

2: مازولیشن LoRa توسط Semtech به صورت Close Source درست شده و جزئیات کمی ازش برونه. مجموعه ای از شرکت ها با منافع مشترک برای درست کردن LoRaWAN هستند و به صورت Open Source LoRa Alliance هستند.

برای این که تفاوت بین MAC و PHY Modulation قرار بگیره ما 2 تا term داریم:

- لایه فیزیکی: LoRa

- استانداردسازی LoRaWAN: یعنی کل مجموعه MAC و لایه فیزیکی که توسط LoRa Alliance خریدش.

3: خود LoRa (لایه فیزیکی) یک از مهم ترین سوالش اینه که مازولیشن اش به چه صورته. ابتدا توسط یک شرکت فرانسوی (Cycleo) درست شد سپس Semtech خریدش.

5: از نظر Urban area ما معمولاً 2 تا 5 کیلومتر رو میتوانیم پوشش بدیم. در Rural Area 5 تا 15 کیلومتر و جایی که خارج از شهر باشیم مثل جاده تهران قم بالاتر از 15 کیلومتر هم میتوانیم ارتباط مبتنی بر LoRa داشته باشیم. با فرض 2 کیلومتر پوشش حدود 56 تا Access Point برای پوشش تهران نیاز داریم. سوالی که باید جواب بدیم اینه که این پوشش زیاد از کجا میاد؟

6: application node های ما با ارتباط بی سیم با یک سری gateway در ارتباط و gateway ها با سرور های Lora و اونم با Server

7: چهار کامپوننت مهم:

- End Device: Actuator هایی که استفاده میکنیم

- Base station: همان

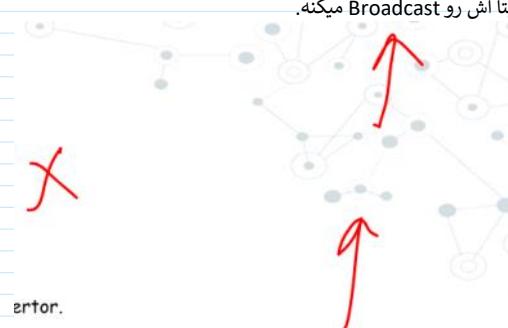
- Lora Server

- Application Server: تصمیمات های هوشمند رو میگیره و میده به End Device

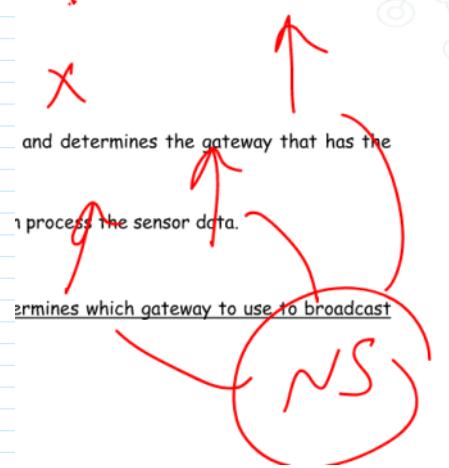
10: معمولاً End node ها با باتری کار میکنند ولی gateway ها به برق شهر وصل میشن. بنابراین مهمه ببینیم در جزئیات LoRaWan چطوری Low Power شده.

11: توپولوژی Star of Star جلوتر توضیح داده میشه هم مهندس بابک جلوتر میگه SS Chirp با توجه رنج بالایی که داریم یکی از هزینه هایی که باید بدیم اینه که دیتا ریتش کم میشه.

12: در شبکه های LoRaWAN ما پیچیدگی رو از سمت End device به سمت Gateway میگیرد. سعی میکنیم ما کمترین پیچیدگی رو داشته باشیم و کمترین وظیفه رو داشته باشیم. هر node ای دیتا اش رو Broadcast میکنه.



یعنی این Node پیامی که میده برای هر 2 تا base station میره. اما پکیشون وظیفه decode کردن و ارسال به core assign میشنیم به یک دیگه (Handover). میگیم در شبکه LoRaWAN چنین اتفاق نمیافته. یعنی end device associate با base station نمیشه به broadcast. یک پیام رو broadcast میکنه و همه میشنون. این gateway ها یدونه نیستن. همچون demodulate میکنن و میفرستن به network server (base station unicast). وقتی broadcast میکنیم ممکنه که:



یک پیام رو از 3 تا gateway مختلف مشاهده کنند. پس یکی از کاراش اینه که متوجه بشه کدومش بدون خطا است و بقیه رو حذف کنند. و اون gateway که بهترین گیرنده بوده رو داخل دیتابیس خودش بذاره.

برای شبیه سازی اون گره ما چشمشو میبنده یه packet ای رو ارسال میکنه (مثلا هوا فلان درجس). هر 3 تا gateway دریافت میکنند و هر کدام میگن با چه شدتی دریافت کردن. (برحسب dB) بعد Network server همشو میشنو و gateway ای که best reception داره رو انتخاب میکنه.

سپس این packet فروارد میشه به Application Server مناسب. حالا اگر Application Server یک تصمیم گرفت مثلا این که شیر آب باز بشه. تا الان این ارتباط رو Uplink میگفتیم. حالا که میخوایم اطلاعات رو از Application Server بفرستیم که Downlink هستش رو NS بفرسته به Gateway. آیا به هر سه تاش میده؟ نه. میده به best reception اگه به هر 3 تاش میداد جه اشکال پیش میمود؟ اون موقع 3 end node 3 تا دریافت میکرد که باشد مشخص رو قبول کنه و پیچیدگی محاسباتی میرفت بالا

یک از مهم ترین پیچیدگی که حذف شده اینه که عملیات Association نمیشه درست کرد. این که user بیاد base station خودش رو تشخیص بده

13: مشاهده میکنیم ارتباط gateway با NS به صورت بی سیم میتوانه بشه. حتی با 3G یا 4G چون به برق شهر وصلن و Low power بودن اهمیت نداره.

14: پس دیدیم که Star هستش ولی هر node به gateway موردنظر Associate نشده. Key feature اش اینه که handover دیگه نیازی نداریم. فایده دیگش خطای کمتره چون چندتا gateway گوش میدن

17: از نظر پروتکل ما در لایه فیزیکی End device و Gateway و NS داریم.
از Gateway به End device ارتباطات توسعه LoRa هستش.
همونطور که گفته شد NS تا Gateway تا ارتباطات همه چی میشه باشه.

19: Water level monitoring سطح آب و آلدگی آب در یک رودخونه رو مشخص میکنه. این مانیتورینگ لازم نیست لحظه ای باشه بلکه هرجند ساعت به بارگزارش کنه کافیه. میشه تنظیم هم کرد وقی تغییر 2 درصدی داشت ارسال کنه
يعني هرموقع اطراف نزدیک بودن چراخ روشن بشه. Smart lighting

27: LoRaWAN: از نظر نسخه 1.0.2 روی SubGHZ هستش چون میخواد هم Low power باشه هم long range.

28: مثلا حول 863 تا 870 ما 8 تا کanal فرکانسی داریم. این کanal هر کدامش یک پهنانی باندی دارن که میتوان 125 تا 500 کیلوهرتزی باشند. بوزر میتوانه هر کدام را برای ارسال انتخاب کنه

29: در شبکه های LoRaWAN یک تابیه تحت عنوان Hop time داریم که زمانی هست که بین تعویض کanal تلف میشه. چرا باید اصن بین کanal ها تعویض کنیم؟ فرکانس متفاوت میشه و ممکنه در یک کanal 10 تا یوزر باشن تداخل زیاد باشه و با تعویض میریم یه جا خلوت تر. تایم Dwell time زمانی هست که برای ارسال داخل یک کanal نیاز داریم. تو شکل میبنیم 2 تا فرکانس هاشون چطوری فرق میکنه.

30: بحث Duty Cycle: در باند های ISM خیلی مهمه (خاص LoRa نیست). چی هست؟ وقتی در آنالاینس کار میکنیم درسته مجوز عمومی صادر شده و پول نمیدیم براش. ولی جای بدون قانونی هم نیست. رگولیشن داره:

- باید Transmit power محدود داشته باشه
- مداوم نمیشه ارسال کرد و باید یه میزان صبر کرد (Duty Cycle)

وقتی میگیم Duty Cycle اش 1 درصد هست یعنی از 100 واحد زمانی 99 واحدش رو باید خاموش باشیم و 1 اش رو روشن باشیم و حق ارسال داریم.

31: بهترین واحدی که برای واحد زمانی میشه در نظر گرفت Time on Air هست. وقتی یک فرستنده سیگنالی رو میفرسته یک زمان طول میکشه که ارسال بشه. این زمان وایسته به فرکانس و سایز پکت وابستس که بهش میگن ToA. وقتی میگیم Duty cycle اش 1 درصده یعنی در این مثال اسالید 530ms * 99 طرف حق نداره ارسال کنه.

35: از نظر ریاضی برای دیوی سایکل 1 درصد

99 - 1 / d - 1 میشه

برای یک دهم درست

999 - 1 / d - 1 میشه

حروفی که مهندس بایک زندن:

Communication and transport technologies

- An essential requirement for implementing a large and cooperating network of devices is the ability to have a flexible, inexpensive and adaptive connectivity layer.
- For these reasons, during the years, several wireless connectivity standards have emerged, ranging from very short-range, low power connections to long-range connectivity solutions based on cellular technologies.
- While short-range protocols can be successfully used for a number of applications (e.g., smart homes, wearable, etc.), for many other applications larger coverage radii become mandatory.

برای تبادل داده نیاز به شبکه ارتباطی داریم. یه سری استاندارد ها Short-range هستن برای کاربرد های مثل خانه هوشمند. ولی یک سری دیگه long range که برد زیادی داره

Communication and transport technologies

- We expect that in 2024 more than the 40% of the wide area M2M connections will be LPWAN. That is why several companies are developing different protocols to implement the LPWAN technology.



علاوه بر این کم برد توان اینها که داریم میبینیم در فرکانس های آنلاینس کار میکنن و LPWAN های معروف هستن که در حال حاضر دارن استفاده میشن

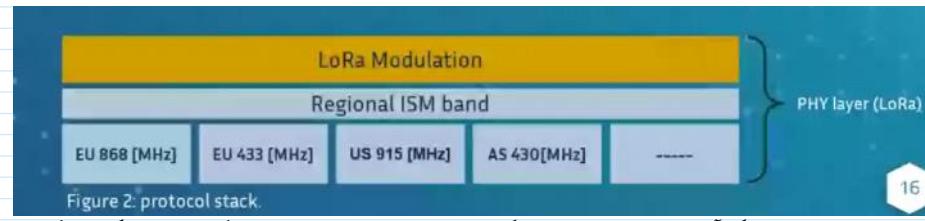
11

LoRaWAN network

- The LoRaWAN network is formed by two core components: LoRa and LoRaWAN, each of which refers to a layer in the protocol stack. LoRa is a physical layer modulation developed by Semtech. LoRaWAN is implemented on top of the LoRa and includes data link and network layers [4].

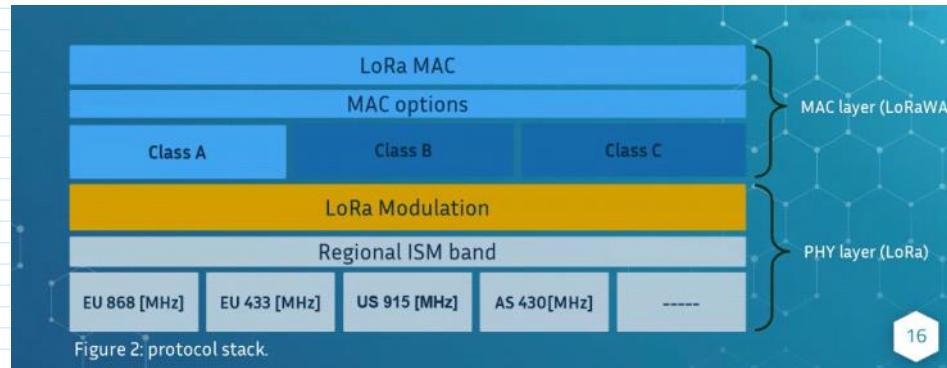
شبکه LoRaWAN برد زیاد و توان کم رو فراهم میکنه. به این دلیل جذابه که تا به حال همچین چیزی نبوده. مزیت هاش نسبت به LPWAN اینه که :

- خیلی بیشتر نسبت به LPWAN ها خیلی low power ترره.
- ارزون هستش
- در همه لایه ها متن باز است (به جز لایه فیزیکی که semtech توسعه داده). و برمی گیریم کل الگوریتم های لایه MAC توش هست.



جمع میشه

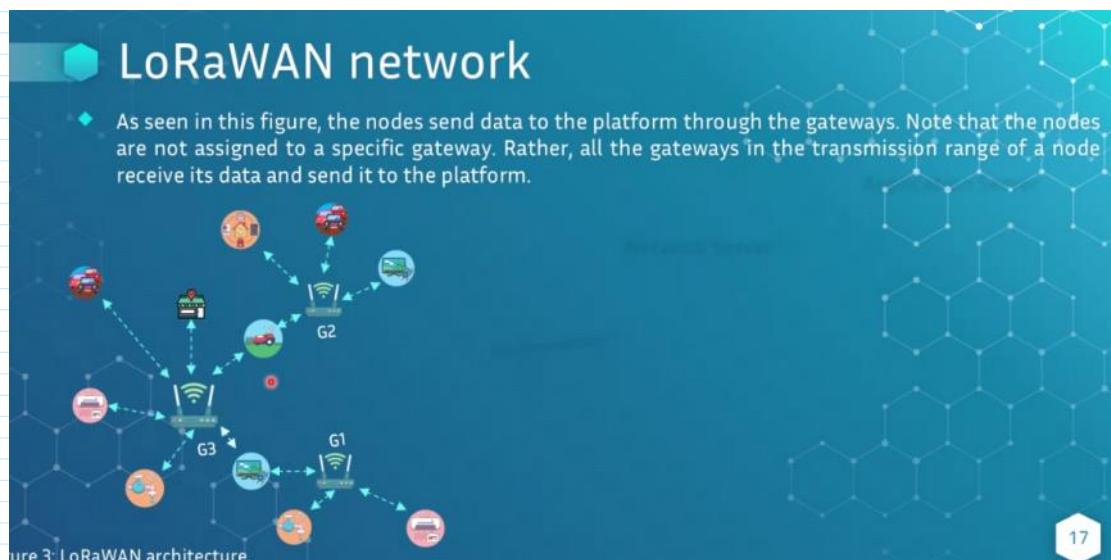
همونطور مبینه‌نیم از ISM که آنلاپسنس هست استفاده میکنه. در مناطق مختلف جهان این باند فرکانسی متفاوته که در شکل مبینه‌نیم. این فرکانس ها یک فرکانس مرکزی هستن و یک پهناهی باند هم داریم که با اینا



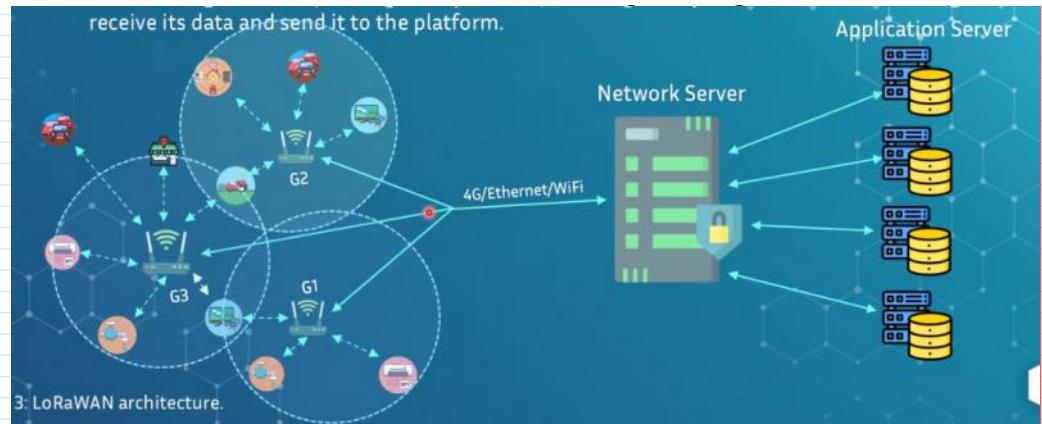
بعد میریم لایه مک که از ALOHA استفاده میشه. Node ها هم 3 تا کلاس بندی کرده و برا هر کدام ویژگی در نظر گرفته



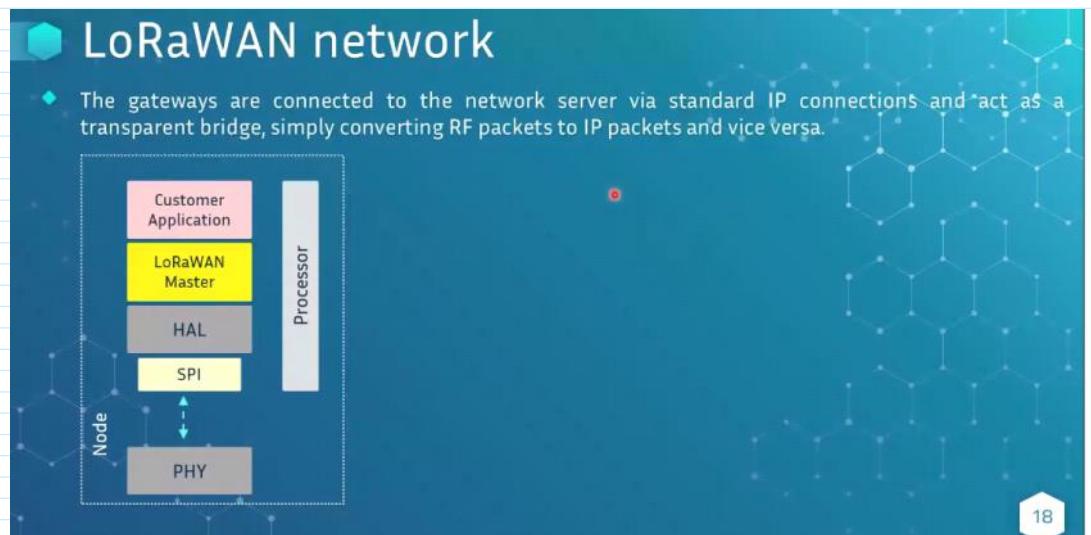
میشه اینطوری گفت یه Application داریم اون پلا بعد زیرش CSS بعد Zیرش ALOHA



ما یک سری Node داریم یک سری gateway که با هم ارتباطن. دیتا هایی که یک گره میفرسته Gateway های مختلف دریافت میکنن که تو شکل مشخصه. ما هرجی Gateway بیشتر داشته باشیم اطمینان بیشتری داریم که بسته برسه. ولی خب هیچ گره ای به هیچ gateway ایassociate نمیشه و gateway ها بر بستر پروتکل IP به Network server و Application server میدن و اونم میده به

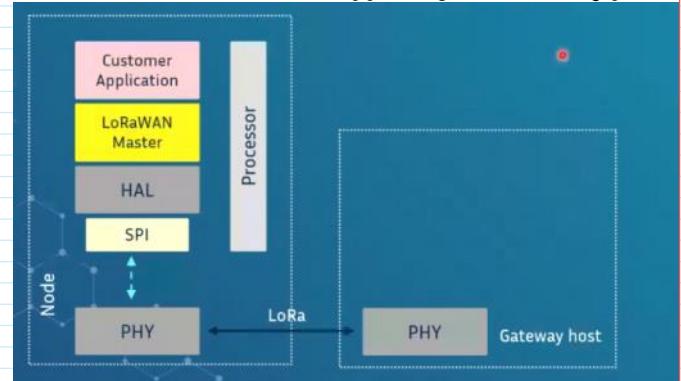


خوبی این پروتکل اینه که راحت میتونیم وسایلشو بگیریم تو آزمایشگاه راه اندازی کنیم.

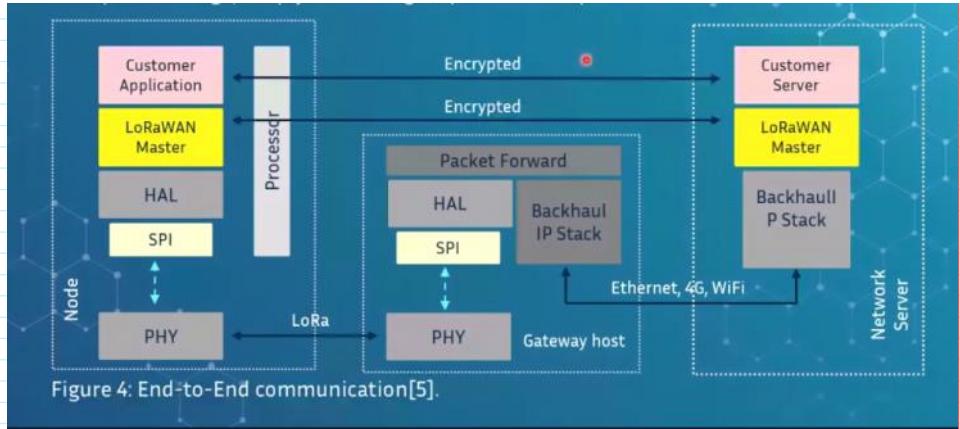


18

دید دیگه اینه که بیايم Component هارو جزو تر ببینیم. هر Node که در LoRaWAN داریم یک کاربردی رو داره انجام میده مثلًا یک سنسور رطوبت رو اندازه گیری میکنه. حالا این باید یک مازول برای ارتباط نیاز داره پس میکروکنترلر نیاز داریم که ارتباط رو برقار کنه. در آخر توسط ارتباط LoRa این ارتباط برقرار میشه:



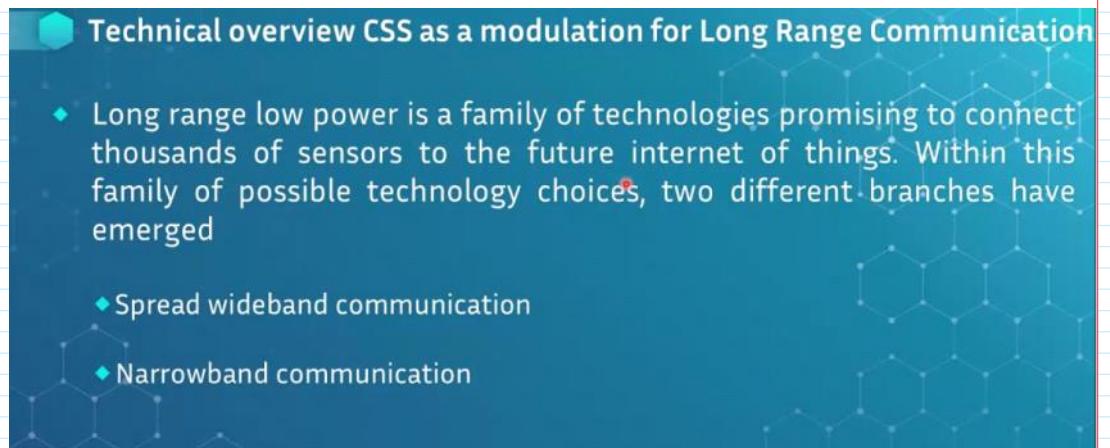
سپس در gateway یک مازول ارتباطی هستش که بر اساس سیگنال دریافتی بسته رو تشکیل میده و Packet Forwarder میاد یک بسته IP ازش میسازه میده به



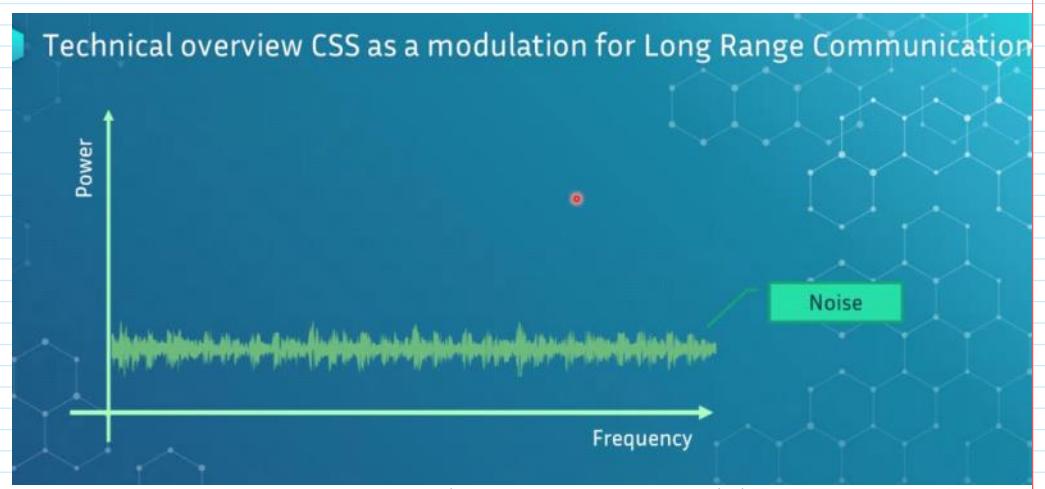
چیزی که هست اینه که بسته ها به صورت end-to-end ارسال میشه و خبری نداره اون payload بسته چیه.



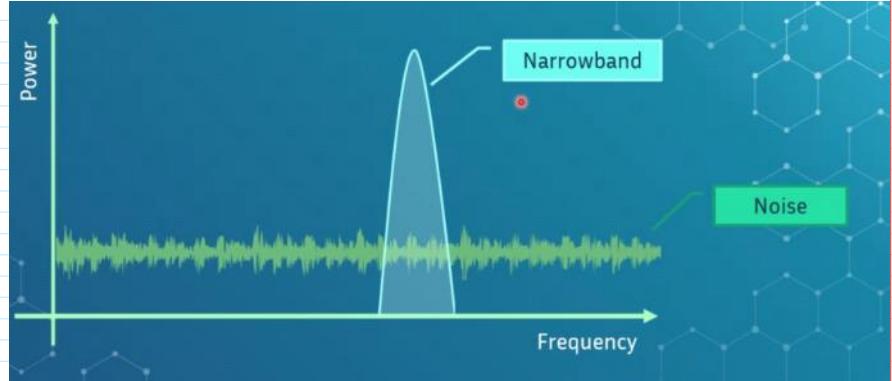
بریم سراغ جزئیات لایه فیزیکی. میخوایم ببینیم چطور ارتباط برقرار میشه و مدل‌اسیون چطوریه



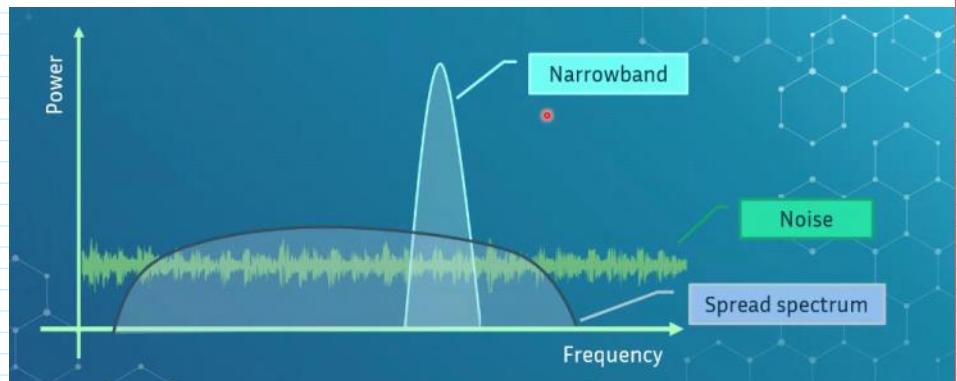
برای شبکه لانگ رنج در حالت کلی مهمه از چه مدل‌اسیون استفاده کنیم که در رنج و باور تأثیر داره. 2 تا تکنیک وجود داره برای لانگ رنج بودن که در شکل می‌بینیم.



بریم یه دید کلی داشته باشیم از این 2 تا تکنیک. نویز در تمام بازه های فرکانسی با یک قدرتی وجود داره. حالا میتوانه نویز سفید باشه یا صورقی یا



سیگنال بر اساس Narrowband مدل میشه از یک پهناهی باند کم استفاده میکنه ولی Power density زیاده در اون محدوده زیاده

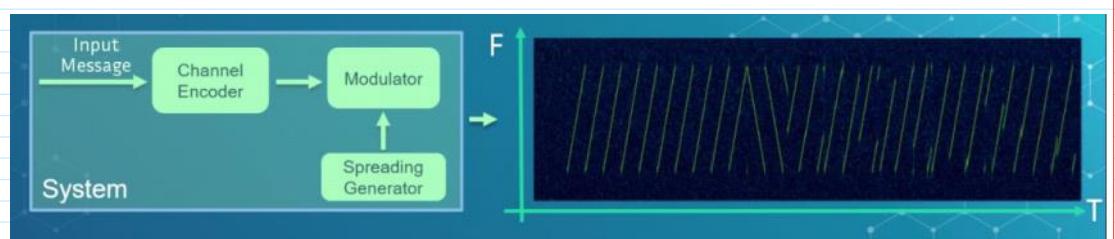


بر عکس Spread spectrum که در تمام بازه فرکانسی کاربرایا میتوون ارسال کنن و power اش پخش میشه.

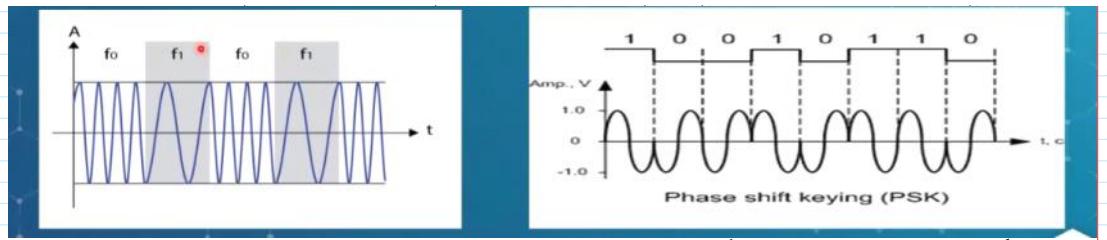
چون Power Density Narrowband زیادی داره از نویز فاصله میگیره و چون نویز تاثیرش کم میشه رنج زیادی رو میتوونیم بوشش بدیم. توی Spread spectrum هم بخشی زیادی زیر نویز هستش و باعث میشه رنج زیاد بشه



شبکه Spread spectrum از LoRaWAN استفاده میکنه که هرکدام روش های مختلف داره. که از CSS استفاده میکنه.

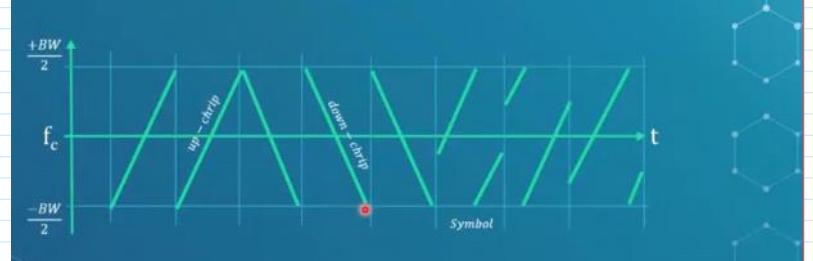


حالا یک گره داریم که سیگنال بر حسب زمانش رو داریم میبینیم. سیگنالی که مشاهده میکنیم سیگنال خروجی از سیستم هستش که پیوسته است که میخود 0 و 1 رو مدل کنه. مثل FSK



توی FSK میگفتیم مثلاً f_0 بود ما منظورمون ۰ عه اگر f_1 بود منظورمون ۱ عه یا PSK که قبلاً دیدیم.

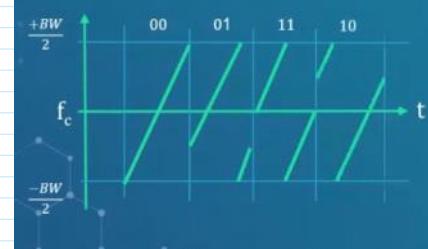
توی سیگنال خروجی داریم میبینیم خط ها مورب هستن. اگر به شکل واضح تر بینیم:



سیگنال LoRa توی بازه فرکانس آنلاینس ارسال میکنه. این بازه حول یک پهنهای باندی در حال تغییر هستش که 128 یا 256 کیلوهرتز هستش. که بازه رو توی محور عمودی میبینیم. سیگنال LoRa از یک سری up-chip و down-chip تشکیل شده. اونایی که از وسط دارن میرن و کامل نیستن shift cycle میگن.

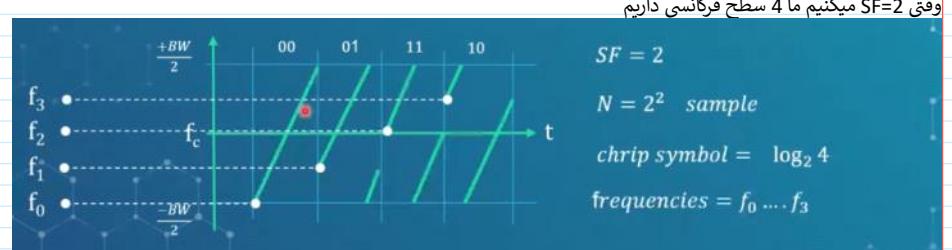
حروف استاد: ما یک فرکانس مرکزی داریم و به اندازه نصف پهنهای باند میتوانیم بالا پایین بریم. (محور γ فرکانس است). این شکل موج نیست. منظور تغییر فرکانسی هستش. فرض کنیم یک سینوسی داریم. فرکانسون یک Amplitude و فرکانس و فاز داریم که هر کدام رو اگه نماینده ۰ و ۱ داشته باشیم FSK و PSK و ASK داشتیم در CSS این فرکانس یک مقدار پیوسته در زمان هستش و فرکانسیش داره بر حسب زمان تغییر میکنه.

مهندس باتک: حالا میخوایم داده هارو ارسال کنیم:
حالا این که در هر chrip چه تعداد بیت رو ارسال کنیم توسط یک پارامتری به اسم Spread Factor تعریف میشه. بر فرض مثال اگر اوونو ۲ بذاریم (که توسط ما تعیین میشه) داریم:
اینطوری در هر chrip میتوانیم ۲ تا بیت رو ارسال کنیم



حالا این که up-chip هارو کی داریم بسته به خودمون داره که چطوری کد بکنیم.

$$\begin{aligned} SF &= 2 \\ N &= 2^2 \text{ sample} \\ \text{chrip symbol} &= \log_2 4 \end{aligned}$$



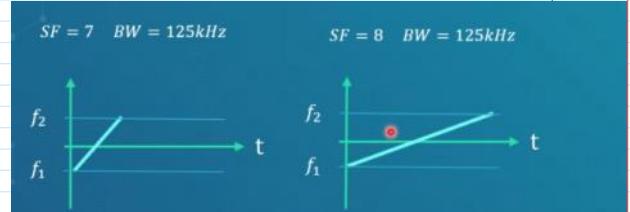
وقتی SF=2 میکنیم ما ۴ سطح فرکانسی داریم

$$\begin{aligned} SF &= 2 \\ N &= 2^2 \text{ sample} \\ \text{chrip symbol} &= \log_2 4 \\ \text{frequencies} &= f_0 \dots f_3 \end{aligned}$$

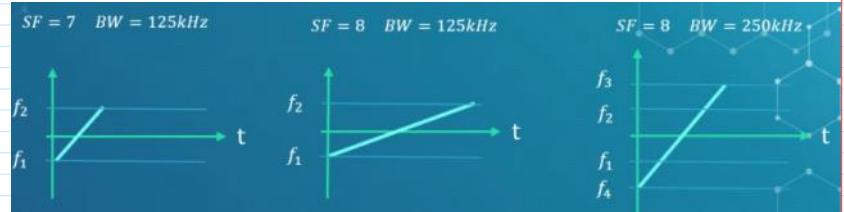
در حالت کلی:

$$\begin{aligned} SF &= s \\ N &= 2^s \text{ sample} \\ \text{chrip symbol} &= \log_2 N \\ \text{frequencies} &= 2^s \end{aligned}$$

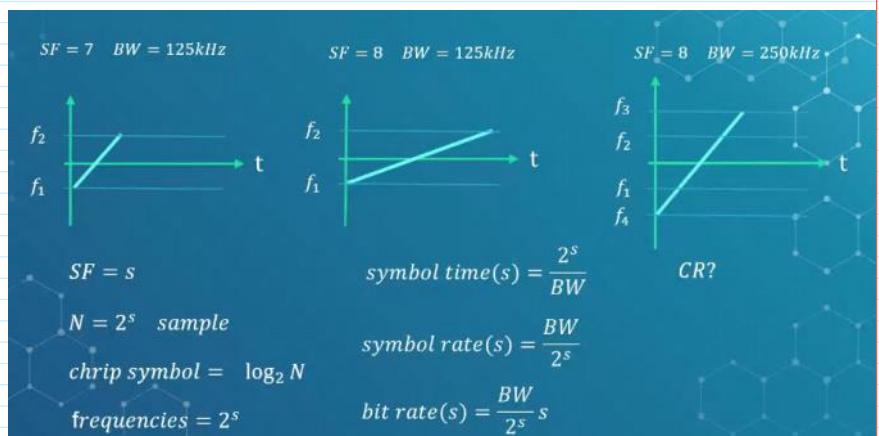
حالا بینیم چه تاثیری روی مصرف انرژی و توان دارد.



