              Autonomic Networking: Definitions and Design Goals

Abstract

   Autonomic systems were first described in 2001. The fundamental goal
   is self-management, including self-configuration, self-optimization,
   self-healing, and self-protection. This is achieved by an autonomic
   function having minimal dependencies on human administrators or
   centralized management systems. It usually implies distribution
   across network elements.

   This document defines common language and outlines design goals (and
   what are not design goals) for autonomic functions. A high-level
   reference model illustrates how functional elements in an Autonomic
   Network interact. This document is a product of the IRTF's Network
   Management Research Group.

Table of Contents

1. Introduction to Autonomic Networking

   Autonomic systems were first described in a manifesto by IBM in 2001
   [Kephart]. The fundamental concept involves eliminating external
   systems from a system's control loops and closing of control loops
   within the autonomic system itself, with the goal of providing the
   system with self-management capabilities, including self-
   configuration, self-optimization, self-healing, and self-protection.

   IP networking was initially designed with similar properties in mind.
   An IP network should be distributed and redundant to withstand
   outages in any part of the network. Routing protocols such as OSPF
   and IS-IS exhibit properties of self-management and can thus be
   considered autonomic in the definition of this document.

   However, as IP networking evolved, the ever-increasing intelligence
   of network elements was often not put into protocols to follow this
   paradigm, but was put into external configuration systems. This
   configuration made network elements dependent on some process that
   manages them, either a human or a network management system.

   Autonomic functions can be defined in two ways:

   o On a node level: Nodes interact with each other to form feedback
      loops.

   o On a system level: Feedback loops include central elements as
      well.

   System-level autonomy is implicitly or explicitly the subject in many
   IETF working groups, where interactions with controllers or network
   management systems are discussed.

   This work specifically focuses on node-level autonomic functions. It
   focuses on intelligence of algorithms at the node level, to minimize
   dependency on human administrators and central management systems.

   Some network deployments benefit from a fully autonomic approach, for
   example, networks with a large number of relatively simple devices.
   Most currently deployed networks, however, will require a mixed
   approach, where some functions are autonomic and others are centrally
   managed. Central management of networking functions clearly has
   advantages and will be chosen for many networking functions. This
   document does not discuss which functions should be centralized or
   follow an autonomic approach. Instead, it should help make the
   decision which is the best approach for a given situation.

Autonomic function cannot always discover all required information;
for example, policy-related information requires human input, because
policy is by its nature derived and specified by humans. Where input
from some central intelligence is required, it is provided in a
highly abstract, network-wide form.

Autonomic Computing in general and Autonomic Networking in particular
have been the subject of academic study for many years. There is
much literature, including several useful overview papers (e.g.,
[Samaan], [Movahedi], and [Dobson]). In the present document, we
focus on concepts and definitions that seem sufficiently mature to
become the basis for interoperable specifications in the near future.
In particular, such specifications will need to coexist with
traditional methods of network configuration and management, rather
than realizing an exclusively autonomic system with all the
properties that it would require.

There is an important difference between "automatic" and "autonomic".
"Automatic" refers to a predefined process, such as a script.
"Autonomic" is used in the context of self-management. It includes
feedback loops between elements as well as northbound to central
elements. See also the definitions in the next section. Generally,
an automatic process works in a given environment but has to be
adapted if the environment changes. An autonomic process can adapt
to changing environments.

This document provides the definitions and design goals for Autonomic
Networking in the IETF and IRTF. It represents the consensus of the
IRTF's Network Management Research Group (NMRG).

2. Definitions

We make the following definitions.

Autonomic: Self-managing (self-configuring, self-protecting, self-
healing, self-optimizing); however, allowing high-level guidance by a
central entity, through Intent (see below). An autonomic function
adapts on its own to a changing environment.

Automatic: A process that occurs without human intervention, with
step-by-step execution of rules. However, it relies on humans
defining the sequence of rules, so is not Autonomic in the full
sense. For example, a start-up script is automatic but not
autonomic. An automatic function may need manual adjustments if the
environment changes.

Intent: An abstract, high-level policy used to operate the network.
Its scope is an autonomic domain, such as an enterprise network. It
does not contain configuration or information for a specific node
(see Section 3.2 on how Intent coexists with alternative management
paradigms). It may contain information pertaining to a node with a
specific role (for example, an edge switch) or a node running a
specific function. Intent is typically defined and provided by a
central entity.

