is used for the SOAP over HTTP communications. The Redirection feature uses secure sockets layer (SSL) to establish a secure connection between the remote console and the Intel AMT platform.

5.1 Q (gd): How does this fit within secure-by-default? Does the service listen only to IN_ADDR_ANY, or does it open up a port accessible to the entire network?

A: LMS only accepts connections from the local machine. Packets from/to the entire network are directly handled by the ME.

5.2 Q (jc): More generally: have you looked at the security questionnaires? How do you comply with them?
http://www.opensolaris.org/os/community/arc/policies/ITS/
http://www.opensolaris.org/os/community/arc/policies/NITS-policy/
http://www.opensolaris.org/os/community/arc/bestpractices/security-questions/

Q (gw): Given the lack of materials in a consumable fashion, I can't tell how this project complies with... policy.

A: We are OK, since LMS only accepts connections from the local machine. And the **administration interface is not provided locally**. See section 4. Remote administration is done directly between the MC and the firmware. The Host OS is not involved, so it's completely beyond our control.

5.3 Q (jc): It's unclear exactly what the daemon and the driver do. What can the driver do to the system? Can it modify system memory? Can it reboot the system? Can it turn off power? Can it force the system to reload a different kernel?

A: As seen in section 3.4 above, the LMS daemon and the HECI driver provide – as a proxy – access to the following features of the firmware: Reading general information, Agent Presence, and third party storage in the AMT EEPROM. It cannot modify system memory, cannot reboot the system, cannot turn off power, cannot reload a different kernel. **It does not provide any privileged operation to the consumers of the XML/SOAP interface**. The consumers are mostly developed by ISVs, not included in this case.

5.4 Q (bs): What's the actual access control and IPC mechanism here? you can't assume all local users are trusted, either.
A: HTTP/HTTPS. Authentication is done by the firmware.

5.5 Q (jc): Are the port numbers involved registered with IANA? What security is provided?

A: The ports are registered with IANA: 16992 and 16993. TLS-PSK or Kerberos authentication is required.

(dc): NICs don't filter data by themselves, there's no NIC related to 127.0.0.1, and I don't understand what "OOB" means in this context. Please explain.

A: OOB – out of band. It's an Intel onboard NIC. AMT currently doesn't work on with a plugged-in PCI NIC. The NIC monitors inbound traffic and filters those on port 16992/16993 to the ME.

5.6 Q (jc): What privileges are required to talk with the kernel driver? Where are the username and password stored.

A: It requires root privileges to talk to the driver. The daemon listens on port 16992/16993 for HTTP requests, accessible to anyone on the host. The driver/daemon just passes the HTTP requests down to the AMT firmware and passes the HTTP responses back up.

The username, password, and certificates are stored in the firmware.

5.7 Q (jc) It sounds like ordinary users could open that node and talk to it. In that case, I don't think the daemon should run with elevated privileges. There's no need. In fact, it should run with the minimum privileges possible.

A: The driver interface is undocumented and has been changing from AMT1.0 ~ AMT 3.x, while the XML/SOAP/HTTP interface is stable, backward compatible, and clearly documented in Intel's AMT SDK. So it'll be better to hide/restrict the driver interface and export only the SOAP/HTTP interface to the consumers.

There have been many applications developed by ISVs using the SOAP/HTTP interface, so maintaining the convention can greatly encourage them to port their applications to Solaris.

5.8 Q(jc) If the HECI interface is innocuous, as you seem to be saying, I _might_ recommend that it not require all privileges ("root") to open it. It'd be better if the LMS daemon could run with minimal privileges. In fact, "Least Privilege" design is a requirement on Solaris. For 5.7, it looks like you may have misunderstood the question. I'm not suggesting at all that the interfaces be documented or kept from changing such that users could reasonably design applications that talk directly to the HECI device node. Instead, I'm talking about ***privileges***. If it's possible to reduce the required amount of privilege for LMS, then I would like to see that path investigated.
A: We'll investigate if Least Privilege and RBAC can help.