وقتی SF رو زیاد کنیم زمان ارسال بیشتر میشه چون بیت های بیشتری برای ارسال داریم.



اگر پهنای باند بیشتر شه سطوح فرکانسی بیشتر میشه



توی فرمول هم میبینیم هرجی BW زیاد شه زمان کم میشه هم بیشتر شه زمان بیشتر میشه

حرف استاد: ما یک پهنای باند در اختیار داریم (f_c) که به اندازه نصف پهنای باند میتوانیم میتوانیم بالا و پایین ببریم. حالا به موقع 250KHz که متفاوت است. هرچه پهنای باند بیشتر باشه فرکانس های بیشتری رو میتونم انتخاب کنیم. در CSSF داریم که یک عددی هست که تعیین میکنه که اولاً چندتا بیت رو منتقل کنیم با بدونه سیگنال. دوم این که مشخص میکنه این چند بیت رو در چه زمانی ارسال میکند.

این به یک نوعی Symbol rate هم داره تعیین میکنه. Symbol time رو که تعریفیشو تو شکل داریم میبینیم. Symbol rate هم که عکسش میشه.

مهندسان پاپی: برای بدست آوردن bit rate هم میشه گفت ما symbol rate رو داشته باشیم تعداد بیت هاشم داشته باشیم ضرب کنیم بدست مید که رابطش رو میبینیم. که ضریلر SF شده

یک چیز دیگه داریم تحت عنوان Coding rate که یک سری بیت برای خطابایی اضافه میشه. با در نظر گرفتن این CR، تعداد بیت های ارسالی میشه $(SF \times CR)$.

با وجود CR چه تاثیری روی انرژی مصرف میشه؟ نخ ارسال کمتر میشه و سمبول کمتری ارسال میشه. ولی انرژی بیشتر میشه چون زمان بیشتری طول میکشه ارسال بشه. چون باید به انگلآل بگیریم روی زمان و زمان هم زیاد داره میشه.

بس جراحت رو میبریم بالا وقتی هم دیتابیت کم میشه هم انرژی زیاد میشه؟ چون از لحاظ Range قضیه فرق میکنه و هرجی بیشتر کنیم Range بیشتر میشه.

ما میتوانیم احتمال موقوف رسیدن یک SF اینطوریه که در انرژی ای که در هر سمبول موجوده ضرب بشه.

$$PSS = symbol\ time \times E_s$$

بس چون هردو داره زیاد میشه احتمال رسیدن بسته به مقصد بیشتر میشه و رنج میره بالاتر.

برای درک بهتر آگه 2 نفر در فاصله زیاد قرار بگیرن. برای این که حرف هم رو بفهمن هم این که داد میزنن (انرژی میره بالاتر) هم شمرده حرف میزنن تا اون یکی بفهمه.

حالا بیایم در مورد دسیبل صحبت کنیم: فرض کنیم یک Amplifier داریم با توان 10 وات. این یک صدای میرسونه. وقتی 20 وات اش کنیم یک نسبت میگیریم میگیم 2 برابر شده چیزی که میشنویم. اما نکته ای در بعضی معیارا هست اینه که وقتی 10 وات رو میکنیم 20 وات، خطی نیست که 2 برابر بشه. مثلاً ممکنه 4 برابر بشه.

Decibel

- ❖ A unit of measurement is a definite magnitude of a quantity, defined and adopted by convention or by law, that is used as a standard for measurement of the same kind of quantity.
- ❖ If a person is listening to a radio with a sound power of 10 watts, the volume will not be doubled when it increases the output power to 20 watts.
- ❖ The decibel (symbol: dB) is a relative unit of measurement corresponding to one tenth of a bel (B). It is used to express the ratio of one value of a power or root-power quantity to another, on a logarithmic scale.

$$x_{dB} = 10 \times \log_{10} \left(\frac{\text{signal}}{\text{ref. signal}} \right)$$

که رابطه لگاریتم رو میتوانیم ببینیم. مقدار 10 که ضرب میشه اینه که مقدار بل که کم هست رو در نظر نمیگیریم و ضرب در 10 میکنیم که برآمون ملموس تر بشه.

این که بخوایم توان رو با dB بگیم باید یک سیگنال مرجع (ref signal) داشته باشیم. برای مرجع 1mW 0 در نظر میگیریم و واحدش هم dBm میدارن که m همون میلی بودن وات رو داره میگه

- ❖ What if we want to measure an absolute power with dB? We have to define a reference.
- ❖ The reference point that relates the logarithmic dB scale to the linear watt scale may be for example this:

$$1mW = 0 \text{ dBm}$$

- ❖ The new m in dBm refers to the fact that the reference is one mW, and therefore a dBm measurement is a measurement of absolute power with reference to 1 mW.

- ❖ To convert power in mW to dBm:

$$P_{dBm} = 10 \times \log_{10} P_{mW}$$

- ❖ To convert power in dBm to mW:

$$P_{mW} = 10^{\frac{P_{dBm}}{10}}$$

(تو رابطه آخر اون dB باید بشه dBm)

حالا برمی مثل ببینیم:

- ❖ Example: mW to dBm, Radio signal power: 100mW

$$P_{dBm} = 10 \times \log_{10}(100)$$

$$100mW = 20dBm$$

- ❖ Example: dBm to mW, Radio signal power: 17dBm

$$P_{mW} = 10^{\frac{17}{10}}$$

$$17dBm = 50 \text{ mW}$$

حالا فرض بشه یک سیگنال 10mW داریم و یک تقویت کننده گذاشتیم که dB10 اضافه کرده (مرجع نداشتیم و dBm نیست). حالا بخوایم mW جدید و dBm رو بدست بیاریم داریم:

- ❖ You can now imagine situations in which:

$$10 \text{ mW} + 10 \text{ dB of gain} = ? \text{ dBm}, ? \text{ mW} \quad 10 \text{ dB} = 10 \times \log_{10} \left(\frac{x_{mW}}{10 \text{ mW}} \right) \quad 100 \text{ mW} = 20 \text{ dBm}$$

تا مثال دیگه:

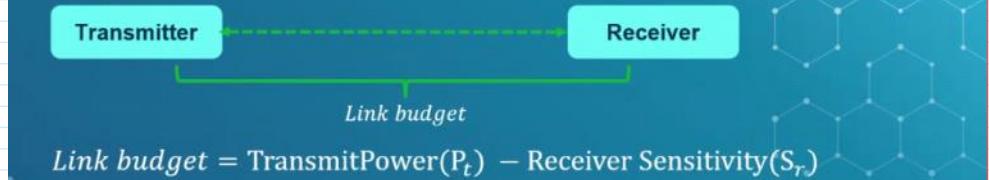
$$\begin{aligned} 50 \text{ mW} + 3 \text{ dB} &= 100 \text{ mW} = 20 \text{ dBm} \\ 17 \text{ dBm} + 3 \text{ dB} &= 20 \text{ dBm} = 100 \text{ mW} \end{aligned}$$

در پایین چون dBm و dB هرچفتیون نسبت ان دیگه تبدیل نیاز نیست میشه جمعشون کرد. میشه loss هم مثال زد و حل کرد فقط کم میشه:

$$20 \text{ dBm} - 10 \text{ dB of loss} = ? \text{ dBm}, ? \text{ mW} \quad 10 \text{ dBm} = 10 \text{ mW}$$

What factors determine RF range and performance?

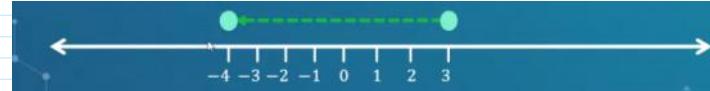
- ❖ **Link budget** : The link budget is the difference between the strength of the transmitted signal and the minimum required signal at the receiver



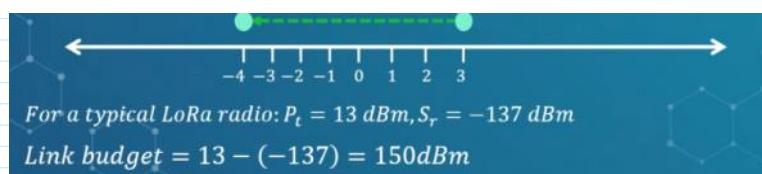
اختلاف بین قدرت سیگنال ارسالی و حداقل میزان دریافتی که گیرنده میتوانه بگیرد. یعنی میخوایم ببینیم اگر یک سیگنال را ارسال کنیم ببینیم اگر یک سیگنال را پاشنودیم، گیرنده هم یک حساسیت دارد. دریافت کننده بتوانه دریافت کنه

حرف استاد: مثل انسان که نسبت به صدای حساسیت نشون میده و هرجی سن میله بالاتر این حساسیت کمتر میشه و باید صدا خیلی بیشتر بشه تا پاشنودیم، گیرنده هم یک حساسیت دارد. اگر واحد ها بر حسب dBm بگیم link budget فاصله این 2 تاکه در شکل میبینیم.

برای مثال:



فرض کنیم از جایگاه 3 میریم به -4. اگر ازم بگن چقدر دویدیم میگیم 7 گام دویدیم. میگه اختلاف اون سیگنالی که ارسال گرفتی با حداقل که میتوانی دریافت کنی چقدر هستش.



در شبکه LoRaWAN مینیمم منفی 137 هستش. دست ما نیست و اون سازمان مقررات میگه حداقل تا 13 dBm میشه گذاشت.

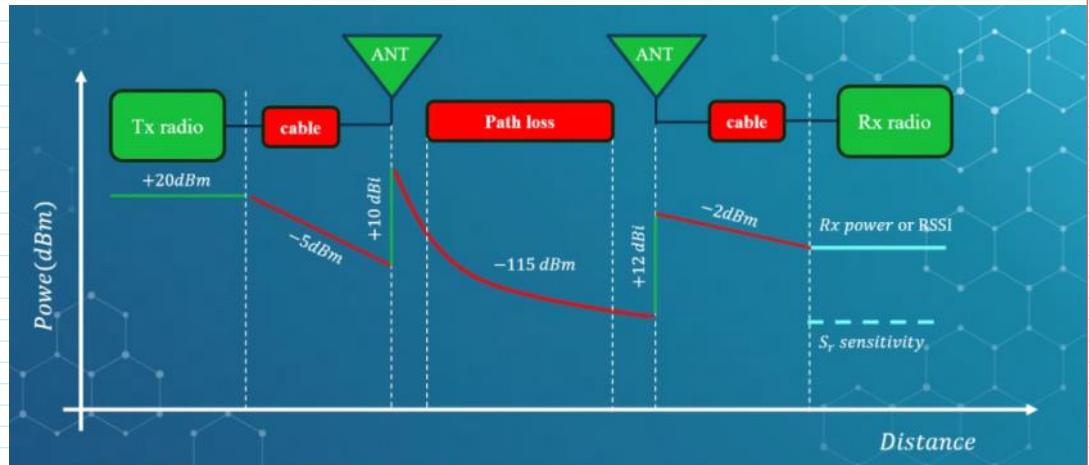


Figure 5: Example link budget calculation [8].

سیگنال وقتی انتشار میشه روی cable یک تضعیفی داره بعد میرسه به آنت و تقویت شد. بعد دوباره بعد انتشار دچار تضعیف میشه و میرسه به آنت و تقویت میشه. بعد به کابل میشه تضعیف میشه و میرسه به گیرنده. حالا سوال اینجاست اگه عوامل loss در مکریم حالت باشند، تاچه حد میتوانند زیاد باشند که گیرنده بگیرد. به این مورد میگن که گیرنده تاچه حد حساس هستش رو میگن Receive sensitivity را. این حساسیت در گیرنده به Spread Factor و به عوامل نویز در محیط بستگی دارد.

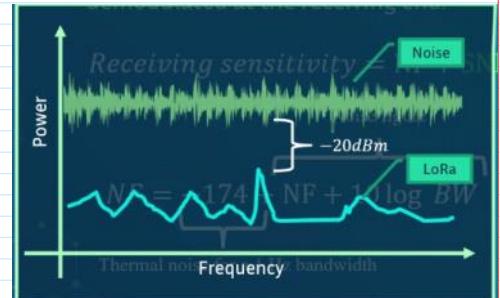
❖ **Receiving sensitivity:** refers to the smallest modulated signal that can be demodulated at the receiving end.

$$\text{Receiving sensitivity} = \text{NF} + \text{SNR}$$

که SNR با SF مرتبط است. SNR میشه نسبت سیگنال به نویز که در اسلاید های جلوتر توضیح داده میشه. NF هم میشه کف اون نویز که شامل:

$$NF = -174 + \frac{\text{Noise figure}}{\text{Thermal noise for a 1 Hz bandwidth}}$$

NF سمت چپ Noise floor ولی سمت راست Noise figure هستش.



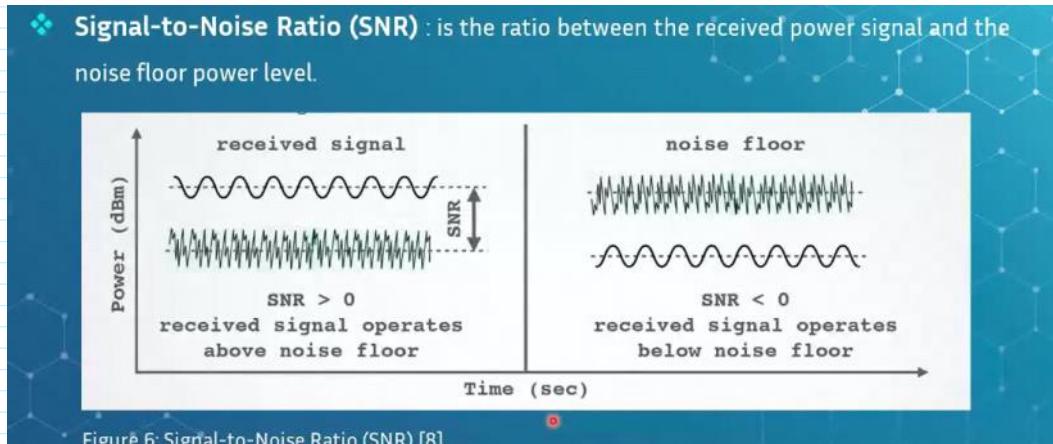
نویز در تمام بازه های فرکانسی وجود دارد و سیگنال LoRa زیر سطح نویز دریافت میشه. این که تاچه حد زیر سطح نویز هست به SNR بستگی دارد. هرچی SF رو بیشتر کنید سیگنال ما بیشتر زیر سطح نویز میره. مثال:

- $Tx\ power = 14\ dBm$
- $BW = 125\ kHz = 10\log_{10}(125000) = 51$
- $NF = 6\ dBm$
- $SNR = -20\ dBm (for SF=12)$
- $Rx\ sensitivity = -174 + 51 + 6 - 20 = -137\ dBm$
- $Link\ budget = 14dBm + (-137dBm) = 151dBm$

توضیحات تکمیلی استاد: حساسیت یک receiver در آخر یک عدد که نشان میده میزان Power سیگنال دریافتی درجه حد باشه که گیرنده بتوانه detect بکنه و demodulate کنه. حالا این عدد هرجی بیشتر باشه بهتر یا کمتر؟ کمتر، چون نشون میده سیگنال کمتر هم میتوانیم بشنویم. هرجی میزان نویز بیشتر باشه عدد receiving sensitivity هم میره بالاتر. این نویز به دمای محیط وابستس به پهنهای باند. چرا پهنهای باند؟ هر گیرنده یک فیلتر فرکانسی میداره که تو اون بازه فرکانسی دریافت کنه. هرجی این دروازه رو بیشتر کنیم نویز بیشتری دریافت میکنیم. اما SNR وابسته است به لایه فیزیکی. مثلًا این که از چه مازولیتوري استفاده میکنیم. ASK بیشتر تحت نویز قرار میگیره تا FSK و PSK. SNR به تضعیف محیط هم ربط دارد (عواملی که سیگنال تو محیط پیش برخورد میکنه). بازه فرکانسی (فرکانس پایینتر تضعیف کمتره) و فاصله (هرچی دورتر تضعیف بیشتر) هم وابسته است.

در LoRaWAN spread spectrum هستش. حتی اگر شدت سیگنال دریافتی از مقدار نویز کمتر باشد هم امکان recovery هستش.

--- اتمام توضیحات استاد



SNR میشه نسبت سیگنال به نویز. اگر از 0 dBm بیشتر باشه یعنی بالای نویزه و اگر زیر صفر باشه یعنی زیر نویزه.
برای محاسبه :

Accurate calculation of SNR:

$$SNR = \frac{P_s}{P_w} \quad P_s = \text{The average power of } s(t) \\ P_w = \text{The average power of noise}$$

به راه دیگه محاسبه اش هم:

Calculate minimum SNR in terms of SF:

$$SNR_0 = \frac{E_{bit}}{NF}$$

انرژی که در هر بیت داره تقسیم بر نویزی که در محیط دارد.
که انرژی هر بیت هم:

$$E_{bit} = RSSI \times T_{bit} \\ T_{bit} = \frac{1}{BW} \times 2^{SF}$$

پس میتوان نوشت:

$$SNR_0 = \frac{E_{bit}}{NF} \quad E_{bit} = RSSI \times T_{bit} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} \quad SNR_0 = \frac{RSSI \times 2^{SF}}{NF \times BW}$$

بر اساس این رابطه RSSI (قدرت سیگنال دریافتی) هم میشه بدست آورد:

$$RSSI = \frac{SNR_0 \times NF \times BW}{2^{SF}}$$

:هم receiving sensitiviy و

$$S_r = BW \times NF \times SNR(SF)$$

حالا اگر کا RSSI و Receiving Sensivity یک در نظر بگیریم:

$$\left. \begin{array}{l} RSSI = \frac{SNR_0 \times NF \times BW}{2^{SF}} \\ S_r = BW \times NF \times SNR(SF) \end{array} \right\} \quad SNR(SF) = \frac{SNR_0}{2^{SF}} \quad SNR(12) = \frac{31}{2^{12}} = 0.007mW = -21dBm$$

حالا میخوایم در مورد Path loss صحبت کنیم که مهمترین عامل تضعیف سیگنال بود:

- ❖ **Path loss (PL):** Path loss (or path attenuation) delineates a decline in power density of any given electromagnetic wave as it propagates through space.
- ❖ Path loss models are developed using a combination of numerical methods and empirical approximations of measured data collected in channel sounding experiments.

هم میشه به صورت عملی در محیط عملی اندازه بگیریم یا با مدل ریاضی اندازه بگیری کنیم. ساده ترین مدل اینه که فقط 2 عامل رو در نظر بگیریم:

$$L_{path} = \left(\frac{4 \cdot \pi \cdot f}{c} \right)^2 \cdot d^n$$

d = distance (meter)

f = signal frequency (MHz)

n = is 2 for free space

صحبت استاد: از سه عامل محیط، فاصله و فرکانس کاری فقط دو عامل رو در نظر میگیریم. تو فرمول مبینیم هرجی فرکانس میره بالاتر loss بیشتر باشه. هرجی فاصله هم بیشتر باشه loss بیشتره که طبیعیه. در این مدل محیط در نظر نگرفته ولی با این توان میشه در نظر گرفت. هرجی محیط پیچیده تر باشه n رو بیشتر میگیریم. چرا دنبال loss هستیم؟ میخوایم link budget و receiving sensitiviy را بررسی کنیم و loss تو ش اهمیت داره. ----- تمام

حالا اگر فاصله 1000 متری و فرکانس 868 هرتز باشه داریم:

$$L_{path(dB)} = 10 \log_{10} \left(\left(\frac{4 \times 3.14 \times 868}{300} \right)^2 \cdot 1000^2 \right) = 91 dB$$

در این مدل حتی اگر تکنولوژی دیگه ای هم داشته باشیم همین loss رو داریم.

حالا ما distance رو بر اساس رابطه در باریم:

Calculate maximum distance in terms of SF:

$$d = \left(\frac{L_{path}}{\left(\frac{4 \cdot \pi \cdot f}{c} \right)^2} \right)^{\frac{1}{n}}$$

حالا با فرض این که link budget برابر با loss محیط باشه داریم:

We assume that there are no antenna gains and set the path loss L_{path} equal to L_{budget}

$$d = \left(\frac{L_{budget}}{\left(\frac{4 \cdot \pi \cdot f}{c} \right)^2} \right)^{\frac{1}{n}}$$

What factors determine RF range and performance?

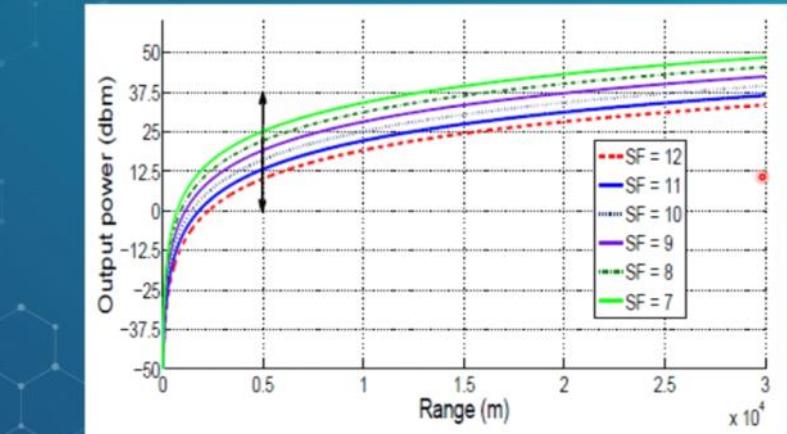


Figure 11. Required output power vs. LoRaWAN range.

این تصویر در شبیه سازی بدست امده. فرض کنیم در فاصله 500 متر هستیم (اون 0.5 که نوشته) میبینیم از توان کمتر از 12.5 dBm8 حدود استفاده کنیم. و از شکل میبینیم هرچی SF بیشتر باشه رنج ارتباط بیشتره.

حالا میخوایم در مورد time on air صحبت کنیم. این که چقدر طول میکشه بسته به مقصد برسه.

❖ Duration of the packet transmission

$$T_{on_the_air} = (N_{payload} + N_{preamble}) \times T_{symbol}$$

که میشه داده اصلی و preamble تعداد بیت های هستش که برای برقراری ارتباط در شروع ارتباط ارسال میشه.

❖ Duration of the packet transmission

$$T_{on_the_air} = (N_{payload} + N_{preamble}) \times T_{symbol}$$

$$T_{symbol} = \frac{2^{SF}}{BW}$$

$$N_{payload} = 8 + \max \left[\text{Ceil} \left(\frac{8PL - 4SF + 28 + 16 - 20IH}{SF - 2DE} \times \frac{CR + 4}{4}, 0 \right) \right]$$

$$N_{preamble} = 8$$

Description	
SF	Spreading Factor
PL	Payload Length
CRC	The use of the Cyclic Redundancy Check (CRC, 1 when enabled 0 when not)
IH	The use of explicit header or not (IH, 1 when enabled 0 when not)
CR	Coding Rate
DE	The Low Data Rate Optimization (DE, 1 when used 0 when not).

توضیح استاد: مثلاً preamble که تو قبیل داشتیمه این که تعداد بیت ها و ... رو مشخص کنن. Payload هم میشه کل بیت هایی که از لایه بالا داره انتقال پیدا میکنه. منظور از N هم میشه تعداد بیت ها.

البته میشه N رو تعداد بیت گرفت اون موقع زمان T آخری که ضرب میشه میشه زمان بیت. میشه هم تعداد symbol گرفت مثل اسلاید که در T_{symbol} ضرب بشه.
-- اتمام

حالا ما محدودیت Duty cycle هم داریم. با توجه به این محدودیت موجود در کانال مشخص میشه بسته هر چند وقت یه بار ارسال بشه:

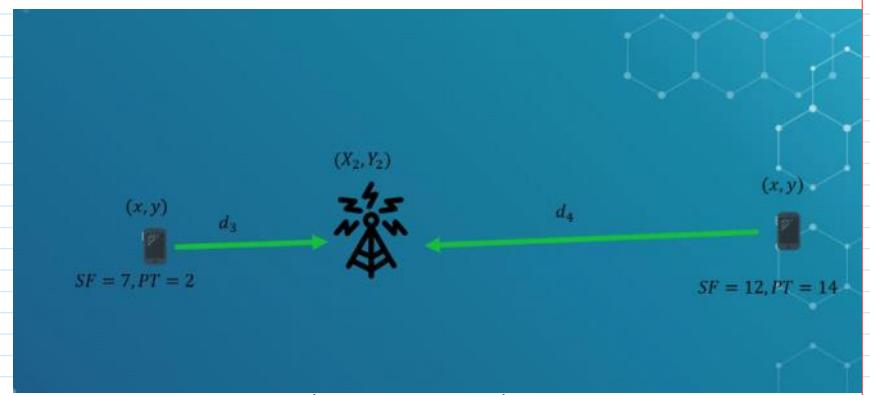
❖ The time that the LoRa Mote has to wait before send the next packet is called T_{off}

$$T_{off} = T_{on_the_air} \times (1 - \frac{1}{DCycle})$$

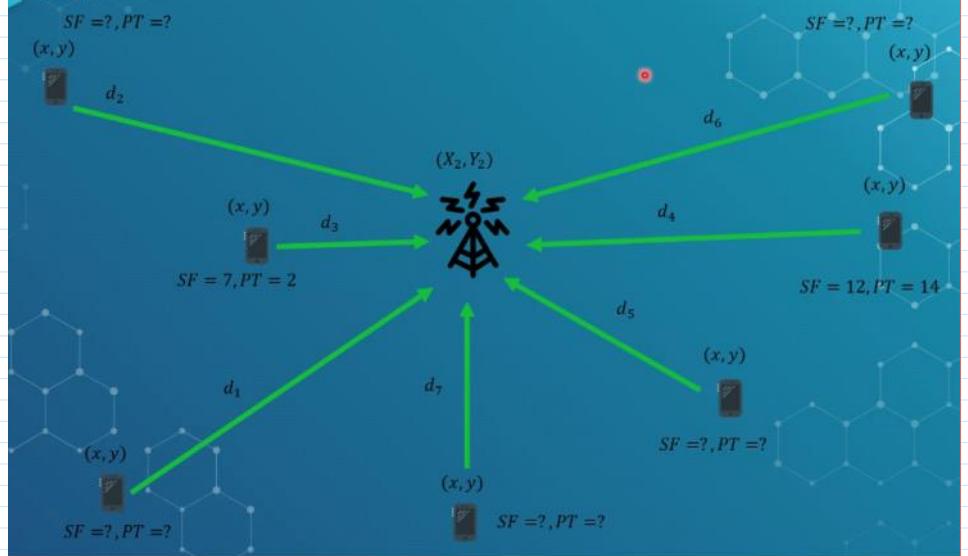
حالا این که هر چند وقت یه بار ارسال کنیم میبینیم به Time on air بستگی داره و Duty cycle و مشخص میشه که چقدر باید بایستیم.

❖ So we can calculate the maximum daily packets

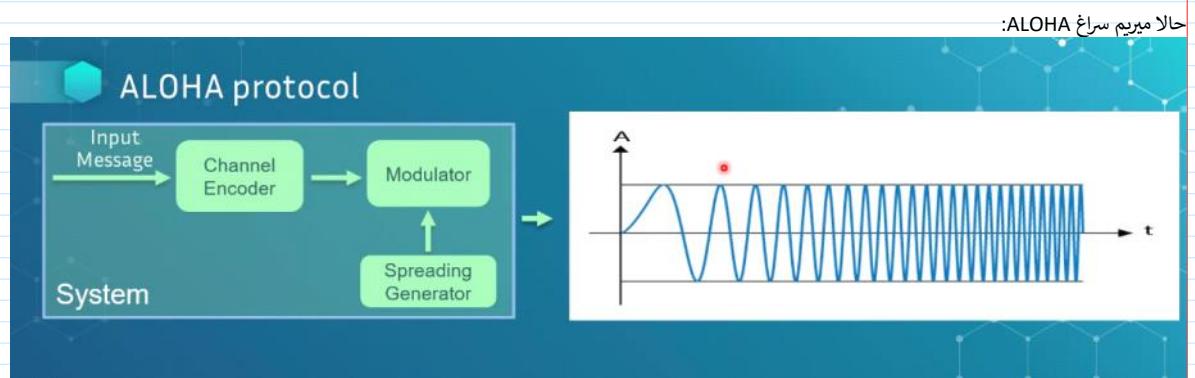
$$\text{Daily packets} = \frac{\text{Seconds in a day}}{T_{off}}$$



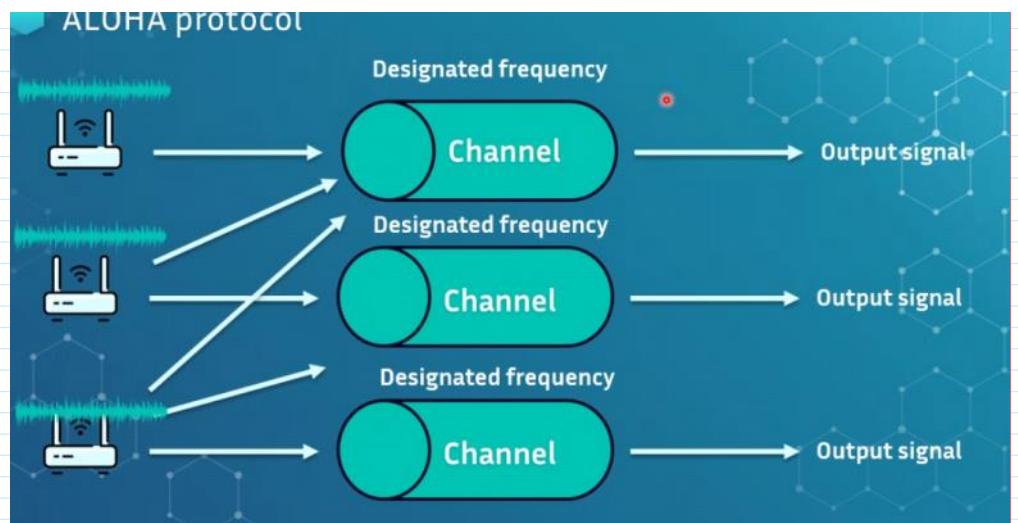
این که جه و توانی استفاده کنیم دیدیم که تاثیر گذاره. در شکل میبینیم که گره ای که نزدیک تر از SF و توان کمتری استفاده کرده.



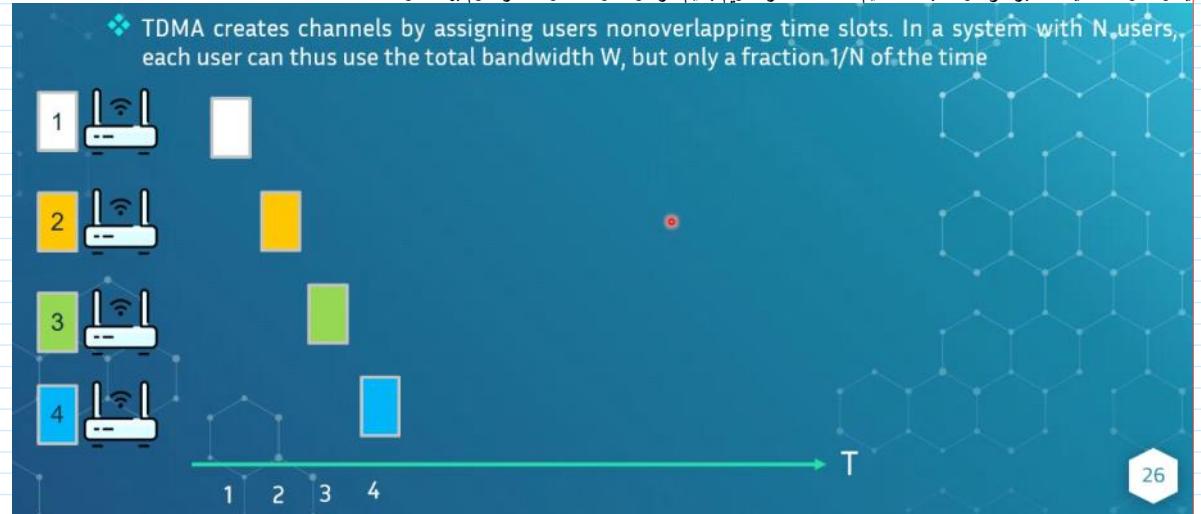
اما حالا هر کدام چه و توانی داشته باشن رو الگوریتم ADR تعیین میکنه.



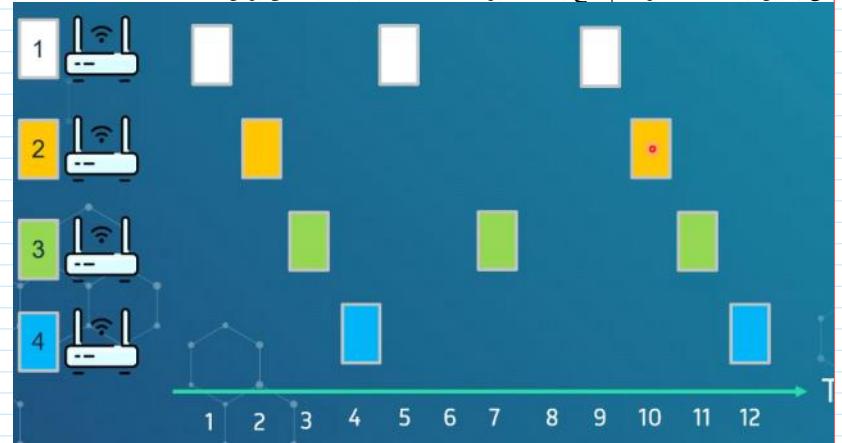
در شبکه LoRaWAN گره های مختلفی داریم. حالا میخوایم ببینیم چطوری به کاتال دسترسی دارن.



