Evaluating 6lowPAN implementations in WSNs

Ricardo Silva, Jorge Sá Silva and Fernando Boavida
Department of Informatics Engineering
University of Coimbra
Pólo II - Pinhal de Marrocos, 3030-290 Coimbra, PORTUGAL
rnsilva@dei.uc.pt, sasilva@dei.uc.pt, boavida@dei.uc.pt

***Abstract*— Wireless sensor networks have been the subject of intensive research over the last years. Although the concept is now well defined, there are still some important topics that must be discussed.**

Over the recent years, 6lowPAN has been proposing the use of IPv6 solutions in WSNs. IP in WSNs has been accepted not only as an extension from the conventional networks to the real world, but also as a solution to use well-known high-level protocols in WSNs, such as UDP or HTTP. Created to adapt IPv6 long packets to IEEE802.15.4 short frames, 6lowPAN has currently three well-known implementations: 6lowpancli, b6lowpan and SICSlowPAN.

The objective of this paper is to analyze the actual panorama of 6lowPAN and evaluate the existent implementations, regarding energy consumption.

***Index Terms*— 6lowPAN, energy consumption, WSN**

I. Introduction

Wireless Sensor Networks (WSNs) are one of the most important technologies in 21st century. Most of the researchers and technological analysts believe that, in the near future, these microsensors will be in everywhere: in our homes, factories, bodies, animals, cars or rivers. However there are several challenges and problems that must be overcome like quality of service, mobility and integration with Internet.

WSNs, independently of all the recognized potential, remain limited in the access. Traditional WSN protocols do not provide the flexibility and interoperability that WSNs demand. As a result, to allow the communication between conventional networks and WSNs, it is necessary to provide specific and complex gateways.

Nevertheless, the integration of IP directly in sensor nodes (or in some of them) will offer several advantages, which can be useful for some applications. Integrating IP in WSNs allows any device connected to the Internet to communicate directly with a specific sensor. Although for most of the cases, it is not necessary to communicate with this specific sensor node but with a group of sensor nodes that has the required information (data-centric approach), there will be some situations in the future that the communication to that specific sensor will be necessary, for instance for maintenance purposes. Integrating IP in WSNs also offers a transparent and an easy approach for developing new generation applications. According to 4th Generation Networks there will be a wide variety of terminals and devices with different capabilities. Although IP has a lot of limitations, it is the common "language" that every device recognizes.

IP in WSNs is still a hot topic because IP is considered energy prohibited to use in WSNs. However integrating IP in WSNs is not only a problem of energy. There are significant differences that it is necessary to combine if we really want to integrate them. As described before, IP networks are address centric while routing in WSNs is data centric. Bandwidth is the most important constrain in IP while in WSN is the energy. Data rate is high in IP networks and low in WSNs. The lifetime of the sensor nodes is low, while such problem does not exist in IP networks.

To integrate IP in WSNs is also necessary to reduce the complexity and the header length of IP as WSNs present low memory and limited processing capabilities.

Another important requirement is to provide enough addresses, which means that more than an ordinary IP support IPv6 is required. Besides that, there are other important advantages of IPv6, which can be valuable to WSNs. Mobility, anycast, security and self-configuration, are examples of IPv6 properties that are not native in IPv4.

Based on such requirements and on such benefits, a WG from IETF called 6lowPAN was created, whose main objective was the integration of IPv6 in lowPANs. Since its formation, several approaches have been proposed as well as some standards (RFCs) and real implementations.

The objective of this paper is to study 6lowPAN and evaluate their most well-known implementations.

Hence, the remaining of this paper is organized as follow: the next section introduces the 6lowPAN from the theoretical point of view; the third section presents the most well known implementations of 6lowPAN, stressing the main differences among them; section 4 evaluates and compares the implementations presented in section 3; and finally, section 5 concludes the paper.