6 Network

AMT Network Discovery:

Intel AMT listens on TCP ports 16992 for HTTP connections and 16993 for TLS connections. The favorite way to detect Intel AMT is to try a connection to port 16992 and see if it works. If the connection is immediately rejected, you can try port 16993 and see if the connection is accepted. If a connection on 16992 is accepted, the computer is setup to accept connections without TLS. While that connection is established, you may as well send a HTTP HEAD request like so:

"HEAD / HTTP/1.1\r\n\r\n"

Intel AMT will return a header and you can parse the "Server:" field to get the version of Intel AMT. If no version number is specified, it's because it's an Intel AMT 1.0 computer.

In both Intel AMT Commander and Intel AMT Director, I could use the built-in HTTP client objects (called WinHTTP) to do this, but it does not work quite right for Intel AMT 1.0 computers. Intel AMT 1.0 requires authentication right away, and you can't get the server header field correctly using that library. As a result, I use direct TCP connections using the "TCPClient" class in .NET.

If you get a rejection on TCP 16992, I try TCP 16993 and if I get a connection accept, you can conclude that Intel AMT is TLS enabled or un-provisioned. At this point, you can't get the version number since you would need to do TLS to try to get it.

6.1 Q (jc): Then please explain exactly how the host name affects kernel routing decisions (given that the kernel TCP/IP stack neither knows nor cares about names) and how packets to 127.0.0.1 are ever routed.

A: Packets sent by other computers to port 16992/16993 are filtered by the NIC as OOB data to ME, so LMS never receives them. These packets never reach the Host OS.

If an application running on the Host OS wants to talk to ME, it needs to send packets to port 16992/16993 on 127.0.0.1 (or the hostname), regardless of what IP address ME is using. If ME is configured an different IP address than the Host OS, the local application should still use 127.0.0.1 rather than ME's dedicated IP address, otherwise the packets can never goes through LMS to HECI to ME to access the web services for **local applications (see 3.4).**

6.2 Q (jc) Do the Host OS and ME share an IP address? IPv4 or IPv6? What about VLANs?

A: AMT and the Host OS share an IP address in DHCP mode.
In static mode, the IP address must be configured to a value different from the operating system IP address. It is recommended that Intel AMT and the host have different hostnames if Intel AMT is to be addressed by its name, rather than by its IP address. This is the case when TLS is enabled. The IP subnet mask must also be configured. The default gateway, DNS servers, and Domain Name are configured optionally.
The Intel AMT and host TCP/IP settings must be compatible with each other:

- If the host is configured for DCHP, then Intel AMT must also be configured for DHCP.
- If the host is configured with a static IP address, then Intel AMT must also be configured with a different static IP address.
- ME can use a VLAN that is different from the host processor, or the host processor can be configured to operate without a VLAN definition.

But when the ME is configured to share IP addresses with the Host, using a DHCP-assigned address, they must be configured to use the same VLAN. The DHCP server should be enabled on this VLAN as well.

6.3 Q(jc) What about IPSec and IPv6 ?
A:

- The ME cannot be assigned an IPV6 address but the host OS can be assigned an IPV6 address
- The ME *does* support System Defense filters on the host traffic, whether or not the host communicates IPV4 or IPV6. This means that the System Defense filters must be configured on the ME network (via IPV4) but the filters are able to filter IPV6 traffic that if that is what the host OS is communicating with.

- If the host OS is assigned an IPV6 address, the ME must be statically assigned an IPV4 address. The assumption is that if the ME is configured in DHCP mode, it shares its configuration with the host OS. If the host OS is deployed in an IPV6 environment, it cannot share its configuration with the ME.
- Platforms deployed with Layer 3 (network) VPNs or IPSec will be unprotected by System Defense filters.