برای این که این Node ها تداخل نکنند ما نیاز به یه پروتکل داریم. در LoRaWAN ما 8 تا کانال فرکانسی داریم. این که ما تصمیم بگیریم در چه زمانی و چه بازه فرکانسی Node حق ارسال داره میره توی لایه MAC یه راه ساده اینه که برashون زمان بندی کنیم. مثل 4 تا گره داریم بگیم گره اول در فلان زمان گره دوم بهمان و ... :



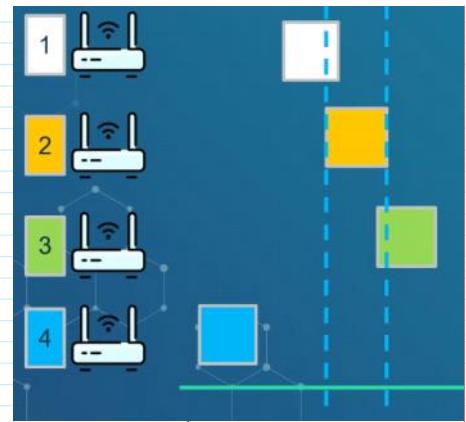
اینطوری بدون تداخل داده ها ارسال میشون
ولی بدیش اینه اگه مثلاً گره دوم هیچ بسته ای برای ارسال نداشته باشه یه زمانی برash تلف میشه:



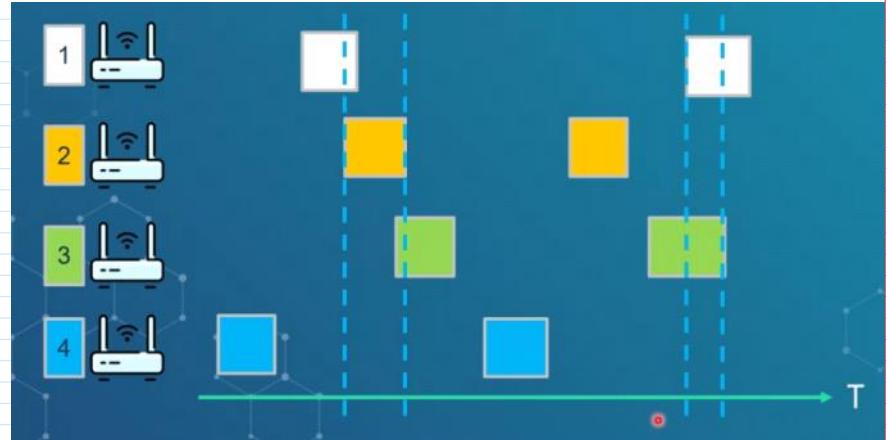
خوبیش اینه تضمین داریم تداخل نداریم ولی بدیش اینه که زمان تلف میشه و زمان بلااستفاده زیاد داریم.
برای بدست آوردن performance(بهره وری) میشه اینطوری گفت:

$$performance = \frac{10}{12} \times 100 = 83.3\%$$

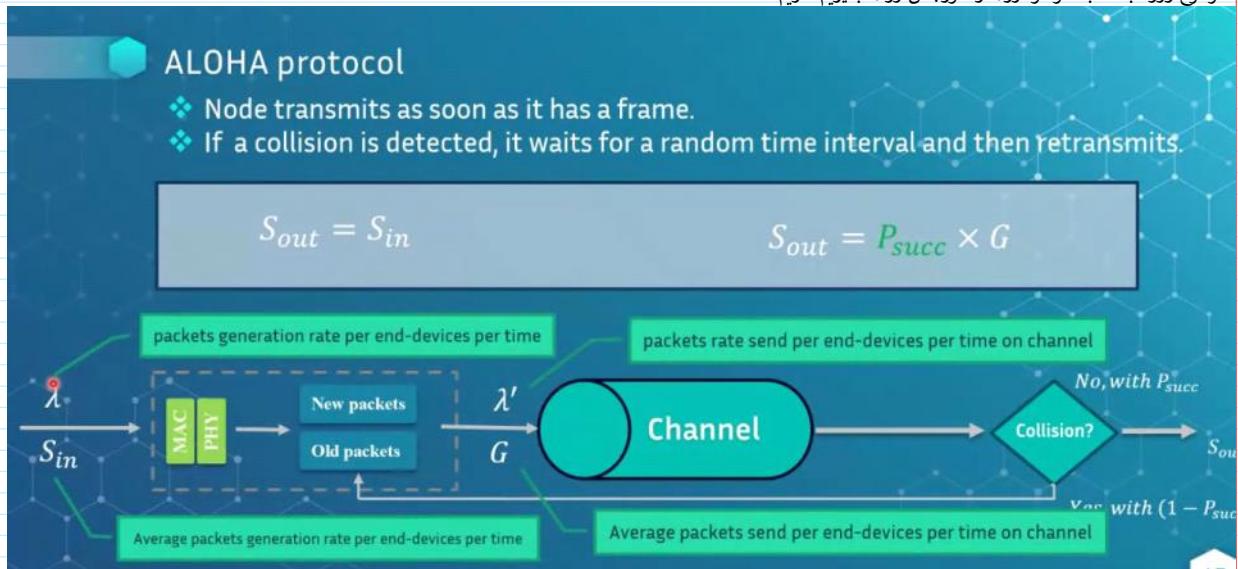
پروتکل ALOHA: هر نودی در هر زمانی داده برای ارسال داره میتوانه ارسال کنه.



ولی خوب ممکنه تداخل بشه و وقتی به گیرنده برسه هر 3 دچار خطأ بشه. در این موقع هریک یک مدت زندگی صبر میکنه و دوباره میفرسته. علت زندم هم اینه که چون اگه ثابت وای میستاندن دوباره تداخل میکدند:



مزیت این روش: اونی که time on air پایین تری داره کانال رو همیشه در اختیار داره حالا performance بیشتر میشه یا چی؟
برای این کار همچنان کانالی رو در نظر میگیریم که یک سری Node دارن ازش استفاده میکنن. یک سری Node که داده ارسال میکنن و ممکنه تداخل داشته باشیم. (اگه SF متفاوت باشه سطح فرکانسی متفاوت هستش و تداخل نداریم). این Node ها از لایه فیزیک و MAC تشکیل شدن.
اگر نزد ورود بسته به هر گره رو λ و خروجش رو λ' بگیریم داریم:



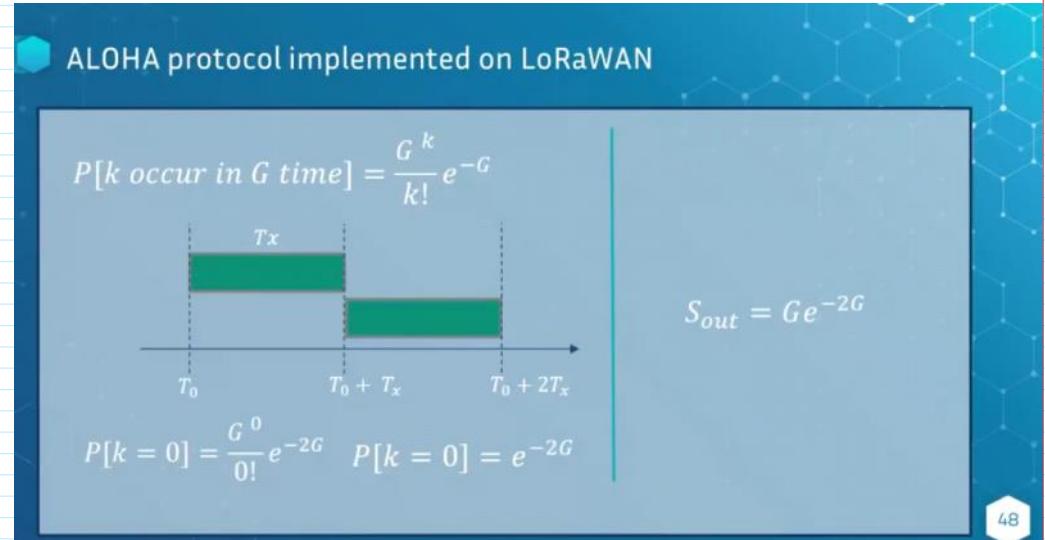
با احتمال $P_{success}$ بسته با موفقیت میرسه مقصد و با احتمال $1 - P_{success}$ ای درست میشه و ارسال میشه

فرایند تولید بسته در این Node ها یک حالت deterministic نداره. این که بگیم همه چی مشخصه و هر کدام در چه زمانی چه جزئی ارسال میکنن همچنان چیزی نداریم! تعداد گره ها خیلی زیاده و عوامل زیادی تاثیرگذار هستش. در واقع داره از توزیع بواسون پیروی میکنه که باعث میشه فاصله زمانی بین تولید بسته ها نمایی باشه.

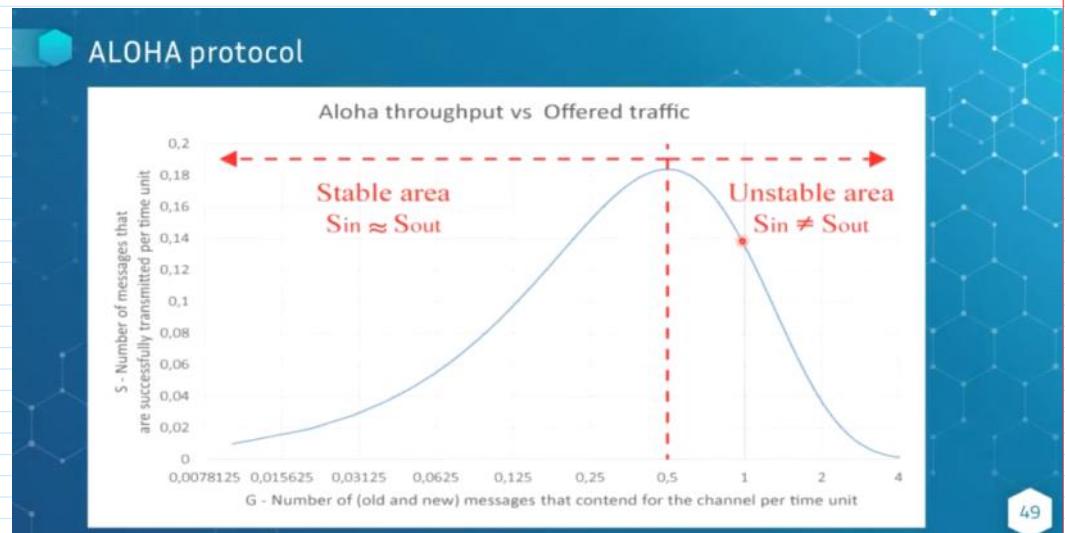
میانگین بسته هایی که دارن دوباره ارسال میشن یا برای تلاش بعدی ارسال میشن رو G در نظر میگیریم.

در چه زمانی $S_{out} = S_{in}$ میشه؟ تداخل نداشته باشیم.
ولی ما در ALOHA امکان تداخل داریم. پس S_{out} میشه احتمال این که تداخل نداشته باشیم ضریب میانگین بسته هایی که دارن از کانال خارج میشن. پس این احتمال موفقیت بودن باعث میشه ما مشخص بشه.

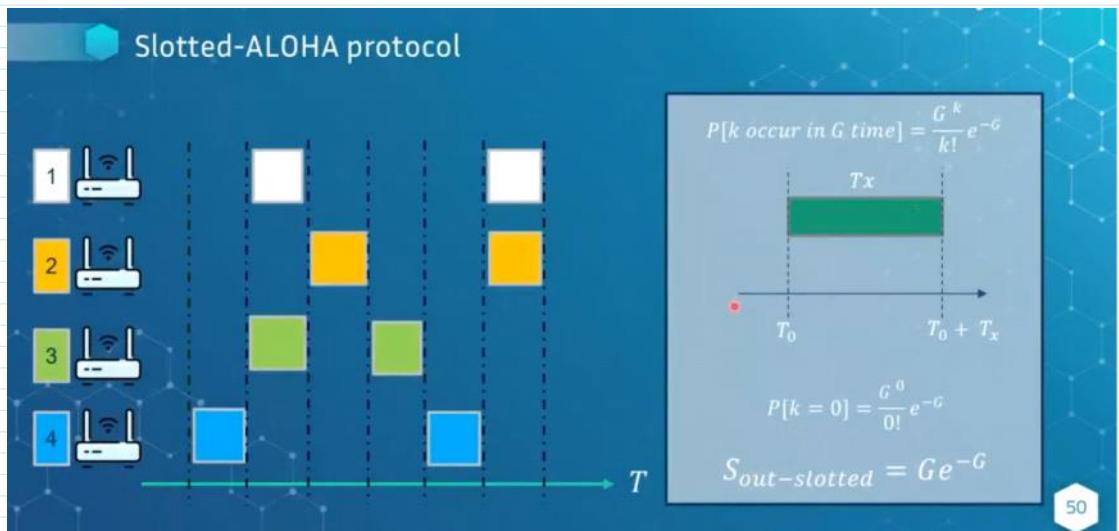
در این مدل هرچه تولید بسته ها بیشتر بشه. میتوان در نظر گرفت با زیاد شدن بسته ها احتمال تداخل بیشتری نیز داریم.



احتمال رخدادن k اتفاق در متوسط زمانی G را با این فرمول بدست می‌اریم وقتی $0 = k$ می‌داریم یعنی هیچ تداخلی رخ نده. یعنی میخوایم متوسط زمانی بدست بیاریم که در اون مدت کاربران دیگه نباید ارسال داشته باشند.

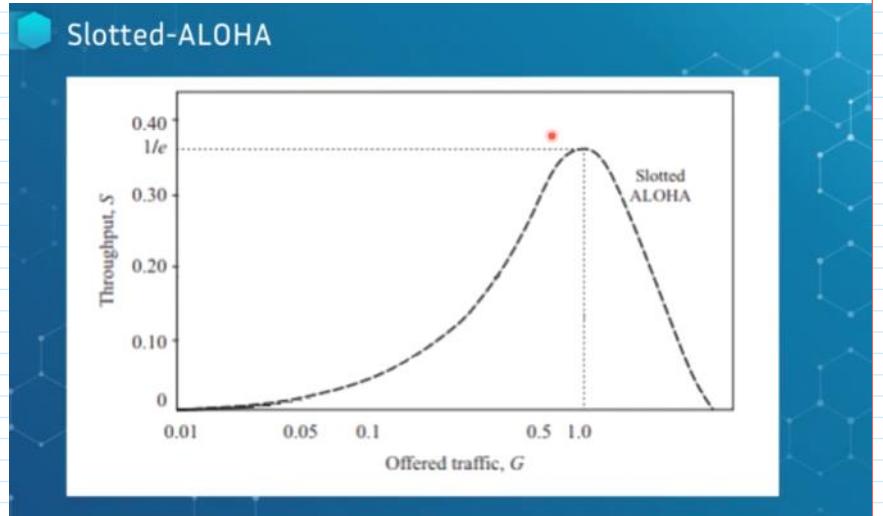


در این نمودار محور x تعداد تلاش هایی که در واحد زمان انجام میشه رو میبینیم و محور y اشم گذرده هستش. میبینیم که G ما وقتی از 0.5 بیشتر بشه گذرده کم میشه و در 0.5 حد اکثره. حالا چرا از 0.5 بیشتر شه کم میشه؟ چون تعداد بسته ها زیاد میشه و تعداد تداخل و old packet بیشتر میشه و تداخل پشت تداخل ایجاد میشه و گذرده کم میشه.

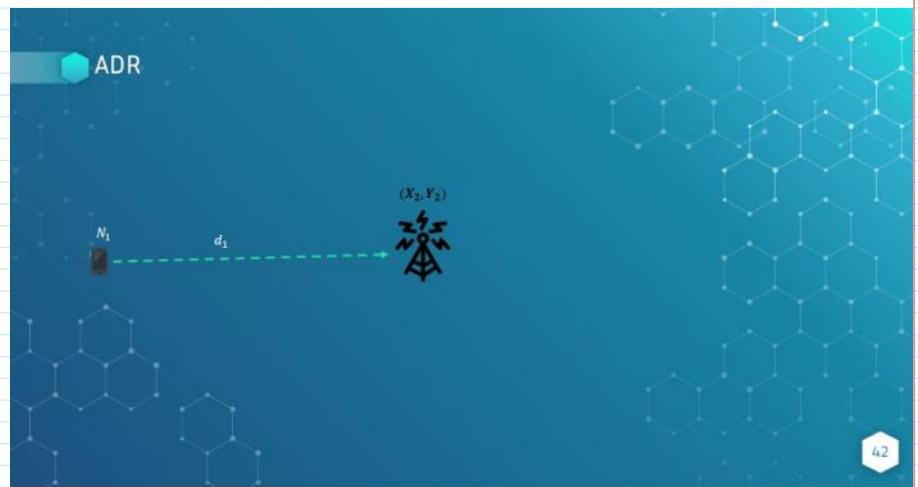


در این حالت وقتی هر گره ای بسته ای برای ارسال در کاتال داشت، تو یه زمانی خاصی حق ارسال دارن که از قبل تعیین شده است. در این صورت موقعی که یک Node وقتی ارسال میکنه برای این که با موقوفیت به دست گیرنده برسه Node های دیگه فقط در آن زمانه که نباید بسته ای را رو ارسال کنند. پس درنهایت گذرهی Slotted Aloha میشه⁶.

فرق این **TDMA** با **Slotted-ALOHA** جیه؟! بسته ها زمانی مخصوص کسی نیست. واین Node ها با هماهنگی هم دارن ارسال میکنن. ولی در حال TDMA یک سنکرون سازی مرکزی انجام میشند.



گرددی ۳۷ درصدی Slotted ALOHA در این شکل قابل مشاهده است.



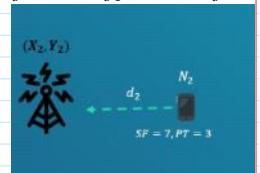
حالا میخوایم در مورد ADR صحبت کنیم

در لایه فیزیکی دیدیم SF های مختلف مزایای مختلف داشتن. در شکل فوق یک Gateway با یک Node که در یک فاصله قرار داره و یک بسته رو ارسال میکنه. SNR بعد دریافت بسته میاد رو محاسبه میکنه و نسبت سیگنال به نویز رو میسنجه.

حالا فرض کیم SNR پایین باشه. یعنی عوامل نویز خیلی تاثیرگذار بوده و پیام به خوبی به دست Gateway نرسیده و کامل دریافت نشده. در این موقع Config یک Node برای Gateway میاد



حالا یک Node در یک فاصله دیگه قرار داره و میبینیم SNR اش خوبه. در این موفق Config براش یک Config ای میفرسته که SF و توان کمتری داشته باشه.



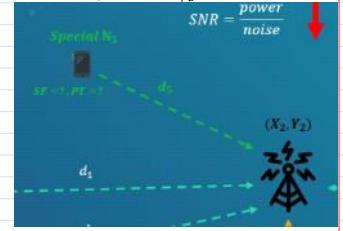
حالا یک Node با یک فاصله دیگه داریم که در نزدیک 1 Node قرار داره. میبینیم SNR اش بد و براش Config ای ارسال میکنیم. ولی این Config مشابه Config اول نیست چون اگه مشابه میشد باعث میشد تداخل داشته باشن



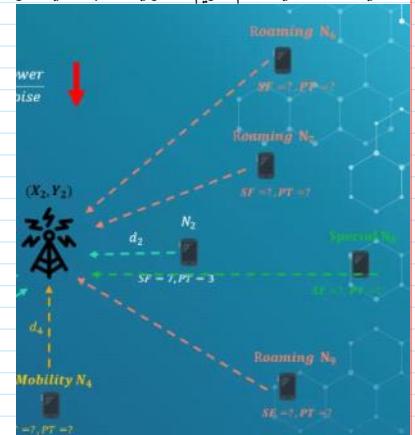
حالا یک Node داریم Mobility داره باید اینجا هوشمندی به خرج بدیم چون نویز اش با دستگاه ثابت متفاوته



حالا یک Node داریم که Special هستش و میخواهد SNR اش همیشه ثابت باشد و اگه کم بشه مارو جریمه میکنه. ما باید برای این Config خاصی رو ارسال کنیم

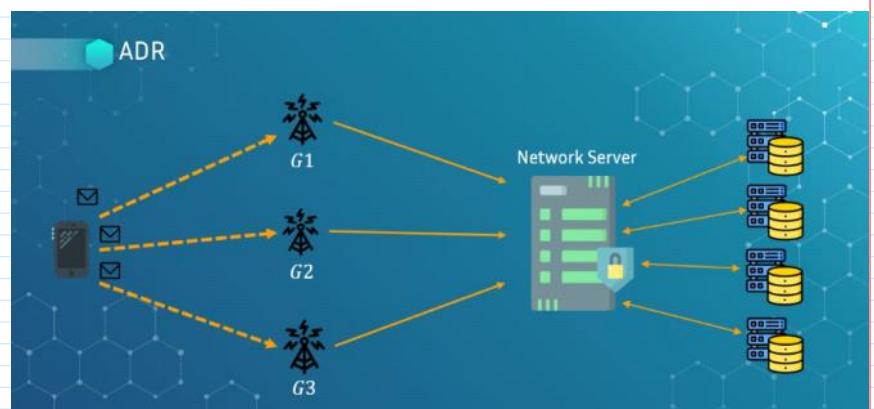


حالا به Node دیگه هم داریم که از یک شبکه دیگه اومده. برای این Node هم باید یک Config جداین با اهداف مشخص در نظر بگیریم (Roaming):

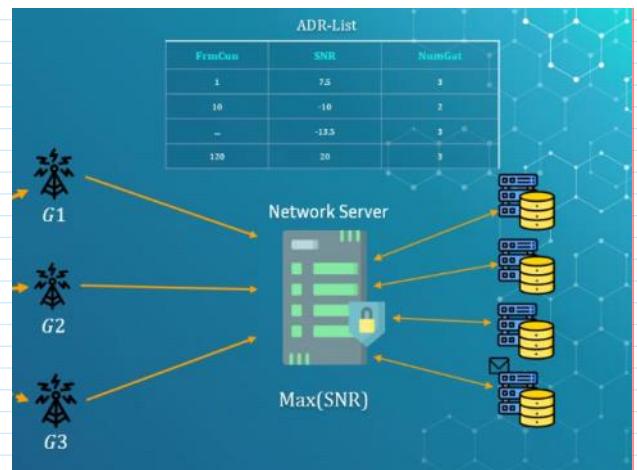


ما باید توان رو برای همه این Node ها به نحو هوشمندانه توزیع کنیم و نیازمندی همه رو برطرف کنیم. پس به یک الگوریتم هوشمند نیاز داریم که همه این کار را برآورده باشد.

پس الگوریتمی تحت عنوان ADR موجود هست که میاد اینرا تنظیم میکنه. حالا میخوایم بینینم چطوری کار میکنه.



در معماری شبکه LoRaWAN یه همچنین چیزی داریم. Node ما وقتی بسته ای میفرسته همه Gateway های در دسترس میتوان دریافت کنن. به محض این که بسته توسعه Network Server دریافت میشه میاد SNR و زمان دریافت بسته، فرکانس و ... را محاسبه میکنه. حالا این اطلاعات رو همراه با بسته به Network Server میفرسته. NS یک Payload مشابه از 3 تا 3 Gateway دریافت میکنه. باید یکی رو ذخیره کنه و بقیه رو دور بندازه. برای این کار میاد بررسی میکنه SNR کدوم Gateway بالاتر دارد. اون بسته که SNR بالاتری دارد رو نگه میداره و بقیه رو میندازه دور.



NS به ازی هر Node یک جدول 20 سطری نگه میدارد و 3 تا ستون دارد که 20 تا بسته اخیر را تو ش قرار میده. از این جدول تصمیم میگیره که مثلا اگر تو 20 تا اخیر همش منفی بود یک Config دیگه پفرسته تا از SF و توان جدیدی گره ارسال کنه. علاوه بر SNR میاد Frame Counter هم در نظر میگیره. وقتی از 1 میرسه به 10 یعنی یک سری بسته ها اصن اون وسط گم شده.

الگوریتم پیشفرض LoRaWAN به این صورته. میاد Max 20 بسته اخیر را میگیره و میرزه توی SNR_{max}. بعد مینیمم SNR که از آخرین بسته داریم رو داخل SNR_{req} میرزیم. بعد SNR_{margin} رو طبق رابطه بذست میایم و داریم:

```

ADR   SNRm ← max(SNR of last 20 frames)
      SNRreq ← demodulation floor(current data rate)
      SNRmargin ← (SNRm - SNRreq - 10)
      step ← floor(SN Rmargin/3)
      while steps > 0 and SF > SFmin do
          SF ← SF - 1
          steps ← steps - 1
      while steps > 0 and TP > TPmin do
          TP ← TP - 3
          steps ← steps - 1
      while steps < 0 and TP < TPmax do
          TP ← TP + 3
          steps ← steps + 1
    
```

بریم با یک مثال بررسی کنیم: فرض کنیم آخرین بسته SF=9 و TP=14 داشته باشه. طبق محاسبات جلسه قبل میفهمیم SNR_{min}=-12.5 میشه. بزرگترین SNR هم 10 اگه داشته باشیم داریم:

```

ADR   SNRm ← max(SNR of last 20 frames)
      SNRreq ← demodulation floor(current data rate)
      SNRmargin ← (SNRm - SNRreq - 10)
      step ← floor(SN Rmargin/3)
      while steps > 0 and SF > SFmin do
          SF ← SF - 1
          steps ← steps - 1
      while steps > 0 and TP > TPmin do
          TP ← TP - 3
          steps ← steps - 1
      while steps < 0 and TP < TPmax do
          TP ← TP + 3
          steps ← steps + 1
    
```

✉ SF = 9 TP = 14
SNR_{MIN} = -12.5
SNR_m = 10
SNR_{req} = -12.5
SNR_{req} = 12 - (-12.5) - 10 = 12.5
step = 4
step = 4 3 2
SF = 9 8 7
step = 2 1 0
TP = 14 11 9
✉ SF = 7 TP = 9

توضیح استاد: کنترل توان در تمام شبکه های بی سیم انجام میشه ولی در IoT و LoRaWAN حساسیت و آزادی بیشتری داریم. ADR میاد SF و توان رو برآمون تنظیم میکنه. به طور حسی بخواهیم بگیم این کارو میکنه که وقتی SF و TP بالاست و وضعیت خوبه میایم کمیش میکنیم. حالا این که چطوری کمیشون کنیم کدوم اول کم بشه بایستی تحلیل بشه. یکی از موضوع های باز در LoRaWAN اینه که ADR چطوری عمل بکنه. این ADR توسط LoRaLicence طراحی شده ولی خب لزومی نیست اینطوری باشه. مثلا اینجا 20 تای اخیر را داره در نظر میگیره یکی ممکنه با 40 تا بره. ممکنه برآمون سوال بشه توی Margin چرا 10- بشه. اون Heuristic میایم منهای یک عدد دیگه میکنه. برای تقسیم بر 3 هم همینطور. در آخر وقتی Step منفی میشه باید TP زیاد بشه.

ALOHA protocol implemented on LoRaWAN

- ❖ λ represents all the packets sent to the network by all the LoRa devices in the same coverage area, so we can rewrite λ as:

$$\lambda = N \cdot \lambda_i$$

$\lambda_i = \text{Represents the throughput of one device}$
 $N = \text{The number of devices on the coverage area}$

- ❖ Each packet sent by one LoRa device, will select randomly one channel from the 8 available, so the total throughput sent by one LoRa devices per channel will be:

$$\lambda_{i \text{ per channel}} = \frac{\lambda_i}{8 \text{ channel}}$$

55

ما در شبکه LoRaWAN 2 تا باند فرکانسی داریم که هر کدام از باند ها 4 تا کاتال داره. پس مجموعاً 8 تا کاتال داره. ای که در هر کاتال داره رو محاسبه کنیم باید اون load را تقسیم بر 8 کنیم.

فرض کنیم یک بسته داریم $= 16B$. Payload 8 تا کاتال فرکانسی هم داریم با پهنای باند $125kHz$. طبق تحلیل هایی که از گذشته داشتیم اگر یک Node بخود حداکثر بسته رو بخواهد تو یه روز بفرسته میتوانه 13 هزار تا بسته بفرسته. این تحلیل برای 4 تا کاتاله که برای 8 تا دویابر میشه و میشه 26 هزار تا. حالا اگر تقسیم بر 86400 یعنی تعداد ثانیه های در روز بکنیم میشه 0.302 یعنی هر 3 دهم ثانیه یک بسته عبور میکنه. حالا بخواهیم برای هر کاتال در نظر بگیریم تقسیم بر 8 میکنیم میشه 3 صدم:

Maximum throughput per end-device and per channel

- ❖ Considering packets of 16B payload, a bandwidth of 125 kHz and two available sub-bands on LoRaWAN (Band0 from 868.1 MHz to 868.5 MHz and Band1 from 867.1 MHz to 867.9 MHz).

	Max. daily packets per end-device [packets/day]	Max. λ_i per end-device [packets/sec]	Max. λ_i per channel [packets/sec]
SF7	26122	0,302	0,037
SF8			
SF9			
SF10			
SF11			
SF12	1080	0,012	0,001

$\text{Max. } \lambda_i = \frac{26122}{86400} = 0,302$

56

Table 3: Theoretical Ton-the-air and maximum number of daily packets [10].

حالا بخواهیم یک تحلیلی انجام بدیم. میدونیم متوسط بسته هایی که به یک کاتال ارسال میشه زمان ارسال بسته ضربدر نرخ ورود. در ALOHA دیدیم وقتی $G=0.5$ بداریم گذردهی حداکثر 0.18 را داریم.

Maximum throughput per end-device and per channel

$$G = T_{tx} \times \lambda$$

$$S(G = 0.5) = 18.4\%$$

$$G_{SF7} = 0.302 \times 0.037 = 0.011174 \quad G_{SF7} < 0.5 \quad 0.5 - G_{SF7} = 0.488826$$

متوسط بسته هایی که در SF7 ارسال میشه رو میبینیم خیلی کمه و یعنی ظرفیت زیادی از کاتال خالی میمونه.

❖ We can obtain the maximum number of end-devices N.

$$N = \frac{0.5}{T_{tx} \times \lambda}$$

We can obtain the maximum number of end-devices in two different Scenario:

$$G_{SF7} = 0.302 \times 0.037 = 0.011174 \quad G_{SF7} < 0.5 \quad 0.5 - G_{SF7} = 0.488826$$

Scenario 1: maximizing the packets send by the end-devices

We can obtain the maximum and throughput of network.

0.5

Scenario 2: considering 1 packet per end-device and per day.

سناریو اول: ماکسیمم packet ارسال داشته باشیم و ظرفیت 100 درصدی داشته باشیم.

سناریو دوم: هر Node در روز یک بسته ارسال کنند و load کانال ماکسیمم باشه اون موقع چقدر میتوانه پوشش بده.

توضیح استاد: از وقتی که وارد ALOHA داشتیم، قبلاً به Node بگیم SF و PT اش چی باشد. که احتمال این که دچار Collision نشود من گیرنده بتونم این رو دریافت کنم.

حالا یه لایه اومدیم بالا. Node های دیگه هم داریم Interference داریم. ها فقط وقتی ارسال میکنن که Packet ای در حال ارسال داشته باشن. پس باید محاسبه کنیم هر Node در روز چندتا بسته میتوانه ارسال کنه.

حالا میخوایم بینینیم ماکسیمم Node های End device throughput از ALOHA میروشه رو بفهمیم. پس 2 تا سناریو تعریف میشه:

سناریو اول: تعداد Packet هایی که توسط هر End device ارسال میشه بیشترین مقداری باشه که Duty cycle اجازه میده.

سناریو دوم: 1 پکت در روز توسط end-device ارسال بشه

قطععاً در سناریو دوم تعداد دیوایس بیشتری میشه پوشش داد. چون در سناریو اول دائم هی ارسال میکنه و به خاطر duty cycle صبر میکنه.

حالا میخوایم سناریو اول رو توضیح بدیم:

Maximum throughput per end-device and per channel Scenario 1				
	Max. daily packets per end-device [packets/day]	T _{on the air} [sec]	Max. λ _i per channel [packets/sec]	N
SF7	26122	0,302	0.00825	198
SF8				
SF9				
SF10				
SF11				
SF12	1080	0,012	0.201625	198

Table 4: Theoretical Ton-the-air and maximum number of daily packets [10].

$$T_{tx_SF7_125kHz_16B} = 0.066$$

$$\lambda_i \text{ per channel} = \frac{0.066}{8} = 0.00825$$

$$N = \frac{0.5}{0.302 \times 0.00825} = 198$$

ماکسیمم تعداد End device رو داریم در حالی که هر Device میتواند در یک روز رو میفرسته با فرض حدکثر بودن throughput در اینجا ما میخوایم کانال رو مaks در نظر بگیریم.

(دقیقه 45)

حرف استاد: Time on air وابستگی داره به میزان بایت Packet و SF. بر اساس این یک جدول تهیه میکنیم که Quantify کنیم پارامتر های LoRaWAN رو. میبینیم چه تعداد Node میتوان در شبکه ارسال پکن.

ما رو Time on air داشتیم:

Maximum daily packets on LoRaWAN															
❖ Duration of the packet transmission															
$T_{on_the_air} = (N_{payload} + N_{preamble}) \times T_{symbol}$															
$T_{symbol} = \frac{2^SF}{BW}$															
$N_{payload} = 8 + \max \left[\text{Cell} \left(\frac{8PL - 4SF + 28 + 16 - 20IH}{SF - 2DE} \times \frac{CR + 4}{4} \right) \right]$															
$N_{preamble} = 8$															
															
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="2">Description</th></tr> </thead> <tbody> <tr> <td>SF</td><td>Spreading Factor</td></tr> <tr> <td>PL</td><td>Payload Length</td></tr> <tr> <td>CRC</td><td>The use of the Cyclic Redundancy Check (CRC), 1 when enabled 0 when not</td></tr> <tr> <td>IH</td><td>The use of explicit header or not (IH, 1 when enabled 0 when not)</td></tr> <tr> <td>CR</td><td>Coding Rate</td></tr> <tr> <td>DE</td><td>The Low Data Rate Optimization (DE, 1 when used 0 when not)</td></tr> </tbody> </table>		Description		SF	Spreading Factor	PL	Payload Length	CRC	The use of the Cyclic Redundancy Check (CRC), 1 when enabled 0 when not	IH	The use of explicit header or not (IH, 1 when enabled 0 when not)	CR	Coding Rate	DE	The Low Data Rate Optimization (DE, 1 when used 0 when not)
Description															
SF	Spreading Factor														
PL	Payload Length														
CRC	The use of the Cyclic Redundancy Check (CRC), 1 when enabled 0 when not														
IH	The use of explicit header or not (IH, 1 when enabled 0 when not)														
CR	Coding Rate														
DE	The Low Data Rate Optimization (DE, 1 when used 0 when not)														

برگردیم به جدول سناریو اول:

Maximum throughput per end-device and per channel Scenario 1				
	Max. daily packets per end-device [packets/day]	T _{on-the-air} [sec]	Max. λ _i per channel [packets/sec]	N
SF7	26122	0,302	0.00825	198
SF8				
SF9				
SF10				
SF11				
SF12	1080	0,012	0.201625	198

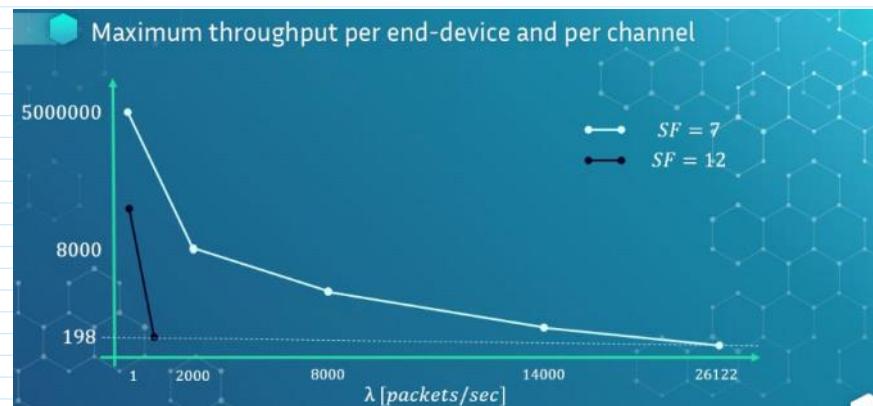
Table 4: Theoretical Ton-the-air and maximum number of daily packets [10].

یک بسته در SF7 بود 0.066 Time on air بود که تقسیم بر 8 تا کانال کردیم دیدیم در هر کانال چقدر.

سناریو دوم فرض بود که یه بسته ارسال میشه:

	Max. daily packets per end-device [packets/day]	T _{on-the-air} [sec]	Max. λ _i per channel [packets/sec]	N
SF7	1	0,066	0.00000144	5172000
SF8				
SF9				
SF10				
SF11				
SF12	1	1,613	0.00000144	214149

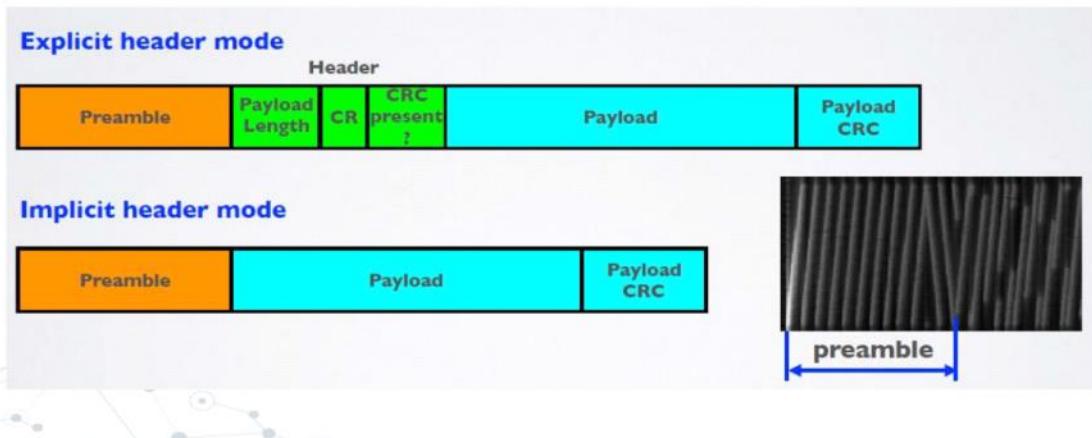
Table 5: Theoretical Ton-the-air and maximum number of daily packets [10].



ددر این نمودار میبینیم وقتی نخ ورود بسته ها 1 بسته در ثانیه باشه 5 میلیون تا Device رو میتوانه در SF=7 پوشش بده و در حالت مаксیمم به 198 میرسه.

LorRa Packet Format

- The LoRa packet comprises of three elements: Preamble, header (optional) and payload.