II. 6LOWPAN

A. The adaptation model

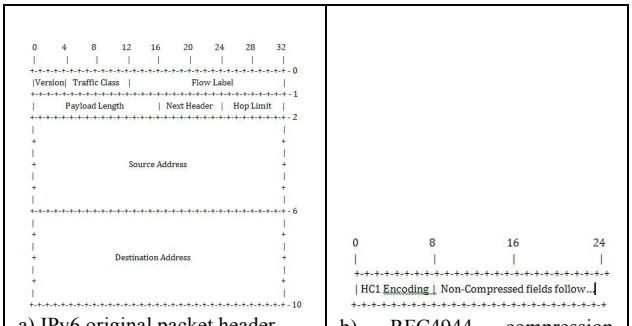
6lowPAN [1] is an IETF working group, whose principal objective is to develop an energy aware adaptation model for the integration of IPv6 packets[2] over Low Power Wireless Personal Area Networks (LowPANs) supported by the MAC Layer protocol IEEE802.15.4 [3].

6lowPAN main work items are:

- IP adaptation/Packet Formats and interoperability.
- Addressing schemes and address management.
- Network management.
- Routing in dynamically adaptive topologies.
- Security, including set-up and maintenance.
- Application programming interface.
- Discovery (of devices, services, etc).
- Implementation considerations.

Based on the list above, two RFCs were released, the RFC 4919 [4] and the RFC4944 [5]. The first presents an overview, defining assumptions, identifying the problems and establishing the goals necessary to support IPv6 over lowPANs. The second RFC approaches the hot topic of 6lowPAN: the transmission of IPV6 Packets over IEEE 802.15.4. This RFC defines a LowPAN adaptation layer and frame format, a fragmentation and reassembly mechanism and an addressing scheme, where nodes can generate the IPv6 link local address from the EUI64bits or 16bits MAC address. The same RFC defines also the support for mesh networks, which means that multi-hop can be, or not, transparent to IP layer.

The MTU, Maximum Transmission Unit, of an IPv6 packet, is 1280 octets, where 40 of those octets belong to the packet header. Considering the 102 maximum octets of IEEE802.15.4 MAC Layer, 40 bytes would be an enormous overhead. Therefore, the solution found was first compress the header and after, in the case of the packet size overcomes the 102 bytes, fragment it in several packets. The fragmentation, in turn, requires the reassembly of all packets' parts at the receiver side. The IPv6 packet header compression can reduce the header size from 40 to 2 octets. Figure 1 presents: a) the IPv6 original header and in b) the maximum compressed header.



a) IPv6 original packet header. b) RFC4944 compression mode.

Fig. 1 Comparasion between the IPv6 packet header and the 6lowPAN compressed header.

To reduce the size in 2 octets, RFC4944 defined that: the version would be always IPv6 therefore it could be omitted; the traffic class and the flow label would be both equal to zero and consequently omitted too; the packet length would be inherited from MAC Layer or from the field "datagram_size" when fragmentation was present; the next header identification since it was limited to ICMPv6, UDP and TCP, could be represented in 2 bits; Hop limit would remained in the same format; and both addresses would be considered as link local and consequently inherited from MAC layer. Consequently, one octet became enough to identify: IPv6 source address (bits 0 and 1); IPv6 destination address (bits 2 and 3); Traffic class and Flow Label (bits 4); Next header (bit 5 and 6); HC2 encoding (bit 7). The second octet is necessary for the Hop Count field, performing the minimum size. HC2 identifies whether the next header is also compressed or not. Each address requires 2 bits, allowing 4 combinations to identify whether the Address Prefix and the Interface Identifier are carried in line or not.

RFC4944 has been pointed as deprecated due to the weaknesses of the compression algorithm [6]. In terms of header length, 2 octets are considered a good result. However, as explained before, to achieve such length, both parts (Prefix + Interface Identifier) of both link local addresses (Source and Destination) must derive directly from the IEEE802.15.4 MAC Layer. Consequently, since it is not usual for applications to use link local addresses, and considering that for route-over and for connection to the exterior the global address is required, 2 octets compression can only be used in limited scenarios. Hence, new solutions have been proposed [6] in order to establish a different compression mechanism. However, at this moment RFC4944 continues the unique standard.

B. Additional IPv6 feature for lowPANs

The support of IPv6 in lowPANs is not only justified by a seamless communication with conventional networks, but also by the support of well-known protocols. Such protocols can be higher-level protocols (UDP, TCP, HTTP, etc.), protocols that are included in the original IPv6 (ND, DHCPv6, etc) or protocols that are extensions of the own IPv6 (MIPv6, etc.).

In the context of 6lowPAN, some drafts aiming to adapt some of those protocols to lowPANs were also proposed.

