InfoWeapons
SECURE INTUITIVE RELIABLE

Opportunities in Application Design Made Possible by IPv6

Lawrence E. Hughes

CTO & Chairman

InfoWeapons Corporation

The Speaker

- Founder, CTO and Chairman of InfoWeapons Corporation; he also currently serves as Chairman and CTO of DualStak Networks Sdn. Bhd., based in Kuala Lumpur, Malaysia.
- He established an IPv6-Ready Product Testing Center in the Philippines which is accredited to test products for IPv6 certification.
- He is an IPv6 Forum Certified Engineer, a member of the IPv6 Forum Education Committee, and an IPv6 Certified Trainer. He has spoken at a number of International IPv6 Summit conferences in Beijing (China), Seoul (Korea), Kuala Lumpur and Langkawi Island (Malaysia), Taipei (Taiwan), Potsdam (Germany), Washington D.C. and San Jose, California (U.S.), and Melbourne (Australia).
- He was the co-founder and initial CTO of CipherTrust in the US and was a Senior Security Consultant at VeriSign where he created and taught their crypto and PKI certification courseware internationally.
- He has written numerous magazine articles, blogs, and two books -- *Internet E-mail: Protocols, Standards and Implementation (1998)*, and *The Second Internet: Reinventing Computer Networks with IPv6 (2010)*

The Major Relevant Differences Between IPv4 and IPv6

- There are numerous technical differences in IPv4 and IPv6, but from the viewpoint of an application designer, the most important are:
  - Vastly larger address space means every node can have a global address.
  - There is no need for NAT. The almost universal presence of NAT on the IPv4 Internet has had a major chilling affect on Internet software design. The End-to-End connectivity of the early (pre-NAT) IPv4 Internet cannot be restored with IPv4. Carrier Grade NAT just makes it even worse. We can now create a true global End-to-End Internet, *but only with IPv6*.
  - We can finally deploy IPsec (the only IETF approved technology for VPNs) since NAT is no longer *in the way*. SSL-VPN was needed in IPv4 *only* because NAT breaks IPsec. IPsec over IPv6 will *revolutionize* VPNs.
  - Working, scalable multicast opens up fascinating new possibilities for automated server discovery (why should you have to tell clients the addresses of servers?) It also makes possible chat, voice and video conferencing, as well as viable global IPTV.

Introduction of Network Address Translation (NAT)

- *Network Address Translation* was introduced into the IPv4 Internet by RFC 1631, “The IP Network Address Translator (NAT)”, May 1994, when we first realized we would shortly run out of IPv4 addresses. From RFC 1631:

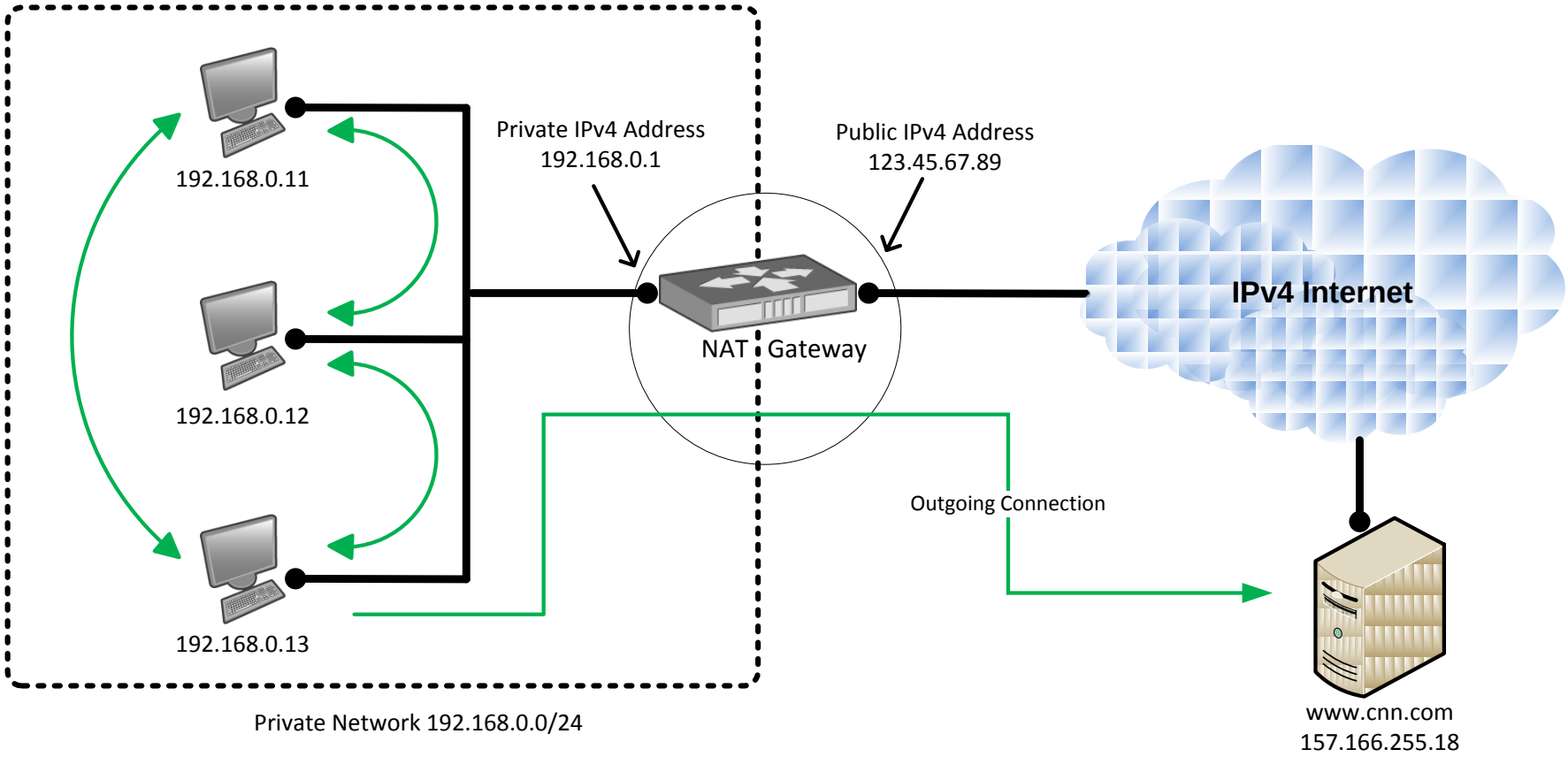
  > “The two most compelling problems facing the IP Internet are *IP address depletion* and scaling in routing. Long-term and short-term solutions to these problems are being developed. The short-term solution is CIDR (Classless InterDomain Routing). *The long-term solutions consist of various proposals for new internet protocols with larger addresses.*
  >
> It is possible that CIDR will not be adequate to maintain the IP Internet until the long-term solutions are in place. This memo proposes *another short-term solution*, address reuse, that complements CIDR or even makes it unnecessary. The address reuse solution is to place Network Address Translators (NAT) at the borders of stub domains.”

- RFC 1631 was updated by RFC 2663, “IP Network Address Translator (NAT) Terminology and Consideration”, Aug 1999.