Autonomic Domain: A collection of autonomic nodes that instantiate
the same Intent.

Autonomic Function: A feature or function that requires no
configuration and can derive all required information through self-
knowledge, discovery, or Intent.

Autonomic Service Agent: An agent implemented on an autonomic node
that implements an autonomic function, either in part (in the case of
a distributed function) or whole.

Autonomic Node: A node that employs exclusively autonomic functions.
It requires (!) no configuration. (Note that configuration can be
used to override an autonomic function. See Section 3.2 for more
details.) An Autonomic Node may operate on any layer of the
networking stack. Examples are routers, switches, personal
computers, call managers, etc.

Autonomic Network: A network containing exclusively autonomic nodes.
It may contain one or several autonomic domains.

3. Design Goals

   This section explains the high-level goals of Autonomic Networking,
   independent of any specific solutions.

3.1. Self-Management

   The original design goals of autonomic systems as described in
   [Kephart] also apply to Autonomic Networks. The overarching goal is
   self-management, which is comprised of several "self" properties.
   The most commonly cited are:

   o Self-configuration: Functions do not require configuration, by
      either an administrator or a management system. They configure
      themselves, based on self-knowledge, discovery, and Intent.
      Discovery is the default way for an autonomic function to receive
      the information it needs to operate.

o Self-healing: Autonomic functions adapt on their own to changes in
   the environment and heal problems automatically.

o Self-optimizing: Autonomic functions automatically determine ways
   to optimize their behavior against a set of well-defined goals.

o Self-protection: Autonomic functions automatically secure
   themselves against potential attacks.

Almost any network can be described as "self-managing", as long as
the definition of "self" is large enough. For example, a well-
defined Software-Defined Networking (SDN) system, including the
controller elements, can be described overall as "autonomic", if the
controller provides an interface to the administrator that has the
same properties as mentioned above (high level, network-wide, etc.).

For the work in the IETF and IRTF, we define the "self" properties on
the node level. It is the design goal to make functions on network
nodes self-managing, in other words, minimally dependent on
management systems or controllers, as well as human operators. Self-
managing functions on a node might need to exchange information with
other nodes in order to achieve this design goal.

As mentioned in the introduction, closed-loop control is an important
aspect of self-managing systems. This implies peer-to-peer dialogues
between the parties that make up the closed loop. Such dialogues
require two-way "discussion" or "negotiation" between each pair or
groups of peers involved in the loop, so they cannot readily use
typical top-down command-response protocols. Also, a discovery phase
is unavoidable before such closed-loop control can take place.
Multiparty protocols are also possible but can be significantly more
complex.

3.2. Coexistence with Traditional Management

For the foreseeable future, autonomic nodes and networks will be the
exception; autonomic behavior will initially be defined function by
function. Therefore, coexistence with other network management
paradigms has to be considered. Examples are management by command
line, SNMP, SDN (with related APIs), the Network Configuration
Protocol (NETCONF), etc.

Conflict resolution between a) autonomic default behavior and Intent
and b) other methods is therefore required. This is achieved through
prioritization. Generally, autonomic mechanisms define a network-
wide behavior, whereas the alternative methods are typically on a
node-by-node basis. Node-based management concepts take a higher
priority over autonomic methods. This is in line with current

examples of autonomic functions; for example, with routing, a
(statically configured) route has priority over the routing
algorithm. In short:

o lowest priority: autonomic default behavior

o medium priority: autonomic Intent

o highest priority: node-specific network management concepts, such
   as command line, SNMP, SDN, NETCONF, etc. How these concepts are
   prioritized is outside the scope of this document.

The above prioritization essentially results in the actions of the
human administrator always being able to overrule autonomic behavior.
This is generally the expectation of network operators today and
therefore remains a design principle here. In critical systems, such
as atomic power plants, sometimes the opposite philosophy is used:
The expectation is that a well-defined algorithm is more reliable
than a human operator, especially in rare exception cases.
Networking generally does not follow this philosophy yet. However,
warnings should be issued if node-specific overrides may conflict
with autonomic behavior.