6.4. Q(gda/bs) IPMP & Link Aggregation ?
A: Up to version3.0, AMT is not available on server platforms. It's only equipped on workstations and laptops. So we haven't seen any use case so far. Intel's documents have no little discussion on this. The only thing we've found is in the release notes (Oct. 19, 2006) of Intel PRO NIC's Windows driver. It says: "If Intel(R) AMT is enabled on an adapter, you will not be able to add it to a team (Link Aggregation)".

6.5. Q(jc) The daemon has to bind that special port so that nobody else can use it. How or if it does so is unclear. Binding ports exclusively (and properly) on Solaris is unnecessarily hard.

A: Firstly, the ports are registered by Intel at IANA, nobody else is supposed to use the ports. Secondly, if the daemon fails to bind the ports, there's not any security impact to AMT.

It is actually very common for the daemon to fail, e.g. when the machine is compromised by a local user, and this is exactly how the firmware's Agent Presence and System Defense features are useful: a pre-configured watchdog timer can be triggered so that a pre-configured action will be taken, like sending an alert, isolating the machine from the network according to a preprogrammed network isolation filter policy, or logging the event to the local event log.

AMT is designed so that it works with or without the OS running, there's no failure mode depending on the state of the operating system.

6.6.Q (pj) So what happens when the host OS stops running suddenly (crash or hardware fault)? Further, it seems the host OS needs to re-enable the ARP intercept in the NIC hardware on shutdown. I'm very curious if that means a sudden host stop will render the AMT LOM inaccessible in short order.
A: AMT will continue to function if the OS stops running. See 3.2 Architecture. AMT has its own TCP/IP stack and NIC driver. It does not depend on the Host OS to enable ARP.

6.7.Q(gda) Interaction between the heci driver and LMS are project private ioctls. I think the project team needs to confirm this, and confirm that heci doesn't expose any knobs which are accessible to non-privileged users except via LMS.
A: exactly. Confirmed.

6.8. Q (rf) What occurs if the machine is suspended and the NIC is still hot (maybe WOL enabled)? (a host of questions about power state)

A: AMT can detect AC(power cord) vs. DC(battery) power supply. AMT can be configured to operate when the machine is in S0 ~ S5 states, alternatively with WOL enabled. For example, "On in S0 WoL S3/AC S4-5/AC" means that the ME is on when the host is on, when the host is in S3 to S5 and the platform is connected to AC power. The ME will shut down after a defined period of time, but will awaken when it receives a network message.

6.9.Q(bs) What happens when the host picks the same port for its own use

(for instance as the local port of an outgoing connection)?

A: The official answer from Intel is that “Intel has registered these ports at IANA and nobody else is allowed to use them.” If this happens the host will not be able to receive the expected packets because they're filtered to ME. Other than that there's no side effect. No security risk to AMT.

6.10.Q(pj) Does enabling AMT degrades snoop's visibility?
A: Yes, you don't get to see traffic destined for AMT. We can document this in the “platform guide” if there is such a thing. Or sticking a reference to it in the e1000g and iwk (or whatever the wlan drivers are that work with AMT) man pages is probably an even better place.

6.11.Q(jc) What is LMS listening to? Is it bound on just 127.0.0.1 port 16992 or something else?
A: Yes. 127.0.0.1 port 16992.

6.12.Q(jc) I don't quite know what "agent presence" means.
A: There's a watchdog in the ME firmware. An **Agent** is any process that is willing to report its **presence** to the watchdog in ME. The idea is that if an Agent doesn't tickle the ME periodically, then the ME can take some other assertive action. (Like sending an alert out over the network)

6.13.Q(jc) It sounds like this device won't work on IPv6-only networks. That probably needs to be explained somewhere in the Solaris documentation, as it's a consideration for Solaris administrators.
A: Yes. Agreed.