Introduction of Private IP Addresses

- *Private addresses* were introduced into IPv4 by RFC 1597, “Address Allocation for Private Internets”, Mar 1994 (updated by RFC 1627 “Network 10 Considered Harmful” Jul 1994; then replaced by RFC 1918, “Address Allocation for Private Internets”, Feb 1996.)
- The idea was to repurpose several formerly *globally unique* netblocks (10.0.0.0/8, 172.16.0.0/12 and 192.168.0.0/16) and allow anyone to use them in isolated (“private”) internets (*lower case i*), consisting of one or more LANs connected via routers. Full end-to-end connectivity is still possible *within* an entire private internet, if no internal NAT gateways are deployed.
- The two concepts were combined to allow many nodes with private addresses to be deployed behind NAT gateways. Nodes in these private internets could make outgoing connections (and accept incoming replies) to nodes on the IPv4 Internet (*capital I*), via a single *public* IPv4 address (e.g. 123.45.67.89). This is called “full cone NAT”. One-to-one mappings are also supported (“BINAT” or “symmetric NAT”). We no longer have a single global Internet. We have millions of tiny private internets connected via NAT gateways.

IPv4 with NAT (NAT44)



The Impact of NAT and Private Addresses

- With the introduction of NAT and private addresses, at each NAT gateway End-to-End connectivity is lost. Nodes in a private internet behind a NAT gateway can easily make *outgoing* connections (to nodes with real public addresses in the public Internet) up to a total of 60,000 or so simultaneous outgoing connections through one gateway. But, it is complicated for nodes outside of the NAT gateway to make *incoming* connections to the nodes inside the private internet. Connectivity between nodes inside private internets and nodes in the public IPv4 Internet has become “one way” (outgoing).
- Some number of nodes in a private internet (e.g. typical workstations) can make *outgoing* connections to public nodes (e.g. cnn.com), but except in certain circumstances, no *incoming* connections to those nodes can happen.
- At the time NAT was designed, most network applications made only 1 or 2 connections at any given time. Current network applications (e.g. iTunes and Google Maps) make 200 or more connections *simultaneously* for higher performance. You can use up the 60,000 possible outgoing connections on one NAT gateway surprisingly quickly today (perhaps 300 nodes).

Incoming Connections to Nodes Behind NAT

- If a node is behind a full cone NAT gateway, incoming connections to that node from external nodes (either nodes on the public Internet, or nodes in other private internets) can be made *only* if one or both of the following is true:
  - The external address of the NAT gateway is a public address *and*
    - The admin has mapped specific ports (e.g. 25, 80 and 443) on the gateway to an internal private address (*port translation*) . A given port can only be mapped to *one* internal node in a given private internet. -OR-
    - The admin has mapped *all* ports from a secondary public *alias* address (in addition to the one used for Full Cone NAT) on the gateway *to and from* an internal address (*1:1 NAT*). Each such mapping requires one public address.
  - Both the client and server applications involved have implemented some variant of NAT Traversal - see RFC 3489, “Session Traversal Utilities for NAT (STUN)”, Oct 2008. This describes a set of network tools used to build traversal solutions for specific applications.

Problems with All Schemes for Connections Thru NAT

- All of these schemes have severe limitations, unnecessary complexity and/or security issues.
  - Port translation – only a single server for any given protocol (e.g. HTTP) is allowed in the entire private internet behind the NAT gateway. You can redirect HTTP to one internal server, and SMTP to another, but you cannot direct HTTP to two different internal servers.
  - 1:1 NAT – a unique public IPv4 address is needed for *each* internal server (and you may recall that we are running very short of public IPv4 addresses!)
  - NAT Traversal (e.g. STUN) – this *greatly* complicates design and implementation of both client and server applications and introduces major security issues. It also introduces efficiency and reliability issues due to breaking TCP streams into UDP datagrams.

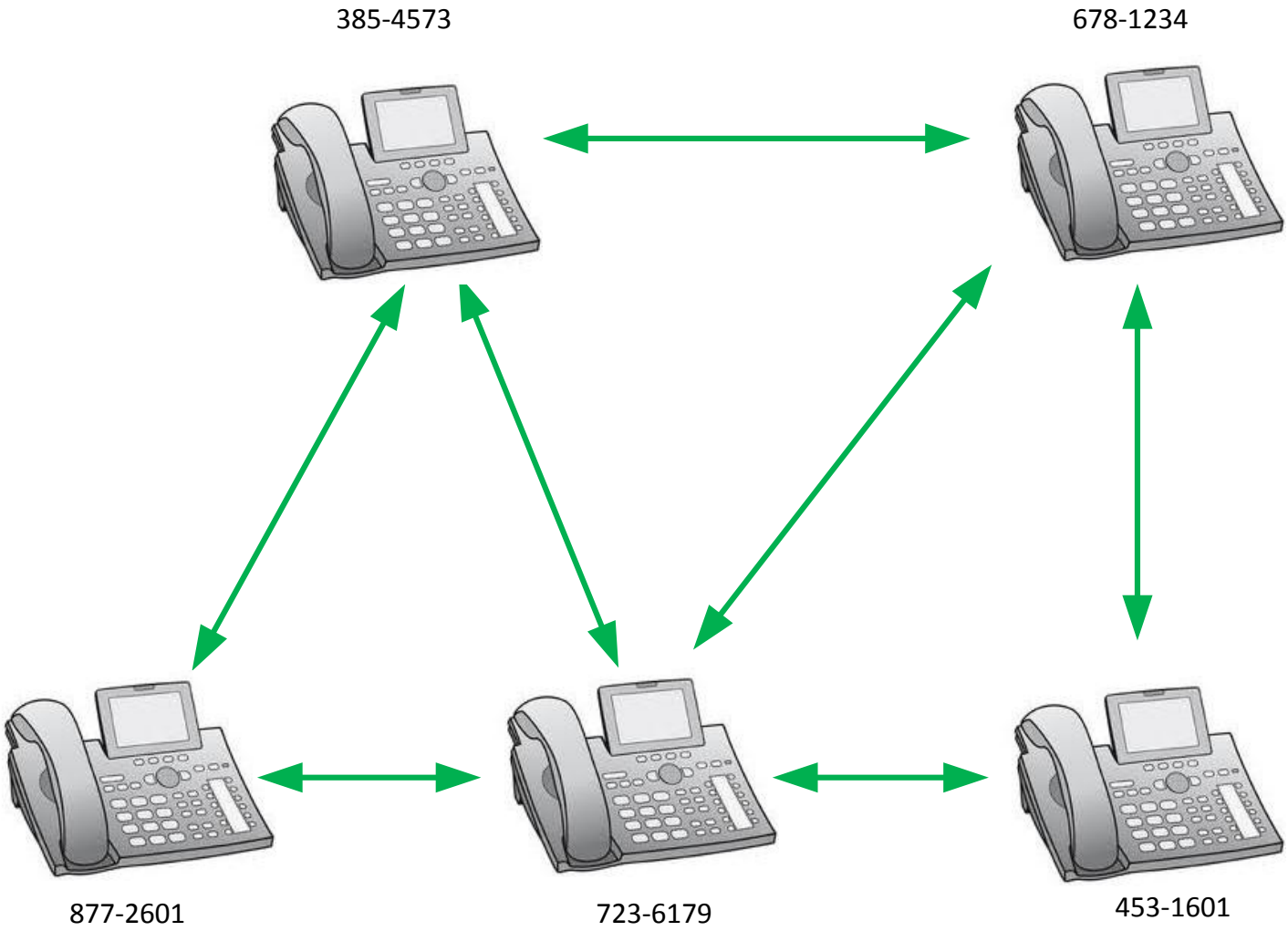
The other “Digital Divide”: Producers vs. Consumers

- With NAT, the IPv4 Internet has divided into two communities – those who *produce* content (e.g. cnn.com) and those who can only *consume* content. This is similar to having a few people produce content for newspapers and television, and many people consuming that content. This is a concept whose time has passed.
- Web 2.0 has addressed that to some extent, by allowing more people to *contribute* content to public sites like YouTube or Facebook, but *you* don’t control those sites - big companies do. Those companies can filter content they don’t like or enforce one political view. Peer-to-peer file sharing is an example of full end-to-end connectivity, but it requires a public address, or employs NAT traversal, which accounts for most of the complexity in its design.
- Most importantly, all applications and sites have to be *designed around* the “one-way” access available to most Internet users on the IPv4 Internet today.