In other fields, autonomic mechanisms disengage automatically if
certain conditions occur: The autopilot in a plane switches off if
the plane is outside a predefined envelope of flight parameters. The
assumption is that the algorithms only work correctly if the input
values are in expected ranges. However, some opinions suggest that
exactly in exceptional conditions is the worst moment to switch off
autonomic behavior, since the pilots have no full understanding of
the situation at this point and may be under high levels of stress.
For this reason, we suggest here to NOT generally disable autonomic
functions if they encounter unexpected conditions, because it is
expected that this adds another level of unpredictability in
networks, when the situation may already be hard to understand.

3.3. Secure by Default

All autonomic interactions should be secure by default. This
requires that any member of an autonomic domain can assert its
membership using a domain identity, for example, a certificate issued
by a domain certification authority. This domain identity is used
for nodes to learn about their neighboring nodes, to determine the
boundaries of the domain, and to cryptographically secure
interactions within the domain. Nodes from different domains can
also mutually verify their identity and secure interactions as long
as they have a mutually respected trust anchor.

A strong, cryptographically verifiable domain identity is a
fundamental cornerstone in Autonomic Networking. It can be leveraged
to secure all communications and thus allows automatic security
without traditional configuration, for example, preshared keys. See
also the document "Making The Internet Secure By Default" [Behringer]
for more information.

Autonomic functions must be able to adapt their behavior depending on
the domain of the node they are interacting with.

3.4. Decentralization and Distribution

The goal of Autonomic Networking is to minimize dependencies on
central elements; therefore, decentralization and distribution are
fundamental to the concept. If a problem can be solved in a
distributed manner, it should not be centralized.

In certain cases, it is today operationally preferable to keep a
central repository of information, for example, a user database on an
Authentication, Authorization, and Accounting (AAA) server. An
Autonomic Network should be able to use such central systems, in
order to be deployable. It is possible to distribute such databases
as well, and such efforts should be at least considered. Depending
on the case, distribution may not be simple replication but may
involve more complex interactions and organization.

3.5. Simplification of Autonomic Node Northbound Interfaces

Even in a decentralized solution, certain information flows with
central entities are required. Examples are high-level service
definitions, as well as network status requests, audit information,
logging, and aggregated reporting.

Therefore, nodes in an Autonomic Network require a northbound
interface. However, the design goal is to maintain this interface as
simple and high level as possible.

3.6. Abstraction

An administrator or autonomic management system interacts with an
Autonomic Network on a high level of abstraction. Intent is defined
at a level of abstraction that is much higher than that of typical
configuration parameters, for example, "optimize my network for
energy efficiency". Intent must not be used to convey low-level
commands or concepts, since those are on a different abstraction
level.

   For example, the administrator should not be exposed to the version
   of the IP protocol running in the network.

   Also on the reporting and feedback side, an Autonomic Network
   abstracts information and provides high-level messages such as "the
   link between node x and y is down" (possibly with an identifier for
   the specific link, in case of multiple links).

3.7. Autonomic Reporting

   An Autonomic Network, while minimizing the need for user
   intervention, still needs to provide users with visibility like in
   traditional networks. However, in an Autonomic Network, reporting
   should happen on a network-wide basis. Information about the network
   should be collected and aggregated by the network itself and
   presented in a consolidated fashion to the administrator.

   The layers of abstraction that are provided via Intent need to be
   supported for reporting functions as well, in order to give users an
   indication about the effectiveness of their Intent. For example, in
   order to assess how effective the network performs with regards to
   the Intent "optimize my network for energy efficiency", the network
   should provide aggregate information about the number of ports that
   were able to be shut down, and the corresponding energy savings,
   while validating current service levels are, on aggregate, still met.

   Autonomic network events should concern the Autonomic Network as a
   whole, not individual systems in isolation. For example, the same
   failure symptom should not be reported from every system that
   observes it, but only once for the Autonomic Network as a whole.
   Ultimately, the Autonomic Network should support exception-based
   management, in which only events that truly require user attention
   actually cause the user to be notified. This requires capabilities
   that allow systems within the network to compare information and
   apply specific algorithms to determine what should be reported.

3.8. Common Autonomic Networking Infrastructure

   [RFC7576] points out that there are already a number of autonomic
   functions available today. However, they are largely independent,
   and each has its own methods and protocols to communicate, discover,
   define, and distribute policy, etc.