6.14.Q(jc) Similar issue for dladm(1M): need to update the man page to indicate that aggregations involving Intel AMT are not possible or not functional.(I don't think that Intel's market segmentation is necessarily identical to the segmentation seen by OpenSolaris users. In other words, aggregations and other features aren't "server only" features on Solaris.)
A: Agreed.

6.15. Q(jc) It'd be very nice if the system could _prevent_ the user from creating an aggregation that includes an AMT-laden port, but if that's not feasible, I guess I understand. That brings up a related architectural question: is there any way for other software in the system (or for administrators) to know that a given network interface is running AMT?
A: It *should* be possible, but it would require changes to Nemo, the e1000g and Intel wifi drivers, and they would have to negotiate with LMS when AMT is enabled.

6.16.Q(jc) I don't think the answer to 6.9 is right. Registration of the ports with IANA means that other applications shouldn't bind them. It doesn't actually prevent the ports from being used, and, in this case, the application will fail in a very strange and hard-to-diagnose way, because AMT appears to break the ability of snoop/ethereal to tell the user what's really happening. I assume there's not much we can do about that, but it would be surprising to folks who tune tcp_smallest_anon_port or who use IP Filter's NAT features and specify locally-unused port ranges.
A: We can document this in some kind of AMT chapter in a platform guide

6.17.Q(jc) On FMA -- I'm a little surprised that it's possible to create a local AMT client application, but that it's not possible to export events received via LMS into a local fault management application. Are none of the "General Info," "Storage (3PDS)," or "Local Agent Presence" features interesting to local fault management?
A: Possibly. But all of these are really just services being exported to ISV applications. I don't think (at least during this initial deployment) that any of them are intrinsic to the platform availability that I normally associate with FMA.

6.18.Q(jc) Perhaps it would help to have some more depth on these "Realms." I don't think I know what they mean.

A: There are only 3 Realms visible to local applications via LMS:
GeneralInfoRealm - Returns general setting and status information. It gives a user Read-only access to the settings.

StorageRealm - Used by user applications to allocate, maintain and specify access rights to non-volatile memory blocks in ME. The data placed in a non-volatile memory block is persistent.

AgentPresenceLocal - Realm used by an application designed to run on the local platform to report that it is running and to send heartbeats to ME periodically.

6.19.Q(bs) One of the things we asked for that I don't see an answer to in this spec is a set of recommendations for our customers regarding using AMT on hosts running solaris. Given the complexity and the impact on expected behavior, I believe we should recommend that, on systems with multiple interfaces, one interface should be dedicated to use by AMT and not used by the host. i.e., "just because you can, doesn't mean you should".

A: You're right we should recommend our customers to run AMT in low-complexity configurations. AMT is currently delivered on low-end PC/laptops with only one network interface. So the recommended way, according to Intel's AMT SDK doc, is to use shared IP (ipv4) in DHCP mode, and separate IP in static mode. See also 6.2.

6.20.Q(bs) section 4: "zero touch" administration involves pre-configuring keying material in the factory. Are we going to produce systems with preloaded trusted keying material? If so, how do we securely deliver that keying material to our customers along with the system?

A: preprogrammed certificates are delivered in ME firmware.

6.21.Q(bs) in section 6.2: is there really no way to assign different ip addresses to the host and the AMT device via DHCP? (for instance, different client id's for AMT device and host).

A: We do not know if it's possible. You may have made a good point. According to Intel's documentation the host and AMT should share an IP address in DHCP mode. See also 6.2 which is copied from Intel's website.