Telephones: True End-to-End Connectivity

- In most cases (unless a PBX is in the way) when I call someone by PSTN phone (wired or wireless), I have full End-to-End connectivity. My phone can make outgoing calls to anyone, and accept incoming calls from anyone.
- This requires a globally unique telephone number for every phone on earth.
  - At the top level there are ITU country codes (like +63 for the Philippines).
  - The next level is in-country regional or city codes (like 2 for Manila or 32 for Cebu).
  - In most regions there are 3 or 4 digit exchange codes (e.g. 123 or 8882)
  - Finally you get to the four digit number within a local exchange (e.g. 4567).
- For example, a phone in Cebu, Philippines might be +63-32-123-4567.
- Being able to directly connect from any phone in the world to any other phone in the world (without any help from telephone company operators) is one of the major achievements of the 20th century.

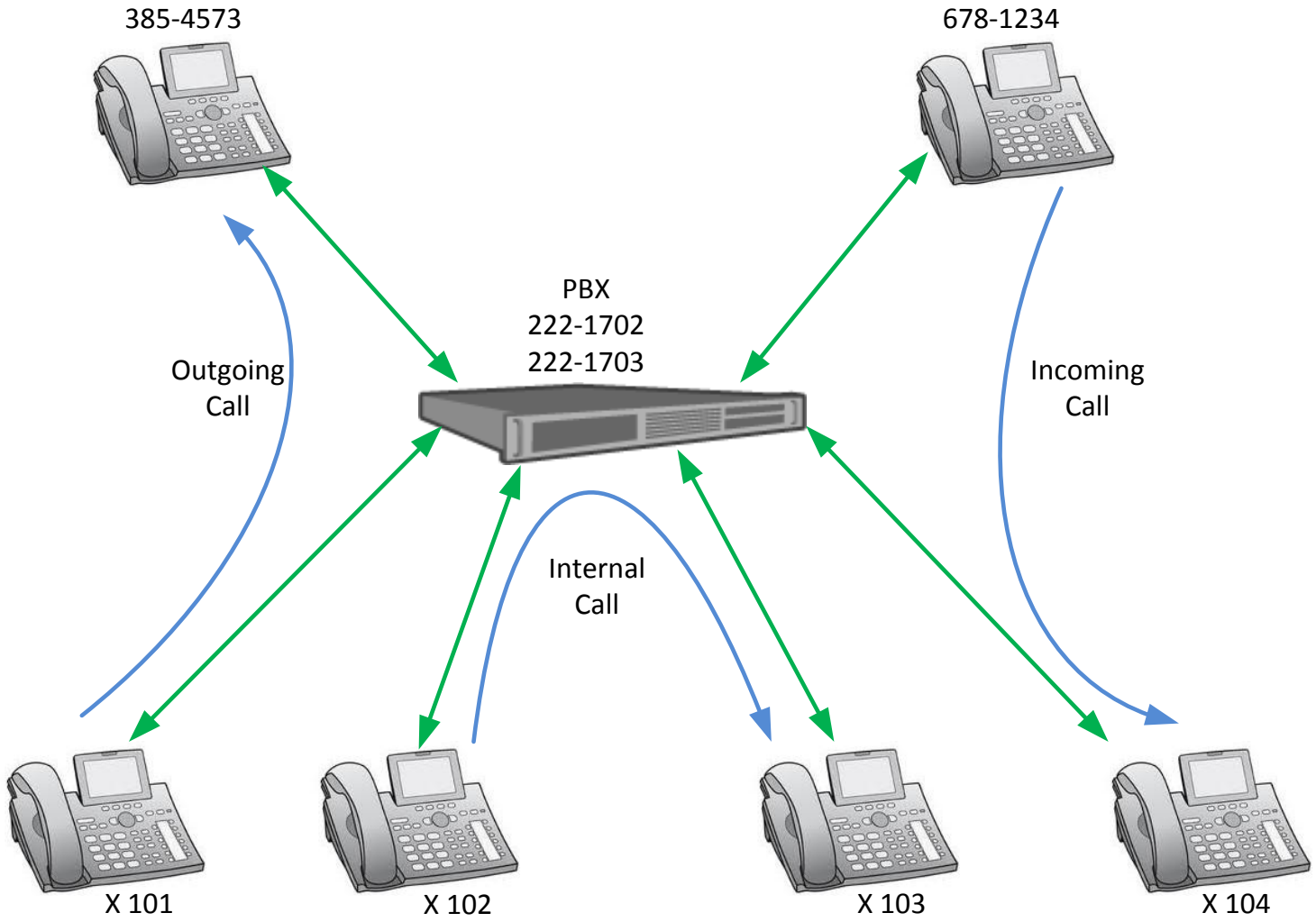
Peer-to-Peer Phone Connectivity



Telephones: PBXes – the Fly in the Ointment

- For extension telephones in an organization, they obtain a few “real” global telephone numbers, and a magic box (called a Private Branch Exchange or PBX) that can let quite a few “local extensions” hide behind the “real” phone numbers.
- Any internal extension can easily call any other internal extension. This is comparable to nodes in a LAN (where there is no NAT).
- Only a certain number of people can simultaneously be having conversations with “external” phones (once all the “real” phone numbers are in use, the next incoming or outgoing call gets a busy signal). Incoming calls must be connected to the right “extension” (*private address*) by an operator, or by an auto-attendant program that lets you dial a few characters of the person’s last name. This is a close equivalent to Internet NAT. Network connections via NAT gateways have comparable restrictions and issues.
- Imagine if telcos told their customers there were no longer enough “real” telephone numbers to go around and almost everyone was going to be put behind “Carrier Grade” PBXes, possibly even behind TWO LEVELS of PBX.

Phones Behind a PBX



So How Does Skype Do End-to-End connections?

- Skype gives the *appearance* of End-to-End connectivity, even when one or both users are behind NAT (have only private addresses).
  - First, the majority of the code and complexity in Skype deals with getting around the limitations of NAT. Few developers are capable of handling this particular complexity. This has had a “chilling effect” on innovation.
  - Second, your Skype client can magically become a “supernode” with no warning and act as a relay agent for many other people’s calls, using up your bandwidth – *especially if you happen to have a public address*.
  - Third, there is no way to achieve true End-to-End privacy and authentication with Skype – it can be secured from Alice to Skype, and from Skype to Bob, but not directly from Alice to Bob.
  - Fourth, use of Skype introduces many complex security issues for the people responsible for a company’s network security. Many companies are now blocking all Skype usage because of these issues.