   The goal of the work on Autonomic Networking in the IETF is therefore
   not just to create autonomic functions but to define a common
   infrastructure that autonomic functions can use. This Autonomic
   Networking Infrastructure may contain common control and management

   functions such as messaging, service discovery, negotiation, Intent
   distribution, self-monitoring, and diagnostics, etc. A common
   approach to define and manage Intent is also required.

   Refer to the reference model below: All the components around the
   "Autonomic Service Agents" should be common components, such that the
   Autonomic Service Agents do not have to replicate common tasks
   individually.

3.9. Independence of Function and Layer

   Autonomic functions may reside on any layer in the networking stack.
   For example, Layer 2 switching today is already relatively autonomic
   in many environments, since most switches can be plugged together in
   many ways and will automatically build a simple Layer 2 topology.
   Routing functions run on a higher layer and can be autonomic on Layer
   3. Even application-layer functionality such as unified
   communications can be autonomic.

   "Autonomic" in the context of this framework is a property of a
   function that is implemented on a node. Autonomic functions can be
   implemented on any node type, for example, a switch, router, server,
   or call manager.

   An Autonomic Network requires an overall control plane for autonomic
   nodes to communicate. As in general IP networking, IP is the
   spanning layer that binds all those elements together; autonomic
   functions in the context of this framework should therefore operate
   at the IP layer. This concerns neighbor discovery protocols and
   other functions in the Autonomic Control Plane.

3.10. Full Life-Cycle Support

   An autonomic function does not depend on external input to operate;
   it needs to understand its current situation and surroundings and
   operate according to its current state. Therefore, an autonomic
   function must understand the full life cycle of the device it runs
   on, from manufacturing and initial testing through deployment,
   testing, troubleshooting, and decommissioning.

   The state of the life cycle of an autonomic node is reflected in a
   state model. The behavior of an autonomic function may be different
   for different deployment states.

4. Not among the Design Goals

   This section identifies various items that are explicitly not design
   goals in the IETF and IRTF for Autonomic Networks; they are mentioned
   to avoid misunderstandings of the general intention. They address
   some commonly expressed concerns from network administrators and
   architects.

4.1. Eliminate Human Operators

   Section 3.1 states that "It is the design goal to make functions
   [...] minimally dependent on [...] human operators". However, it is
   not a design goal to completely eliminate them. The problem targeted
   by Autonomic Networking is the error-prone and hard-to-scale model of
   individual configuration of network elements, traditionally by manual
   commands but today mainly by scripting and/or configuration
   management databases. This does not, however, imply the elimination
   of skilled human operators, who will still be needed for oversight,
   policy management, diagnosis, reaction to help-desk tickets, etc.
   The main impact on administrators should be less tedious detailed
   work and more high-level work. (They should become more like doctors
   than hospital orderlies.)

4.2. Eliminate Emergency Fixes

   However good the autonomous mechanisms, sometimes there will be fault
   conditions, etc., that they cannot deal with correctly. At that
   point, skilled operator interventions will be needed to correct or
   work around the problem. Hopefully, this can be done by high-level
   mechanisms (adapting the policy database in some way), but, in some
   cases, direct intervention at the device level may be unavoidable.
   This is obviously the case for hardware failures, even if the
   Autonomic Network has bypassed the fault for the time being. "Truck
   rolls" will not be eliminated when faulty equipment needs to be
   replaced. However, this may be less urgent if the autonomic system
   automatically reconfigures to minimize the operational impact.

4.3. Eliminate Central Control

   While it is a goal to simplify northbound interfaces (Section 3.5),
   it is not a goal to eliminate central control, but to allow it on a
   higher abstraction level. Senior management might fear loss of
   control of an Autonomic Network. In fact, this is no more likely
   than with a traditional network; the emphasis on automatically
   applying general policy and security rules might even provide more
   central control.

5. An Autonomic Reference Model

   An Autonomic Network consists of Autonomic Nodes. Those nodes
   communicate with each other through an Autonomic Control Plane that
   provides a robust and secure communications overlay. The Autonomic
   Control Plane is self-organizing and autonomic itself.