Neighbor Discovery (ND) has been the most approached, so that several drafts have been released trying to identify and propose an acceptable solution. Neighbor Discovery is one mechanism where routers periodically broadcast Router Advertisements (RA) and nodes multicast Router Solicitation (RS) on deployment or Neighbor Solicitation (NS) to get any neighbor address. These messages share a problem in WSN: all are broadcasts at the MAC layer. Therefore, the network is constantly flooded with broadcasts, which will consume extra energy and decrease nodes lifetime.

[7] proposes the suppressing of the Neighbor Discovery Router Advertisements. This draft applies to the unnecessary knowledge of the global address by the sensor node. Nodes should be configured with the previously defined anycast L2 address, which would be substituted whether the node

received any Router Advertisement (RA).

In [8] a new version of Neighbor Discovery route-over 6lowPAN Networks is described. This draft introduces the concept of Router Edges (RE) sharing a common backbone, by which Neighbor Solicitations or Duplicate Address Detection can move to all connected lowPANs. It allows the use of stateless address assignment, Neighbor Discovery proxy (to allow the interoperability among all lowPANs) and optimization of the Router Advertisements.

Besides the ND, MIPv6 have been approached as well. [9] is an expired draft about mobility in 6lowPAN that identified the mobility scenarios, the main challenges and the security issues. [10], the latest draft about mobility, proposes an adaption model for the original MIPv6. Based on the same mode that RFC4944 compresses the IPv6 header, [10] proposes compressed version for the MIPv6 header, all mobility messages and all mobility options.

Also, the SNMP has been approached within the 6lowPAN WG [11], identifying only the requirements.

Besides all the work done within the scope of the IETF, some 6lowPAN implementations have appeared. The next section presents the most well known.

III. Implementations

Currently there are three well-known implementations of 6lowPAN. Two were developed for the operating system TinyOS-2.x and the other one was developed for ContikiOS.

The first implementation, also known as 6lowpancli, was released in 2007 as a result of an MSc thesis [12], being now integrated as a native application in TinyOS-2.x. 6lowpancli supports header compression (accomplishing with the HC1 format), fragmentation and addressing, as well as IPv6 Stateless Configuration. Such Stateless Configuration requires the previous manual setting of the network address prefix. Regarding high-level protocols, 6lowpancli supports UDP, accomplishing with the HC_UDP format and ICMPv6 messages. Hence, it is possible to ping motes and use UDP sockets from any Internet point to a small sensor node.

To implement the bridge between the WSN and the conventional network, the same implementation provides a daemon to emulate an IPv6 tunnel over IEEE802.15.4, allowing the communication, through, for instance, the USB port of any laptop.

Therefore, 6lowpancli is completely static requiring manual configuration. It doesn't support any type of Node discovery mechanism, or mobility, like MIPv6. Besides, the support for mesh network is not provided and when a packet with different destination address is received, it is just dropped.

B6lowpan[13], now renamed to *blip*, was the second released implementation to TinyOS-2.x. The code is not included in the TinyOS-2.x core, but can be found in the contributions from Berkeley. Implementing the basic features defined in RFC4944, namely header compression, fragmentation and addressing, b6lowpan also supports ICMPv6 and UDP packet. Besides that, the last version introduced the first prototype of a TCP stack.

B6lowpan, or blip, implements Neighbor Discovery, in a light version [sensys08], configuring a link local address on nodes' boot and a global address if a Router Advertisement is received.

As 6lowpancli, b6lowpan includes a daemon to create an IPv6 tunnel to connect conventional networks and WSNs. As part of the Sink Node application, the daemon can run in any Unix based machine. Since the last released version (20/3/2009) that ND is included in the daemon. Before that, radvd was required to run separately.

Unlike 6lowpancli, b6lowpan supports mesh networks, which as defined by RFC4944 are known as "mesh under". Mesh under means that from the IP level perspective all nodes are one-hop distance, being that multi-hop is only performed by IEEE802.15.4 (MAC layer).

SicslowPAN [14] is the first implementation released for ContikiOS. SicslowPAN is based on RFC4944, implementing header compression, fragmentation and addressing. Besides that, it goes further and implements interoperability support based on [15] and new header compression mechanisms defined in [16].