What Does the Popularity of Skype Prove?

- Clearly people want (and are ready for) true Peer-to-Peer applications. They are willing to put up with some really bad issues (vulnerabilties, becoming a supernode, etc) in order to have that today on the NAT-infested IPv4 Internet.
- It is my contention that many more true Peer-to-Peer applications would be just as enthusiastically received, but they are *very* difficult to create and deploy with NAT in the way.
- Such applications are WAY easier to create and deploy if NAT is not in the way (for example, on the IPv6 Internet).
- Massive Multiplayer Role Playing Games (like World of Warcraft) are another area where many strange things are being done today to “fake” real public IP addresses. Check to see if your kids have installed something called *Hamachi* on their computers – this is actually a serious security vulnerability. It allows them to join Multiplayer Games from behind NAT.
- Without NAT to overcome, almost any network developer can create true Peer-to-Peer applications, and no “Hamachi” type kludges are needed.

Voice over IP – the Classic *Peer-to-Peer* Application

- Most applications you use today (e.g. email & web) are *client-server* architecture. Clients make outgoing connections to a central server (which must have a public IP address). In some cases, servers may exchange information with other servers (e.g. e-mail MTAs). But servers *never* try to connect to your client. Nor will another e-mail or web client. *The whole client-server concept evolved the way it did due to the limitations of the IPv4+NAT Internet.*
- VoIP is simply telephony done over the Internet. VoIP is based on SIP (Session Initiation Protocol) and RTP (Real-time Transport Protocol). A VoIP client (e.g. an IP hardphone or softphone) is called a **User Agent** (UA). Any UA can connect directly to any other UA, if no firewall or NAT blocks that connection. No e-mail or web client can connect directly to another e-mail or web client – they can only connect to a *server*. A VoIP "server" is really just a pair of User Agents connected "back to back" (the technical name for a VoIP server is B2BUA – Back to Back User Agent). VoIP has *lots* of problems working through NAT. It *assumes* that full end-to-end connectivity is available. If it isn't you have to use VoIP in a degraded client-server manner.

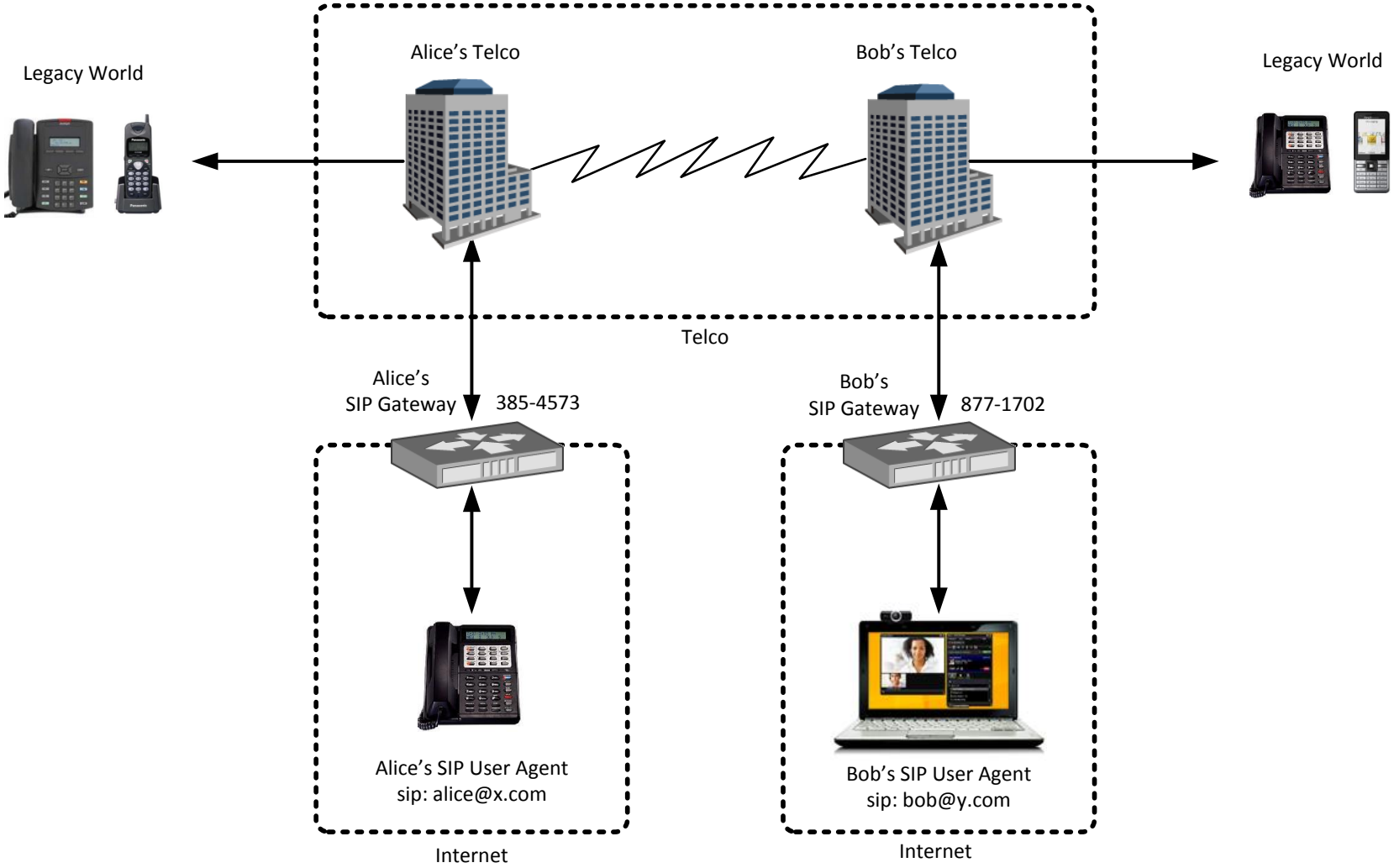
Let's Think "Out of the Box"

- Imagine an Internet *with no NAT* – true End-to-End connectivity. This exists within single subnets and most private internets today. It existed on the entire IPv4 Internet until the mid 1990s. It will *always* exist on the IPv6 Internet (no need for NAT66 – there are plenty of global addresses). Any node can connect to any node (unless specifically blocked by a firewall rule or authentication failure).
- Second, let's imagine you have an essentially unlimited number of public addresses (18 *quintillion* public addresses in a single IPv6 subnet (a "/64") qualifies as *practically* unlimited in my book). The IETF recommends that ISPs provide every home user with 16 or more /64 blocks. Business users get a minimum of 65,536 /64 blocks (a /48 allocation block). There are enough /48 blocks in IPv6 for *every human alive today* to get over 5,000 of them.
- With IPv6, *anyone* can be both a consumer and a producer of content (a *prosumer*). No more "digital divide" between producers and consumers of content. Anyone can run as many web, email or other servers as they want to.
- More importantly, VoIP and other *Peer-to-Peer* applications work great!