   An Autonomic Node contains various elements, such as autonomic
   service agents that implement autonomic functions. Figure 1 shows a
   reference model of an autonomic node. The elements and their
   interaction are:

   o Autonomic Service Agents: They implement the autonomic behavior of
      a specific service or function.

   o Self-knowledge: An autonomic node knows its own properties and
      capabilities

   o Network Knowledge (Discovery): An Autonomic Service Agent may
      require various discovery functions in the network, such as
      service discovery.

   o Feedback Loops: Control elements outside the node may interact
      with autonomic nodes through feedback loops.

   o An Autonomic User Agent, providing a front-end to external users
      (administrators and management applications) through which they
      can receive reports and monitor the Autonomic Network.

   o Autonomic Control Plane: Allows the node to communicate with other
      autonomic nodes. Autonomic functions such as Intent distribution,
      feedback loops, discovery mechanisms, etc., use the Autonomic
      Control Plane. The Autonomic Control Plane can run in-band, over
      a configured VPN, over a self-managing overlay network as
      described in [ACP], or over a traditional out-of-band network.
      Security is a requirement for the Autonomic Control Plane, which
      can be bootstrapped by a mechanism as described in [BOOTSTRAP].

```
+---------------------------------------------------------------+
|                        +-----------+                          |
|                        | Feedback  |                          |
|                        |   Loops   |                          |
|                        +-----------+                          |
|                              ^                                |
|                   Autonomic User Agent                        |
|                              V                                |
| +-----------+      +-----------+      +-----------+           |
| | Self-     |      | Autonomic |      | Network   |           |
| | knowledge |<----->| Service   |<----->| Knowledge |          |
| |           |      | Agents    |      | (Discovery)|          |
| +-----------+      +-----------+      +-----------+           |
|                         ^                  ^                  |
|                         |                  |                  |
|                         V                  V                  |
|---------------------------------------------------------------|
|                 Autonomic Control Plane                       |
|---------------------------------------------------------------|
|              Standard Operating System Functions              |
+---------------------------------------------------------------+
```
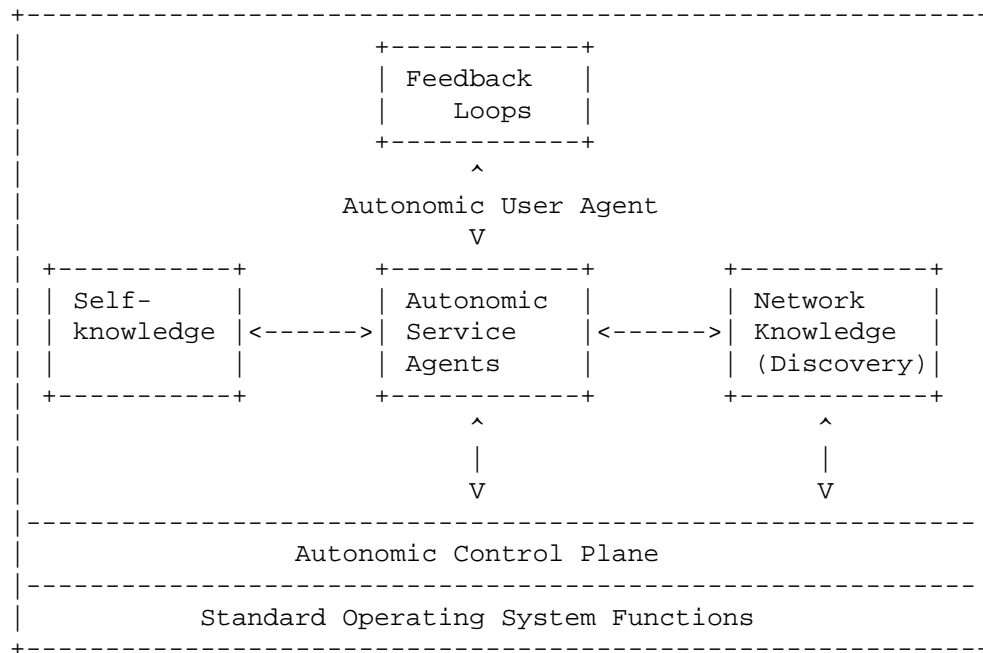
                Figure 1: Reference Model for an Autonomic Node

   At the time of finalizing this document, this reference model is
   being worked out in more detail. See [Reference-Model] for more
   details.

6. Security Considerations

   This document provides definitions and design goals for Autonomic
   Networking. A full threat analysis will be required as part of the
   development of solutions, taking account of potential attacks from
   within the network as well as from outside.