ما در LoRa به دو نوع میتوانیم packet داشته باشیم:

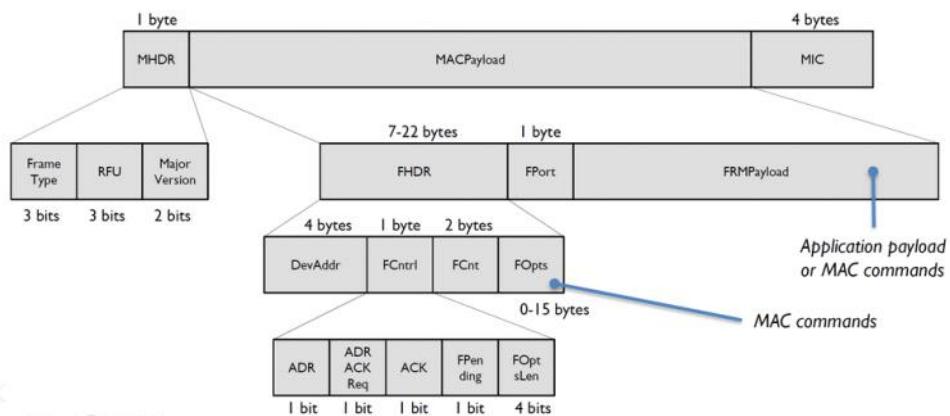
- **Explicit:** اول ارسال میشه که یک سیگنال با فرمت مشخص برای سنتکرون شدن بین گزند و فرستنده است. بعدش اولین پام اینه که میگیم این چیزی که ارسال میکنیم طولش چقدر است. CRC Present یعنی این که روشنی برای تخصیص خط داریم یا نه (0 باشه نیست 1 باشه هست). اگر CRC Present یک باشه اون آخر Payload CRC داریم و گزنه نداریم.
- **Implicit:** میاد Payload length را حذف میکنه و باعث میشه Payload طول ثابتی داشته باشه. ولی بدیش اینه که بسته Flexible نداریم که بتونیم با طول های مختلف داشته باشیم.

پس وقتی بسته ها طول متغیر نداریم میریم سراغ Implicit و طوری طول ثابت روتیین میکنیم که loss نداشته باشیم. ولی وقتی multi sensor داریم یه موقع میخوایم برای 2 تارو ارسال کنیم یه موقع 3 تا اون موقع ما با طول متغیر سر و کار داریم و رو انتخاب Explicit header mode میکنیم.

از نظر فیلد پس ما Preamble و header و payload به طور کلی و Payload CRC تهش میتوانه برای تصحیح خط اضافه بشه.

LorRa Frame Format

- The LoRa packet comprises of three elements: Preamble, header (optional) and payload.

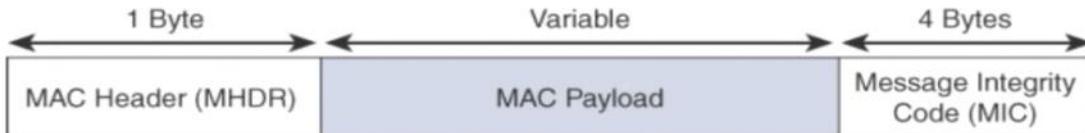


که 4 هستش برای کارای کنترلی هستش مثل DevAddr که آدرس دیوایس هستش با 4 بایت. هم میشه اون داده که میخوایم از لایه بالاتر بیایم ارسال کنیم.

LoRaWAN- MAC Layer



- LoRaWAN- MAC layer
 - LoRaWAN messages, have a
 - The MAC payload size depends on the frequency band and the data rate, ranging from 59 to 230 bytes for the 863-870 MHz band and 19 to 250 bytes for the 902-928 MHz band.



MAC Payload متغیره. ولی حالا باید زیاد بذاریم یا کم بذاریم؟ در ارتباطات IoT و با توجه به ویژگی های LoRaWAN باید کم بذاریم.

Direct Communication Between Devices

- The LoRaWAN protocol does not support direct communication between end nodes.
- If you want direct communication between LoRa devices without the use of gateways, use the RadioHead Packet Radio library for embedded microprocessors. It provides a complete object-oriented library for sending and receiving packet sized messages via a variety of radios such as LoRa on a range of embedded microprocessors:
<https://www.airspayce.com/mikem/arduino/RadioHead/>

RadioHead does not have an official GitHub repo but several people have cloned the Radiohead library on GitHub.

ما در LoRaWAN Direct Communication نباریم. در 802.15.4 full function device داشتیم میتوانستن حق بدون Coordinator با هم صحبت کنن. در LoRaWAN همچنین چیزی تعریف نشده ولی یه سری گروه اومدن یه چیزی ساختن تو گیت هایم گذاشتن که 2 تا Node چجوری peer-to-peer صورت LoRaWAN صحبت کنن.

LoRaWAN- Security

- LoRaWAN considers two layers of security, one for the network and another for the applications.
- Each end-device has key assignments done by device manufacturers or the application owners.
 - **Other systems use a single key for encryption and authentication,** compared to LoRaWAN.
- Authentication and encryption are separate, so it is **possible to authenticate packets and provide integrity protection.**

J. Silva, et al, *LoRaWAN - A Low Power WAN Protocol for Internet of Things: a Review and Opportunities*, 2th Int. Multidisciplinary Conf. on Computer and Energy Science, 2017

بخش امنیت همیشه مهمه. LoRaWAN بخلاف بقیه پروتکل ها مثل 802.15.4 از همون اول برای IoT درست شد. سعی شده دیزاینچ سبکی داشته باشه مثلا هیچ Association ای نداشتم بین گره و Gateway.

بحث Low Power هم به دیزان ربط داره هم به duty cycle هم این که بایت های کمی برای ارسال داریم. من وقتی بخواهم امنیت رو برم بالا باید محاسبات رو برم بالا که پیام محترمانه بمونه و در سطح Application کسی اون وسط نتونه بخونه. اگر الگوریتم های سنگین استفاده کنیم Node کشش نداره چون از لحاظ سخت افزاری ضعیفهن. ولی نمیشه در نظرش هم نگرفت چون وقتی همه گیر بشه محلی برای آسیب پذیری و ممکنه سو استفاده بشه ازش.

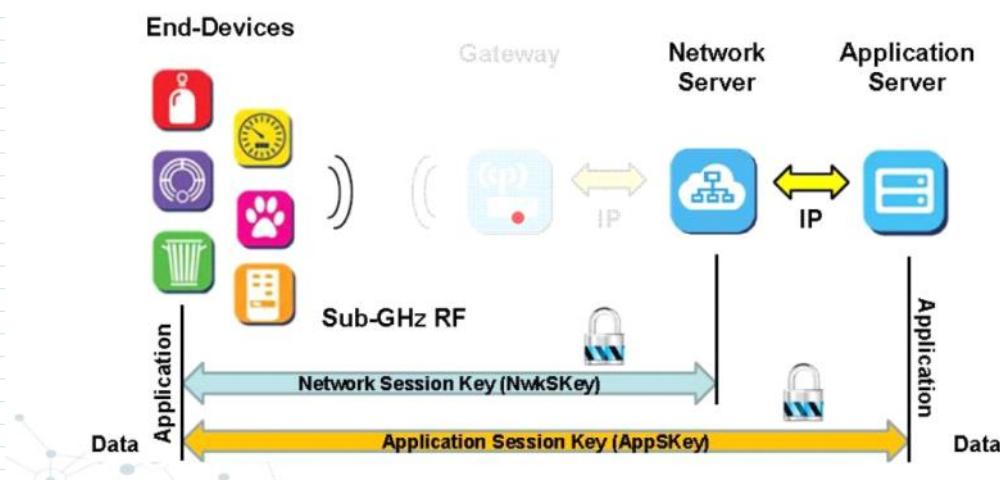
لایه امنیتی درست شده:

- سطح اپلیکیشن
- سطح شبکه

هر یک کلیدی بهش assign میشه که با hard code میشه (سازنده) یا میشه config کرد روش (صاحب اپلیکیشن).

LoRaWAN- Security

- Logical data Flow:



در معماری ما یک سری End-device که ارسال میکن به یک سری Gateway که حالا اینجا بدونه کشیده شده ارتباط لایه فیزیکی LoRaWAN فقط از physical layer تا Gateway عه و از این به بعد منطقی نیست که LoRa باشه چون به برق شهر وصلن از اون به بعد و با TCP/IP بهتره ارسال بشه. از طرفی ها کلی داده از کلی Node دریافت کردن و منطقی نیست این همه داده با LoRaWAN ارتباط داشته باشه. اصطلاحا به شبکه Gateway به بعد میگن Backhaul که ارسال میکن به Network server و اونم به Application

حالا که در مورد امنیت صحبت میکنیم 2 سطح امنیتی داریم

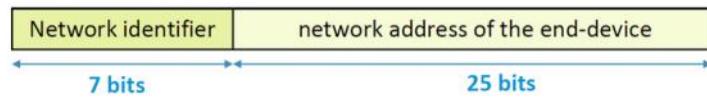
- **Gateway** ممکنه برای ما نباش ه. مثلا در آینده اپراتور های IoT داریم. یک node ای رو میخیریم که gateway داره و Device address میتونه تنظیم شه که فقط از دیواپس های خاصی پکت دریافت کنه.

با توجه به این که gateway از آن مانیست مهمه که ارتباطات بینمون Confidential gateway باشند. یعنی Network Session Key (NwkSKey) موجود میشند و یک رمزگاری این بین انجام میشود.

- حقیقت ممکنه Network Server هم برای مانیشا استفاده میکنیم. در اینجا از Application Session Key (AppSKey) استفاده میشند. این کار باعث میشود بخشی از packet صرف رمزگاری بشود و بخش کمتری صرف payload بشود.

IoT wireless technologies overview

- **End-device address (DevAddr):**



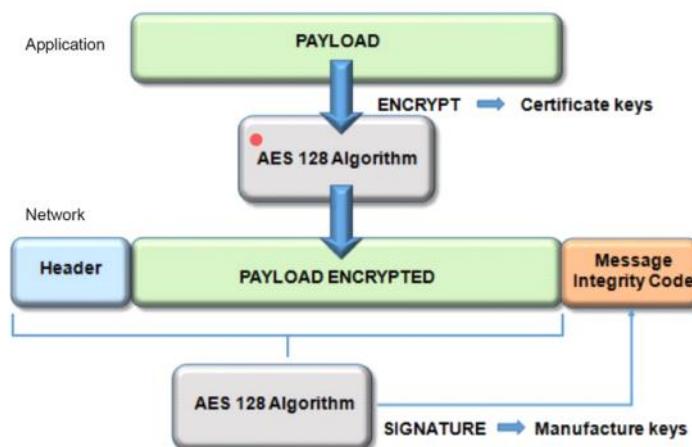
- **Application identifier (AppEUI):** A global application ID in the IEEE EUI64 address space that uniquely identifies the owner of the end-device.
- **Network session key (NwkSKey):** A key used by the network server and the end-device to calculate and verify the message integrity code of all data messages to ensure data integrity.
- **Application session key (AppSKey):** A key used by the network server and end-device to encrypt and decrypt the payload field of data messages.

از نظر آدرسی که استفاده میشود یک DevAddr داریم. قطعاً ممکنه تکراری هم باشد. در یک محدوده مشخص اگر نخوایم تکراری باشند میتوانیم 20 بایت برای فرستنده و 20 بایت برای گیرنده استفاده کیم. ولی خب ما کلا مثلاً 200 بایت داریم که 40 بایت میشود آدرس گیرنده و فرستنده که به صرفه نیست. پس 4 بایت استفاده میکن که توزیعشو تو شکل مییند.

AppEUI: بر اساس یک استاندارد اپلیکیشن های مثل شهر هوشمند مثل مدیریت بازیافت، مدیریت روشنایی معابر... تفکیک میشون.

LoRaWAN- Security

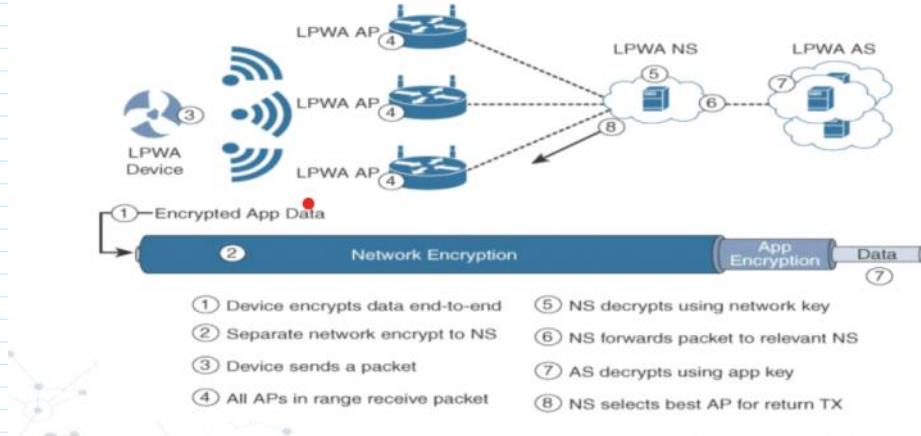
- LoRaWAN endpoints must implement two layers of security:



(48 دقیقه : صدا قطعی)

LoRaWAN- Security

- LoRaWAN endpoints must implement two layers of security, protecting communications and data privacy across the network.



LoRaWAN- Security

- LoRaWAN endpoints attached to a LoRaWAN network must get registered and authenticated. This can be achieved through one of the two join mechanisms:
- **Activation by personalization (ABP):**
 - Endpoints don't need to run a join procedure as their individual details, including DevAddr and the NwkSKey and AppSKey session keys, are preconfigured and stored in the end device.
 - The same information is registered in the LoRaWAN network server.
- **Over-the-air activation (OTAA):**
 - Endpoints are allowed to dynamically join a particular LoRaWAN network after successfully going through a join procedure. The join procedure must be done every time a session context is renewed.
 - During the join process, which involves the sending and receiving of MAC layer join request and join accept messages, the node establishes its credentials with a LoRaWAN network server, exchanging its globally unique DevEUI, AppEUI, and AppKey.
 - The AppKey is then used to derive the session NwkSKey and AppSKey keys.

جزی که مرتبط با Security هم هست بحث Activation نود هاست. ما یک Node را می‌داریم داخل فضای سیز امیرکیم. میخواهیم بینیم چطوری میشه آدرس دهی رو مشخص کرد
دو نوع مود داریم:

- ABP: شخصی سازی End Point ها نیاز نیست یک روالی انجام بدن برای این که خودشون رو برای Gateway ها معرفی کنن.
- OTAA: روی هوا این کار انجام میشه. یعنی Dynamically End point خودشون رو به Gateway های اطراف معرفی میکنن و DevAddress و.. رو میگن (همون لحظه که نود فعال شد).
- نکته مهم در این قسمت اینه که یک Renew Session ای Join Procedure که یک هر موقع که باید انجام بشه. در این فرایند Join شدن یک سری اطلاعات مثل DevEUI, AppEUI, AppKey منتقل میشه.

LoRaWAN- OTAA Activation Method

- In OTAA, a device is given a DevEUI, an AppEUI and an AppKey. The AppKey is used to generate the session keys, NwkSKey and AppSKey.
- To activate, the device sends a join request and uses the join response to derive the session keys NwkSKey and AppSKey
- The device may store those keys and continue to use them to communicate. If they are lost or the network chooses to expire them, the device must re-join to generate new keys.

پیداست که OTAA سریع تر از قبل نیازی به کانفیگ نیست. ولی یک سریار داره. هی Node ها وصل میشن خارج میشن باید بریم
ولی وقتی کشاورزی همش از قبل هست ABP بهتره که از اول کانفیگ بشه

LoRaWAN- ABP Activation Method

- Pros:

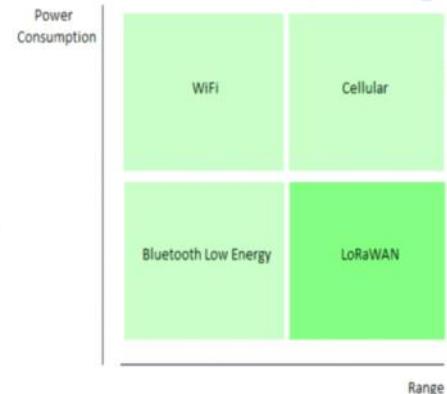
- The device does not need the capability or resources to perform a join procedure.
- The device does not need to decide whether a join is necessary at any point, since it is never necessary.
- No scheme is necessary to specify a unique DevEUI or AppKey

- Cons:

- The scheme must be secure to prevent the keys being obtained or derived by rogue parties.
- If the device is compromised at any time, even before activation, the keys may be discovered.
- Network settings cannot be specified at join time.
- Events that warrant a change of keys (for example, moving to a new network, the device being compromised, or the keys being expired) require a re-programming of the device..

LoRaWAN- Battery Lifetime

- Synchronization network usually **consumes** significant **energy**. In LoRaWAN, nodes are asynchronous and communicate via **events** or in **prescheduled opportunities**.
- The **ADR** (Adaptive Data Rate) scheme is used for LoRa network infrastructures for manage the individual **data rates** and **maximize** the **battery life** of each connected device through RF output.
- A recent research study performed by **Scientific Research Publishing**, Inc revealed that LoRaWAN showed an **advantage of 3 up to 5-fold in the energy economy compared** to all the others LPWAN technologies.



حتی رنج بالاتر از Cellular هم میتوانه پوشش بده و یکم توی شکل Range اشو بیشتر کرد.
LoRaWAN اتمام -----

SigFox -----

SigFox

- SigFox
 - The Sigfox technology was developed in 2010 by the startup Sigfox (in Toulouse, France),
 - Sigfox operates and commercializes its own IoT solution in 31 countries and is still under rollout worldwide owing to the partnership with various network operators

هر Sigfox LoRaWAN زادگاهشون از فرانسه بوده.
این اطلاعات برای 2 سال پیشہ الان قطعاً از 31 کشور بیشتره

سیگفاکس پارترشیپ داره با اپراتور های شبکه های متفاوت. یعنی چی پارتر شیپ داره؟
در LoRaWAN همین امروز تصمیم بگیریم LoRaWAN اختصاصی خودمون رو داشته باشیم، میریم Gateway اشو میخیریم Node هاشم داریم و بهم وصل میشن. Network server هم با خودمون مینویسم یا Cloud based App هم اوکی میکیم و شبکه اختصاصی خودمون رو مینوینم تشکیل بدیم.
مینوینم چندین اپراتور LoRaWAN داشته باشیم. چرا؟ چون یک استاندارد و پروتکل هست که جریات مشخصه. (البته لایه فیزیک اش پینت شده و باید بخریم) بعد یک اپراتور تشکیل میدیم و تعریفه براش مینذاریم.
اما در سیگفاکس خودش اپراتوره و میاد ایران مجوز میگیره و شبکه خودشو درست میکنه. پس میشه گفت سیگفاکس یک Communication Provider عه و ارتباط رو برقرار میکنه ولی Protocol Design یک LoRaWAN هستش.

SigFox: Technical Specification

- Binary phase-shift keying (BPSK) modulation in an ultra-narrow band (100 Hz) sub- GHZ ISM band carrier
- It causes SigFox experiences
 - very low noise levels,
 - leading to very low power consumption,
 - high receiver sensitivity,
 - and low-cost antenna design
 - all, at the expense of maximum throughput of only 100 bps.

SigFox: Technical Specification

- The downlink communication can only occur following an uplink communication.
- The number of messages over the uplink is limited to 140 messages per day.
- The maximum payload length for each uplink message is 12 bytes.
- However, the number of messages over the downlink is limited to 4 messages per day, which means that the acknowledgment of every uplink message is not supported.

در SigFox میگه جدا از بحث Duty cycle و باقی پارامتر ها حق ارسال بیشتر از 140 مسیج uplink در روز رو نداریم.

Payload length اش نسبت به LoRaWAN کمتره
این که 140 تا مسیج میشه ارسال کرد ولی 4 تا میشه دریافت کرد. این یعنی مکانیزم ACK رو بخوایم بیاده سازی کنیم در SigFox نمیشه این کارو کرد. در صورتی که در LoRaWAN قابلیتش وجود دارد.

SigFox: Technical Specification

- Without the adequate support of acknowledgments, the uplink communication reliability is ensured using time and frequency diversity as well as transmission duplication.
- Each end-device message is transmitted multiple times (three by default) over different frequency channels (three by default) chosen randomly.
 -
- This simplifies the end device design and reduces its cost

سنسر SigFox وققی یک چیزی رو اندازه گرفت، وقتی میخواهد ارسال کنه این پیام 3 بار در سه کانال مختلف که به صورت زندم انتخاب شده اند ارسال میشه. این کار یک Duplication به وجود میاره.

multiple times (three
inels (three by default)

reduces its cost

با این کار وقتی 3 تا gateway بخوان ارسال کنن 9 بار ارسال میشه. درسته که ACK نمیگیریم ولی خب به تعداد زیادی داریم ارسال میکنیم و احتمال خطأ خیلی ناچیز میشه.



SigFox

- SigFox
 - The Sigfox technology was developed in 2010 by the startup Sigfox (in Toulouse, France),
 - Sigfox operates and commercializes its own IoT solution in 31 countries and is still under rollout worldwide owing to the partnership with various network operators



یک از رقیب های LoRaWAN است که همزمان با LoRaWAN Close Source هستش

NB-IoT and Other LTE Variations



- Existing cellular technologies,
 - 2G (GPRS, Edge), 3G, and 4G/LTE,
 - are not particularly well adapted to battery-powered devices and small objects specifically developed for the Internet of Things.
- While industry players have been developing unlicensed-band LPWA technologies, 3GPP and associated vendors have been working on evolving cellular technologies to better address IoT requirements.
 - low throughput and low power consumption, and decrease the complexity and cost of the LTE devices.
 - This resulted in the definition of the LTE-M work item.

IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Thing, Cisco press, 2017

اپراتور های موبایل یک صنعت موفق در دنیا هستند. با به وجود آمدن LPWAN های مثل LoRaWAN و NB-IoT اپراتور ها احساس کردن که بازار شبکه رو دارن از دست میدن. به همین دلیل اون گروه هایی که شبکه های سلوولار رو دارن استاندارد هاشو مینویسند، پروتکل هایی که دارن برای دیوایس هایی که باتری ضعیف دارن طراحی نشده اند. در واقع در حالی که صنعت داشت نقش مهندسی برای Unlicenced-band LPWA 3GPP روی این کار کردن که شبکه های سلوولی نیاز های IoT را بهتر پاسخ بده. دنبال این بودن که یک پروتکل رو درست کنن که گزرنده کمی داشته باشد، مصرف توان کم و پیچیدگی کمی داشته باشد و هزینه دستگاه های 4G را کم کنه.

این کار باعث شد یک گروهی تحت عنوان LTE-M درست بشه که M اش Machine to Machine هستش

NB-IoT and Other LTE Variations

- Because the new LTE-M device category was not sufficiently close to LPWA capabilities, in 2015 3GPP approved a proposal to standardize a new narrowband radio access technology called Narrowband IoT (NB-IoT).
- NB-IoT specifically addresses
 - the requirements of a massive number of low throughput devices,
 - low device power consumption,
 - improved indoor coverage,
 - and optimized network architecture.

در سال 2015، 3GPP چندتا استاندارد که IoT هست را تعریف کرد. هدف‌شون این بود که کارهای IoT را در فرکانس‌های لایسننس دار انجام بدن که رقابت بتوان کن. بازار Unlicensed

چون در دست پروتکل‌های IoT بودولی خوب Reliable نیست و برای یک سری کاربرد‌ها که خیلی برامون مهمه مشکل برامون به وجود می‌آید.

در دوچرخه‌های بی‌بود از شبکه‌های سولوی استفاده می‌کنند و اطلاعات کمی رو منتقل می‌کنند. Low Power

Sigfox و LoRaWAN برای بی‌بود خوب نیست. چرا؟ چون Reliable نیست و می‌بشه سمت شبکه‌های سولوی NB-IoT بایم که reliable باش.

• بایم سراغ شبکه‌های IoT را بگیری که دوچرخه هست بتونیم با حرکتش انرژی الکتریک تولید کنیم. با این کار دیگه نیاز کمتر به شارژ داریم و کار راحت تر می‌شود.

• با توجه به این که دوچرخه هست بتونیم با حرکتش انرژی الکتریک تولید کنیم. با این کار دیگه نیاز کمتر به شارژ داریم و کار راحت تر می‌شود.

الان بی‌بود یه کاری که می‌کنیه اینه که وقتی 10 درصد مونده باتری تموم پشه متصلی می‌آونا که با تریشون کمه رو جمع می‌کنن و بر می‌گردونن.

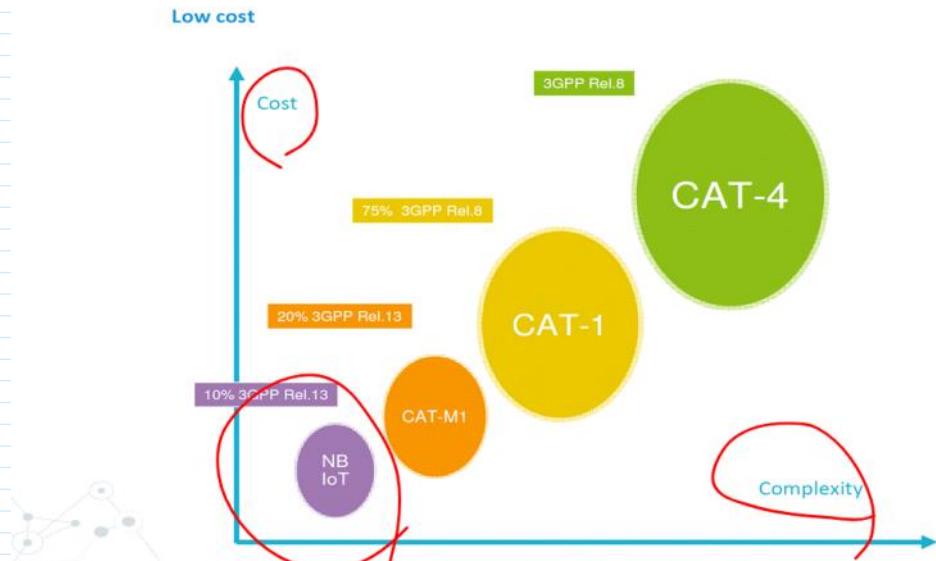
NB-IoT and Other LTE Variations

- Mobile vendors and service providers are not willing to lose leadership in this market of connecting IoT devices

- Licensed LPWAN:
 - LTE Cat 0
 - LTE-M
 - NB-IoT

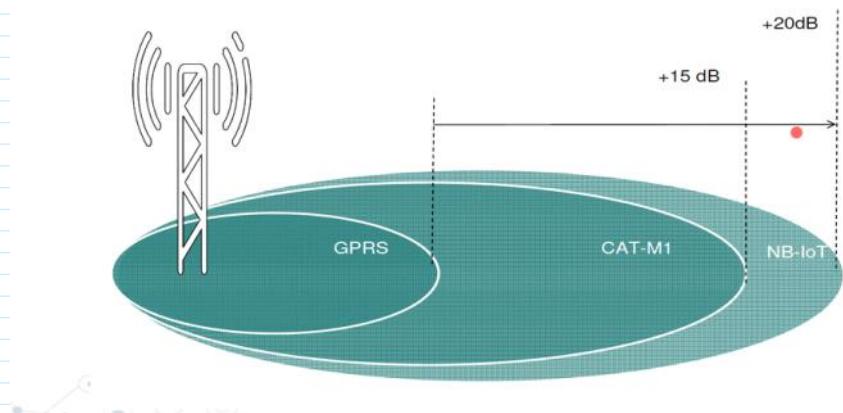
از بین 3 تا LPWAN لایسننس دار که ارائه شده معروف ترینش NB-IoT هستش.

NB-IoT and Other LTE Variations



از نظر Cost و Complexity که در جایی که دورش خط کشیدیم قرار بگیریم. ممکن است با این کار دیتاریت و اینوارو از دست بدیم ولی خوب میصرفه. به همین دلیل مهم ترین پروتکل LPWAN لایسنس دار میشه NB-IoT

NB-IoT and Other LTE Variations



Link Budget اش هم به نحوی هست که Coverage خوبی نسبت به بقیه شون دارد.

LTE Cat 0



- The first enhancements to better support IoT devices in 3GPP occurred in LTE Release 12.
- A new user equipment (UE) category, Category 0, was added, with devices running at a maximum data rate of 1 Mbps.
- Category 0 includes important characteristics to be supported by both the network and end devices.
- Meanwhile, the UE still can operate in existing LTE systems with bandwidths up to 20 MHz.

از لحاظ تاریخی بخواهیم بررسی کنیم: اولین ریلایز، ریلایز 12 بود که از لحاظ دسته نبندی UE، دسته نبندی 0 ایجاد شد که دیتابیت تا 1Mbps مکسیمم هستش. Cat 0 یک سری ویژگی داره که هم توسط end-device Network و end-device ساپورت میشه، ما نمیشه فقط بگیم اسلیپ پشه و بلند پشه، باید شبکه هم این رو ساپورت کنه و وقتی در مورد یک پروتکل صحبت میکنیم هم در سمت end device هم در سمت network باید حواسمن بهش باشه. علاوه بر این، یک Software Update Node هایی که در Unlicenced سیستم LTE کار کنن. یعنی تاکید این بود که با یک Software Update بتونیم مستول Base Station را بتوانیم بهش باشیم.

LTE Cat 0



These Cat 0 characteristics include the

- Power saving mode (PSM):**

- being similar to "powered off" mode, but the device stays registered with the network. By staying registered, the device avoids having to reattach or reestablish its network connection.
- with PSM, a device can be practically powered off but not lose its place in the network

- Half-duplex mode:**

- This mode reduces the cost and complexity of a device's implementation because a duplex filter is not needed.
- Most IoT endpoints are sensors that send low amounts of data that do not have a full-duplex communication requirement.

Cat 0 یک از کاربرد هاش پیشنهاد Power off هستش. ما وقتی در گوشی هامون Power off اطلاع میده که به Base station اطلاع میده دارم خاموش میشم و به همین دلیل وقتی بهش زنگ میزنیم بهمون اطلاع میده که موبایل خاموش و بیگامش با این که در دسترس نیستیم متفاوته. مود off رو پس همین الانم داریم. اما نکته مهم اینه وقتی روشن میکنیم همه کارها از نظر Register و انتقال اطلاعات سیمکارت و .. انجام میشه. اما در فاینده Registration یعنی یک سری Task انجام دادن و مار IoT میخواهیم تا جا که میشه کم بشن این Task ها. پس در Cat 0 به بعد Mode ای تعریف شد که PSM هستش و دستگاه خاموش میشه ولی همچنان در شبکه Register می مونه. با این کار وقتی دیوایس روشن شد دیگه نمیخواد reestablish کنه و جایگاهش در شبکه از دست نمیده.

یک ویژگی دیگر Half-duplex هستش. یعنی دیوایس یا داره ارسال میکنه یا دریافت و در آن واحد نمیتوونه جفتشو انجام بده. با این کار هزینه و پیچیدگی کم میشه. اما در Category های دیگه مثل گوشی های موبایل Full-duplex هستش و همزمان هم میشه ارسال کرد هم دریافت.

LTE-M

- Following LTE Cat 0, the next step in making the licensed spectrum more supportive of IoT devices was the introduction of the LTE-M category for 3GPP LTE Release 13.
- LTE-M requires new chipsets and additional software development.
- Commercial deployment is expected in 2017.
- Mobile carriers expect that only new LTE-M software will be required on the base stations, which will prevent re-investment in hardware.

در ادامه گام بعدیش این بود که IoT Device رو برای Licensed spectrum را آزاد کنیم.
انیاز دارن که یک new chipset و software development انجام بشے
اش سال 2017 انجام شد Deployment
Mobile carrier ها میتوانستن با یک Update نرم افزاری این کارا رو بکن و دیگه روی سخت افزارا re-investment نکنن.

LTE M

- These are the main characteristics of the LTE-M category in Release 13:
 - Lower receiver bandwidth:**
 - Bandwidth has been lowered to 1.4 MHz versus the usual 20 MHz, further simplifying the LTE endpoint.
 - Lower data rate:**
 - Data is around 200 kbps for LTE-M, compared to 1 Mbps for Cat 0.
 - Half-duplex mode:**
 - Just as with Cat 0, LTE-M offers a half-duplex mode that decreases node complexity and cost.
 - Enhanced discontinuous reception (eDRX)**
 - This capability increases from seconds to minutes the amount of time an endpoint can “sleep” between paging cycles.
 - A paging cycle is a periodic check-in with the network.
 - This extended “sleep” time between paging cycles extends the battery lifetime for an endpoint significantly.

Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Thing, Cisco press, 2017

در ریلیز 13 LTE-M رو داشتیم باعث شد موارد فوق رو داشته باشیم.
نکته ای که در LTE-M اضافه شد اینه که eDRX رو دارد. در ابراتور ها ما یک paging cycle داریم. یک گوشی موبایل در یک دیقیه چندین بار داره paging انجام میده و به موبایل مدام میگه من اینجا من اینجا، هدفش برای بحثی Handover هستش. وقتی ما حرکت میکیم از Base station خارج میشیم میریم سراغ اون یک و هر یک قدرت beacon رو میسنجن و 1 BS1 میگه این داره قدرت داریم beacon. در واقع به کمک قدرتی که ما به BS های مختلف داریم beacon میشه داره زیاد میشه پس Handover میکن و BS2 میشه Base station User association میگیم. درین دلیل بهش میگیم. الان در گوشی ها در یک ثانیه نتوونه paging کنه میشه IoT Node Unreachable. ولی وقتی میشیم درونکل درش تهیه شده که وقتی sleep رفتیم و Base station دیتا رو نگه میداره و نمیگه Unreachable. واچ میسته از sleep در بیایم بعد میفرسته و باعث میشه از لحاظ مصرف باتری خیلی بهمن کمک کنه

NB-IoT

- Recognizing that the definition of new LTE device categories was not sufficient to support LPWA IoT requirement, 3GPP specified Narrowband IoT (NB-IoT).
- The work on NB-IoT started with multiple proposals pushed by the involved vendors, including the following:
 - Extended Coverage GSM (EC-GSM), Ericsson proposal
 - Narrowband GSM (N-GSM), Nokia proposal
 - Narrowband M2M (NB-M2M), Huawei/Neul proposal
 - Narrowband OFDMA (orthogonal frequency-division multiple access), Qualcomm proposal
 - Narrowband Cellular IoT (NB-CIoT), combined proposal of NB-M2M and NB-OFDMA
 - Narrowband LTE (NB-LTE), Alcatel-Lucent, Ericsson, and Nokia proposal
 - Cooperative Ultra Narrowband (C-UNB), Sigfox proposal

او Cat 0 نتوسنت خیلی کاربردی و همه گیر بشن. پس 3GPP امتد NB-IoT را ایجاد کرد و چندین Proposal را از از Nokia و Ericsson و ... که در اسلاید میگیریم دریافت کرد.

NB-IoT

- Three modes of operation are applicable to NB-IoT:

– Standalone

- A GSM carrier is used as an NB-IoT carrier, enabling reuse of 900 MHz or 1800 MHz.

– In-band

- Part of an LTE carrier frequency band is allocated for use as an NB-IoT frequency.

– Guard band

- An NB-IoT carrier is between the LTE or WCDMA bands.
- This requires coexistence between LTE and NB-IoT bands.

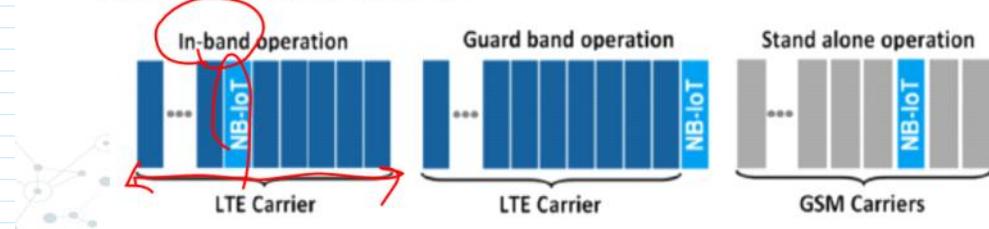
تا Mode 3 برای NB-IoT تعریف شده. الان مثلا ایرانسل اگر بخواهد NB-IoT داشته باشد، با یک software upgrade میتوانه بهش برسه ولی مثلا 40 مگاهرتز فقط پهنهای باند دارد و باید یک بخشی رو آزاد کنند برای NB-IoT که تقاضای خیلی کم براش و ضرر نمیکند. به همین دلیل سمتش نمیزنند.
برای این که فرکانس کجا باشد 3 تا مود داریم

NB-IoT



- NB-IoT occupies a frequency band width of 200 KHz, which corresponds to one resource block in GSM and LTE transmission

- Operation Modes of NB-IoT

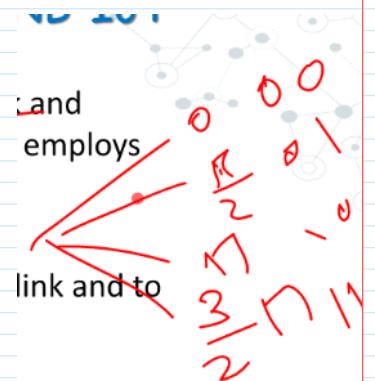


در in-band یک بخش از داخل پهنانی باند میگیره. Guard-band هم یک بخشی از GSM را برای احتصاص میده. که پهنانی باند NB-IoT 200 کیلوهرتز است. ازش استفاده میکنه. Stand alone هم یک استفاده از GSM میکنه. چون بین هر 2 تا پهنانی باند یک فضای هست برای این که پرش فرکانسی نشه و اون فاصله رو ازش استفاده میکنه.