VoIP with NAT in the Way (How It Works Today)

- Let's say we have two users, Alice and Bob (both behind NAT) that want to talk with each other.
- On the IPv4 Internet with NAT, it is difficult for anyone to call Bob directly via SIP (no incoming connections for most users). Most people today obtain SIP accounts from a SIP service provider (probably at a telco, who can also connect you to zillions of legacy PSTN users via gateways, for a small fee).
- If Alice wants to call Bob via SIP, she dials the ITU numeric phone number that Bob's provider assigned to him. Her outgoing connection goes to her SIP service connection, over intervening telco infrastructure, and finally into Bob's SIP service provider at the same or another telco. To Alice and her telco, Bob is just another legacy PSTN user with an ITU numeric phone number, even though he is using a SIP UA. Unless *both* of their SIP UAs have unique public IPv4 addresses, it is almost impossible for them to call each other.
- If Alice and/or Bob have SIP servers, they probably obtain SIP Connect service from those to some telco, and things work pretty much as above.

VoIP on IPv4 with NAT in the Way



An Analogy to E-mail

- Imagine if to send someone an e-mail, you had to go through the Post Office.
  - Alice would use her computer to compose a message to Bob and send it electronically to her neighborhood Post Office.
  - Her Post Office would print out her message, then fax (or maybe even physically deliver) that printout to a Post Office near Bob.
  - Bob's Post Office would receive the fax (or physical printout) and deliver it to Bob's home. If they are *really* advanced, maybe they receive the fax to disk, or scan in the physical printout, and allow Bob to retrieve it electronically from his local Post Office.
- Using SIP between two SIP UAs via legacy Telcos using ITU phone numbers makes about as much sense as this "e-mail via Post Office" system just described. That system might be useful for getting messages to legacy users who don't *have* computers, but if Alice and Bob both have computers connected to the Internet, it would be insane. I'm sure the Post Office wouldn't do this for *free*. Telcos won't relay your SIP calls for free either.

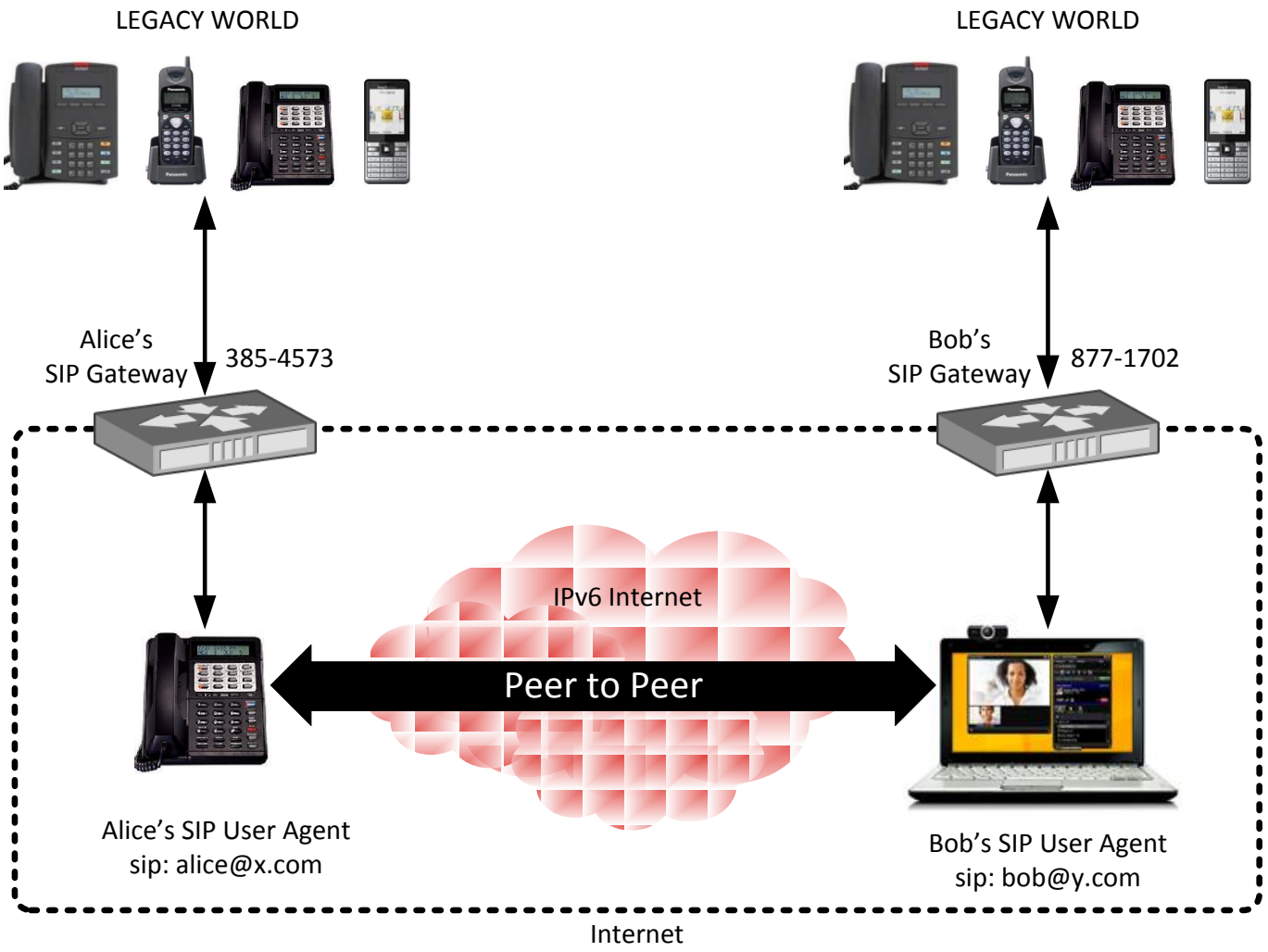
VoIP Without NAT

- Real VoIP phone numbers are *not* strings of decimal digits (that is *so* 20th century). Real VoIP phone numbers are *SIP URIs*. These look kind of like web URIs or email addresses, and leverage the same worldwide DNS infrastructure for scalability and decentralization. Lets say Alice and Bob each have SIP URIs: *sip:alice@x.com*, and *sip:bob@y.com*.
- If Alice knows Bob's IPv6 address (maybe saved in her address book from a previous conversation), and he is online, she can simply connect directly from her SIP UA to his SIP UA using VoIPv6. Just like "real" telephones!
  - If Alice knows the domain name of the node where Bob's SIP UA is running (*bobphone.y.com*), she can use DNS to resolve it.
  - If she knows his SIP URI, she can ask DNS for the preferred SIP server for his domain (found in a SRV record) and connect to him via that server.
  - She can lookup his SIP URI in his LDAP server. She can even ask DNS for the preferred Directory (LDAP) server for his domain (found in another SRV record), in which she can look up his SIP URI.