7 Use Cases (copied from http://softwarecommunity.intel.com/articles/eng/1032.htm)

Intel® Active Management Technology (Intel® AMT) is a silicon-resident management mechanism for remote discovery, healing, and protection of computing systems. It provides the basis for software solutions to address key manageability issues, improving the efficiency of remote management and asset inventory functionality in third-party management software, safeguarding functionality of critical agents from operating-system (OS) failure, power loss, and intentional or inadvertent client removal:

- Remotely Discover Computing Assets in Any Operational State: Intel® AMT stores hardware asset information in flash memory that can be read anytime, even if the PC is powered off or has an inoperable OS. Intel AMT does not rely on software agents to prevent accidental data loss. It also provides management applications with a general-purpose, non-volatile data store that accepts local or network-based storage commands.
- Remotely Heal Computing Assets: Proactive alerting notifies IT of a system problem, even when the system is down. Intel AMT provides out-of-band (OOB) access to remotely diagnose, control, and repair PCs after software, OS, or hardware failures. Alerting and event logging assists IT to diagnose problems quickly to reduce end-user downtime. Intel AMT also supports IDE-Redirection and Serial-Over-LAN capabilities for management applications.
- Remotely Protect Computing Assets: Through Out of Band communication, each system's software version numbers are checked and, if necessary, system software and virus protection are remotely updated with the most recent patches and virus definitions. Viruses and worms can also be contained at their source, if needed, by means of built-in circuit-breaker functionality.

Intel AMT infrastructure supports the creation of setup and configuration interfaces for management applications, as well as network, security, and storage administration. The platform provides standards-1.1 based encryption support by means of Transport Layer Security (TLS), as well as robust authentication support via Kerberos.

The technical capabilities and business value of Intel AMT are summarized in the use cases linked to the descriptions below:

Use Case	Purpose	Intel AMT Features Implemented (Typical)
UC1 (Discover) : Platform Auditing	Reduce or eliminate manual inventory audits by being able to locate systems regardless of power state or health. Improve asset management.	Out of Band (OOB) access, Power Status Control/ Monitoring, Intel® AMT Flash, Remote platform inventory, Tamper-resistant agent, Network Admin Interface
UC2 (Discover) : Software Inventory Management	Improve the software-inventory process; optimize maintenance contracts, licensing, and configurations inventory through firmware (FW) resident SW info.	Out-of Band (OOB) access, Remote software inventory, 3rd Party Data Store,Tamper-resistant agent, Network Admin Interface

UC3 (Discover): Hardware Inventory Management	Reduce manual audits and better manage hardware inventories, recalls, warranties. Efficiently manage hardware inventories.	Out-of Band (OOB) access, Intel® AMT Flash, Remote Hardware Inventory, Tamper-resistant agent, Network Admin Interface
UC4 (Heal): Remote Diagnosis, Remote Repair	Remotely diagnose and repair client machines, reducing on-site visits to resolve SW problems, even when OS is down.	Out-of Band (OOB) access, Remote troubleshooting and recovery, Tamper-resistant agent, Alert Handling, Read Event Logs, Network Admin Interface
UC5 (Heal): Remote Diagnosis, Local Repair	Reduce visits to resolve HW problems with improved remote diagnosis and hardware information.	Out-of Band access, Remote troubleshooting and recovery, Remote field-replaceable unit(FRU) inventory, Intel® AMT Flash, Tamper-resistant agent, Event Logs, Alert Handling, Network Admin Interface
UC6 (Protect): Software Version Compliance	Ensure up-to-date software versions, virus signatures, etc. Improve accuracy, speed and efficiency of anti-virus software updates regardless of OS or power state.	Out-of Band (OOB) access, IDE-R/SOL, 3rd Party Data Storage, System Defense, Agent Presence, Alert Handling, Read Event Logs, Network Admin Interface
UC7 (Protect): Hardware-based Isolation and Recovery	Detect and stop malware from propagating. Suspicious activity detected at a node, alert sent to console, IT quarantines system and updates policy out of band. Monitors out-bound traffic by comparing a timeslice of network traffic to enhanced filters in the system defense engine to obtain data on the timeframe and number of occurrences of a particular network traffic event.	IDE-R/SOL, System Defense, Alert Handling, Read Event Logs, Network Admin Interface, Wired and/or Wireless Network Filters, Flash Memory for Enhanced Filter Storage, Worm-Detection Filters