SICSlowPan does not implement mesh under.

Moreover SICSlowPAN is located between the uIPv6 layer and the MAC Layer. Therefore, on one hand, when the MAC layer receives an IPv6 packet, it calls the SICSlowPan to adapt the packet from the MAC to the IP layer. On the other hand, when the uIPv6 needs to send a packet, it also uses the SICSlowPan layer to adapt the packet.

ContikiOS implements by default a non-IP protocol called RIME at the MAC layer, a uIP stack implementing IPv4 and recently a uIPv6 stack. Theoretically, the uIPv6 is independent of any layer below [16], being possible to run over 802.15.4, 802.11 or 6lowPAN. However, over IEEE802.15.4, to fully support the IPv6 integration, SICSlowPAN is required. The uIPv6 supports ICMPv6, implements Neighbor Discovery and supports UDP and TCP.

Although, ContikiOS implements IPv6 and 6lowPAN as completely different platforms, the 6lowPAN WG is currently working on issues that are included in the uIPv6 (eg. ND), but not in the SICSlowPAN. By the same reason, b6lowPAN is now called blip, justifying that it is more than 6lowPAN. The next table summarizes the three implementations.

TABLE 1 Summary of the three analyzed implementations.

	6lowpancli	**B6lowpan**	**Sicslowpan**
OS	TinyOS-2.x	TinyOS-2.x	ContikiOS
ICMPv6	YES	YES	YES
UDP	YES	YES	YES
TCP	NO	Prototype	YES
ND	NO	YES	YES
Mesh under	NO	YES	NO
Route over	NO	YES	NO
MIPv6	NO	NO	NO

As it is possible to conclude, 6lowpancli is the simplest

implementation supporting only the basic ICMPv6 and UDP. On one hand it means that for developers it can be a good platform to work in. However, on the other hand, it can be really poor for final users, and/or quite difficult to be optimized. B6lowPAN, or Blip, is the most complete solution, implementing not only the RFC4944 but also a light Neighbor Discovery mechanism to automate the network. SICSlowPAN has the advantage of supporting TCP connections. However TCP is an undefined issue, considered too complex for WSNs. 6lowPAN WG has not approached TCP support. Besides, SICSlowPAN does not support mesh under, but according to the authors, new route over techniques will overcome the issues of routing/forwarding.

The next section evaluates the performance of these three implementations considering ROM, RAM, time and energy required per packet's size.

IV. Evaluation

In order to evaluate the performance and efficiency of the three presented implementations, 6lowpancli, b6lowpan and SICSlowPAN, practical tests were performed in our lab. Changing the data length of an UDP packet, from 0 to 1024 bytes, we a) measured the time required to send the message and b) calculated the energy spent based on the current and voltage required for the used hardware.

The hardware used was the mote telosB from Crossbow, equipped with a MSP430 microprocessor.

To measure the time needed, we used native functions from the respective operating systems (TinyOS or Contiki), to compute the difference between the time when the UDP packet was created, until the time it was completely sent by the MAC Layer.

Hence, considering the three implementations: 6lowpancli, b6lowpan and SICSlowPAN; and UDP packets with data length of 0, 16, 32, 64, 128, 256, 512, 768 and 1024; the time and energy required is outlined by figures 2 and 3, respectively. The time values presented in figure 2 are the average of thirty measurements, with a confident interval of 95%.

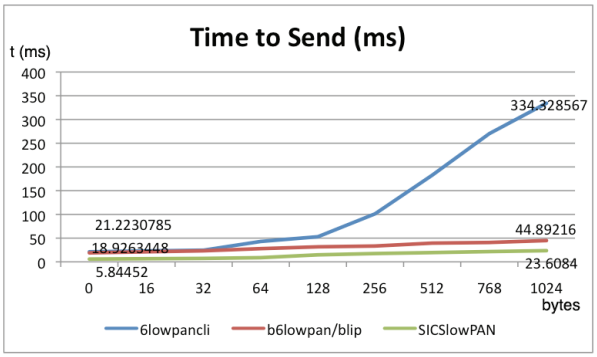


Fig. 2 Time needed to send an UDP packet per amount of data.