VoIP Without NAT – Connecting to the PSTN

- What if Bob (the person you are trying to call) is still using VoIPv4 behind NAT? You can't connect directly to him unless you happen to be in the same private internet (or both of you have your own public IPv4 addresses), even if *you* have IPv6. Bob may have a SIP server (with a real public address), and it could relay a call to him if he happens to be in the home network. What if he is at home or on the road? He will probably be behind *yet another* NAT gateway. Not only can YOU not connect to him, his SIP server can't either. With VoIPv6, he could register his current address from anywhere in the world (maybe via dynamic DNS registration), and anyone can connect directly to him.
- If Bob only has PSTN service (or even if he has a SIP account via a telco), you can still use SIP connect service at a telco to route calls between your SIP phone and legacy users like Bob. It will be a long time before all of the billions of telephone users worldwide have made the leap to real 4G telephony using VoIPv6. Eventually though, *all telephony will be over IPv6!*

VoIP on IPv6 – *Peer to Peer!*



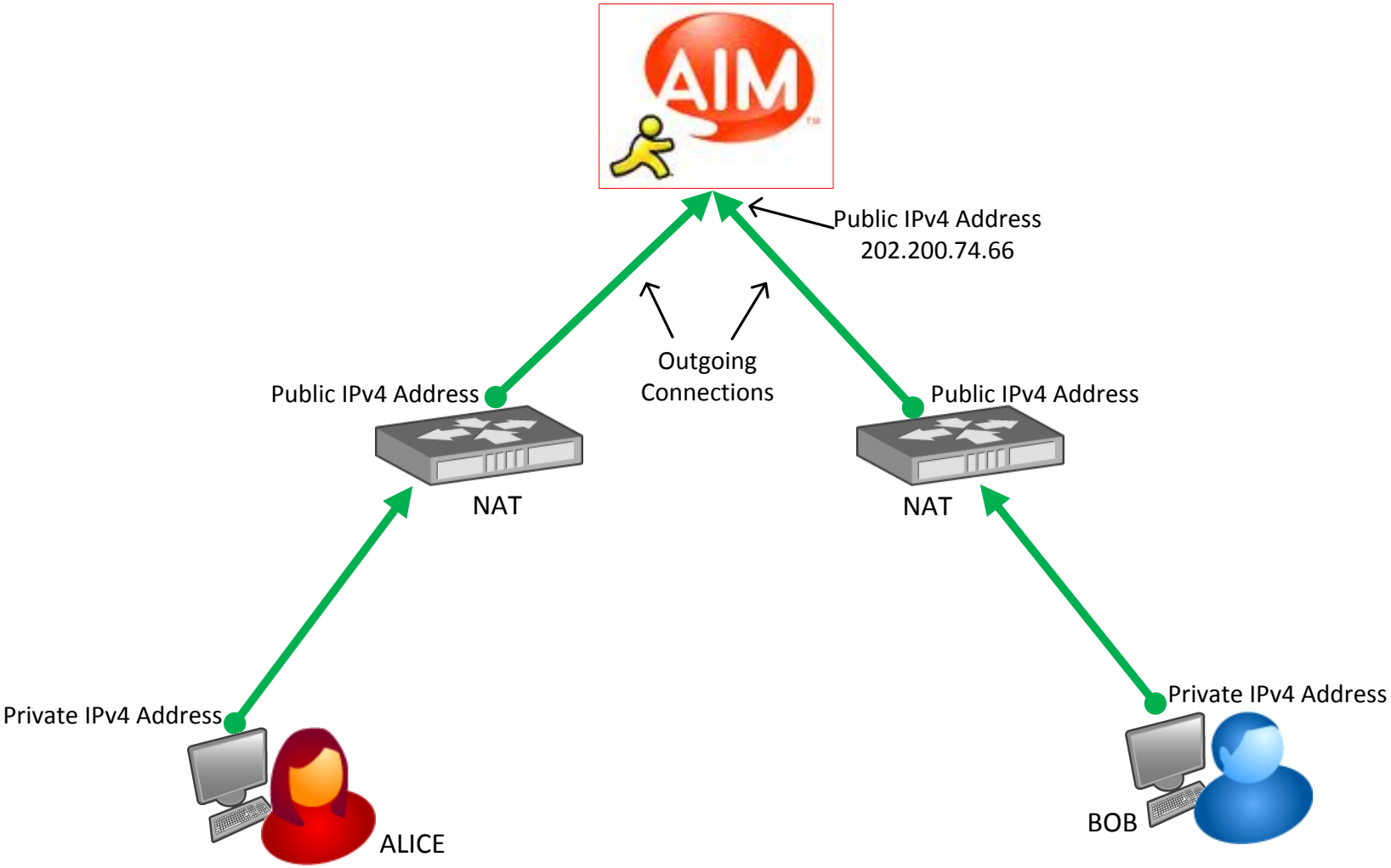
Future Telephony – Fourth Generation (4G)

- Real 4G begins with *LTE Advanced* – "LTE" (which is being deployed now) can be thought of as *3.9G*
  - With real 4G, wireless and wired systems converge, both IP based
- Real 4G uses *only* packet switched technology (no more circuit switched)
  - Essentially a subsystem of the Second Internet, real 4G telephony uses Internet infrastructure and protocols such as routers, DNS, ENUM, QoS, and VoIP, all over IPv6
  - Unlike 3G hub-and-spoke architecture, 4G is *true End-to-End*
- Real 4G must support IPv6
  - By time 4G is deployed, the public IPv4 address pool will be ancient history
  - It's not possible to do real End-to-End telephony with IPv4+NAT
- Verizon already requires all 4G devices to support IPv6 – not optional

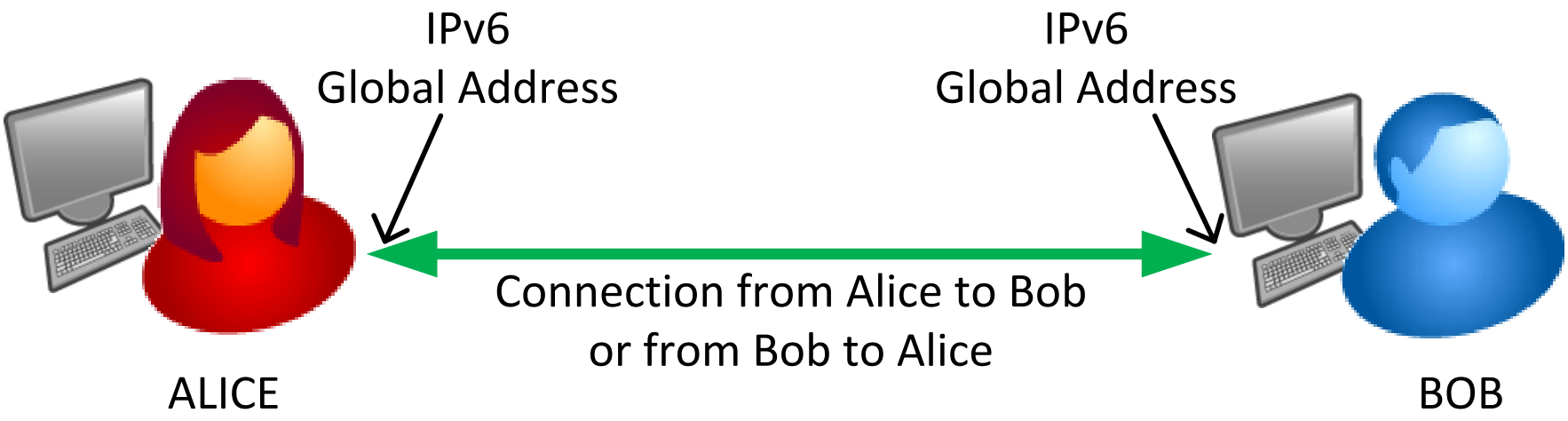
Let's Think WAY Outside of the Box

- So IPv6 (with no NAT) is a major win for VoIP. What about for other applications? Let's start thinking WAY outside of the box.
- What about Instant Messaging (chat)? Today, with NAT, Alice and Bob both make outgoing connections to AIM, and AIM shuttles chat messages back and forth between their two outgoing connections. Here are some issues with that:
  - Just how many simultaneous conversations *can* AIM handle?
  - AOL can see (and even record) everything Alice and Bob are saying to each other.
  - Third, Alice and Bob are not even really sure that they are chatting with each other. Maybe "Alice" is really some 50 year old guy in Russia!
  - There is no way to achieve true End-to-End privacy or authentication.
  - AIM has unique identifiers different from Alice and Bob's other identifiers (the AIM identifiers are in a unique *namespace*).
- We can build a next generation true End-to-End Instant Messaging system based on SIP and IPv6. It works the same was as VoIPv6, just for text messaging. It could be done using SIMPLE (SIP based IM).