Physical and Link Layers Protocols- NB-IoT

- NB-IoT uses the single-carrier FDMA in the uplink and orthogonal FDMA (OFDMA) in the downlink, and employs the QPSK modulation (QPSK)
- The data rate is limited to 200 kbps for the downlink and to 20 kbps for the uplink.
- The maximum payload size for each message is 1600 bytes.

جزئیات فنی: از FDMA استفاده میکنه. ماثولیشن اشت QPSK هستش:



دیتاریت اش 200 کیلوبیت در ثانیه است برای downlink و uplink اشم 20 کیلوبیت در ثانیه با توجه به این که IoT بیشتر uplink-oriented هستش چرا اینجا downlink بیشتره؟ یک کاربردش میتوانه این باشه node هارو بگیره و در short range بده به بقیه ها. یکی از کاربرد های مهمش: مثلا ببود فرض کنیم بخواهد یک الگوریتمی اجرا کنه و آپدیت نرم افزاری رو بخواهد در سمت Node هاش انجام بده. باید تمام دوچرخه هاشو جمع کنه ببره یک جایی و کل اون رو آپدیت کنه. در بسیاری از کاربرد ها این کار سخته. مثلا در کشاورزی هوشمند سخته ما بروم از خاک بیاریم شون بیرون برای آپدیت. اما با link downlink بالا میشه update over the air کرد.

Physical and Link Layers Protocols- NB-IoT

- NB-IoT uses the single-carrier FDMA in the uplink and orthogonal FDMA (OFDMA) in the downlink, and employs the QPSK modulation (QPSK)
- The data rate is limited to 200 kbps for the downlink and to 20 kbps for the uplink.
- The maximum payload size for each message is 1600 bytes.

اشو میبینیم در مقایسه با LPWAN ها و 802.15.4 خیلی بیشتره.

Physical and Link Layers Protocols- NB-IoT

- Based on the LTE protocol, but with minimal functionalities
 - LTE backend system is used to broadcast information that is valid for all end devices within a cell.
 - As the broadcasting backend system obtains resources and consumes battery power from each end device, it is kept to a minimum, in size as well as in its occurrence.
 - It was optimized to small and infrequent data messages and avoids the features not required for the IoT purpose, e.g., measurements to monitor the channel quality, carrier aggregation, and dual connectivity.

بس NB-IoT بر مبنای LTE Protocol با کمترین Functionality هستش. و برای داده های کوچک و غیر پشت سر هم بهینه سازی شده.

Physical and Link Layers Protocols- NB-IoT

- NB-IoT technology is a new air interface from the protocol stack point of view, while being built on the well-established LTE infrastructure
 - NB-IoT can be supported with only a software upgrade in addition to the existing LTE infrastructure
- NB-IoT allows connectivity of up to 100 K end devices per cell with the potential for scaling up the capacity by adding more NB-IoT carriers.
- 10 years of battery lifetime when transmitting 200 bytes per day on average.

از خوبی های NB-IoT اینه که در هر سلول میتوانه تا 100 هزار دستگاه رو ساپورت کنه و باتری هاشم در صورتی که روزانه 200 بایت به طور متوسط ارسال کنیم طراحی پروتکل اش به نحویه که ادعا میشه باتریش تا 10 سال کار میکنه.

مقایسه بین NB-IoT و LoRaWAN و Sigfox

Comparison of Main LPWAN Technologies

	Sigfox	LoRaWAN	NB-IoT
Modulation	BPSK	CSS	QPSK
Frequency	Unlicensed ISM bands (868 MHz in Europe, 915 MHz in North America, and 433 MHz in Asia)	Unlicensed ISM bands (868 MHz in Europe, 915 MHz in North America, and 433 MHz in Asia)	Licensed LTE frequency bands
Bandwidth	100 Hz	250 kHz and 125 kHz	200 kHz
Maximum data rate	100 bps	50 kbps	200 kbps
Bidirectional	Limited / Half-duplex	Yes / Half-duplex	Yes / Half-duplex
Maximum messages/day	440 (UL), 4 (DL)	Unlimited	Unlimited
Maximum payload length	12 bytes (UL), 8 bytes (DL)	243 bytes	1600 bytes
Range	10 km (urban), 40 km (rural)	5 km (urban), 20 km (rural)	1 km (urban), 10 km (rural)
Interference immunity	Very high	Very high	Low
Authentication & encryption	Not supported	Yes (AES 128b)	Yes (LTE encryption)
Adaptive data rate	No	Yes	No
Handover	End-devices do not join a single base station	End-devices do not join a single base station	End-devices join a single base station
Localization	Yes (RSSI)	Yes (TDOA)	No (under specification)
Allow private network	No	Yes	No
Standardization	Sigfox company is collaborating with ETSI on the standardization of Sigfox-based network	LoRa-Alliance	3GPP

منظور از Unlimited این نیست که محدود نیستیم. اینه که شبکه خودش محدودیتی روشن نداشته. در SigFox جدا از محدودیت های Duty cycle خودش نمیذاره.

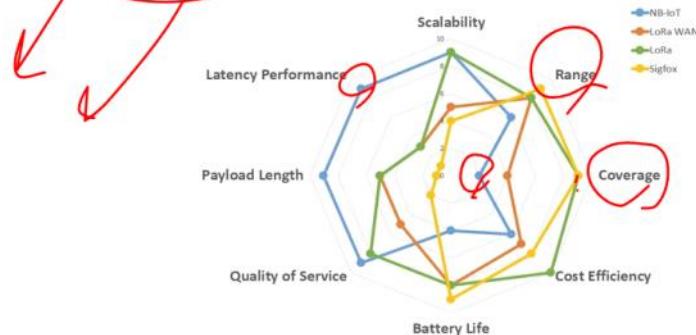
از لحاظ رنج Sigfox > LoRaWAN > NB-IoT

در Sigfox یوزر ها به نمیان Associate base station پشن ولی در 2 تای دیگه میشن.

شبکه Private فقط در LoRaWAN داریم.

Communication Technologies Comparison

- Comparison of SigFox, LoRa, LoRaWAN, and NB-IoT in terms of different criteria



از لحاظ Coverage میبینیم NB-IoT از همه کمتره ولی از لحاظ تاخیر و طول Payload و کیفیت بیشتره و از همه گرون تره که دلیل همش License دار در یک فرکانس بالا است.

Physical and Link Layers Protocols- IEEE 1901-2a: PLC

- Power Line Communication (PLC) is the use of electrical wires to provide data transmission capabilities
- PLC networks provide a number of advantages that make them both a useful complement and a strong competitor to wireless networking solutions.
 - low deployment cost when an electrical wired infrastructure is already in place.
 - PLC networks allow communication through obstacles that commonly degrade wireless signals, while delivering high data rates.
 - PLC also provides a low-cost alternative to complement existing technologies when aiming for ubiquitous coverage.
 - PLC provides the possibility of re-using the existing wired electrical network to provide communication capabilities

PLC میشه ارتباط هایی که از طریق برق شهر داریم و روی آن بخوایم داده ارسال کنیم. از خوبی وقت پیش بوده این و سیم های برق مناسب ارتباطات نیستن چون افت ولتاژ روش خوبی زیاده. ولی یک حسنه دارن و اینه که در همه شهر ها وجود دارن و یک داده با حجم کم رو میشه با PLC منتقل کرد. PLC ای که 50 هرتز روبا برق AC ارسال میکنه.

Physical and Link Layers Protocols- IEEE 1901-2a: PLC

- A classification of PLC systems is according to frequency bands:
 - ultra-narrowband (UNB) operating between about 125-3000 Hz,
 - narrowband (NB) operating between about 3-500 kHz , and
 - broadband (BB) operating between about 1.8-100 MHz

میبینیم تمام پیشنهاد ها از 50 هرتز بیشتره ولی خوب همه گیر نشده چون تضعیف خیلی زیاده.

The Business Case for IP

- Previous lecture focused on connectivity at Layer 1 (PHY) and Layer 2 (MAC).
- In this chapter, we move up the protocol stack and focus on network layer connectivity, which is commonly referred to as Layer 3.

میخوایم ببینیم میشه IPv4 و IPv6 در لایه 3 برای گره های IoT استفاده کنیم یا خیر. در این لکچر لایه سوم رو بررسی میکنیم.

The Business Case for IP

- The key advantages of the IP suite for the Internet of Things:
 - Open and standards-based
 - Versatile
 - Ubiquitous
 - Scalable
 - Manageable and highly secure
 - Stable and resilient
 - Consumers' market adoption
 - The innovation factor

مزیت های IP رو میبینیم که همه جاگیر شده. Scalable هستش و اگر تعداد Node ها به 50 میلیارد برسه اون موقع ممکنه مجبور بشیم بیم سراغ IPv6. ولی دیگه حال حالا ها به اندازه کافی فضا داره. Stable هستش و اون چیزی که مهمه اینه که وارد بازار شده و در لایه 3 جز IP کسی رو چیزی صحبت نمیکنه. نوآوری بالای هم داره. چیزی که باید دقت بکنیم اینه که نکته Optimum هستش. یعنی اگر کسایی که الان متخصصن برگردان زمانی که موقع طراحی IP بود آیا همین رو طراحی میکردن؟ در اون زمان بیشتر هدفشوں برقراری ارتباط بوده و خیلی جاهاش Optimal نیست و الان میشه یک پروتکل دیزاین کرد که از اون بهتر باشه. حقی برای ارتباطات فعلی (جدای از IoT) هم Optimum نیست.

ولی چرا عوضش نمیکنن؟ چون فرآگیر شده و تمام تجهیزات و دستگاه ها که برای ارتباط و اتصال به اینترنت استفاده میشه مبتنی بر این پروتکل هستش. جایگزین این ها هزینه بره چون کارت شبکه کامپیوتر ها و روتور ها و .. همگی باید عوض بشن و با این فرآگیری که داره اگه 20% هم بهبود بدیم ولی سرمایه گذاری مجدد 1.5 برابر باشه دیگه به صرفه نیست.

همین IPv6 هم که لایه Network اش عوض شده که فضای آدرس دهی بیشتری داشته باشن همین باعث شده یک Protocol Convertor اون وسط باشه که IPv4 رو به IPv6 تبدیل کنه و بالعکس

Adoption or Adaptation of the Internet Protocol

- Typically, one of two models, adaptation or adoption, for IP in IoT is proposed:

– Adaptation

- means application layered gateways (ALGs) must be implemented to ensure the translation between non-IP and IP layers.

– Adoption

- involves replacing all non-IP layers with their IP layer counterparts, simplifying the deployment model and operations.

ما در IoT یک سری ALG میداریم که بین IP Non و IP Layer یک ترجمه انجام بد. مثلا در لایه 1 و 2 داریم یه چیزی استفاده میکنیم که IP نمیتونه روشن اجرا بشه. میایم یک مبدل کنترلی بین بخش Non IP و IP قرار میدیم. خودمون رو Adapt میکنیم و IP رو میفهمیم و هم میتوانیم بسته بگیریم هم بفرستیم.

تمام non-IP Layer هارو با IP جایه جا میکنیم. کاملا IP رو میپذیریم. چون میخوایم Deployment ساده تری داشته باشیم و مستقیما به اینترنت وصل بشیم.

Adoption or Adaptation of the Internet Protocol

- IP adaptation versus adoption model still requires investigation for particular last-mile technologies used by IoT



هر جفت شون دارن استفاده میشن. NB-IoT, SigFox, LoRaWAN, BLE, 802.11ah ... اینا همه میشن

Adoption or Adaptation of the Internet Protocol

- You should consider the following factors when trying to determine which model is best suited for last-mile connectivity:
 - **Bidirectional versus unidirectional data flow:**
 - While bidirectional communications are generally expected, some last-mile technologies offer optimization for unidirectional communication.
 - IoT devices, may only infrequently need to report a few bytes of data to an application.
 - For these cases, it is not necessarily worth implementing a full IP stack.
 - **Overhead for last-mile communications paths:**
 - IPv4 has 20 bytes of header at a minimum, and IPv6 has 40 bytes at the IP network layer
 - For the IP transport layer, UDP has 8 bytes of header overhead, while TCP has a minimum of 20 bytes.
 - If the data to be forwarded by a device is infrequent and only a few bytes, you can potentially have more header overhead than device data
 - It needs to decide whether the IP adoption model is necessary and, if it is, how it can be optimized.

یه سری عوامل در تعیین انتخاب بین این ۲ موثه.

- **Data flow** به صورت **Bidirectional** هستش یا **unidirectional**. یعنی Node بیشتر داره داده ارسال میکنه یا هم داره ارسال میکنه هم دریافت. چرا؟ ما عموماً انتظار داریم و در last-mile technologies یک سری Optimization های برای Unidirectional انجام دادن و یک طرفش سنگین تره. IoT اینجا uplink-oriented هستش و بیشتر میخواهد داده ارسال کنه.

در TCP/IP ارتباط دو طرفی. دستگاه های IoT هم میخوان یک سری داده محدود ارسال کنن. به همین دلیل شاید نیاز نباشه ما در IoT Full stack یک IP رو بپاده سازی کنیم چون ممکنه اصن بیش نیاز نداشته باشیم. این که پیاده سازیش کنیم عین این میمونه یک بنز آوردیم برای کسی که بکی میخواهد هر 2 ماه یه بار ماشین سواری کنه.

- سریاری که TCP/IP داره برای نود هایی که در last-mile قرار دارن. IoT IPv4 20 بایت هدر و IPv6 40 بایت هدر داره. در لایه Transport هم UDP 8 بایت و TCP 20 بایت هدر داره. اگر دیتا که میخواهد ارسال بشه، شما بیشتر Overhead داری با خودت حمل میکنی تا خود داده. مثلًا وقتی داده 2 بایته با این حجم هدر حداقل 46 بایتش هدره و اصن نمیصرفه

ولی یه موقع ارتباط دو طرفه باشه یا یک طرفه مهم نیست و این هدر های زیاد هم برآمون مهم نیست. در sigfox payload داریم منطق نیست با IP بریم ولی در 802.15.4 که 127 بایت داریم یکم معقول تره.

Adoption or Adaptation of the Internet Protocol

- **Data flow model:**
 - One benefit of the IP adoption model is that any node can easily exchange data with any other node in a network
 - However, in many IoT solutions, a device's data flow is limited to one or two applications.
 - In this case, the adaptation model can work because translation of traffic needs to occur only between the end device and one or two application servers.
 - Depending on the network topology and the data flow needed, both IP adaptation and adoption models have roles to play in last-mile connectivity.

در مورد Data flow model یکی از مزیت های IP Adoption اینه که گره به راحتی میتوانه داده رو انتقال کنه و با هر گره ای در شبکه ارتباط برقرار کنه.

اما در کاربردهای IoT به صورت کامپیوتر و گوشی موبایل نیست که با اپلیکیشن های مختلف ارتباط برقرار کنیم. Data flow محدود میشه به یک یا 2 تا اپلیکیشن. مثلاً سطل آشغال هوشمند که پر بودن رو تشخیص میده و جاش رو هم ریپورت کنه. این که دیگه قرار نیست با گوگل سرج انجام بده و داده بگیره. نهایتاً میخواهد با motion detector داره روشن میشه نهایتاً میخواهد با یک اپلیکیشن برای مدیریت روشنایی و قسمت های امنیتی ارتباط داشته باشه که یکی رد شد. پس اون IP Adoption نیاز ما نیست. پس در مواقعی که اپلیکیشن محدوده Adaptation بپرسه و سنگینی پروتکل IP رو نداریم و یک مترجم میداریم.

The Need for Optimization of IP for IoT



• Constrained Nodes

- Devices that are very constrained in resources, may communicate infrequently to transmit a few bytes, and may have limited security and management capabilities.
 - This drives the need for the IP adaptation model, where nodes communicate through gateways and proxies.
- Devices with enough power and capacities to implement a stripped down IP stack or non-IP stack
 - In this case, you may implement either an optimized IP stack and directly communicate with application servers (adoption model) or go for an IP or non-IP stack and communicate through gateways and proxies (adaptation model).
- Devices that are similar to generic PCs in terms of computing and power resources but have constrained networking capacities, such as bandwidth
 - These nodes usually implement a full IP stack (adoption model), but network design and application behaviors must cope with the bandwidth constraints.

دیوايس ها ما در IoT در واقع از نظر Resource محدودن. به ندرت ارتباط برقرار میکنن و همچنان Security و Management capability کمی نیاز دارن. این مارو تحریک میکنه که IP adaptation روانخواه کنیم. از Gateway Node تا Non-IP باشه یا به نحوی در Gateway یک تبدیل کنن بذاریم که از اون به بعد IP بشه. اما دیوايس هایی که پاور خوبی دارن به برق شهر وصلن مثل خانه هوشمند و طرفیتاشم بد نیست اون موقع میتوانه IP رو Implement کنه. دیوايس های شبیه به کامپیوچر هم که دیگه میشه IP رو روش پیاده سازی کرد. Full stack

The Need for Optimization of IP for IoT



• Constrained Networks

- Constrained networks are limited by
 - low-power,
 - low-bandwidth links (wireless and wired).
- Constrained networks operate between a few kbps and a few hundred kbps and may utilize a star, mesh, or combined network topologies, ensuring proper operations.
- In contrast, highly stable and fast links are available for typical IP networks

حالا ممکنه Node مشکلی نداشته باشه ولی یک سری محدودیت در شبکه داشته باشیم. مثل payload کم هستش. دیتا ریت کمتر از کیلوبیت در ثانیه است. در حالتی که در IP هم سرعتش زیاده و هم stable شده.

The Need for Optimization of IP for IoT

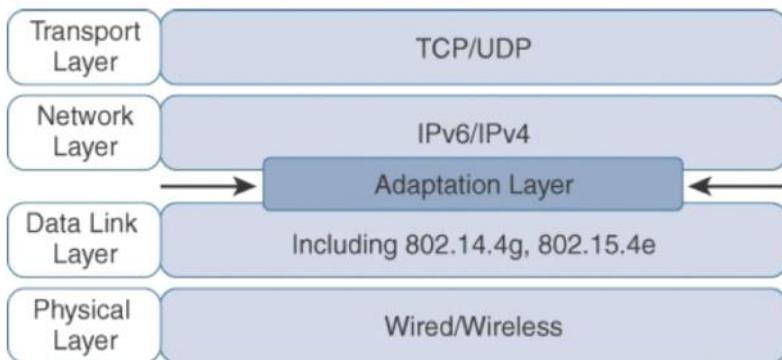
- Constrained Networks

- Packet delivery rate (PDR) oscillate between low and high percentages
- Large bursts of unpredictable errors and even loss of connectivity at times may occur
- Control plane traffic must also be kept at a minimum; otherwise, it consumes the bandwidth that is needed by the data traffic.
- Finally, you have to consider the power consumption in battery-powered nodes.
 - Any failure or verbose control plane protocol may reduce the lifetime of the batteries.

يعني از تعداد بسته هایی که ارسال شده چقدر دریافت شده. پروتکلی که کلی کارای کنترلی انجام میده. مثل HTTP که 404 داره و ... و در IoT دیگه انقدر نیاز نداریم.

Optimizing IP for IoT

- constrained nodes and constrained networks mandate optimization at various layers and on multiple protocols of the IP architecture.



من در لایه 1 و 2 هرجی که داریم. اگر کاربرد IoT مد نظرمون هست نیاز هست یک Adaptation layer نیاز داریم چون لایه های بالا اگر این لایه را قرار ندمیم بسته ای رو تحويل پایین میده که کلی مشکل داره.

Optimizing IP for IoT

- In the IP architecture, the transport of IP packets over any given Layer 1 (PHY) and Layer 2 (MAC) protocol must be defined and documented. The model for packaging IP into lower-layer protocols is often referred to as an *adaptation layer*.
- An adaptation layer designed for IoT may include some optimizations to deal with constrained nodes and networks

The Header Size Problem



- Worst-case scenario calculations
 - Maximum frame size in IEEE 802.15.4: 127 byte
 - Reduced by the max. frame header (25 byte): 102 byte
 - Reduced by highest link-layer security (21 byte): 81 byte
 - Reduced by standard IPv6 header (40 byte): 41 byte
 - Reduced by standard UDP header (8 byte): 33 byte
 - This leaves only 33 byte for actual payload
 - The rest of the space is used by headers

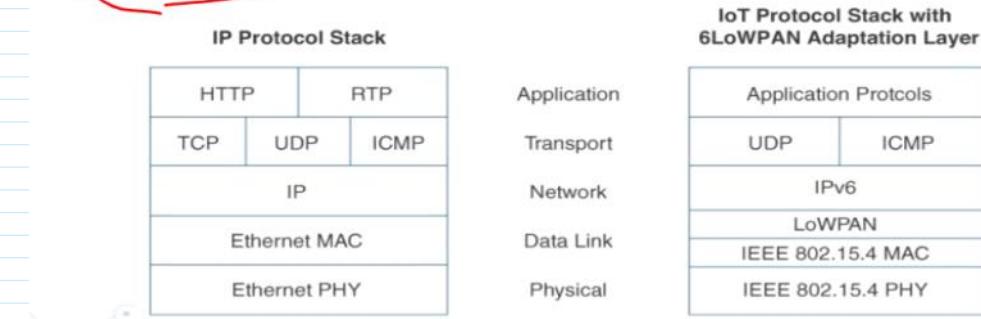


بایام 802.15.4 رو بررسی کنیم. ماکسیمم سایز فریم 127 بایت هستش. اگر این 127 بایت از 802.15.4 به بخشیش payload که از لایه بالاتر میتوانه بگیره. امنیت هم حساب کنیم 81 بایت باقی میمانه. 40 بایت هم میشه برای هدر IPv6. اگر بالا ش هم UDP بگیم میشه 33 بایت. پس تنها 33 بایت Payload باقی میمانه از 127 بایت.

Optimizing IP for IoT



- 6LoWPAN as IP Adaptation Layer for IoT



6LoWPAN با فرض این که میخواهد در یک PAN که نود ها محدودیت دارن استفاده بشه. Transport Application Protocols که در Lecture 5 بررسی میشه. یک لایه Adaptation تحت عنوان LoWPAN اضافه کرده. پس هرجا 6LoWPAN گفتیم یعنی یک IP Protocol داریم که یک LoWPAN اضافه داره.



Movement Towards IP

- All IoT protocols are moving towards IP
- TCP/IP is not one size fits all
 - Adaptations needed for MTU size
 - Reduce of header overhead
 - UDP instead of TCP to avoid latencies

عموماً UDP استفاده می‌شود چون ACK فرستادن سخته و میخوایم سرعت بیشتری داشته باشیم



Optimizing IP for IoT

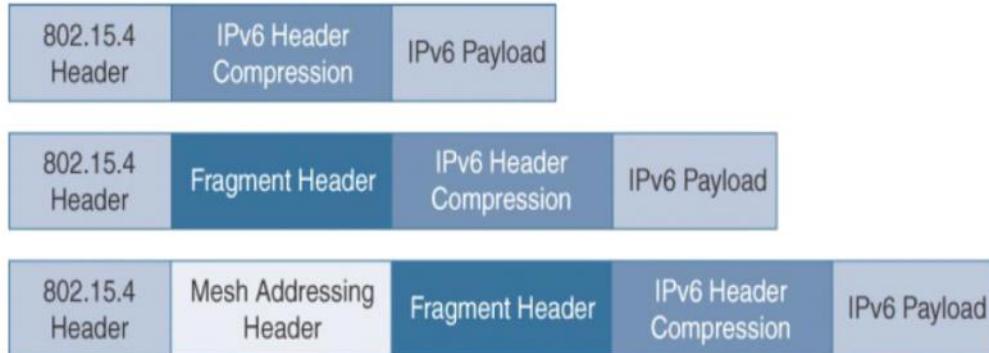
- The IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) working group focused on enabling IPv6 over IEEE 802.15.4 networks.
- The group started its work in 2005 and concluded in 2014 after working through the following goals:
 - Defining a fragmentation and reassembly layer to allow adaptation of IPv6 to IEEE 802.15.4 links
 - Introduce an IPv6 header compression mechanism to avoid excessive fragmentation and reassembly, since the IPv6 header alone is 40 bytes long, without optional headers.
 - Examining mesh routing protocol suitability to 802.15.4 networks, especially in light of the packet size constraints.



Optimizing IP for IoT

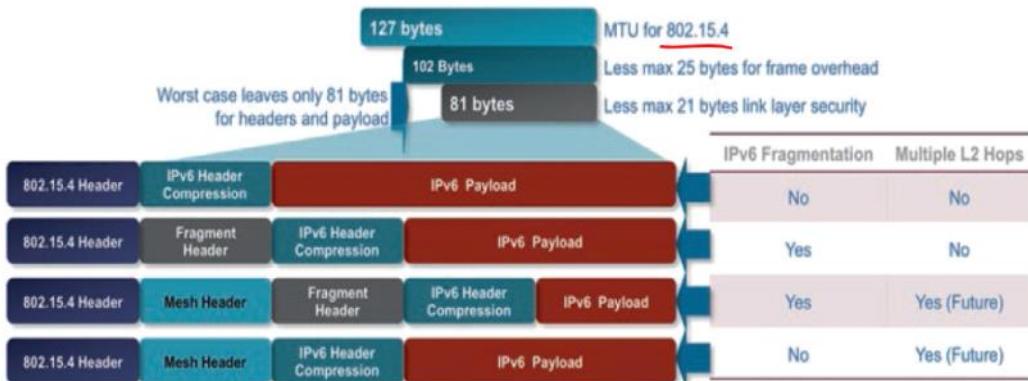
- 6LoWPAN as IP Adaptation Layer for IoT
 - Header compression
 - Fragmentation
 - Mesh addressing

Optimizing IP for IoT



بالی و قتیه که فقط Compression داریم. پایینی Fragment هم اضافه شده و آخری هم Mesh addressing هم اضافه شده (برای مسیریابی). همه اینا payload رو کمتر میکن.

Optimizing IP for IoT



Optimizing IP for IoT-6LoWPAN Header compression

- At a high level, 6LoWPAN works by taking advantage of shared information known by all nodes from their participation in the local network.
- In addition, it omits some standard header fields by assuming commonly used values.

اگه بخوایم سطح بالا حرف بزنیم. 6LoWPAN میاد از مزیت این که اطلاعات همه نود ها معلومه استفاده میکنه. 802.15.4. یک شبکه محدوده هر چند PAN ها با Coordinator ها میتوان برقرار کنن ولی شبکه در کل Coverage کمی دارد. در آدرس IP برخیش آدرس Gateway عه و چند بایت اول یکیه بین IP افراد مختلف. پس میان این بخش رو حذف میکن

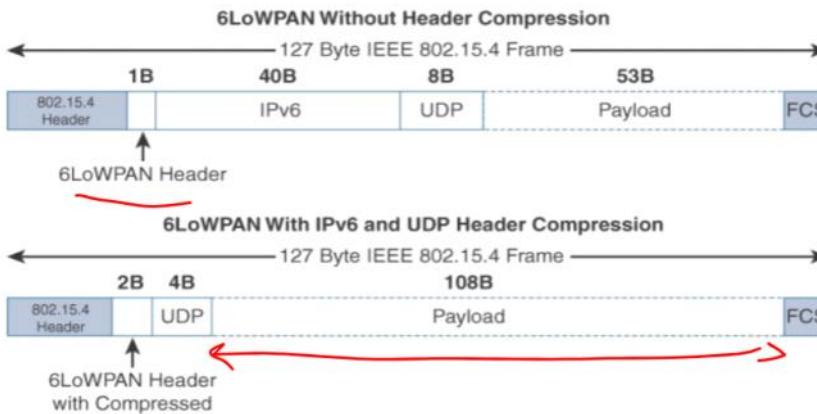
Optimizing IP for IoT-6LoWPAN Header compression

- A 6LoWPAN frame without any header compression enabled:
 - The full 40-byte IPv6 header and 8-byte UDP header
 - The 6LoWPAN header is only a single byte
 - Notice that uncompressed IPv6 and UDP headers leave only 53 bytes of data payload out of the 127-byte maximum frame size in the case of IEEE 802.15.4.
- A 6LoWPAN frame with a header compression enabled:
 - The 6LoWPAN header increases to 2 bytes to accommodate the compressed IPv6 header, and UDP has been reduced in half, to 4 bytes from 8.
 - Most importantly, the header compression has allowed the payload to more than double, from 53 bytes to 108 bytes, which is obviously much more efficient.

اگر یک 6LoWPAN frame را بدون استفاده کنیم 53 بایت از 127 رو داریم، ولی وقتی از Header compression استفاده کنیم، یک هدر خودش اضافه میکنیم که 2 بایت هستش. اون موقع UDP از 8 بایت تبدیل به 4 بایت میشه، و باعث میشه 108 بایت payload داشته باشیم

Optimizing IP for IoT- 6LoWPAN Header compression

- Header compression

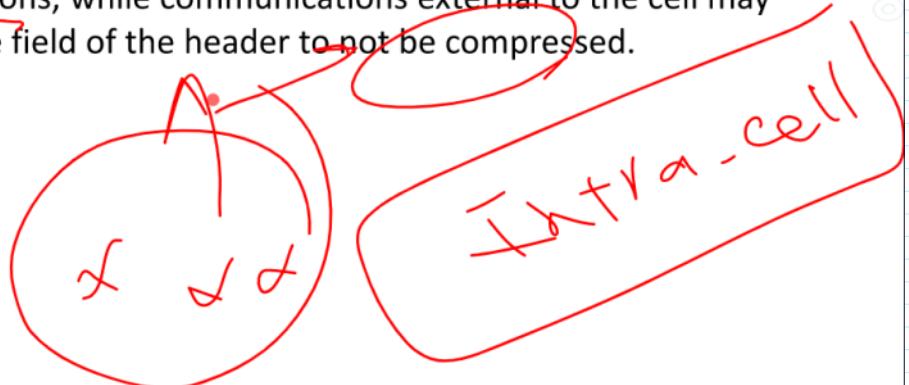


اینجا بهتر میشه فهمید.

Optimizing IP for IoT- 6LoWPAN Header compression

- Header compression

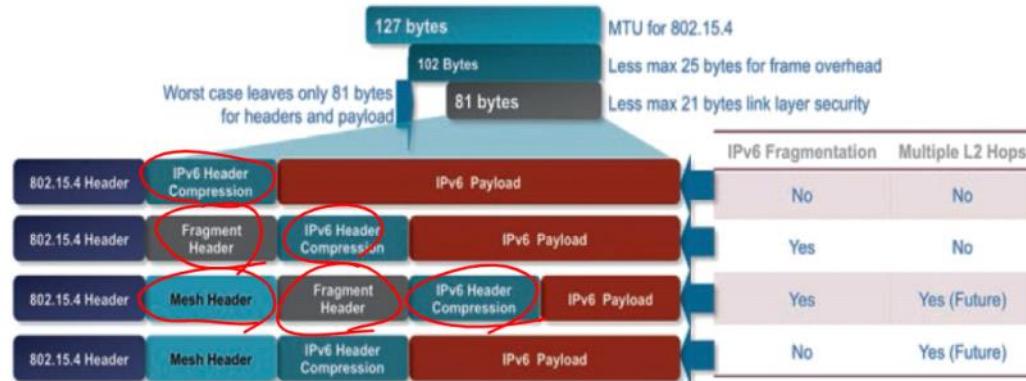
- Note that the 2-byte header compression applies to intra-cell communications, while communications external to the cell may require some field of the header to not be compressed.



Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Thing, Cisco press, 2017

نکته قابل توجه اینه که این 2 بایت compression در ارتباطات درون سلولی هستش و خارج از این سلول باید از gateway (وظیفه خارج شه)

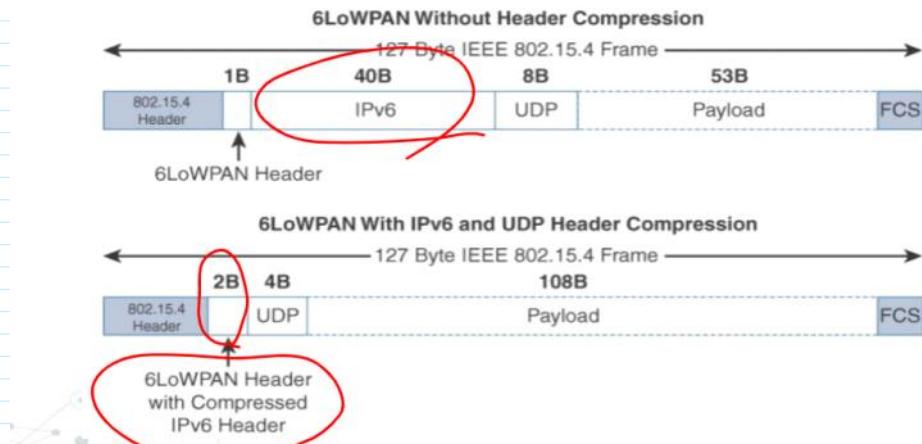
Optimizing IP for IoT



چرا ما Compression نداریم؟ از نظر تئوری میشه ولی از نظر منطقی نمیشه. چون ما زمانی میریم سراغ Payload که لایه بالا یک Fragmentation زیادی تحويل داده و لایه های پایین 102 بایت ارسال میکنه و 1020 رو مثلًا به 10 تا 1020 بایت تقسیم میکنه. که همینا مال یک Message است که از لایه بالا اومند. گیرنده هم باید این را متوجه بشه که بعدش اینارو به هم بچسبوئه. اگر استفاده نکنیم این 1020 بایت رو با توجه به هدرشون باید به 20 تا به جای 10 تا تقسیم کنیم. پس با Compress کردن تعداد Fragmentation کم میشه.

Optimizing IP for IoT- 6LoWPAN Header compression

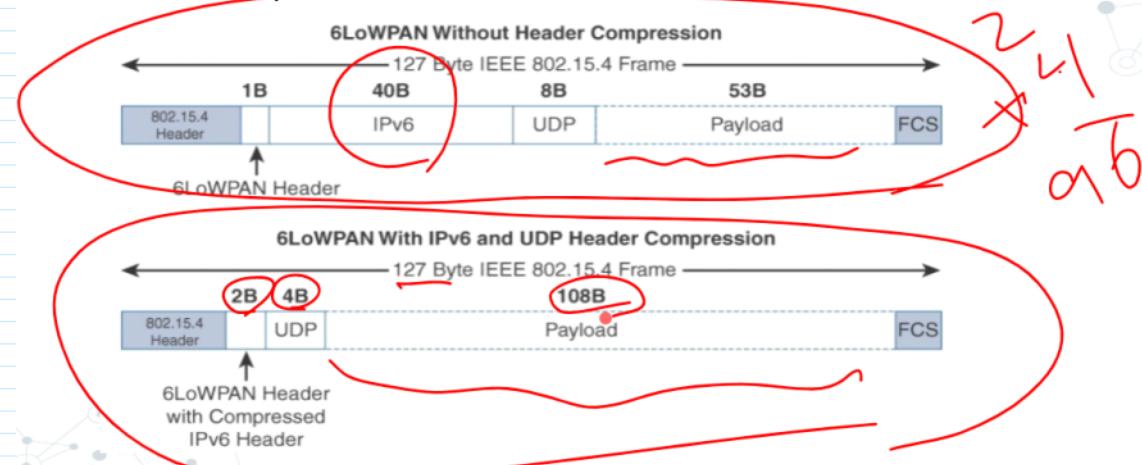
- Header compression



این 2 بایت که برای Header Compression اضافه میشه نقش 40 بایت IPv6 رو داره و 38 بایت به نفعمنه. استاد محاسبه کرد دید شد 96 بایت ولی نوشته 108 بایت:

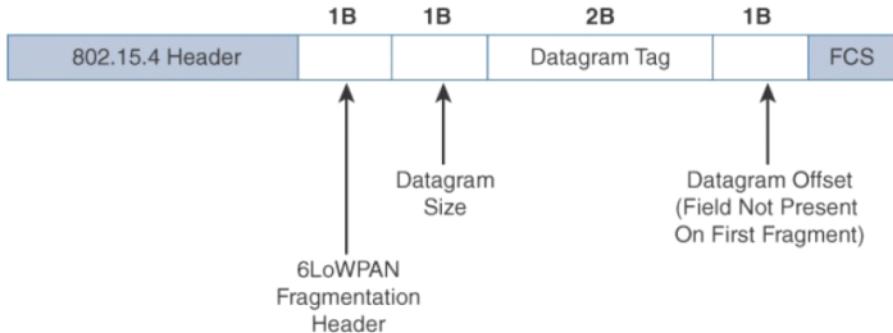
Optimizing IP for IoT- 6LoWPAN Header compression

- Header compression



Optimizing IP for IoT-6LoWPAN Fragmentation

- Fragmentation: 6LoWPAN Fragmentation Header



در Fragmentation هدر نیاز داریم و حداقل چیزی که توش نیاز داریم اینه که Offset چیه و بسته چندم داره ارسال میشه.