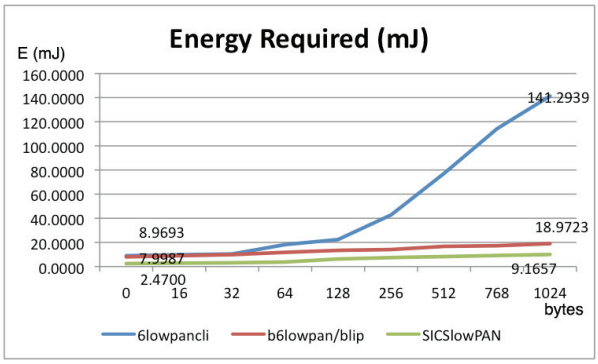


Fig. 3 Energy needed to send an UDP packet per amount of data.

It is possible to conclude that the energy required is proportional to the time spent. Considering that the telosB in our tests required 18.7mA (measured using a multimeter in series) to send with a voltage of 2.6Volts, the energy in joules is given by:

$E = t \times 18.7 \times 22.6$, where t is the time measured.

According to the results SICSlowPAN is the implementation that requires less time and energy. Although, the newest implementation for TinyOS-2.x, b6lowPAN is really close to the consumptions of SICSlowPAN, the 6lowpancli was somewhat below. In fact, considering the maximum MTU of 102bytes in IEEE802.15.4, we can observe that the performance of 6lowpancli started to decrease when fragmentation was required. It is possible to see this behavior at the 128bytes of data. From that value until the end, the time/energy increases significantly when compared with the other solutions. Under the same conditions, b6lowpan and SICSlowPAN, also increased the required time, but in a constant and controlled mode.

To better analyze the two best solutions, the next figure presents the correlation between them, based on the Pearson coefficient.

		SICSlowPAN	b6lowPAN
SICSlowPAN	Pearson Correlation	1.000	.972
	Sig. (2-tailed)		.000
	N	8.000	8
b6lowPAN	Pearson Correlation	.972	1.000
	Sig. (2-tailed)	.000	
	N	8	8.000

Fig. 4 Pearson correlation between SICSlowPAN and b6lowPAN.

As we can see in figure 4, the correlation between these two cases is very significant, with a value of 0.972, which means a positive linear association. Based on this value we prove that the performance of these two solutions is almost the same. Although, SICSlowPAN requires less energy than b6lowPAN, it does not scale better. We can conclude that SICSlowPAN, in general, is more energetically efficient. However, considering packet size, b6lowPAN performs as well as SICSlowPAN.

We also evaluated the required ROM and RAM for each case. Figures 5 and 6 outline the results.

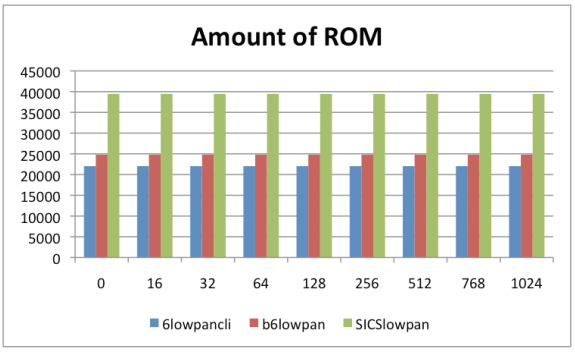


Fig. 5 Amount of ROM required per data length.

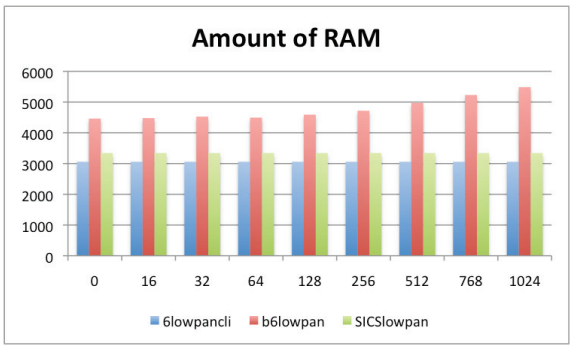


Fig. 6 Amount of RAM per data length.

As we can observe in figure 5, there is no difference between the amounts of ROM, when we change the packet size. However, the requirements of each implementation is significant, being SICSlowPAN, the one that requires more ROM. Regarding RAM requirements (figure 6), whereas SICSlowPAN and 6lowpancli required always the same amount independently of the packet size, b6lowPAN increased the RAM requirements each time the packet size increased.