Instant Messaging on IPv4 with NAT



Instant Messaging on IPv6 - *Peer-to-Peer*



Why Stop There?

- Current E-mail is hopelessly broken, thanks to spammers. SMTP is 1982 technology. It's almost as old as IPv4 (1981). It's older than many of you in the audience!
- With a slight generalization of the IPv6 based Peer-to-Peer IM client to handle larger messages (including MIME and even S/MIME), and optional store and forward nodes (for when the recipient is offline), we could make a truly scalable, secure and authenticated replacement for e-mail. It could use the same SIP URI addresses as VoIP and IM.
- If I'm online when you send me a large message, why should it have to go through a central store and forward system, especially if I have a global address? You could even use IPv6 multicast to deliver messages to multiple recipients simultaneously. If I'm offline, the message could go into a store and forward node temporarily, for delivery to me the next time I go online.
- This could be generalized just a bit more to handle file transfer (not just as MIME attachments, but in native binary format).

Generalizing VoIP to Unified Messaging

- Working, scalable multicast will allow sophisticated conferencing
  - UAs could create fully meshed connections using IPv6 multicast
  - SIP has good support for conference setup
- You would still need some centralized infrastructure, in each domain
  - A directory to locate people, including their nodename and their client digital certificate (for asynchronous methods), plus a *presence server* to track and publish their current status (available, offline, etc)
  - A PKI to issue client certificates to each user (for privacy and authentication)
  - A conferencing server to schedule or manage conferences
  - Gateways to/from legacy systems like IM, Skype or SMTP mail
  - A message store for asynchronous methods (like mail and file transfer)

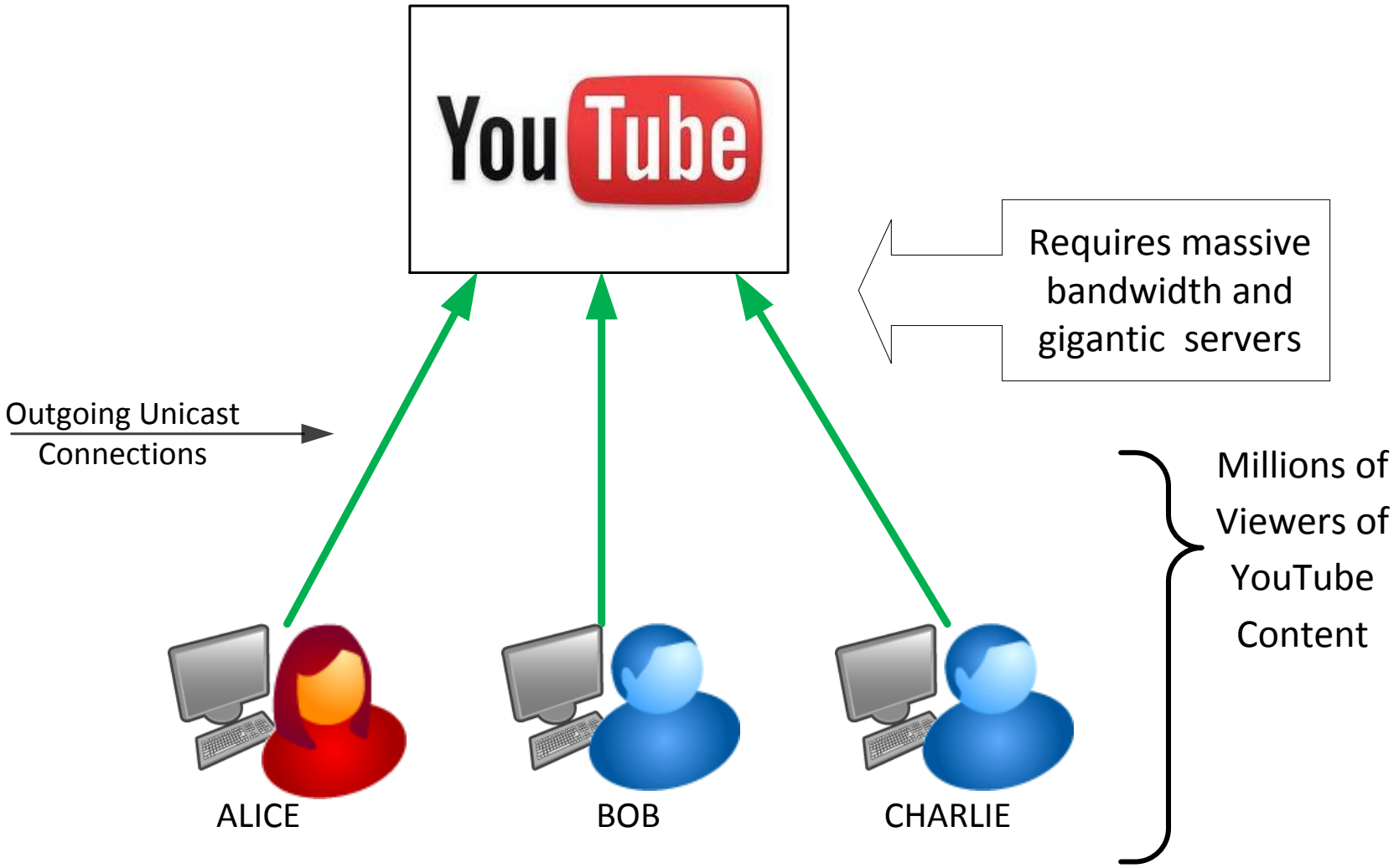
Potential IPv6 “Killer App” to Drive IPv6 Adoption

- People will not get IPv6 for its own sake – it is *infrastructure.* If you don’t have a car (or use taxis) you don’t need roads. People will get IPv6 if it makes possible something they actually *need*.
- If all they can do over IPv6 are things they can do over IPv4, they won’t see any benefit, and will not be willing to pay for it, or demand it from their ISP.
- IPv6 will only be widely adopted when there are applications that can do things that are *not possible* over IPv4 with NAT.
  - When TV was first released, they mostly showed *movies* over it (the previous technology). When someone realized they could do live news and live sporting events over TV, that was something compelling that *could not be done with movies*, and adoption quickly went mainstream.
  - Today, most demos of IPv6 show things that can easily be done over IPv4 (web surfing, file download, etc). No excitement here!
- An Peer-to-Peer decentralized communications system like the one described here can drive IPv6 adoption.