Optimizing IP for IoT-6LoWPAN Fragmentation

- The maximum transmission unit (MTU) for an IPv6 network is 1280 bytes.
 - The term *MTU* defines the size of the largest protocol data unit that can be passed.
- For IEEE 802.15.4, 127 bytes is the MTU. IPv6, with a much larger MTU, is carried inside the 802.15.4 frame with a much smaller one.
- To remedy this situation, large IPv6 packets must be fragmented across multiple 802.15.4 frames at Layer 2.

Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Thing, Cisco press, 2017

رو تقسیم بر 127 کنیم یعنی به همون تعداد بسته نیاز داریم.

Optimizing IP for IoT-6LoWPAN Fragmentation

- The fragment header utilized by 6LoWPAN is composed of three primary fields:
 - Datagram Size: specifies the total size of the unfragmented payload.
 - Datagram Tag: identifies the set of fragments for a payload
 - Datagram Offset: delineates how far into a payload a particular fragment occur

• سایز دیتگرام داره تحويل 6LoWPAN داده میشه.

• شماره گذاري Fragment ها

• چندمي هست يا چندتا مونده (بسته به قرارداد)

Optimizing IP for IoT-6LoWPAN Fragmentation

• Fragmentation

- 6LoWPAN fragmentation header field itself uses a unique bit value to identify that the subsequent fields behind it are fragment fields as opposed to another capability, such as header compression.
- Also, in the first fragment, the Datagram Offset field is not present because it would simply be set to 0.
 - This results in the first fragmentation header for an IPv6 payload being only 4 bytes long.
 - The remainder of the fragments have a 5-byte header field so that the appropriate offset can be specified.

به کمک مشخص میکنند پکت ها فرگشت شده. توی بسته اولی هم offset نمیاریم چون 0 عد.

6LoWPAN: Mesh Addressing

- The purpose of the 6LoWPAN mesh addressing function is to forward packets over multiple hops.
- Three fields are defined for this header:
 - Hop Limit,
 - Source Address,
 - Destination Address
- Analogous to the IPv6 hop limit field, the hop limit for mesh addressing also provides an upper limit on how many times the frame can be forwarded.
 - Each hop decrements this value by 1 as it is forwarded.
 - Once the value hits 0, it is dropped and no longer forwarded.

وظیفه سوم بحث Mesh addressing هست. منظور از Mesh در شبکه اینه که یک سری نقاط داریم که میخوان با هم در ارتباط باشن. ممکنه با بیرون محیط هم مثل اینترنت در ارتباط باشن. مش یعنی مثل توری به هم وصل کنیم تا ارتباط برقرار بشه.

- The purpose of the 6LoWPAN mesh addressing function is to forward packets over multiple hops.
- Three fields are defined for this header:
 - Hop Limit,
 - Source Address,
 - Destination Address
- Analogous to the IPv6 hop limit field, the hop limit provides an upper limit on how many times the frame can be forwarded. کارشن ارتباط Node های داخلی با بیرون هست رو میگن .Mesh router

مشن یه چیزی بین Ad hoc و شبکه Cellular هستش. در Ad hoc ما هیچ زیرساختی نداریم. یک سری Node Mobility که میتوانن Routing داشته باشن و وقتی از 1 به 3 میخوایم بفرستیم باید به 2 بگیم قبول کرد بعد بفرسته. توی Cellular هم یک Base station میباشد که بهش وصل میشیم و ارتباط پرقرار میشه. در مشن یه چیزی بین ایناس و هم با هم در ارتباطن هم پک Mesh router که ارتباط با خارج رو فراهم میسازه.

حالا وقتی میگیم Mesh addressing یعنی از یکی از مهم ترین بحثای مهم در Routing بخش Addressing هستش که مشخص کنه به چه کسی میخوایم بفرستیم. که در قالب IP Source و Destination IP هستش. آدرس ها هم سلسله مراتبی هستش مثل آدرس خونه. مثلاً قسمت اولش قاره رو بگه بعد کشور و همینطور جزئی تر. وقتی میگیم Mesh addressing یعنی این آدرس دهی چگونه صورت بگیرد.

- 3 تا فیلد تعريف میشه:
یک بسته مثلاً فقط 10 بار از یک گره تا گره دیگه بره و دچار لوب بینهايت نشه.

Mesh-Under Versus Mesh-Over Routing

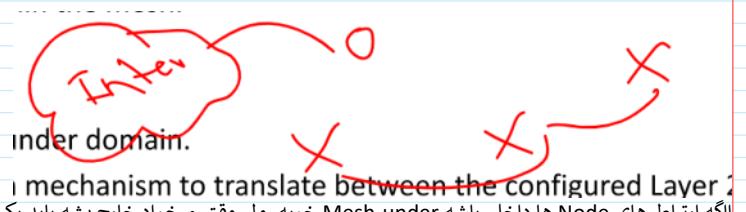
- Two main options exist for establishing reachability and forwarding packets.
 - Mesh-under:
 - Routing of packets is handled at the 6LoWPAN adaptation layer.
 - Mesh-over” or “route-over,”
 - Utilizes IP routing for getting packets to their destination

روتینگ در خود لایه 6LoWPAN میافته. •
از همون IP routing استفاده کنیم. •
جداب تره چون دیگه به اون حجم داده نیاز نداریم. •
Mesh-under

Mesh-Under Routing

- With mesh-under routing, the routing of IP packets leverages the 6LoWPAN mesh addressing header discussed in the previous section to route and forward packets at the link layer.
- Nodes have a Layer 2 forwarding table that they consult to route the packets to their final destination within the mesh.
- An edge gateway
 - terminates the mesh-under domain.
 - must also implement a mechanism to translate between the configured Layer 2 protocol and any IP routing mechanism implemented on other Layer 3 IP interfaces.

منظور از Forwarding یعنی هر بار که مثلاً میریم جلو Hop limit رو کم کنیم و Routing اینه که مسیریابی بشه به مقصد برسه در لایه 2 یک forwarding table داریم که یوزر ها با اون جدول میتوانن مشورت کنن و بینن چطوری میتوانن به مقصد برسن (شبیه routing table در لایه سه ولی خلاصه تر)



اگه ارتباط های Node ها داخلی باشه Mesh-under خوبه. ولی وقتی میخواد خارج بشه باید یک NAT یعنی IPv6 نرمال در باره. اگر بخوایم قیاس انجام بدیم یه چی شبیه

Mesh-Over Routing



- In mesh-over or route-over scenarios, IP Layer 3 routing is utilized for computing reachability and then getting packets forwarded to their destination, either inside or outside the mesh domain.
- Each full-functioning node acts as an IP router, so each link layer hop is an IP hop.
- When a LoWPAN has been implemented using different link layer technologies, a mesh-over routing setup is useful.
- While traditional IP routing protocols can be used, a specialized routing protocol for smart objects, such as IPv6 Routing Protocol for Low Power and Lossy Networks (RPL), is recommended.
 - RPL is discussed in more detail later in this chapter.

در لایه 3 انجام میشه و Full-function nodes میتونه تحت عنوان IP Router عمل کنه. در لایه 3 به طور کامل پیاده سازی میشه ولی در Mesh-under ما در لایه 2 میایم این کار را میکنیم.
یک سری پروتکل مثل RPL برای شبکه های power Low و Lossy تعریف شده که در IPv6 برای Smart object ها هستش.

6Lo Working Group



- With the work of the 6LoWPAN working group completed, the 6Lo working group seeks to expand on this completed work with a focus on IPv6 connectivity over constrained-node networks.
- While the 6LoWPAN working group initially focused its optimizations on IEEE 802.15.4 LLNs, standardizing IPv6 over other link layer technologies is still needed

6Lo Working Group

- Therefore, the charter of the 6Lo working group, now called the IPv6 over Networks of Resource-Constrained Nodes, is to facilitate the IPv6 connectivity over constrained-node networks.
- In particular, this working group is focused on **using 6LoWPAN technologies (RFC4944, RFC6282, RFC6775) for link layer technologies:**
 - IPv6 over Bluetooth Low Energy
 - Transmission of IPv6 packets over near-field communication
 - IPv6 over 802.11ah
 - Transmission of IPv6 packets on WIA-PA (Wireless Networks for Industrial Automation–Process Automation)
 - Transmission of IPv6 packets over DECT Ultra Low Energy

همین کاری که در 6LoWPAN انجام دادیم، یک سری Work group هستش به اسم 6Lo که میان IPv6 و LPWAN را برای بقیه میاره. برای آوردن شسخته چون payload مثلا در SigFox به اندازه 12 بایتی که کمه.

Optimizing IP for IoT

- In fact, based on the work of the 6LoWPAN working group and now the 6Lo working group:
 - the 6LoWPAN adaptation layer is becoming the de facto standard for connecting constrained nodes in IoT networks.

Forwarding

وقتی میگیم Routing و Forwarding یعنی Connecting

RPL

- The IETF chartered the RoLL (Routing over Low-Power and Lossy Networks) working group to evaluate all Layer 3 IP routing protocols and determine the needs and requirements for developing a routing solution for IP smart objects
- A new routing protocol should be developed for use by IP smart objects, given the characteristics and requirements of constrained networks.
- This new distance-vector routing protocol was named the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL).

RPL

- In an RPL network, each node acts as a router and becomes part of a mesh network.
- Routing is performed at the IP layer.
- Each node examines every received IPv6 packet and determines the next-hop destination based on the information contained in the IPv6 header.
- No information from the MAC layer header is needed to perform next-hop determination.
 - this is referred to as mesh-over routing

در RPL هر Node خودش میتوانه به عنوان Router میتوانه نقش ایفا کنه و میتوانه بخشی از شبکه Mesh بشه. مسیریابیش در لایه 3 انجام میشه و موارد دیگه که در اسلاید میبینیم.

RPL

- To cope with the constraints of computing and memory that are common characteristics of constrained nodes, the protocol defines two modes:

- Storing mode:**

- All nodes contain the full routing table of the RPL domain.
- Every node knows how to directly reach every other node.

- Non-storing mode:**

- Only the border router(s) of the RPL domain contain(s) the full routing table.
- All other nodes in the domain only maintain their list of parents and use this as a list of default routes toward the border router.
- When communicating in non-storing mode, a node always forwards its packets to the border router, which knows how to ultimately reach the final destination

برای این که RPL بتوانه با محدودیت های IoT مچ بشه 2 تا مود کاری داره:

- Storing mode: همه نود ها یک Full routing table ذخیره میکن و هر نود میدونه چطوری ارتباط برقرار کنه.
- Non-storing mode: تنها Mesh router که ارتباط با فضای خارج از شبکه رو فراهم میکنه نیاز هست که جدول رو ذخیره کنه و بقیه لیست parent هاشون به سمت Border router رو ذخیره کنن. یعنی فقط مسیر برای رسیدن به Border router رو دارن.

DAG رو خیلی سریع رد کرد و گفت جز درس نیست و برای علاقه مندان قرار گرفته شده.

خلاصه جلسه پیش و این جلسه:

IP مهمه و تا جایی که میتوانیم باید مچ بشیم چون بسته نود IoT در بسیاری از کاربرد ها باید وارد اینترنت بشه. ولی IP دیدیم مناسب نیست چون سنگینه و خیلی نود ها نمیتوان Full stack پیاده سازی کنن و هم از لحاظ تعداد بایت efficient نیست. اکه میخواستیم بناریم کنار محدود میشدیم به یک شبکه اختصاصی که از Z A تا Z رو باید خودمون درست میکنیم. مثلا در یک Tracking مبتنی بر LoRaWAN که مثلا در یک قلاوه سگ قرار میگیره و با یک دقیق میگه کجا قرار داره. فرض کنیم حالا فرد بره یک کشور دیگه اونجا باید حتما از طریق اینترنت ارتباط برقرار بشه.

پس Adaptation میاریم و یک Convertor بین لایه IoT و LoWPAN میکنه یک جا میاد Compression میکنه یک جا هم Routing را کاستومایز شده انجام میده.

LECTURE 5

وارد لایه Application و Transport میشیم. هرجی از لایه پایین جدا میشیم بیشتر شبیه میشیم به اون چیزی که در اینترنت وجود داره. تغییرات هست ولی تغییراتش دیگه نسبت به لایه پایین که همه چیزش متفاوته کمتره.

Review of Application and Transport Layer Protocols

- The transport layer:** takes application messages and transmits those message segments into Layer 3, the networking layer
 - is responsible for end-to-end communication over a network.
 - It provides logical communication between application processes running on different hosts
- The application layer:** is responsible for data formatting and presentation

Review of Application and Transport Layer Protocols

• The Transport Layer:

- IP-based networks use either TCP or UDP.
- However, the constrained nature of IoT networks requires a closer look at the use of these traditional transport mechanisms.

• IoT Application Transport Methods:

- The application layer in the Internet is typically based on HTTP.
- However, HTTP is not suitable in resource constrained environments because it is fairly verbose in nature and thus incurs a large parsing overhead.
- Many alternate protocols have been developed for IoT environments such as CoAP (Constrained Application Protocol) and MQTT (Message Queue Telemetry Transport).

ما نیاز داریم یک نگاه دقیق تر داشته باشیم بینیم UDP و TCP برای IoT مناسب هستن یا نه. HTTP مناسب دیوایس هایی که از لحاظ پردازشی محدوده خوب نیست چون یک پروتکل شلوغیه و یک overhead بالای داره که در IoT نیازشون نداریم. پس یک سری جایگزین هایی مثل CoAP و MQTT تعریف شدن.

Review of Transport Layer Protocols

• Transport Layer

– Transmission Control Protocol (TCP)

- Connection-oriented protocol
- It is an equivalent to a traditional telephone conversation, in which two phones must be connected and the communication link established before the parties can talk

– User Datagram Protocol (UDP)

- Connection-less protocol
- This is analogous to the traditional mail delivery system
- No guarantee of delivery

از این نظر میگن UDP مثل Mail delivery system هستش چون وقتی میریم اداره پست دیگه مقصد اطلاع نداره لزوماً که قراره داده براش ارسال کنیم.

Review of TCP



- With the predominance of human interactions over the Internet, TCP is the main protocol used at the transport layer
- This is largely due to its inherent characteristics, such as its ability to transport large volumes of data into smaller sets of packets
- In addition, it ensures reassembly in a correct sequence, flow control and window adjustment, and retransmission of lost packets
- These benefits occur with the cost of overhead per packet and per session, potentially impacting overall packet per second performances and latency.

وقتی میخوایم تضمین کنیم داده به همون ترتیب ارسال دریافت بشه و گارانتی بشه که دریافت بشه میریم سراغ TCP. هم انجام میده تا جلوی data loss و congestion رو بگیره. ولی در UDP همچنان با یه نزخ ثابت ارسال میکنه.

Review of UDP



- In contrast, UDP is most often used in the context of network services, such as
 - Domain Name System (DNS),
 - Network Time Protocol (NTP),
 - Simple Network Management Protocol (SNMP), and
 - Dynamic Host Control Protocol (DHCP), or
 - for real-time data traffic, including voice and video over IP.
- In these cases, performance and latency are more important than packet retransmissions because re-sending a lost voice or video packet does not add value.
- When the reception of packets must be guaranteed error free, the application layer protocol takes care of that function.



از کاربرد های UDP میبینیم.
جاهایی که Latency خیلی مهمه و مثلا ویدیو کال و ویس کال که یک بیت گم بشه اهمیت نداره میریم سراغ UDP.

Review of UDP



- While the use of TCP may not strain generic compute platforms and high data-rate networks, it can be challenging and is often overkill on constrained IoT devices and networks.
 - This is particularly true when an IoT device needs to send only a few bytes of data per transaction.
 - When using TCP, each packet needs to add a minimum of 20 bytes of TCP overhead, while UDP adds only 8 bytes.
- TCP also requires the establishment and potential maintenance of an open logical channel

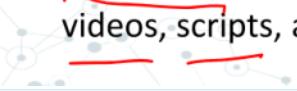


Review of HTPP

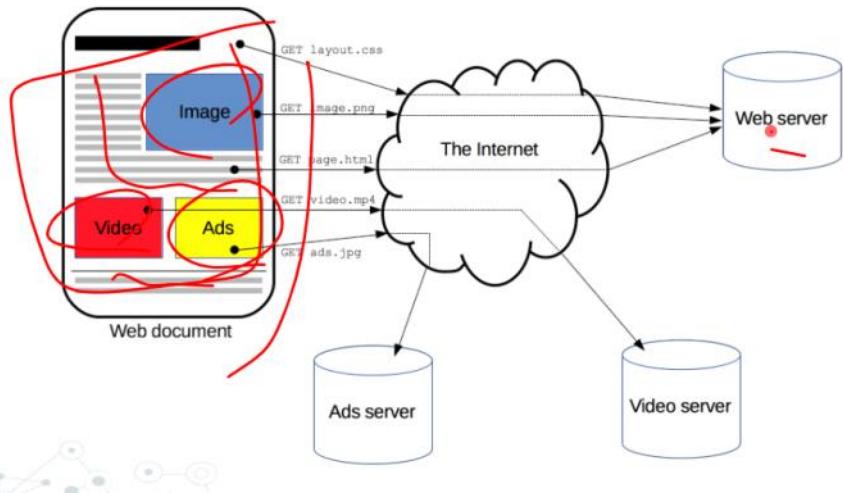


- **HTTP** is a protocol which allows the fetching of resources, such as HTML documents.

- It is the foundation of any data exchange on the Web and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser.
- A complete document is reconstructed from the different sub-documents fetched, for instance text, layout description, images, videos, scripts, and more.



Review of HTPP



Review of HTPP

- URLs

- The URL (Uniform Resource Locator) is probably the most known concept of the Web.
- A URL is a web address used to identify resources on the Web.
- The idea of the web is structured around resources.
- From its beginnings the Web was the platform for sharing text/HTML files, documents, images etc, and as such it can be considered a collection of resources.

`http://www.example.com/search?item=vw+beetle`

Protocol

Domain

Path

Parameters

Review of HTPP

`http://www.example.com/search?item=vw+beetle`

Protocol

Domain

Path

Parameters

IP

DNS

- **Domain** — Name that is used to identify one or more IP addresses where the resource is located.
- **Path** — Specifies the resource location on the server. It uses the same logic as a resource location used on the device where you are reading this article (i.e. /search/cars/VWBeetle.pdf or C:/my cars/VWBeetle.pdf).
- **Parameters** — Additional data used to identify or filter the resource on the server.

Review of HTPP

- **HTTP Requests**

- In HTTP, every request must have an URL address.
- Additionally, the request needs a method.
- The four main HTTP methods are:
 - GET
 - PUT
 - POST
 - DELETE

Review of HTPP

- **HTTP flow**

- When a client wants to communicate with a server, either the final server or an intermediate proxy, it performs the following steps:

- Open a TCP connection:

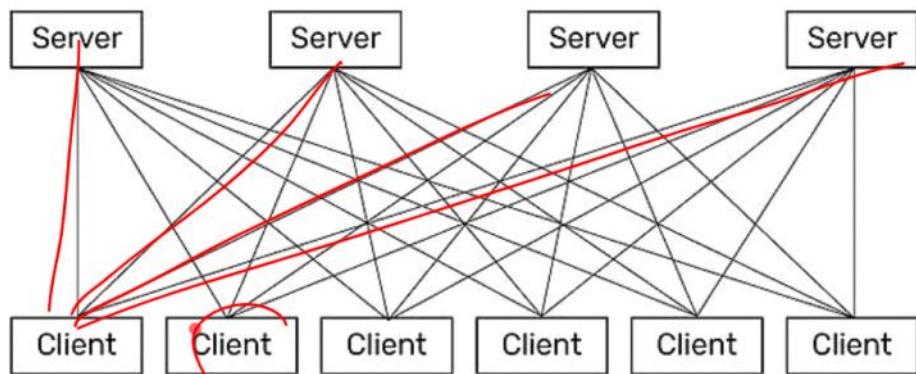
- The TCP connection is used to send a request, or several, and receive an answer.
 - The client may open a new connection, reuse an existing connection, or open several TCP connections to the servers.

- Send an HTTP message:

Review of HTPP

- HTTP functions as a request–response protocol in the client–server computing model.
- A web browser, for example, may be the *client* and an application running on a computer hosting a website may be the *server*.
- The client submits an *HTTP request* message to the server.
- The server, which provides *resources* such as HTML files and other content, or performs other functions on behalf of the client, returns a *response* message to the client.

Review of HTPP



IoT Application Transport Methods

- As an example, consider the Device Language Message Specification/Companion Specification for Energy Metering (DLMS/COSEM) application layer protocol
 - a popular protocol for reading smart meters in the utilities space, is the de facto standard in Europe.
- Adjustments or optimizations to this protocol should be made depending on the IoT transport protocols that are present in the lower layers.
- For example, if you compare the transport of DLMS/COSEM over a cellular network versus an LLN deployment, you should consider the following:
 - Select TCP for cellular networks because these networks are typically more robust and can handle the overhead.
 - For LLNs, where both the devices and network itself are usually constrained, UDP is a better choice and often mandatory.

در صورت استفاده از کنترلر های هوشمند، مصرف به صورت لحظه‌ای می‌شود. فایده هاش:

- ما در لحظه میتوانیم اطلاعات زیاد داشته باشیم
- Big data داریم میتوانیم آنالیز کنیم

- صرف زیاد گاز رو میتوانیم پیشیبینی کنیم و برنامه داشته باشیم

بسته به این که پروتکل ارتباطی چیه، خیلی وابسته به این که در لایه پایین داره چی می‌شود. مثلاً یک لیاس شوی روش میکنیم میبینیم زیاد مصرف کرده.

اگر TCP برای شبکه سلولی انتخاب کنیم باعث می‌شود ارتباط robust بشود و overhead هم هندل بشود.

در LLN که دو ویژگی دارد: دیواپس ها و شبکه محدود. در این شبکه ها قطعاً UDP انتخاب بهتری و بعضی مواقع اجباریه و گزینه دیگه ای نداریم.

IoT Application Transport Methods

- DLMS/COSEM can reduce the overhead associated with session establishment by offering a “long association” over LLNs.
 - Long association* means that sessions stay up once in place because the communications overhead necessary to keep a session established is much less than is involved in opening and closing many separate sessions over the same time period.
 - Conversely, for cellular networks, a short association better controls the costs by tearing down the open associations after transmitting.
- When transferring large amounts of DLMS/COSEM data, cellular links are preferred to optimize each open association.
- Smaller amounts of data can be handled efficiently over LLNs.
 - Because packet loss ratios are generally higher on LLNs than on cellular networks, keeping the data transmission amounts small over LLNs limits the retransmission of large numbers of bytes.

DLMS/COSEM میتوانه Overhead های session establishment را کم کنه. کارشم اینطوریه که session هارو طولانی میگیره

- Session ها باقی میمانن و این باعث می‌شود close نکنیم هارو و در یک long session این کارو بکنیم

- در شبکه سلولی short association داریم و ارتباطات بسته می‌شوند

اگر میخواهیم داده های زیادی رو در قالب DLMS/COSEM میخواهیم منتقل کنیم، اون موقع cellular link ممکنه ترجیح داده بشود.

ولی برای داده های کوچیک که در LLN قراره منتقل بشوند

IoT Application Transport Methods

- The following categories of IoT application protocols and their transport methods are explored in the following sections:
 - Application layer protocol not present:**
 - In this case, the data payload is directly transported on top of the lower layers.
 - No application layer protocol is used.
 - Generic web-based protocols:**
 - Generic protocols, are found on many consumer- and enterprise-class IoT devices that communicate over non-constrained networks.
 - A web protocol like HTTP can also be used on small devices.
 - In the Arduino boards area, for example, there are many examples of smart sensors that have their own web server.
 - The user can direct his browser to the IP address of the device and receive data directly in the form of a web page
 - IoT application layer protocols:**
 - IoT application layer protocols are devised to run on constrained nodes with a small compute footprint and are well adapted to the network bandwidth constraints on cellular or satellite links or constrained 6LoWPAN networks.
 - Message Queuing Telemetry Transport (MQTT) and Constrained Application Protocol (CoAP), covered later in this chapter, are two well-known examples of IoT application layer protocols.

سه رویکرد برای پروتکل app و transport داریم:

- به صورت صریح لایه اپلیکیشن نداریم. مثلاً یک سنسور IoT داریم. اونو در LoRaWAN Payload قرار میدیم و هیچ چی در اپلیکشن قرار نمیدیم. یه چیز من درآورده میشه که خودمون تعیین کنیم بیت هاش چه معنی میده و در کاربرد خودمون استفاده میشه فقط
- همین پروتکل های وب معمول رو در IoT استفاده کنیم. مثلاً در برد Arduino دیدیم مثال های هست میشه Web server باشه و یوزر با browser وصل بشه و داده هارو در قالب page از سنسور بگیره
- پروتکل app مخصوص IoT داریم. با فرض محدود بودن نود و قدرت پردازشی کم میایم یک پروتکل مخصوص IoT تعریف میکنیم. یکیش MQTT یکیش CoAP که جلوتر میبینیم.

Application Layer Protocol Not Present

- As introduced before, IETF RFC 7228 devices defined as class 0 send or receive only a few bytes of data
- For myriad reasons, such as processing capability, power constraints, and cost, these devices do not implement a fully structured network protocol stack, such as IP, TCP, or UDP, or even an application layer protocol.
- Class 0 devices are usually simple smart objects that are severely constrained.
- Implementing a robust protocol stack is usually not useful and sometimes not even possible with the limited available resources.

در IETF RFC 7228 که توان پردازشی محدود دارن و IP رو نمیتونن بپاده سازی کنن.

ما نمیتونیم برای یک دیواس 0 class که داره از LoRaWAN و یا Sigfox استفاده میکن، بیایم TCP و UDP و .. داشته باشیم.

چون از لحاظ منابعی محدود هستن این که بیایم یک protocol stack درست بکیم مفید نیست و حتی بعضی موقع ها غیر ممکنه و به جاش معمولاً لایه اپلیکیشن تعریف نمیکن.

Application Layer Protocol Not Present

- For example, consider low-cost temperature and relative humidity (RH) sensors sending data over an LPWA LoRaWAN infrastructure.
- Temperature is represented as 2 bytes and RH as another 2 bytes of data. Therefore, this small data payload is directly transported on top of the LoRaWAN MAC layer, without the use of TCP/IP.

در اینجا هم یک مثالی میبینیم که 2 بایت داده دما رو بدون این که از لایه Application و Transport داشته باشیم میتوانیم به صورت شخصی بالای لایه Mac در LoRaWAN کد اش کنیم و بفرستیم.
(Payload Mac Layer)

Application Layer Protocol Not Present

- Following Example shows the raw data for temperature and relative humidity and how it can be decoded by the application.

Temperature data payload over the network: Tx = 0x090c
Temperature conversion required by the application
 $T = Tx/32 - 50$ to $T = 0x090c/32 - 50$ to $T = 2316/32 - 50 = 22.4^\circ$
RH data payload over the network: Rx = 0x062e
RH conversion required by the application:
 $100RH = Rx/16-24$ to $100RH = 0x062e/16-24 = 74.9$ to RH = 74.9%

نشون میده داده دما چطوری داره میره و در گیرنده چطوری decode میشه.

Application Layer Protocol Not Present

- While many constrained devices, such as sensors and actuators, have adopted deployments that have no application layer, this transportation method has not been standardized.
- This lack of standardization makes it difficult for generic implementations of this transport method to be successful from an interoperability perspective.

پس تعداد زیادی Constrained device ممکنه از این approach استفاده کنن. ولی این باعث میشه که Standardization نداشته باشیم. که باعث میشه برای Generic implementationoon Interoperability سخت میشه و نمیشه داشته باشه (هر اپلیکیشنی با هر سنسوری نمیتونه کار کنه)

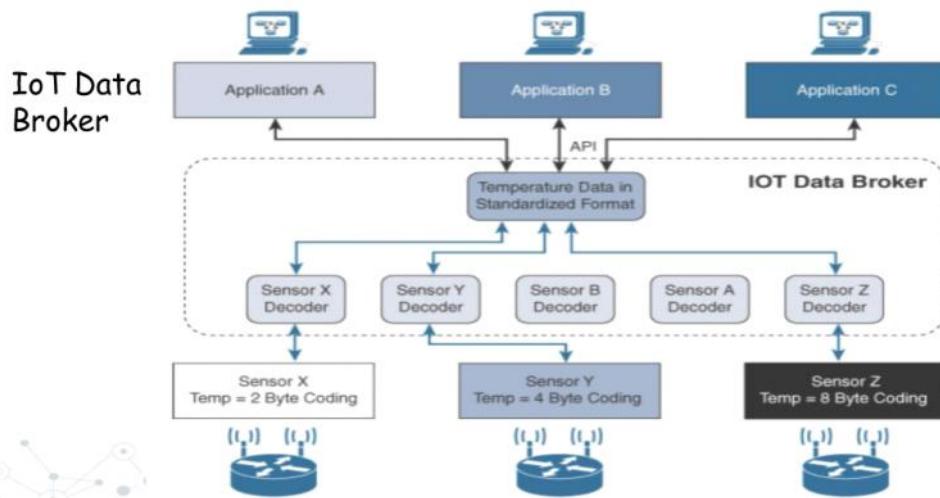
Application Layer Protocol Not Present

- Consider different kinds of temperature sensors from different manufacturers.
 - These sensors will report temperature data in varying formats.
 - A temperature value will always be present in the data transmitted by each sensor, but decoding this data will be vendor specific.
- If you scale this scenario out across hundreds or thousands of sensors, the problem of allowing various applications to receive and interpret temperature values delivered in different formats becomes increasingly complex.
- The solution to this problem is to use an IoT data broker, as detailed in Figure of next slide
 - An IoT data broker is a piece of middleware that standardizes sensor output into a common format that can then be retrieved by authorized applications.

مشکل دیگش اینه که فرض کنیم سنسور های مختلفی داریم. هر کدام temperature value های مختلفی را با فرمات های مختلفی پیورت میکنن. وجود داره ولی decode کردنش به vendor ریط داره. اگر همینو up کنیم برای 100 ها سنسور مشکلی که به وجود میاد اینه که ما یک سری مقادیری در فرمات های مختلف دریافت میکنیم و اپلیکیشن و مدیریت کار مارو سخت میکنه.

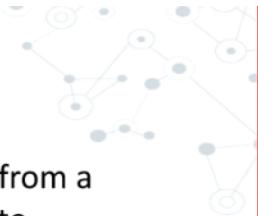
پس یک middleware میخوایم که یک data broker هستش که خروجی سنسور هارو به یک فرمی که application ها بفهمن در میاره

Application Layer Protocol Not Present



الان مثلث سنسور هارو میبینیم هر کدام byte coding متغیر است. در scale زیاد نمیتوانیم مستقیماً وصل کنیم و بهتره data broker داشته باشیم که یک middleware هستش در بینشون قرار میده و Temperature data را به یک استاندارد در میارد و به اپلیکیشن ها میده. این کار باعث میشه دیگه Application ها درگیر نشن چه فرمی داره.

Application Layer Protocol Not Present



- You should note that IoT data brokers are also utilized from a commercial perspective to distribute and sell IoT data to third parties.
 - Companies can provide access to their data broker from another company's application for a fee.
- This makes an IoT data broker a possible **revenue stream**, depending on the value of the data it contains.

از نظر commercial، باید به این توجه کنیم صاحب سنسور و data broker و اپلیکیشن یکیه یا نه؟. لزوماً در آینده IoT اینطوری نیست. یه عده سنسور درست میکنن یه عده اپلیکیشن یه عده هم data broker پس data broker یک جریان درآمدی (revenue system) در آینده درست میکنه و بیزنس های در IoT داریم که data broker اون.

Application Layer Protocol Not Present



- Directly transporting data payload without a structured network stack clearly optimizes data transmission over low-data-rate networks,
- However, the **lack of a data model** implies that each application needs to know how to interpret the data-specific format.
 - This becomes increasingly complex for larger networks of devices with different data payload formats.
- Furthermore, it makes the IoT application environment challenging in terms of evolution, development, interoperability, and so on, and often calls for structured data models and data broker applications.

به عنوان یک جمع بندی، این که دیتا رو transport کنیم بدون استفاده از structured network باعث میشه در شبکه های low-data-rate interoperability و development و evolution اشونو سخت میکنه. ولی نبود یک data model برای شبکه های بزرگ کار رو سخت میکنه.

Web Protocol for the Internet of Things



- A web protocol like HTTP can also be used on small devices.
 - In the Arduino boards area, for example, there are many examples of smart sensors that have their own web server.
 - The user can direct his browser to the IP address of the device and receive data directly in the form of a web page.
- Why don't all small devices automatically become "web services"?

در رویکرد دوم میخوایم web protocol مثل HTTP را میخوایم در smart device هامون استفاده کنیم. این که یک web server بیاریم بالا و داده هارو از مرورگر دریافت کنیم. سوال اینه که چرا نباید دستگاه های small رو به صورت web service استفاده کنیم؟

Web Protocol for the Internet of Things

- Let's take a closer look at the way HTTP works
 - The protocol is used to interact with web resources (e. g. web pages or forms) on a server.
 - Several methods are available for this purpose.
 - Most inquiries are made to request data ("GET").
 - Other inquiries data can be transferred, e. g. when an order form has to be filled out in the online shop (method "POST").
- In principle, HTTP methods can also be used for interaction with devices:
 - Getting data from the device ("GET /temperature"), or
 - Control things/actuators ("POST /fan/control").
- Most developers are already familiar with HTTP web services. So, why not just have IoT devices connect to web services?

باید یه نگاه دقیق تر به HTTP داشته باشیم. این پروتکل برای تعامل بین web resource درست شده و یک سری متد مثل GET برای دریافت داده و POST برای ارسال داره. اگر بخوایم با بین دستگاه استفاده کنیم، مثلاً با GET داده رو از سنسور بخونیم و با POST بگیم فن با یک سرعی بچرخه این که خیلیاً با HTTP آشنان این که در IoT استفاده کنیم خوبه چون کار با developer ها ساده تره.

HTTP: Not necessarily suitable for small devices

- There are several reasons why HTTP is not well suited to interacting with constrained devices:
 - HTTP is a heavy weight protocol with many headers and rules
 - HTTP is a chatty protocol from the point of view of resource restriction.
 - For example, meta information such as the accepted formats or the desired language are transmitted to the server in plain text form.
 - The header of an HTTP request can already be large, and the header of an HTTP response can easily contain several hundred bytes.
 - For the query of a web page with several megabytes of size this is not bad, but for the query of a temperature value in the range of approx. 5 bytes it is significant whether the transmission protocol is slim or rather broad.

اگرچه از نظر تئوری امکان پذیره، چند دلیل هست که میگه برای constrained device ها خوب نیست:

- پروتکل تقریبا سنگینیه و هدر و قوانین زیادی دارد. زمانی که HTTP نوشته میشد برای ارتباطات بین انسان ها بود. چون مثلا google.com رو نسبت به IP اش راحت تر میفهمیم. ولی دیگه برای دیواپس های IoT ما دیگه DNS نیاز نداریم.
- یک پروتکلی هست که بین کلاینت و سرور کلی داده رد و بدل میشه. مثلا accepted format داره که عدد و متننت فارسیه یا انگلیسی هدراش هم خیلی زیاده و میتوشه تا چندین صد بایت باشه
- برای کوئری یک temperature value که نهایتا 5 بایته اصن منطقی نیست که چندصد بایت هدر HTTP بداریم.

HTTP: Not Necessarily Suitable for Small Devices

– HTTP is a synchronous protocol.

- The client waits for the server to respond.
- That is a requirement for web browsers, but it comes at the cost of poor scalability.
- In the world of IoT, the large number of devices and most likely an unreliable / high latency network have made synchronous communication problematic.
- An asynchronous messaging protocol is much more suitable for IoT applications. The sensors can send in readings, and let the network figure out the optimal path and timing for delivery to its destination devices and services.

مشکل دیگش اینه که یک پروتکل سنکرونیه. کلاینت برای سرور باید منتظر بمونه که respond کنه. اما در دنیای IoT ما تعداد زیادی دیواپس داریم که استفاده unreliable/ high latency network میکنیم و ممکنه داده نرسه. پس یک پروتکل آسنکرون بهتره. اینطوری نباشه که timeout بخوره و سنسور داده رو میفرسته به شبکه بسپره.