Hence, based on ROM and RAM, 6lowpancli is the most efficient implementation. B6lowPAN does not scale well considering RAM, but requires less ROM than SICSlowPAN.

V. Conclusion

Although the IP support in WSNs is nowadays a reality, the application in real scenarios remains untrusted and therefore inexistent. The IP protocol, specifically IPv6, allows the use of well-known and well-tested high-level protocols. Those protocols could be considered as energetically undesirable, in part due to the enormous packets size. However, in this paper we evaluated the existent 6lowPAN implementations for different packet sizes.

We concluded that for all implementations, the packet size and, consequently the need of fragmentation, increased the required energy. However, the novelty is that for b6lowPAN and SICSlowPAN such increase is not significant, being the impact smooth and controlled. Regarding RAM and ROM, the main problem of these two solutions is that both can be prohibitive for some existent devices. Therefore, based on our evaluation results and crossing them with the data of Table 1, we can affirm that nowadays there are already some efficient solutions to support IPv6 both in TinyOS and in ContikiOS.

References

[1] C. Bormann and G. Mulligan, “IPv6 over low power wpan (6lowpan),” August 2008, http://www.ietf.org/html.charters/6lowpan-charter.html

[2] R. H. S. Deering, Nokia and Nokia, “Internet Protocol, version 6,” RFC 2460, Dec. 1998.

[3] IEEE, “IEEE standard for information technology-telecommunications and information exchange between systems- local and metropolitan area networks- specific requirements – wireless medium access control (mac) and physical layer (phy) specifications for wpan and lr-wpan (lowpan),” August 2007, http://standards.ieee.org/getieee802/802.15.html

[4] N. Kushalnagar, G. Montenegro, and C. Shumacher, “IPv6 over lowpower wireless personal A´ rea networks (6lowpans),” RFC 4919, Aug. 2004.

[5] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, “Transmission of IPv6 packets over ieee802.15.4 networks,” RFC 4944, Sep. 2007.

[6] J. Hui and P.Thubert, “Compression Format for IPv6 Datagrams in 6lowPAN Networks” draft-ietf-6lowpan-hc-04, 2008

[7] L. Toutain, G. Chelius, Y. Lee, and Y. Dong, “Neighbor discovery suppression draft-toutain-6lowpan-ra-suppression-00.txt,” 2008.

[8] Z. Shelby, P. Thubert, J. Hui, S. Chakrabarti, and E. Nordmark, “Neighbor discovery for 6lowpan draft-ietf-6lowpan-nd-03,” 2009.

[9] Mulligan, Williams, and D. Huo, “6lowpan architectural consideration for mobility” draft-williams-6lowpan-mob-01.txt,” 2008.

[10] R. Silva and J. Sa Silva, “An Adaptation Model for MIPv6 support in lowPAN” draft-silva-6lowpan-ipv6-00.txt”, 2009.

[11] M. Harvan, J. Schönwälder: *A 6lowpan Implementation for TinyOS 2.0*.6th GI/ITG KuVS Fachgespräch "Wireless Sensor Networks", Aachen, July 2007.

[12] http://smote.cs.berkeley.edu:8000/tracenv/wiki/blip

[13] Hui, J. W. and Culler, D. E. 2008. IP is dead, long live IP for wireless sensor networks. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems* (Raleigh, NC, USA, November 05 - 07, 2008). SenSys '08. ACM, New York, NY, 15-28. DOI= http://doi.acm.org/10.1145/1460412.1460415

[14] http://www.sics.se/projects/sicslowpan

[15] draft-hui-6lowpan-interop-00 Interoperability Test for 6LoWPAN

[16] draft-hui-6lowpan-hc-01 Compression format for IPv6datagrams in 6lowpan Networks

[17] Durvy, M., Abeillé, J., Wetterwald, P., O'Flynn, C., Leverett, B., Gnoske, E., Vidales, M., Mulligan, G., Tsiftes, N., Finne, N., and Dunkels, A. 2008. Making sensor networks IPv6 ready. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems* (Raleigh, NC, USA, November 05 - 07, 2008). SenSys '08. ACM, New York, NY, 421-422. DOI= http://doi.acm.org/10.1145/1460412.1460483