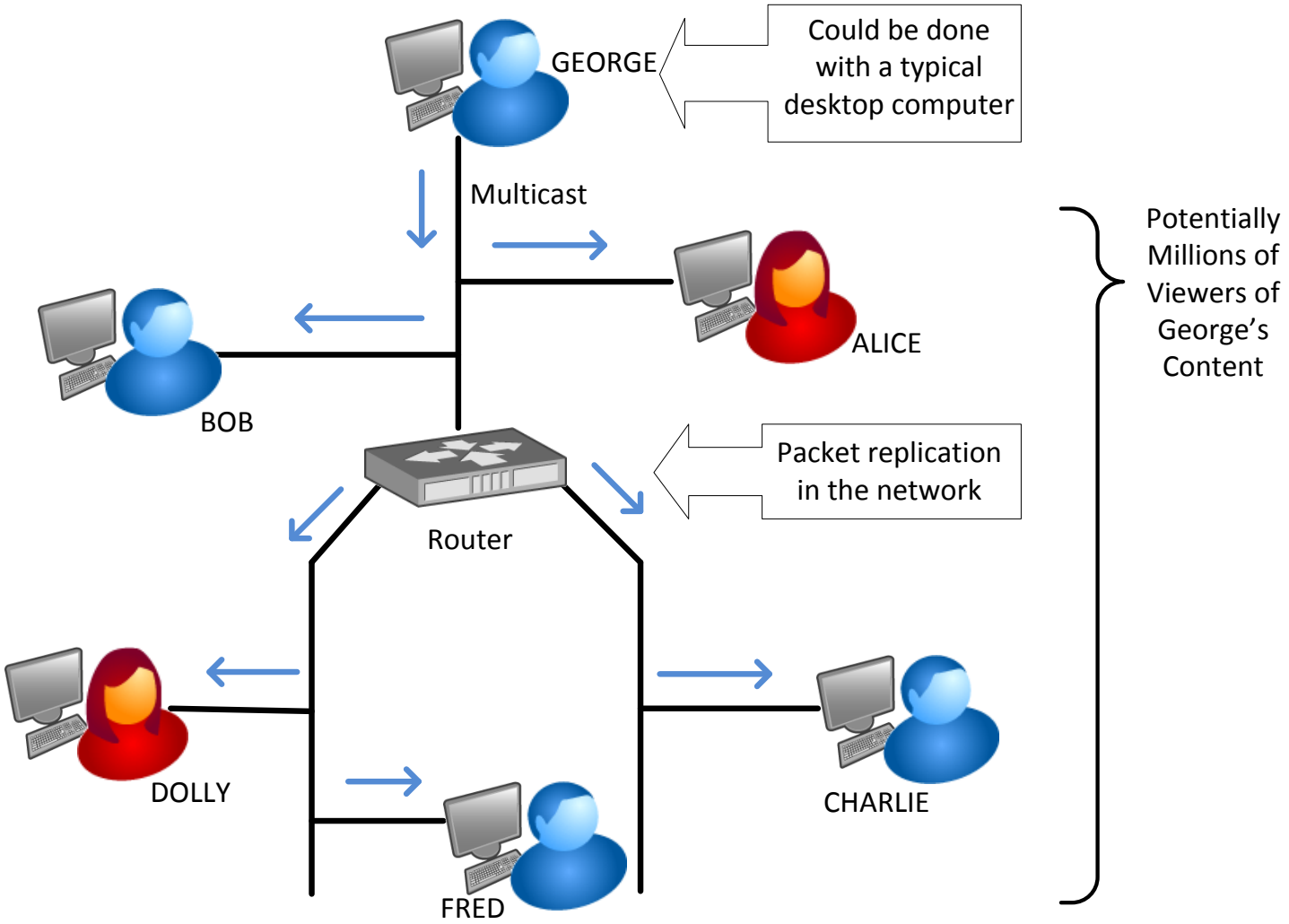
IPv6 - The Key to Commercially Viable IPTV

- Most people trying to do IPTV today are using *unicast* connections, from one client to one server (e.g. YouTube). This *does* allow everyone to be watching something different (“video on demand”), but the bandwidth consumed increases linearly with the number of viewers. You also need powerful servers to support a large number of clients. No conventional broadcast or cable TV business could survive if it provided unique programs to each customer.
- IPv6 has great support for *multicast*, and in all scopes (link local, organization local, global, etc). The mechanisms that a client uses to *join* a multicast group work better and are more scalable than in IPv4 as well.
- Working multicast and a large number of unique multicast addresses are the keys to commercially viable broadcast audio (“Internet radio”) and video (“IPTV”). There are *far* more multicast addresses in IPv6 than in IPv4 (millions per human alive).
- An IPTVv6 broadcast source needs only the power of a typical personal computer, and 2-4 Mbit/sec bandwidth for SD (10 Mbit/sec) for HD, for the outgoing multicast stream. There could be *millions* of viewers of that stream.

IPTV in the IPv4 Internet – Unicast Only



IPTV in the IPv6 Internet – *with Multicast*



The Key to Commercially Viable IPTV (cont.)

- With unicast, replication to each user takes place in the broadcast server. With multicast, replication takes place *in the network* (actually in the routers).
- With legacy cable TV, there may be a few hundred channels available. In IPTVv6, there could be literally *millions* of channels available, from all over the world. Perhaps there will be 500 channels just of Bollywood music videos!
- Legacy TV channels are typically 3 digit integers (e.g. 135). Next generation TV “channels” are IPv6 multicast addresses of which there are *trillions* available. Of course those could be mapped to symbolic names via DNS.
- On existing cable TV, they must provide bandwidth to each user for *all* available channels. On IPTV, you only need bandwidth to a given user for the number of channels being watched at any given time. In Japan *today*, home ISP service is typically 100 Mbit/sec. You could have 10 people in one home, each viewing different HD channels with that kind of bandwidth.
- The technical capabilities of IPv6 can make IPTV not only commercially *viable*, but actually even more cost effective than legacy TV networks.

This Is Why Developers Like Me are Excited about IPv6

- IPv6, with its essentially unlimited number of public addresses, allows us to get rid of NAT once and for all – return to the true End-to-End model envisioned by the creator of IPv4 (Vint Cerf). We can ditch the “two tier” addressing system currently in use (public IP address + private address) and go back to a “one tier” flat model. Peer-to-Peer EVERYTHING.
- NAT was a necessary evil for about 10 years from the mid 1990s to 2005 (or *at most* 2010) while IPv6 was being developed and refined. IPv6 is ready for prime time today and being deployed worldwide. We can throw away the crutches (NAT) that allowed us to continue hobbling around even with broken legs (a scarcity of new IPv4 address).
- The opportunities for creating a whole new generation of true Peer-to-Peer and multicast applications is *incredibly* exciting. But we can’t do it until there is a true Next Generation foundation to build them on. That foundation is the *Second Internet*, based on IPv6.
- It’s time to build this revolutionary Internet and connect billions of people *and even more devices* to it. The King (IPv4) is dead. Long live the King (IPv6)!

Thank You

…and welcome to the Second Internet!

www.infoweapons.com
www.v6address.com
www.secondinternet.org