UC8 (Protect): Presence Checking of User Partition Agents	Virtually eliminate the ability of users or malware to circumvent protection. If the user disables agents, that action triggers alerts, quarantines the system, and re-initializes agent.	Agent Presence, Alert Handling, Read Event Logs, IDE-R/SOL, Network Admin Interface
UC9 (Protect): Endpoint Access Control (EAC)	Limit network access by visitor, rogue systems, and systems that do not conform to company policies for virus protection, OS patches, etc. Force systems that do not meet corporate policy onto a remediation network.	NAC server plug-in to read posture, verify AMT signature and return health statement; posture is created by Intel AMT firmware from system and BIOS data and then given to the Intel AMT Posture Plugin in Host OS
UC10 (Configure): One-Touch Configuration	Perform automated setup and configuration of an Intel AMT device, either using credentials stored on a USB key storage device or by keying credential information manually into BIOS.	Intel AMT firmware image, LMS driver, MEI driver, Intel Setup and Configuration Service (if a corresponding service is not provided by third-party software)
UC11 (Configure): Remote (Zero-Touch) Configuration	Automatically set up and configure an Intel AMT device upon connection to the network, either using a third-party management software agent resident on the client OS or from a 'bare-metal' state, without requiring a host OS.	Intel AMT firmware image, LMS driver, MEI driver, Intel Setup and Configuration Service (if a corresponding service is not provided by third-party software)

8 Solaris HECI driver interface.

The HECI (Host Embedded Controller Interface) driver can be accessed (by LMS) through an interface (IOCTLs) defined by Intel. We classify the IOCTL interface as **project private** because it is volatile, undocumented, and subject to change in the future.

9 Solaris LMS proxy

9.1 Q (gd): Does LMS export any interface over the network? (If its a web proxy....)
A: LMS is a web proxy, but only for clients running on the same machine. It only accepts connections from the local machine.

9.2 Q (gd): Q(jc) What starts the daemon? Is it an inetd service or something in SMF? Either way, please provide the service name that users will see.
A: I added it to SMF. The service name is “svc:/network/lms”.

10 Virtualization

10.1 Phase I is only concerned about global zone and dom0. -- We tested the HECI driver under xVM in Dom0 using snv_76 on a Toshiba Tecra M9 laptop.

10.2 Phase II targets non-global and domUs. Currently, we're ***not*** aware of any customer requirements to access heci driver from a non-global zone. I'm not familiar with virtualization - please advise if there are areas that we need to take notes.

10.3 In addition to its role in improving server capacity utilization, Intel® Virtualization Technology (VT) also is designed to work in a complementary fashion with Intel AMT to manage the health of the PC environment. In a normally executing environment (i.e., "in band"), VT is the primary player. It allows for options such as running multiple OSs or running a user partition and a management partition which simultaneously share the same hardware. The value of Intel AMT is most evident when the in-band channel is not accessible, either because the host OS has gone down or the host platform is in a lower power state. Intel AMT enables remote management of the PC regardless of the host platform's state, while the VT management partition uses the host processor and supports more elaborate and CPU-intensive, in-band management tasks.

11 FMA

Q (dg) Fault management is a purpose of AMT. Should AMT be considered for integration with Solaris's existing fault management framework, FMA? Do the two really need to be separate which necessitates distinct management methods/infrastructure?

A: As seen above, AMT and Solaris FMA address very different problems. And there's no way to integrate the firmware with the Host OS.

12 Potential Customers

There are already a lot ISVs making money on AMT. They include LANDesk, Novell, Altiris, Cisco, BMC Software, Check Point Software, StarSoftComm, etc. Their management software support Windows, Linux, Mac OS etc. With our integration of HECI/LMS into ON, it can greatly encourage the ISVs to migrate their AMT applications to Solaris.