HTTP: Not Necessarily Suitable for Small Devices

– HTTP is one-way.

- The client must initiate the connection.
- In an IoT application, the devices or sensors are typically clients, which means that they cannot passively receive commands from the network.

– HTTP is a 1-1 protocol.

- The client makes a request, and the server responds.
- It is difficult and expensive to broadcast a message to all devices on the network, which is a common use case in IoT applications.

در HTTP پروتکل یه طرفس و کلاینت داده رو شروع میکن. در IoT سنسور ها میشن کلاینت و این یعنی از سرور نمیتونن دستوری دریافت بشن. ولی در IoT شاید بخوایم اینطوری باشه که اپلیکیشن تقاضا بده که داده رو برام بخون ارسال کن و در این نوع کاربرد های IoT مشکل به وجود میاد.

پروتکل 1-1 هم هست یعنی کلاینت رکوئست میده سرور جواب میده. اینطوری نیست که کلاینت یه درخواست بده و سرور 10 تا جواب بده. مثل حالی که broadcast داریم و در IoT نیاز هست. مثلا سرور میخواهد 100 تا نود بگه داده هارو ارسال کنه. وقتی broadcast کنه خیلی بهتره تا این که 100 تارو دونه دونه ارسال کنه.

IoT Protocol Stack- Application Layer Protocols

- When considering constrained networks and/or a large-scale deployment of constrained nodes, verbose web-based and data model protocols, as discussed in the previous section, may be too heavy for IoT applications.
- To address this problem, the IoT industry is working on new lightweight protocols that are better suited to large numbers of constrained nodes and networks.
- Two of the most popular protocols are CoAP and MQTT.
- Before introducing CoAP and MQTT, we review two data exchange models

موقعی که Constrained network و high scale هم داریم. یک پروتکلی که جزئیات زیادی داره و data model پیچیده ای داره برای IoT سنگینه. پس در صنعت IoT روی پروتکل های lightweight کار میشه. که 2 تا از شهرهار هاش CoAP و MQTT هستند.

قبل وارد شدن به این 2 تا وارد رویکرد سوم میشیم که مشکلات application not present و HTTP رو نداشته باشه.

ما برای exchange کردن داده 2 مدل داریم:

Request-Response vs. Publish-Subscribe

- How does a spreadsheet get to the printer, a YouTube video get to your smartphone, or—most important for automation engineers—a value from a sensor get to your HMI?
- Two Communication (Data Exchange) Models:
 - Request and Response
 - Publish and Subscribe

ما وقتی یک فایل اکسل را داریم میخواهیم پرینت کنیم یا یک ویدیو یوتیوب را میخواهیم مشاهده کنیم یا یک عددی را از یک سنسوری رو از یک میخواهیم بخونیم، چه communication model ای استفاده میکنیم؟ عموماً ۲ تا داریم

- HTTP: Request response •
- Publish and Subscribe •

Request-Response vs. Publish-Subscribe

- Request and Response Model
 - A client computer or software requests data or services, and a server computer or software responds to the request by providing the data or service
 - Examples:
 - Send a spreadsheet to the network printer
 - your spreadsheet program is the client and print server, responds to the request and allocates resources for printers on the network.
 - Watching the YouTube video on your smartphone
 - your web browser or YouTube app is the client, YouTube's web server receives the request and responds by serving the video page to you

Request-Response vs. Publish-Subscribe

- Publish and Subscribe
 - A central source called a broker (also sometimes called a server) which receives and distributes all data.
 - Pub-sub clients can publish data to the broker or subscribe to get data from it—or both.
 - Clients that publish data send it only when the data changes (report by exception, or RBE).
 - Clients that subscribe to data automatically receive it from the broker/server, but again, only when it changes.
 - The broker does not store data; it simply moves it from publishers to subscribers.
 - When data comes in from a publisher, the broker promptly sends it off to any client subscribed to that data.

در این مدل یک سری کلاینت داریم که هم میتوانن publish کن و دیتابی رو در broker قرار بدن. هم مثل این که ثبت نام کنیم وقتی دیتا publish شد برآمدون ارسال بشه.

هر موقع دیتا عوض شد کلاینت ها subscribe میکنند و اونایی که subscribe کردن وقتی دیتا تغییر کرد دریافت میکنند. نقش storage نمیکنند بلکه اینه که مدل data exchange رو تسهیل کنه.

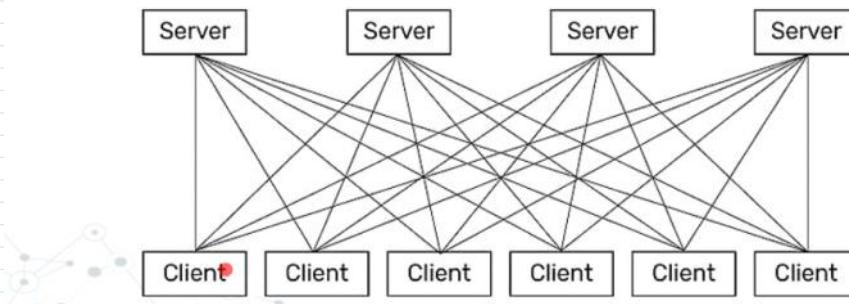
Request-Response vs. Publish-Subscribe-Which to Use?

- In a request-response architecture, each client opens a direct connection to each server, because the client requests data directly from the server.
 - In automation, as fast as multiple times per second—and servers repetitively respond:
 - Q: What's the sensor value? A: 10
 - Q: What's the sensor value? A: 9
 - Q: What's the sensor value? A: 9
- If your network is robust and has few servers, request-response model works very well.
 - As long as the server has the capacity to respond to client demands and the network can handle the volume of traffic, request-response is a proven, reliable communication method. It's particularly useful for communications over a secure internal network.

تفاوت چیه؟ در client-response هر کلاینت مستقیماً به سرور درخواست رو میده. ولی نمیدونه تغییر کرده یا نه و دوباره درخواست ارسال میکنه ولی در pub-sub هرموقع عوض شد براش ارسال میشه و نیازی به این همه درخواست نیست.

Request-Response vs. Publish-Subscribe-Which to Use?

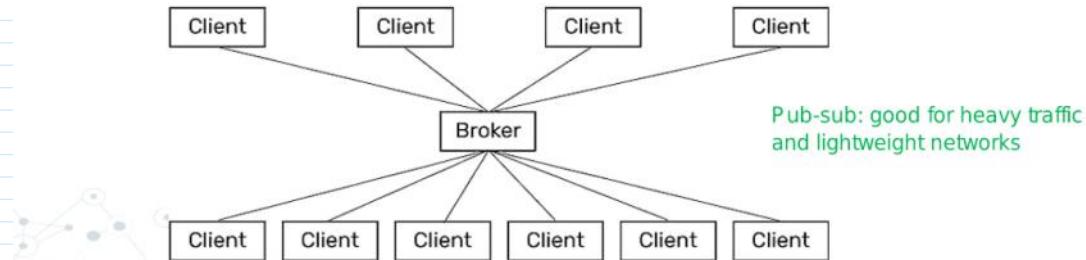
- What about traffic volume?
 - If you have multiple servers with multiple clients, however, the volume of traffic in a request-response model can quickly become a problem.



در کلاینت سرور وقتی تعدادشون زیاد بشه حجم داده زیاد میشه و مشکل ساز میشه

Request-Response vs. Publish-Subscribe-Which to Use?

- In contrast, a pub-sub architecture simplifies communications.
 - Direct connections and repetitive requests for data are not needed.
 - The web of links is replaced by a single link from each device to the broker (also called a server).
- The connection between client and broker is kept open and is incredibly lightweight.
 - Only two things travel over this connection: changed data, and a tiny heartbeat to let the broker know that the client is still there.



ولی در pub-sub بالایا داده هاشونو در broker قرار میدن و اگر client ها subscribe کرده باشن برashون ارسال میشه و درخواست های تکراری حذف میشه و ارتباط ساده تر میشه.

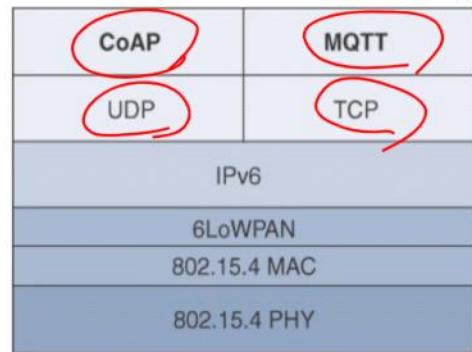
Request-Response vs. Publish-Subscribe-Which to Use?

- So a pub-sub model can make sense if you have many servers and many clients that need to share data and services.
- Since the broker is the central clearinghouse for data, individual servers don't have to strain to serve multiple clients, and clients don't have to connect to multiple servers.
- In addition, network traffic is reduced overall, because data is published and sent on a report-by-exception (RBE) basis.
 - that is, only when the data changes—rather than at regular intervals.
- Pub-sub can also make sense when it's difficult to set up a direct connection between a client and a server, or when the network is low-bandwidth, expensive, or unreliable—for example, when monitoring equipment in remote locations.

وقتی تعداد زیادی سرور و کلاینت داریم برای اون کار مناسبه. Pub-sub model

IoT Application Layer Protocols

- Constrained Application Protocol (*CoAP*)
 - Uses both client-server and pub-Sub methods
 - UDP-based
- Message Queuing Telemetry Transport (*MQTT*)
 - Based on pub-sub
 - TCP-based



جلسه بعد وارد CoAP و MQTT میشیم. CoAP مبتنی بر UDP و MQTT هم TCP هم

- Constrained Application Protocol (CoAP)
 - Uses both client-server and pub-Sub methods
 - UDP-based
- Message Queuing Telemetry Transport (MQTT)
 - Based on pub-sub
 - TCP-based

CoAP	MQTT
UDP	TCP
IPv6	
6LoWPAN	
802.15.4 MAC	
802.15.4 PHY	

ازین پروتکل‌های لایه اپلیکیشن، CoAP و MQTT را میخوایم بررسی کنیم. برای حل مشکل reliability در CoAP تمهدیات قرار داده شده است.

IoT Application Layer Protocols-CoAP

- CoAP has been created close to HTTP in several aspects:
 - It is basically a request/response protocol,
 - It integrates the URIs known from the web world to name resources.
 - It also adopts some of the query methods (such as "GET", "PUT", "POST", etc.) and defines response codes similar to those of HTTP (e.g. 4.04 "Not found" if a resource could not be found).
- Web developers who have grown up with HTTP can quickly find their way around CoAP.

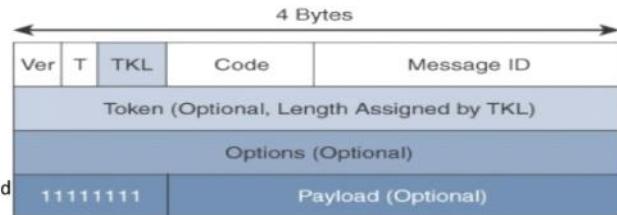
پروتکل CoAP تا حد امکان شبیه HTTP درست شده. بر اساس request/response هستش. مفهوم URI رو برای نامگذاری منابع قرار داده. متدهای که در HTTP داریم از GET و POST در اینجا هم هست.

همین Response code 404 هم دارد مثل همین تشابهات باعث شده کسایی که با HTTP آشنا هستن خیلی راحت با CoAP کار کنن.

IoT Protocol Stack- Application Layer Protocols

Application Layer Protocols- CoAP

- The CoAP framework defines simple and flexible ways to manipulate sensors and actuators for data or device management
- A CoAP message is composed of
 - a short fixed length Header field (4 bytes),
 - a variable-length but mandatory Token field (0-8 bytes),
 - Options fields if necessary, and the Payload field



از نظر framework، این پروتکل روش ساده و flexible برای ارتباط سنسور ها و actuator برای مدیریت داده و دیوایس دارد. (دیوایس منجر مثلاً میشه خوندن باتری و اگر actuator هست یک کاری رو بکنه)

پیام CoAP رو که message میگیم. از چند قسمت تشکیل شده:

- یک Header که 4 بایته
- که یک نقش اصلیش Correlation بین ریکوئیست و رسپانس هستش. (0 تا 8 بایت)
- یک قسمت Optional داره که ما میتونیم تو ش هرجی میخوایم تعريف کنیم
- و در آخر یک payload داره

CoAP Message Fields

CoAP Message Field	Description
Ver (Version)	Identifies the CoAP version.
T (Type)	Defines one of the following four message types: Confirmable (CON), Non-confirmable (NON), Acknowledgement (ACK), or Reset (RST). CON and ACK are highlighted in more detail in Figure 6-9.
TKL (Token Length)	Specifies the size (0-8 Bytes) of the Token field.
Code	Indicates the request method for a request message and a response code for a response message. For example, in Figure 6-9, GET is the request method, and 2.05 is the response code. For a complete list of values for this field, refer to RFC 7252.
Message ID	Detects message duplication and used to match ACK and RST message types to Con and NON message types.
Token	With a length specified by TKL, correlates requests and responses.
Options	Specifies option number, length, and option value. Capabilities provided by the Options field include specifying the target resource of a request and proxy functions.
Payload	Carries the CoAP application data. This field is optional, but when it is present, a single byte of all 1s (0xFF) precedes the payload. The purpose of this byte is to delineate the end of the Options field and the beginning of Payload.

هدر 4 بایته که چند بیت اولش version رو مشخص میکنه. ما وقیعی به پروتکل درست میکنیم تمام سعیمون اینه که کامل بشه. ولی در عمل با مشکلاتی برخورد میکنیم که ممکنه توسعه داده بشه. پس یه کاری که میکنیم اینه که یه بیت رزرو میذاریم مثلاً تایپ هاش اگه 4 تاست با 3 بیت میذاریم که بشه مشخص کرد.

ولی بعضی موقع ها هست که پروتکل با 1 بیت 2 بیت کارش راه نمیافته. پس یک version میذاریم که اول version خونده بشه و بعد باقی بیت ها بسته به این که version چیه تفسیر بشه.

فیلد بعدی type که مشخص میکنه کدام یک از 4 تا نوع مسیح هستن:

- CON •
- NON •
- ACK •
- RST •

که بعد تو پیچشون میدیم.

طول token (بین 0 تا 8 بایت) مشخص میکنه فیلد توکن چقدریه

یک code داریم که مشابه کدهای HTTP هستن

یک id message که میده که اگر بعداً 2 بار دریافت کردیم بفهمیم duplicate شده. همچنین استفاده میشه برای این که ACK و RESET رو مجکنیم به CON و NON

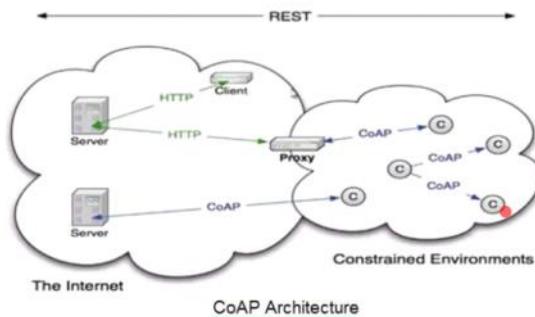
هم که بین ۰ تا ۸ بایت هست بعدش میاد و درخواست و response رو پک همیستگی بینشون برقرار میکنه.

در فیلد بعدی، یک توانمندیه برای این که یک سری کار بکنیم. مثلاً مشخص کنیم target resource برای رکوست با پروکسی رو مشخص کنیم. پس کارای پروکسی رو اینجا میتوانیم بذاریم Options

CoAP اصلی: Payload

• Application Layer Protocols- CoAP

- CoAP Architecture



میبینیم به طرف پراکسی HTTP عه به طرفش node های IoT عه که با CoAP کار میکنن.

IoT Application Layer Protocols-CoAP

- Through the exchange of asynchronous messages, a client requests an action via a method code on a server resource.
- A uniform resource identifier (URI) localized on the server identifies this resource.
- The server responds with a response code that may include a resource representation.
- The CoAP request/response semantics include the methods GET, POST, PUT, and DELETE.

در CoAP مسیج های ما آسنکروننه. یعنی مثل HTTP نیست منتظر ACK باشیم نیومد دوباره بدیم. کلینت به کمک METHOD CODE رو از سرور تقاضا میکنه.

هر سرور هم یک URI داره که مشخص میکنه کجاست.

سرور هم با یک Response code جواب میده

IoT Application Layer Protocols-CoAP

- CoAP Messages Type

1. Confirmable

- a reliable transmission of messages on UDP protocol
- basic congestion control with a default time-out
- simple stop and wait retransmission with exponential back-off mechanism
- detection of duplicate messages through a message ID
- while running over UDP, CoAP offers a reliable transmission of messages when a CoAP header is marked as "confirmable."

2. Non-Confirmable

- not require reliable transmission

اگر مسیح CON باشد، یک ارتباط reliable در پست UDP رخ میده. ما اونجه که از UDP میدونیم اینه که سریارش از TCP کمتره ولی خیلی CoAP میتوانیم اگر بخواهی Reliable نیست. اما اگر بخواهی CoAP هم داشته باشیم و بخواهی reliability CON نیست. یک سری کارای Congestion control انجام میده مثلاً اگر دریافت نشد بعد به مدت دوباره بفرسته. Stop and wait هم دارد و اگر بعد به مدت دید دریافت نمیکنند Congestion control بعد ارسال میکنند. با این که UDP ران میشه، وقتی در قسمت اپلیکیشن CON باشد یک reliable transmission رو تضمین میکنند.

- CoAP Messages Type

3. Acknowledgement

- the recipient must explicitly either acknowledge or reject the confirmable message using the same message ID

4. Reset

- a recipient sends a reset message when can't process a confirmable or non-confirmable message

مسیح از نوع ACK هم میتوانه باشد. وقتی CON باشد که بخود ACK بفرسته باید تایپش موجود باشد. اگر بنا بر هر دلیلی ارور بخوریم یک reset میناریم که بگیم من دریافت کدم ولی این متده که فرستادین رو نتونستیم پردازش کنیم.

IoT Application Layer Protocols-CoAP

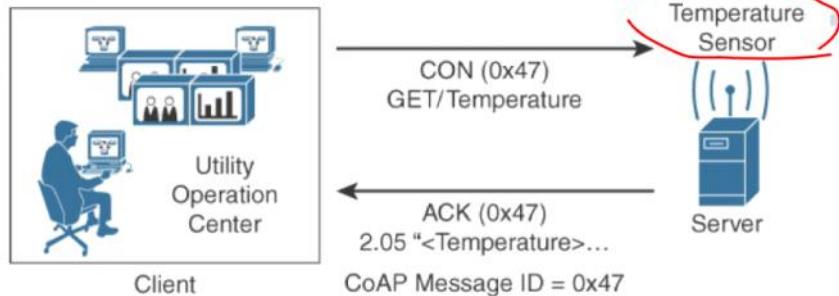
- CoAP Messages semantics

1. Get
2. Post
3. Put
4. Delete

- Method codes and response codes included in some of these four-types messages make them carry requests or responses

این متده و رسپانس کد در هر 4 نوع مسیح CON و NON و ACK و RESET موجوده.

IoT Application Layer Protocols-CoAP



الان مثلا میخوایم دما رو بخونیم. باید یک REQUEST بفرستیم (کلاینت). مثلا نوعش GET/Temperature هست که میخواد داده دما رو بگیره. نوعش CON گذاشته پس نیاز به ACK داریم. که یعنی هم من رکوسرست شمارو دریافت کردم هم ای response code میده.

IoT Application Layer Protocols-CoAP

- **CoAP vs HTTP**
 - CoAP is upgraded version of HTTP.
 - It is designed for resource constrained applications such as IoT/WSN/M2M etc.
 - CoAP is based on UDP.
 - It uses ACK messages so that it will become reliable like TCP.
 - CoAP has low latency and consumes lesser power compare to HTTP.

در مقایسه بین CoAP و HTTP آنکه CoAP آبگرفت ورژن HTTP عه برای اپلیکیشن هایی که Resource constrained هستند. CoAP مبتنی بر UDP عه اگر چه میتوانه از CON استفاده کنه که Reliable بشه.

در MQTT هر مسیحی مبتنی بر TCP عه ولی در CoAP مبتنی بر UDP عه و تها زمانی از CON استفاده میکنیم که برآمدون مهم باشه.

Latency کمتری داره و Power کمتری نسبت به HTTP داره.

IoT Application Layer Protocols-CoAP

- For example:
 - If a temperature sensor transmits the temperature 1x per minute, a failure of a single message can be merged, it could be mapped as a non-confirmable.
 - If a smart door lock is supposed to lock the door, the message behind it is surely to be classified as “Confirmable”.

در مثال قفل در هوشمند، اگر بخوایم مبتنی بر CoAP باشه بین قفل (Actuator) و اپلیکیشن ما یک ارتباط CON نیاز داریم. چون اگر یه بار بخوایم در باز شه نشه برای مشتری یعنی این یک محصول بد و نمیرن سمتیش. ولی وقتی CON باشه با هریار درخواست دیگه اطمینان برای انجامش داریم

ولی مثلا در زمین کشاورزی یک سری نود داریم میخواد رطوبت خاک اطلاعاتش ارسال بشه، اون موقع بحث امنیق و real time بودنش مهم نیست و میشه NON ارسال کرد.

• CoAP vs HTTP

- The asynchronicity of message exchange via UDP also gives CoAP a device the possibility of responding to a request in a resource-saving and time-delayed manner, as well as of transmitting a response in several small (partly “smallest”) blocks.
- The latter is particularly useful for lossy networks with small packet sizes such as those in the WPAN and LPWAN.

در دیگه CoAP 3 way handshake داریم و یک ویژگی به CoAP میده که دیواپس های CoAP-based بتوون پاسخ یک تقاضا رو به صورت resource-saving بدن (اول خواب باشن بیدار بشن جواب بدن برن تو خواب) و هم این که time-delayed manner هستن یعنی چند لحظه بعد جواب بد. همچنین توانایی این رو داره که رسپانس رو به چندتا بلاک کوچیک تقسیم کنه و ارسال کنه. این که داده کوچیک ارسال کنه به درد شبکه های Lossy میخوره چون وقتی میخوایم مخصوصاً در CON دوباره ارسال کیم یک پکت کوچیک رو ارسال کنیم.

• CoAP vs HTTP

Feature	CoAP	HTTP
Protocol	It uses UDP.	It uses TCP.
Network layer	It uses IPv6 along with 6LoWPAN.	It uses IP layer.
Multicast support	It supports.	It does not support.
Architecture model	CoAP uses both client-Server & Publish-Subscribe models.	HTTP uses client and server architecture.
Synchronous communication	CoAP does not need this.	HTTP needs this.
Overhead	Less overhead and it is simple.	More overhead compare to CoAP and it is complex.
Application	Designed for resource constrained networking devices such as WSN/IoT/M2M.	Designed for internet devices where there is no issue of any resources.

از لحاظ Multicast (میسیج هایی که میخوایم به بار ارسال بشه ولی طیف زیادی بگیرن)، HTTP سایپورت نمیکنه ولی CoAP میکنه. بنابراین یک پیام رو میخوایم بدیم مثل اپلیکیشن کشاورزی داریم میخوایم کسایی که در یک محدوده هستن داده هاشونو دریافت کنیم. در HTTP باید برای هر یک سرور یک داده ارسال بشه ولی در CoAP همه میتوون به یکی بفرستن.

از نظر CoAP Architecture در معماری Pub-sub هم میتوونه به کار بگیره ولی HTTP حتماً باید request-response باشه

از لحاظ سنکرون بودن هم CoAP حتماً نیاز بهش نداره ولی HTTP نیاز داره. چون باید در HTTP یک کانکشن اتفاق بیافته

از لحاظ پیچیدگی CoAP ساده تره

از لحاظ کاربرد CoAP برای دیواپس های محدود هستش ولی HTTP برای دستگاه های موجود در اینترنت

خلاصه مطلب که CoAP ویژگی های HTTP رو داره ولی خاص دیواپس های محدود درست شده مثل سنکرون بودن رو برداشته یا CON و NON گذاشته.

IoT Application Layer Protocols-MQTT

- At the end of the 1990s, engineers from IBM and Arcom (acquired in 2006 by Eurotech) were looking for a reliable, lightweight, and cost-effective protocol to monitor and control a large number of sensors and their data from a central server location, as typically used by the oil and gas industries.
- Their research resulted in the development and implementation of the Message Queuing Telemetry Transport (MQTT) protocol that is now standardized by the Organization for the Advancement of Structured Information Standards (OASIS).

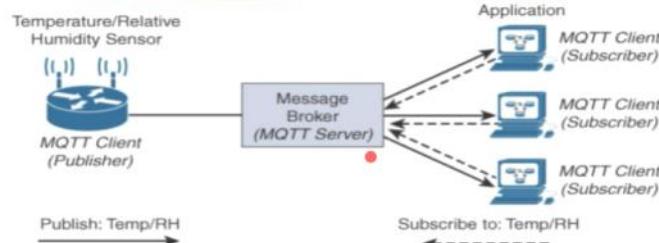
از انتهای سال 1990 مهندسان IBM و شرکت Arcom (که در سال 2006 توسط Eurotech خریداری شد) برای یک پروتکل (که در تضادن) و کم هزینه دنبال یک پروتکل بودن. تا بتون یک سنسور های زیادی در سرور مرکزی در صنعت نفت و گاز مورد استفاده قرار بگیرد. صنعت نفت و گاز قبل IoT از نوع OT(Operational Technology) هستش. اینا خط لوله هاشون طوریه که گاز باید به طور فیزیکی منتقل بشن. گاز هم باید به نقاط مختلف کشور و یک مسافت طولان رو طی کنه. در مسیر هم که مدام نمیشه آدم گذاشت چون کیلومتر های زیادی هستش. یک سری شبکه دارن که مخصوص خودشونه بعد یک سری سنسور مثل میزان آب و نشستی و تست فشار و .. دارن. این شبکه های کنترل صنعتی با وجود IoT دارن Merge میشن باهاش در صورتی که تاریخچه بیشتری از IoT دارن.

تحقیقی که انجام دادن منجر به تولید پرونک MQTT OASIS استاندارد شده و در کاربرد های IoT هم استفاده میشه

IoT Application Layer Protocols-MQTT

Application Layer Protocols- MQTT

- MQTT Publish/Subscribe Framework



- Is like Twitter

میتبی بر pub-sub هستش. یک سری نود داریم به broker وصل میشن. این نود ها یا تحت عنوان publisher میان publish subscriber دارن subscribe میکنن. مثل توییتر که وقتی یک رو فالو کن و توییت کنه برامون نمایش داده میشه.

IoT Application Layer Protocols-MQTT

- With MQTT, clients can subscribe to all data (using a wildcard character) or specific data from the information tree of a publisher.
- In addition, the presence of a message broker in MQTT **decouples the data transmission between clients acting as publishers and subscribers**.
 - In fact, publishers and subscribers do not even know (or need to know) about each other.
 - A benefit of having this decoupling is that the MQTT message broker ensures that information can be buffered and cached in case of network failures.
 - This also means that publishers and subscribers do not have to be online at the same time.

کلاینت ها میتوانند برای تمام داده ها یا یک داده خاص **subscribe** کنند.
از خوبیای broker هم اینه که دیگه subscriber و publisher ها نیازی نیست همو بشناسن و فقط داده رو به کمکش منتقل میکنند.
مزیت این decoupling اینه که آسیکرون میکنه و نیازی نیست هر 2 تا نود بیدار باشن برای برقراری ارتباط.

اگر من رو TCP میکردیم با توجه به request-response جفتشون باید هم زمان آنلاین میبودند. پس UDP کردیم که هرجند وقت به بار بفرسته تا این که طرف آنلاین بشه و بگیره در MQTT ولی چون بروکر داریم TCP کردیم که اگر طرف خاموش بود هم مشکل برآمون پیش نیاد (چون TCP فاز برقراری ارتباط داره از اول) و هر موقع که طرف subscriber آنلاین شد داده براش ارسال بشه.

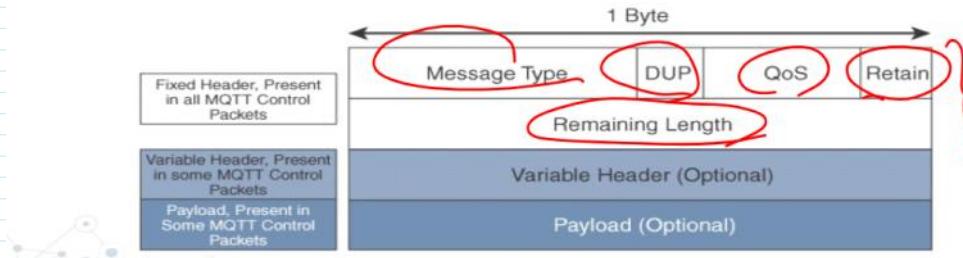
IoT Application Layer Protocols-MQTT

- MQTT is extremely lightweight**
 - it takes up almost no space in a device, so that even small devices with very little computing power can use it.
- Each control packet consists of a 2-byte fixed header with optional variable header fields and optional payload.
 - Note that a control packet can contain a payload up to 256 MB.

IoT Application Layer Protocols-MQTT

MQTT Message Format

- Compared to the CoAP message format, MQTT contains a smaller header of 2 bytes compared to 4 bytes for CoAP



از 2 بایت فیکس که داده یک باشند برای اون بالایه یک باشند هم remaining length داریم و در آخر payload که داده را منتقل میکنند.

IoT Application Layer Protocols-MQTT

Message Type

- identifies the kind of MQTT packet within a message
- Fourteen different types of control packets are specified in MQTT version 3.1.1.
- Each of them has a unique value that is coded into the Message Type field.

از نظر هر کدامیکننده چه نوع MQTT داریم، ما 14 تا داریم که هر کدامیکنده unique value دارند.

IoT Application Layer Protocols-MQTT

MQTT Message Types

Message Type	Value	Flow	Description
CONNECT	1	Client to server	Request to connect
CONNACK	2	Server to client	Connect acknowledgement
PUBLISH	3	Client to server Server to client	Publish message
PUBACK	4	Client to server Server to client	Publish acknowledgement
PUBREC	5	Client to server Server to client	Publish received
PUBREL	6	Client to server Server to client	Publish release
PUBCOMP	7	Client to server Server to client	Publish complete
SUBSCRIBE	8	Client to server	Subscribe request
SUBACK	9	Server to client	Subscribe acknowledgement
UNSUBSCRIBE	10	Client to server	Unsubscribe request
UNSUBACK	11	Server to client	Unsubscribe acknowledgement
PINGREQ	12	Client to server	Ping request
PINGRESP	13	Server to client	Ping response
DISCONNECT	14	Client to server	Client disconnecting

نوع مسیح هارو اینجا میتویم.

- Connect, CONNACK: برای برقراری ارتباط
- PUBLISH: برای ارسال پیام پابلیش

- PUBACK • این که بگه پایلیش با موفقیت انجام شد
- PUBLISH • دیکه release کنیم و ارسال نشه
- SUBREL • این که تقاضا کنه میخواهد ساپسکرایب کنه
- SUBSCRIBE • که شون بده با موفقیت انجام شد
- SUBACK • این که بینیم هست یا نه
- PINGREQ • مسیح های دیگه هم خودمون میتوئیم اضافه کنیم ولی اینا به صورت استاندارد قرار داده شده.

IoT Application Layer Protocols-MQTT

- The MQTT protocol defines two types of entities in the network:
 - a message broker
 - a number of clients.
- The broker is a server that receives all messages from the clients and then routes those messages to relevant destination clients.
- A client is anything that can interact with the broker to send and receive messages.
 - A client could be an IoT sensor in the field or an application in a data center that processes IoT data.
 - Both publishers and subscribers are called as clients

ما تعدادی کلاینت دائم که میتوانن pub و sub بکنن. Broker هم از لحاظ ماهیتی یدونس ولی میشه چندتا هم گذاشت.

IoT Application Layer Protocols-Terminology of MQTT

- Client:

- Clients can be persistent or transient.
 - Persistent clients maintain a session with the broker
 - transient clients are not tracked by the broker.
- Clients often connect to the broker through libraries and SDKs.
 - There are over a dozen libraries available for C, C++, Go, Java, C#, PHP, Python, Node.js, and Arduino.

کلاینت ها میتوانن دائمی و موقت باشن، اونا که Persistent session خستن را با broker فعال نگاه میدارند. ولی اونا که transient session یه لحظه به برقرار میشه و داده رو میگیره و میره. نکته ای که هست اینه که broker اونایی که transient رو track نمیکنه، هارو track میکنه ببینه میشه فرستاد یا نه.

IoT Application Layer Protocols-Terminology of MQTT

- Broker

- The broker is the software that receives all the messages from the publishing clients and sends them to the subscribing clients.
- It holds the connection to persistent clients.
- Since the broker can become the bottleneck or result in a single point of failure, it is often clustered for scalability and reliability.
 - It is up to the implementers to decide how to create a scalable broker layer.
- MQTT brokers include
 - Commercial: [HiveMQ](#), [Xively](#), [AWS IoT](#), and [Loop](#),
 - an open source: [Mosquitto](#).

برای این که broker دچار bottleneck نشود هم باید از لحاظ پردازش قوی باشد هم باید پهنای باند زیادی داشته باشد.

برای این که broker ماscalable باشد به شکل reliable عمل کند که به کاری که می‌شده کرد اینه که redundancy رو برد بالا چون متلا اگر دیتابیس یک از دست رفت در یک دیگه باشد و ... یک دیگه از کاربرد های redundancy هم distributed load balancing هست که 10 تا 20 تا سور فعال داریم و یک load balancing مامنیاریم که اگر یکی پردازش 40 درصدی داشته باشد 0 درصد و درخواست 40 درصدیه همینطور داره میره بالا میایم پخش میکنیم که از 50 درصد بالاتر نرن.

IoT Application Layer Protocols-MQTT

- With MQTT, clients can subscribe to all data (using a wildcard character) or specific data from the information tree of a publisher.
- In addition, the presence of a message broker in MQTT decouples the data transmission between clients acting as publishers and subscribers.
 - In fact, publishers and subscribers do not even know (or need to know) about each other.
 - A benefit of having this decoupling is that the MQTT message broker ensures that information can be buffered and cached in case of network failures.
 - This also means that publishers and subscribers do not have to be online at the same time.

در MQTT دینا بین subscriber و publisher را couple میکنیم که مزینش.

این که ما به عنوان client ساپسکرایب کنیم به ازای تمام دینا های داخل broker یا بخشیش، نیاز مندی یک فرمت استاندارد تعريف کنیم که میگن topic.

IoT Application Layer Protocols-Terminology of MQTT

• Topic

- A topic in MQTT is an endpoint to that the clients connect.
- It acts as the central distribution hub for publishing and subscribing messages.
- A topic is a well-known location for the publisher and subscriber.
- Topics are simple, hierarchical strings, encoded in UTF-8, delimited by a forward slash.
 - For example, building1/room1/temperature and building1/room1/humidity are valid topic names. Subscribers can choose to subscribe to a specific topic or all the subtopics through wildcards.
 - A subscription to building1/+/temperature will automatically subscribe to the temperature topic of all the rooms in the building.
 - Similarly, building1/#/ will match all the topics available under building1.

به عنوان مثال فرض شود میخوایم ساختمان کامپیوتر building1 باش، طبق سوم بشه floor3، اتاق شماره 1 اشو میخوایم subscribe کنیم میشه building1/floor3/room1/temperature. نگاه کنیم یه جا نوشته building1/+/temperature، جای + پاید room قرار میگرفت وقتی + میداریم یعنی همه room ها. وقتی هم #/ building1/#/ تمام sub topic ها رو شامل میشه.

فرض کنیم میخوایم آنسانسور رو مبتنی بر MQTT هوشمند کنیم. به کاری که میشه کرد اینه که بین طبقه های مختلف efficient بره، مثلا اگر در طبقه 4 عه و یکی در طبقه 1 نیازش دارد، در طبقه 2 هم یه نفر داره نزدیک به آنسانسور میشه، بهتره که اول 2 رو سوار کنه بعد به 1 برسه. پس یک motion detector میذاریم، اگر دیتا یک ماه رفت و آمد یک آدم رو ضبط کنیم مشخص میشه 10:45 ترک میکنه 12:15 برمیگردد. و learn میشه که کیا چه فردی چه حرکتی میکنه. پس متوجه میشه 4 دقیقه دیگه قراره بیاد دم آنسانسور و میتوانه الگوریتمی اجرا کنه که حرکت کنه یعنی کمترین مصرف انرژی و تاخیر رو داشته باشه. (اگر چندتا آنسانسور باشه یه طوری هماهنگ بشن که در کل بهینه بشن).

حالا ما که میخوایم ورویدی بگیریم با Motion detector میشوند آنسانسور بگه مثلا تمام طبقات motion detector هاشو میخواهون یا مثلا نه کسایی که نزدیک آنسانسور هستن رو میخواهون، برای همه اینا بر اساس topic میشه subهای مختلف کرد.