7. Informative References

   [ACP]   Behringer, M., Bjarnason, S., BL, B., and T. Eckert, "An
              Autonomic Control Plane", Work in Progress,
              draft-behringer-anima-autonomic-control-plane-02, March
              2015.

   [Behringer]
              Behringer, M., Pritikin, M., and S. Bjarnason, "Making The
              Internet Secure By Default", Work in Progress,
              draft-behringer-default-secure-00, January 2014.

   [BOOTSTRAP]
             Pritikin, M., Behringer, M., and S. Bjarnason,
             "Bootstrapping Key Infrastructures", Work in Progress,
             draft-pritikin-anima-bootstrapping-keyinfra-01, February
             2015.

   [Dobson]   Dobson, S., Denazis, S., Fernandez, A., Gaiti, D.,
             Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt,
             N., and F. Zambonelli, "A survey of autonomic
             communications", ACM Transactions on Autonomous and
             Adaptive Systems (TAAS), Volume 1, Issue 2, Pages 223-259,
             DOI 10.1145/1186778.1186782, December 2006.

   [GANA]   ETSI, "Autonomic network engineering for the self-managing
             Future Internet (AFI); Generic Autonomic Network
             Architecture (An Architectural Reference Model for
             Autonomic Networking, Cognitive Networking and Self-
             Management)", ETSI GS AFI 002, April 2013,
             <http://www.etsi.org/deliver/etsi_gs/
             AFI/001_099/002/01.01.01_60/gs_afi002v010101p.pdf>.

   [Kephart] Kephart, J. and D. Chess, "The Vision of Autonomic
             Computing", IEEE Computer, vol. 36, no. 1, pp. 41-50,
             DOI 10.1109/MC.2003.1160055, January 2003.

   [Movahedi] Movahedi, Z., Ayari, M., Langar, R., and G. Pujolle, "A
             Survey of Autonomic Network Architectures and Evaluation
             Criteria", IEEE Communications Surveys & Tutorials, Volume
             14, Issue 2, Pages 464-490,
             DOI 10.1109/SURV.2011.042711.00078, 2012.

   [Reference-Model]
             Behringer, M., Ed., Carpenter, B., Eckert, T., Ciavaglia,
             L., and B. Liu, "A Reference Model for Autonomic
             Networking", Work in Progress, draft-behringer-anima-
             reference-model-02, June 2015.

   [RFC7576] Jiang, S., Carpenter, B., and M. Behringer, "General Gap
             Analysis for Autonomic Networking", RFC 7576,
             DOI 10.17487/RFC7576, June 2015,
             <http://www.rfc-editor.org/info/rfc7576>.

   [Samaan]   Samaan, N. and A. Karmouch, "Towards Autonomic Network
             Management: an Analysis of Current and Future Research
             Directions", IEEE Communications Surveys and Tutorials,
             Volume 11, Issue 3, Page(s) 22-36,
             DOI 10.1109/SURV.2009.090303, 2009.

Acknowledgements

Authors' Addresses

   Michael H. Behringer
   Cisco Systems
   Building D, 45 Allee des Ormes
   Mougins 06250
   France

   EMail: mbehring@cisco.com

   Max Pritikin
   Cisco Systems
   5330 Airport Blvd
   Boulder, CO 80301
   United States

   EMail: pritikin@cisco.com

   Steinthor Bjarnason
   Cisco Systems
   Mail Stop LYS01/5
   Philip Pedersens vei 1
   LYSAKER, AKERSHUS 1366
   Norway

   EMail: sbjarnas@cisco.com

Alexander Clemm
Cisco Systems
170 West Tasman Drive
San Jose, CA 95134-1706
United States

EMail: alex@cisco.com

Brian Carpenter
Department of Computer Science
University of Auckland
PB 92019
Auckland 1142
New Zealand

EMail: brian.e.carpenter@gmail.com

Sheng Jiang
Huawei Technologies Co., Ltd
Q14, Huawei Campus
No.156 Beiqing Road
Hai-Dian District, Beijing 100095
China

EMail: jiangsheng@huawei.com

Laurent Ciavaglia
Alcatel Lucent
Route de Villejust
Nozay 91620
France

EMail: laurent.ciavaglia@alcatel-lucent.com