IoT Application Layer Protocols-Terminology of MQTT

• Topic

- The pound sign (#) is a wildcard character that matches any number of levels within a topic. The multilevel wildcard represents the parent and any number of child levels.
 - For example, subscribing to `adt/lora/adeunis/#` enables the reception of the whole subtree, which could include topic names such as the following:
 - `adt/lora/adeunis/0018B2000000E9E`
 - `adt/lora/adeunis/0018B2000000E8E`
 - `adt/lora/adeunis/0018B2000000E9A`
- The plus sign (+) is a wildcard character that matches only one topic level.
 - For example, `adt/lora/+` allows access to `adt/lora/adeunis/` and `adt/lora/abeeway` but not to `adt/lora/adeunis/0018B2000000E9E`.
- Topic names beginning with the dollar sign (\$) must be excluded by the server when subscriptions start with wildcard characters (# or +).
 - Often, these types of topic names are utilized for message broker internal statistics. So messages cannot be published to these topics by clients. For example, a subscription to `+/monitor/Temp` does not receive any messages published to `$SYS/monitor/Temp`. This topic could be the control channel for this temperature sensor.

IoT Application Layer Protocols-MQTT



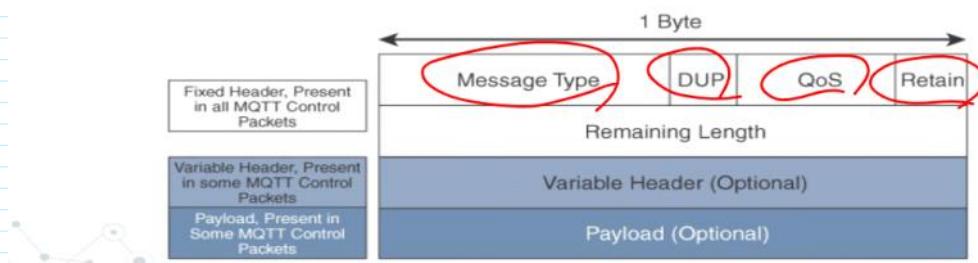
- MQTT is extremely lightweight
 - it takes up almost no space in a device, so that even small devices with very little computing power can use it.
- Each control packet consists of a 2-byte fixed header with optional variable header fields and optional payload.
 - Note that a control packet can contain a payload up to 256 MB.

خیلی سبکه، در سمت کلاینت دیوایس با قدرت پردازشی بسیار کم میتوانن کلاینت اش باشن، 2 بایت هدر فیکس داره و یک سری optional MQTT

IoT Application Layer Protocols-MQTT



- MQTT Message Format
 - Compared to the CoAP message format, MQTT contains a smaller header of 2 bytes compared to 4 bytes for CoAP



جلسه پیش در موردش یکم توضیح دادیم

IoT Application Layer Protocols-MQTT

- MQTT Message Types

Message Type	Value	Flow	Description
CONNECT	1	Client to server	Request to connect
CONNACK	2	Server to client	Connect acknowledgement
PUBLISH	3	Client to server Server to client	Publish message
PUBACK	4	Client to server Server to client	Publish acknowledgement
PUBREC	5	Client to server Server to client	<u>Publish received</u>
PUBREL	6	Client to server Server to client	Publish release
PUBCOMP	7	Client to server Server to client	<u>Publish complete</u>
SUBSCRIBE	8	Client to server	Subscribe request
SUBACK	9	Server to client	Subscribe acknowledgement
UNSUBSCRIBE	10	Client to server	Unsubscribe request
UNSUBACK	11	Server to client	Unsubscribe acknowledgement
PINGREQ	12	Client to server	Ping request
PINGRESP	13	Server to client	Ping response
DISCONNECT	14	Client to server	Client disconnecting

• Publish Receive: مطمئن میکنند ما داده publish شده رو دریافت کردیم
 • Publish complete: برای برقراری یک سرویسی هست و میخوایم مطمئن بشیم یک بار دریافت شده و duplicate نشده.

IoT Application Layer Protocols-MQTT

- *MQTT Message Types*
 - PINGREQ/PINGRESP control packets are used to validate the connections between the client and server.
 - Similar to ICMP pings that are part of IP, they are a sort of keep alive that helps to maintain and check the TCP session.

IoT Application Layer Protocols-MQTT

- DUP (Duplication Flag) field:
 - This flag, when set, allows the client to note that the packet has been sent previously, but an acknowledgement was not received.
- The QoS header field:
 - allows for the selection of three different QoS levels.
- Retain flag field:
 - Only found in a PUBLISH message, the Retain flag notifies the server to hold onto the message data.
 - This allows new subscribers to instantly receive the last known value without having to wait for the next update from the publisher.

• DUP: موقعی سمت میشه که کلینت بخواهد اعلام کنه که پکی قبلاً ارسال شده ولی ACK اش رو دریافت نکرده. حالا این که چرا این کار مهمه توضیح داده میشه. ما QoS داریم و میخوایم مطمئن بشیم یک بار ارسال میشه اونجا از DUP استفاده میکنیم
 • QoS Header Field: مشخص میکنه 3 تا سطح QoS رو که توضیح داده میشه جلوتر
 • Retain flag field: جاش اینجاست:



یک flag ایه که به سرور میگه به اون مسیح دسترسی داره و به subscriber last known value که اجازه میده که اون را دریافت کنه. ما یک لحظه Subscriber انجام میدیم همزمان با درخواست subscription یک داده ای میفرستیم این فلگ بهش میگه last known value.

IoT Application Layer Protocols-MQTT

- Remaining Length:
 - Specifies the number of bytes in the MQTT packet following this field.

IoT Application Layer Protocols-MQTT

• QoS 0:

- This is a best-effort and unacknowledged data service referred to as “at most once” delivery.
- The publisher sends its message one time to a server, which transmits it once to the subscribers.
- No response is sent by the receiver, and no retry is performed by the sender.
- The message arrives at the receiver either once or not at all.

• QoS 1:

- This QoS level ensures that the message delivery between the publisher and server and then between the server and subscribers occurs at least once.
- In PUBLISH and PUBACK packets, a packet identifier is included in the variable header.
- If the message is not acknowledged by a PUBACK packet, it is sent again.
- This level guarantees “at least once” delivery.

ما سه سطح داریم برash. در کاربردهای CoAP که مبتنی بر UDP بود ولی برای Reliability اوردمیم CON و NON رو درست کردیم که در CON میگرفتیم مطمئن شیم بسته دریافت شد. یک نوع QoS هست در واقع که UDP فرآهم نمیکرد.

MQTT در لایه اپلیکیشن ممکنه برای یک سری کیفیت سرویس ها که در برخی کاربردها مورد نیاز هست میباشد سطح درست میگیرند برashون. • QoS 0: مثل حالت best-effort ع. وقتی میگیم یک ترافیک best-effort عه یعنی تضمین شده نیست ولی بهترین تلاششو ممکنه که پکت ارسال بشه. پس نهایتاً ممکنه یک بار به گیرنده برسه

مثل پابلیشر یک بار داده رو به سرور ارسال میکنه و سرور (broker) یک بار به subscriber میفرسته و حالا این با دریافت میشه یا نمیشه و هیچ تلاشی برای ارسال دوباره موجود نیست. یعنی همین داده دوباره ارسال نمیشه (وقتی آپدیت بشه مسلماً ارسال میشه)

• QoS 1: یک سطح بالاتر و مطمئن میشه که یک مسیج بین publisher و سرور، وین server و subscriber و حداقل یک بار برسه. برای این کار PUBACK ارسال میشه تا اطمینان حاصل بشه که ارسال شده.

• QoS 2: برای اپلیکیشن های حساس و critical، برای موقعی که میخوایم مطمئن بشیم نه loss داریم نه duplication. یعنی میخوایم مطمئن باشیم دقیقاً یک بار دریافت شده. برای این کار در گام اول یک PUBLISH/PUBREC دریافت میشه و در فاز دوم PUBREL/PUBCOMP رو داریم برای اینکه اطلاع بدیم که بار دریافت کردیم و complete کنه قضیه رو. پس یک guaranteed service داره که اطمینان بده دقیقاً یکبار ارسال شده

IoT Application Layer Protocols-MQTT

- The QoS process is symmetric in regard to the roles of sender and receiver, but two separate transactions exist.

- One transaction occurs between the publishing client and the MQTT server,
- and the other transaction happens between the MQTT server and the subscribing client.

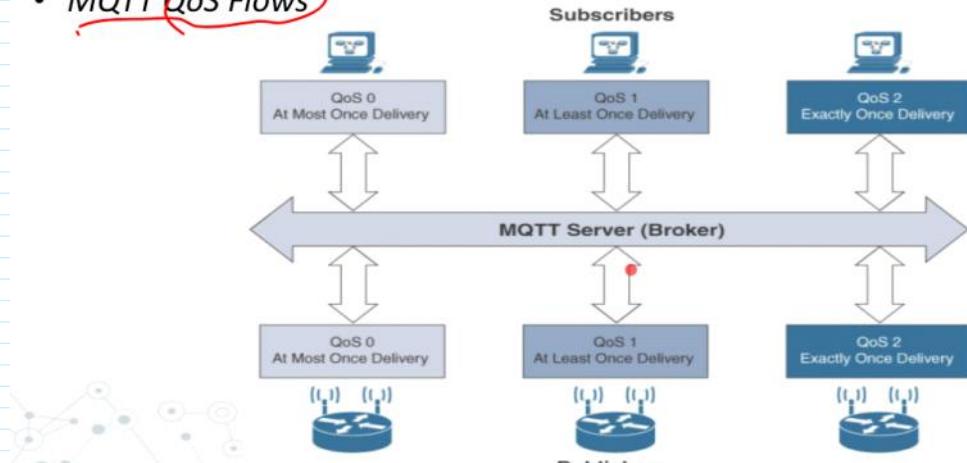
- The publishing client side sets the QoS level for communications to the MQTT server.

- On the other side, the client subscriber sets the QoS level through the subscription with the MQTT server.
 - In most cases, QoS remains the same between clients and broker end to end.
 - However, you should be aware that in some scenarios, QoS levels change and are not the same end to end.

فرایند میتوانه symmetric باشه، یعنی ارتباط هم بین publisher و broker باشه هم بین subscriber و broker باشه، البته معماری اجازه اینو میده که QoS بین 2 طرف متفاوت هم باشه.

IoT Application Layer Protocols-MQTT

- MQTT QoS Flows



IoT Application Layer Protocols-MQTT

- MQTT sessions between each client and server consist of four phases:
 - session establishment,
 - authentication,
 - data exchange, and
 - session termination.
- Each client connecting to a server has a unique client ID, which allows the identification of the MQTT session between both parties.
 - When the server is delivering an application message to more than one client, each client is treated independently.

وقی میخواه یک session بین client و server درست بشه 4 تا فاز داره. فاز آخر برای اونا ک transient هستن اتفاق میافته برای بقیه سعی بر اینکه برقرار بمونه ارتباط.

IoT Application Layer Protocols- MQTT Security

- Securing MQTT connections through TLS is considered optional because it calls for more resources on constrained nodes.
- When TLS is not used, the client sends a clear-text username and password during the connection initiation.
- MQTT server implementations may also accept anonymous client connections (with the username/password being "blank").
 - When TLS is implemented, a client must validate the server certificate for proper authentication.
 - Client authentication can also be performed through certificate exchanges with the server, depending on the server configuration.

ما میتوانیم ارتباط MQTT را بر اساس TLS امنش بکنیم. یک فیلد Optional قرار دادن که قرار بدیم میشه امنش کرد. ولی حواسمنوں باید باشه و قی از الگوریتم رمزگاری و رمزگشایی داریم استفاده میکنیم نیاز به منابع بیشتر در constrained node داریم. در وقتی که TLS نداریم یک clear-text داریم و کسی اون وسط شنود کنه میتوانه مورد استفاده قرار بده.

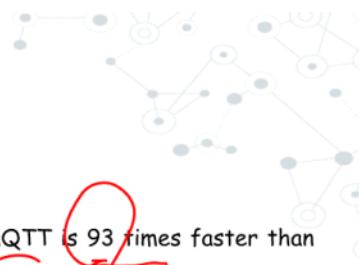
البته بنابر کاربرد میشه anonymous client داشته باشیم یعنی بوزنیم پسوردش خالی باشه و مهم برامون نباشه طرف authenticate باشه.

IoT Application Layer Protocols

- MQTT vs HTTP
 - MQTT is data centric whereas HTTP is document-centric.
 - HTTP is request-response protocol for client-server computing and not always optimized for mobile devices. Whereas MQTT is lightweight protocol (MQTT transfers data as a byte array) and publish/subscribe model, which makes it perfect for resource-constrained devices and help to save battery.
 - MQTT has pretty short specification. There are only CONNECT, PUBLISH, SUBSCRIBE, UNSUBSCRIBE and DISCONNECT types that are significant for developers. Whereas HTTP specifications are much longer.
 - MQTT has a very short message header and the smallest packet message size of 2 bytes

عموما MQTT data centric است. اگر چه HTTP document-centric هست (مثل عکس و ...) منظور از short specification همین 5 تاں ولی در HTTP کلی چیز هست چون برای ارتباط بین انسان ها درست شده.

IoT Application Layer Protocols



MQTT vs HTTP

- According to measurements in 3G networks, throughput of MQTT is 93 times faster than HTTP's.
- Besides, in comparison to HTTP, MQTT Protocol ensures high delivery guarantees. There are 3 levels of Quality of Services:
 - at most once: guarantees a best effort delivery.
 - at least once: guaranteed that a message will be delivered at least once. But the message can also be delivered more than once.
 - exactly once: guarantees that each message is received only once by the counterpart

یک تستی که در شبکه 3G انجام شده، MQTT 93 برابر سریع تر از HTTP است. طبیعیم هست چون هدراش کمتره

IoT Application Layer Protocols



MQTT vs HTTP

Features	MQTT	HTTP
Full form	Message Queue Telemetry Transport	Hyper Text Transfer Protocol
Architecture	It has publish/subscribe architecture. Here devices can publish any topics and can also subscribe for any topics for any updates.	It has request/response means Client/Server architecture.
Upper layer protocol	It runs over TCP.	It runs over TCP and UDP.
<u>message size</u>	<u>small</u> ,	<u>Large</u> .
Message format	binary with 2Byte header	ASCII format.
Data distribution	1 to 0/1/N	one to one only , more POST
<u>Data security</u>	Yes, It uses SSL/TLS for security	NO, hence <u>HTTPS</u> is used to provide data security
Complexity	<u>Simple</u>	Client more complex (ASCII parser)
Encryption	<u>It encrypts</u> payload i.e. it is payload agnostic	data are not encrypted before transmission
When to use	if your project is to let the fridge to communicate with the thermometer to adapt the engine pump, you can use the MQTT easily	if you need to collect big data from around the world, then you can think to use HTTP

<https://intduinis.com/mqtt-and-http/>

IoT Application Layer Protocols

- **CoAP vs MQTT**

Factor	CoAP	MQTT
Main transport protocol	UDP	TCP
Typical messaging	Request/response	Publish/Subscribe
Effectiveness in LLNs	Excellent	low
Security	DTLS	SSI/TLS
Communication model	one-to-one	Many-to-many
Strengths	Lightweight, fast, low overhead, support for multicasting messages	Robust communication, simple management, scalability
Weakness	Not as reliable as MQTT	Higher overhead, no multicasting support

شبکه در لایه 1 و 2، ما شبکه که Loss زیاده، کدوم بکی بهتره؟ درسته که MQTT از TCP استفاده میکنه ولی انتخاب مناسبی نیست. چون مثلما Sigfox ما محدودیت در ارسال داریم و ممکنه به روز طول بکشه تا اطمینان داشته باشیم دریافت بشه. در شبکه های lossy lossy 60 درصد بسته ها تو راه از بین میرن. یعنی از 10 تا 6 تاش ممکنه تو مسیر رفت مشکل بشه. ممکنه از 10 تا ACK در برگشت هم 6 تاش از بین بره. بعد از اول بسته ارسال میشه و همینطوری بدتر میشه و loss خوبی زیاد میشه. ولی در UDP میگیم حالا شاید اصن مهم نیست رطوبت داریم میفرستیم یه ساعت بعد میفرستیم. بنابراین MQTT برای LLN اصلا خوب نیست.

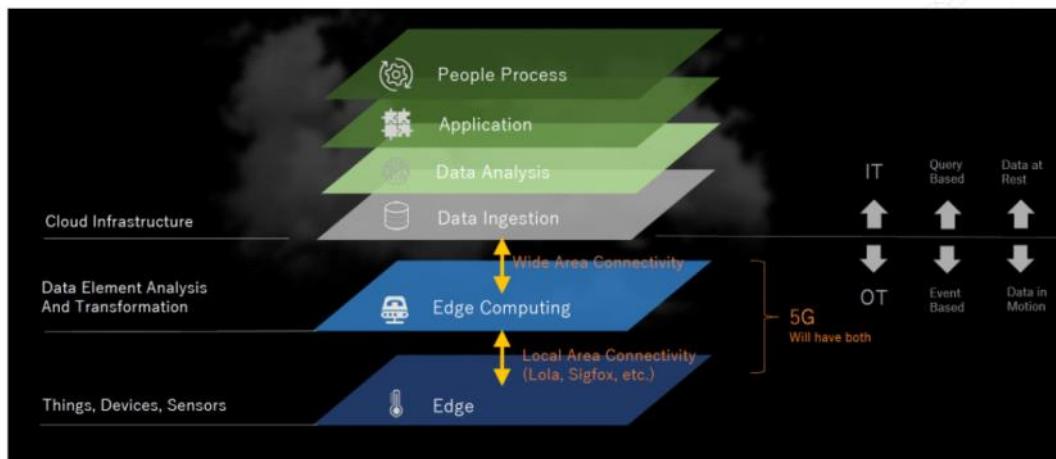
Factor	CoAP	MQTT
Main transport protocol	UDP	TCP
Typical messaging	Request/response	Publish/Subscribe
Effectiveness in LLNs	Excellent	low
Security	DTLS	SSI/TLS
Communication model	one-to-one	Many-to-many
Strengths	Lightweight, fast, low overhead, support for multicasting messages	Robust communication, simple management, scalability
Weakness	Not as reliable as MQTT	Higher overhead, no multicasting support

Convergence of Technologies

- IoT
- Big Data
- Artificial Intelligence
- Blockchain
- Edge Computing and Fog Computing
- 6-Generation (6G) Wireless Networks
- Privacy and Security

ما فناوری های مختلفی داریم به خصوص در مهندسی کامپیوتر.
حالا میخوایم بینهم IoT در مقابل باقیه تکنولوژی ها به چه صورت هستش.

Convergence of Technologies



اگر IoT را بررسی کنیم. ما یک سری things داریم و یک دیتابی رو تولید میکنیم.
یک جای ذخیره شده: **Data at rest**
در شبکه ارتباطی در حال ارتباطه و داره از یک Node میره به اون یکی: **Data at motion**

ما 3 نوع Computing داریم:
• Cloud: همه جی تحت عنوان سرویس باشن و در اختیار Node ها قرار میگیرن.

مزایا:

- قدرت پردازشی در یک جا انجام میشه به صورت قدرمند
- هزینه ها کاهش پیدا میکنه

معایب:

- تاخیر: یک الگوریتم هوش مصنوعی مثلا داره speech detection انجام میشه. مثلا وقتی میگیم چراغ خاموش شو بتونه تشخیص بده ما چه کسی هستیم (Authentication) و سپس متوجه بشه چه چیزی رو گفتیم خاموش کنه (چراغ). بایستی الگوریتمی اجرا بشه. این الگوریتم اگر در Cloud اجرا بشه دچار یک تأخیر میشه تاخیر ایجاد شده ناشی از فاصله زیاد بین Node تا Cloud است.

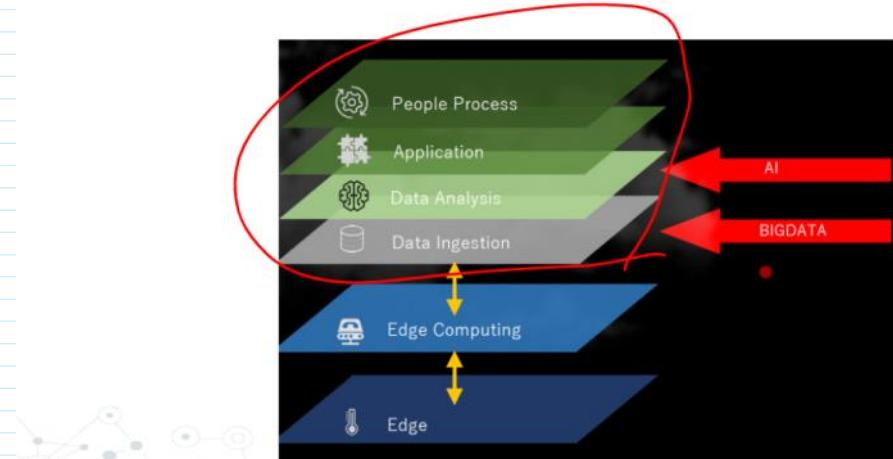
اگر بخواهیم cloud استفاده کنیم تاخیرم داشته باشیم باید در IoT طرف رو مجبور کنیم یک سرور قدرمند مثلا در خانه هوشمندش نصب کنه که به صرفه نیست.

• وقی کارای پردازشی رو در gateway بسپاریم که فاصله زیادی هم داشته باشه و تاخیر به وجود نیاد میشه edge computing

- Fog: تفاوت بین Edge و Fog اینه که Edge عموما در لیه شبکه است ولی Fog یک Node کوچیک رو موبایل هم میتونه Fog node باشه و محاسبات کوچیک رو انجام بده ولی اگر محاسبات بیشتر شد دیگه برون سپاری میکنه و به کلاه میده.

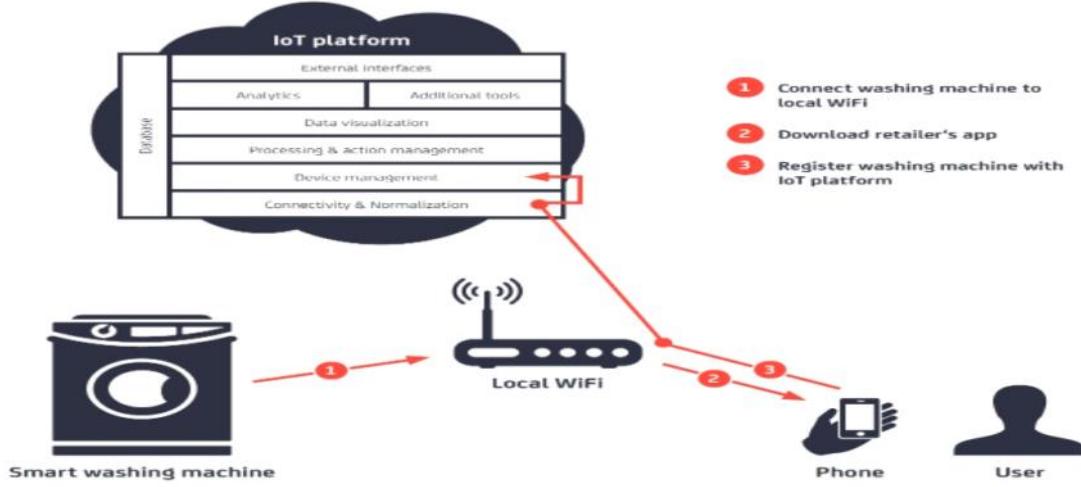
داده های زیادی توسط IoT تولید می شه. این داده ها آنالیز و تحلیلی می خواه و بعضی از real-time Edge computing و Fog computing عه پس نیاز به داریم.

Convergence of Technologies



این که آنالیز چطوری انجام بشه میشه بحث های AI و Big data

IoT Platforms



پلتفرم IoT، سرویس های یکسان و مشترک رو در قالب Open API یا API به اپلیکیشن های مختلف میدن. کارایی مثل big data analysis, Data management, device management. و بحث های

Introduction

- Traditional data management systems are simply unprepared for the demands of what has come to be known as "big data."
- The real value of IoT is not just in connecting things but rather in
 - the data produced by those things,
 - the new services you can enable via those connected things,
 - and the business insights that the data can reveal.
- However, to be useful, the data needs to be handled in a way that is organized and controlled.
 - Thus, a new approach to data analytics is needed for the Internet of Things.

IoT پک از درایور های اصلی Big data است.
نکته مهم: ارزش اصلی IoT، در دیناتی است که در things ها تولید می شون.
برای این که این داده ها مفید باشند یک approach برای data analysis نیاز داریم.

Data Analytics for IoT

- Quote from a Machina Research (is now part of Gartner) recent report:

"The significance of the Internet of Things is not that more and more devices, people and systems are 'connected' with one another. It is that the data generated from these 'things' is shared, processed, analysed and acted upon through new and innovative applications, applying completely new analysis methods and within significantly altered timeframes. The Internet of Things will drive big data, providing more information, from many different sources, in real-time, and allow us to gain completely new perspectives on the environments around us."

Data Analytics for IoT

- Quote from a [Machina Research](#) (is now part of Gartner) recent report:

"The significance of the Internet of Things is not that more and more devices, people and systems are 'connected' with one another. It is that the data generated from these 'things' is shared, processed, analysed and acted upon through new and innovative applications, applying completely new analysis methods and within significantly altered timeframes. The Internet of Things will drive big data, providing more information, from many different sources, in real-time, and allow us to gain completely new perspectives on the environments around us."

Data Analytics for IoT

- Analyzing the massive amount of data in the most efficient manner possible falls under the umbrella of data analytics
- Data analytics must be able to offer actionable insights and knowledge from data, no matter the amount or style, in a timely manner, otherwise, the full benefits of IoT cannot be realized.

یک حجم زیادی داده داریم که باید به شکل کارآمدی آنالیز بشه. این آنالیز باید یک بینشی ایجاد کنه که بعد پشه روش یک اکشنی انجام داد و یک دانشی بین داده رو برامون ایجاد کنه.

Data Analytics for IoT

- Before diving deeper into data analytics, it is important to define a few key concepts related to data.
- For one thing, not all data is the same; it can be categorized and thus analyzed in different ways.
- Depending on how data is categorized, various data analytics tools and processing methods can be applied.
- Two important categorizations from an IoT perspective are whether the data is structured or unstructured and whether it is in motion or at rest.

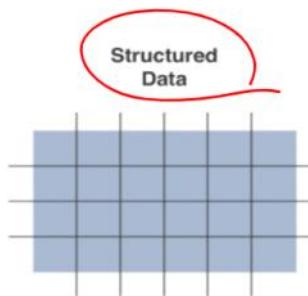
Data Analytics for IoT



- Before diving deeper into data analytics, it is important to define a few key concepts related to data.
- For one thing, not all data is the same; it can be categorized and thus analyzed in different ways.
- Depending on how data is categorized, various data analytics tools and processing methods can be applied.
- Two important categorizations from an IoT perspective are whether the data is structured or unstructured and whether it is in motion or at rest.

Data Analytics for IoT

Structured Versus Unstructured Data



Organized Formatting
(e.g., Spreadsheets, Databases)



Does not Conform to a Model
(e.g., Text, Images, Video, Speech)

Data Analytics for IoT

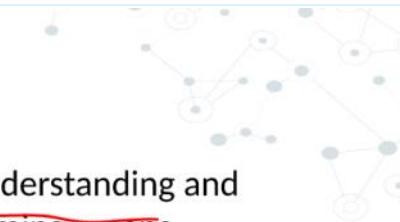
- Structured data means that the data follows a model or schema that defines how the data is represented or organized, meaning it fits well with a traditional relational database management system (RDBMS).
 - from banking transaction and invoices to computer log files and router configurations.
 - Structured data is easily formatted, stored, queried, and processed; for these reasons, it has been the core type of data used for making business decisions.
- IoT sensor data often uses structured values, such as temperature, pressure, humidity, and so on, which are all sent in a known format.

Data Analytics for IoT



- Because of the highly organizational format of structured data, a wide array of data analytics tools are readily available for processing this type of data.
- From custom scripts to commercial software like Microsoft Excel and Tableau, most people are familiar and comfortable with working with structured data.

Data Analytics for IoT



- Unstructured data lacks a logical schema for understanding and decoding the data through traditional programming means.
 - Examples of this data type include text, speech, images, and video. As a general rule, any data that does not fit neatly into a predefined data model is classified as unstructured data.
- According to some estimates, around 80% of a business's data is Unstructured.

Data Analytics for IoT



- Because of this fact, data analytics methods that can be applied to unstructured data, such as **cognitive computing and machine learning**, are deservedly garnering a lot of attention.
 - With machine learning applications, such as natural language processing (NLP), you can decode speech.
- With image/facial recognition applications, you can extract critical information from still images and video.

ما در Structured data روش های آنالیزهای تقریباً روش های مشخصه if-then باشند. مثلاً بگیم اونایی که معدله شدن از یه حد بالاتر هست یک تصمیمی روش انجام شه.

اما در مورد Unstructured data، ما نمیتوانیم خیلی از مکانیزم های Rule based را استفاده کنیم. مثلاً در فرمانی صوتی نمیتوانیم بگیم اگر این روگفت یعنی این. چون این معلوم نیست. یک فرمتنداره و باید learn بشه. پس از Machine learning و cognitive computing استفاده میکنیم.

Data Analytics for IoT

• Data in Motion Versus Data at Rest

– As in most networks, data in IoT networks is either in transit ("data in motion") or being held or stored ("data at rest"):

- Examples of data in motion include traditional client/server exchanges, such as web browsing and file transfers, and email.
- Data saved to a hard drive, storage array, or USB drive is data at rest.

داده ها یا یه چا ذخیره شدن یا در حال حرکتند.
وقتی در حال حرکت میشه همین داده که در MQTT یا CoAP در حال حرکتند
در حال ذخیره هم که تو اسلاید نوشته

Data Analytics for IoT

• From an IoT perspective, the data from smart objects is considered data in motion as it passes through the network and route to its final destination.

– This is often processed at the edge, using fog computing.

– When data arrives at the data center, it is possible to process it in real-time, just like at the edge, while it is still in motion.

• Tools with this sort of capability, such as Spark, Storm, and Flink, are relatively nascent compared to the tools for analyzing stored data.

• Data at rest in IoT networks can be typically found in IoT brokers or in some sort of storage array at the data center.

– Myriad tools, especially tools for structured data in relational databases, are available from a data analytics perspective.

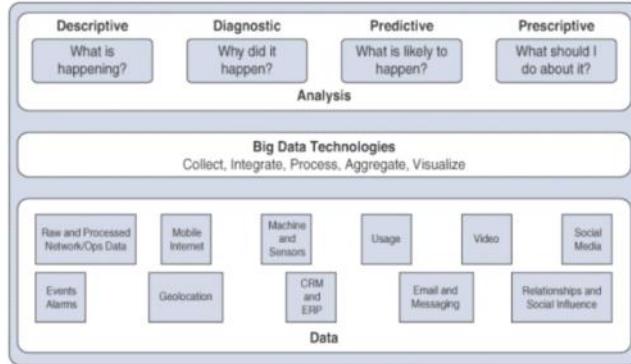
• The best known of these tools is Hadoop.

در IoT، اگه هایی که از smart object ها ارسال میشن رو میگیم in motion که معمولاً بایستی در edge یک پردازشی روشون رخ داده بشه.
اینا ابزار های خودشون دارن مثل Spark, Storm

در data at rest میشه اونایی که در IoT Broker platform ها با IoT ها ذخیره شده. یکی از ابزار های مهم برای آنالیزهای میشه Hadoop.

IoT Data Analytics Overview

- There are four types of data analysis results:



مستقل از این که از چه ابزاری داریم استفاده میکنیم آنالیز هامون 4 نوع هستش.
داده هامون 4 نوع دارند:

- Descriptive
- Diagnostic
- Predictive
- Prescriptive

IoT Data Analytics Overview

- **Descriptive:** Descriptive data analysis tells you what is happening either now or in the past.
- **Diagnostic:** When you are interested in the “why,” diagnostic data analysis can provide the answer.
- **Predictive:** Predictive analysis aims to foretell problems or issues before they occur.
- **Prescriptive:** Prescriptive analysis goes a step beyond predictive and recommends solutions for upcoming problems.

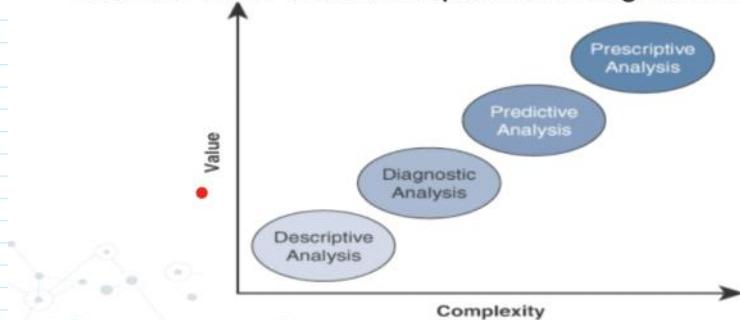
داره میگه داده چه چیزی (What) هست.

میگه یه اتفاق افتاده. جرا؟ (Why)

پیشیبی آینده. مشکلات که در آینده رخ میده رو قبل این که اتفاق بیافته رو پیشیبی میکنه. در واقع یک نوع What هستش ولی what ای که هنوز رخ نداده.

IoT Data Analytics Overview

- Both predictive and prescriptive analyses are more resource intensive and increase complexity, but the value they provide is much greater than the value from descriptive and diagnostic analysis.



از نظر ارزش و پیچیدگی، پیداست که Prescriptive Analysis هم پیچیدگی زیادی دارد هم ارزش زیادی دارد. و بقیه هم که دائم مشاهده میکنیم

یک سوال: عموماً در داده های که دوست داریم با تاخیر کم انجام بشن (Real time) و Unstructured هم هستن. معمولاً از چه ابزاری برای آنالیز استفاده میکنیم؟ ابزار های هوش مصنوعی. چون

1. ابزار های غیر هوش مصنوعی مثل rule based در مورد داده های Unstructured ممکن نیست
2. ابزار های Rule based و بهینه سازی پیچیدگی بالا و تاخیر زیادی دارند. پس میریم سراغ هوش مصنوعی که پیچیدگی کمتر دارد (در حالت کلی!)

IoT Data Analytics Overview

• Machine Learning Overview

– Supervised Learning:

- In supervised learning, the machine is trained with input for which there is a known correct answer

– Unsupervised Learning:

- In some cases, supervised learning is not the best method for a machine to help with a human decision.

– Reinforcement Learning

Big Data Analytics Tools and Technology

- Generally, the industry looks to the “three Vs” to categorize big data:
 - Velocity:** refers to how quickly data is being collected and analyzed.
 - Hadoop Distributed File System is designed to ingest and process data very quickly.
 - Smart objects can generate machine and sensor data at a very fast rate and require database or file systems capable of equally fast ingest functions.
 - Variety:** refers to different types of data.
 - Different database technologies may only be capable of accepting one of structured, semi-structured, or unstructured types.
 - Hadoop is able to collect and store all three types.

دیتا های IoT را میگیریم Big data هستن. که اونا هم مکانزیم و ابزار های خودش داره. عموما در 3 تا 7 داریم.

- Velocity: فرآینس تولید داده چقدر. در ماشین خودران Velocity خیلی بالا و بالعکس در کشاورزی خیلی کم.
- Variety: تنوع. مثلا به جا structure دائم یه جا semi-structure یه جا unstructured.
- Volume: مثلا در ماشین خودران این که فرمان وضعیتش چه مدلیه یک وضعیت Structured هستش (بر حسب درجه) یا Unstructured داره مثل دوربین که فاصله هرگذوام از اشیاه چقدر از ماشین حجم داده. مثلا دما و رطوبت حجم زیادی نداره ولی ویدیو HD خیلی حجم زیادی داره.

یه موقع هم هست مثل کشاورزی هوشمند، با این که رطوبت volume کمی داره. ولی از پس تعداد سنسورها زیاده باعث بشه volume بره بالا و اون موقع میشه گفت big data داریم.

*سامانه Courses رو نمیشه Big data گفت. درسته که میشه گفت Volume زیاده ولی خوب تو یه حجم ثابتی در یک peak ای قرار داره.

Big Data Value Chain



Collection – Structured, unstructured and semi-structured data from multiple sources

Ingestion – loading vast amounts of data onto a single data store

Discovery & Cleansing – understanding format and content; clean up and formatting

Integration – linking, entity extraction, entity resolution, indexing and data fusion

Analysis – Intelligence, statistics, predictive and text analytics, machine learning

Delivery – presenting results to end users based on their needs and availability

Need for Standardized Approaches At Each Step

Source O'Reilly Strata 2012
23

زنگره ارزش. باید در هر مرحله استاندارد سازی بشه.